COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

DESCRIPTIONAL COMPLEXITY OF FORMAL SYSTEMS

DISSERTATION THESIS

2019 Ing. Michal Hospodár

COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

DESCRIPTIONAL COMPLEXITY OF FORMAL SYSTEMS

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, 2019 Ing. Michal Hospodár

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

POPISNÁ ZLOŽITOSŤ FORMÁLNYCH SYSTÉMOV

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, 2019 Ing. Michal Hospodár

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

71429395

THESIS ASSIGNMENT

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

Title: Descriptional complexity of formal systems

Annotation: We study the state complexity of the concatenation operation under the assumption that both languages are accepted by deterministic finite automata with more final states, and as a result, we get the exact complexity of this operation on alternating finite automata. We completely solve the magic number problem for the cut operation for every alphabet size by getting some unattainable (magic) values in the unary case and showing that all possible values are attainable for an alphabet of size at least two. We describe the possible ranges of accepting state complexities for several operations. We get the exact nondeterministic state complexity of power and positive closure in all considered subclasses of convex languages. We consider the descriptional complexity of the forever operator on six different models of finite automata, and in 32 of 36 cases, we get the exact trade-off for this combined operation.

Aim:

1) Provide the state-of-the-art concerning descriptional complexity of formal systems.
2) Investigate the complexity of the concatenation operation on deterministic and alternating finite automata.
3) Find the range of possible state complexities of languages resulting from the cut operation.
4) For some regular operations, find the range of possible accepting state complexities.
5) Obtain the nondeterministic state complexities of power and positive closure on subclasses of convex languages.
6) Examine the descriptional complexity of the forever operator.

Literature:

1) Birget, J.-C.: Intersection and union of regular languages and state complexity. Inform. Process. Lett. 43(4), 185-190 (1992)
2) Birget, J.-C.: The state complexity of the forever operator and its connection with temporal logic. Inform. Process. Lett. 58(4) (1996) 185-188.
3) Brzozowski, J.A., Jirásková, G., Li, B.: Quotient complexity of ideal languages. Theoret. Comput. Sci. 470, 36-52 (2013)
4) Brzozowski, J., Jirásková, G., Li, B., Smith, J.: Quotient complexity of bifix-, factor-, and subword-free regular languages. Acta Cybernetica 21(4), 507-527 (2014)

5) Brzozowski, J.A., Jirásková, G., Zou, C.: Quotient complexity of closed languages. Theory Comput. Syst. 54(2), 277-292 (2014)
6) Brzozowski, J.A., Shallit, J., Xu, Z.: Decision problems for convex languages. Inform. And Comput. 209(3), 353-367 (2011)
7) Chandra, A.K., Kozen, D., Stockmeyer, L.J.: Alternation. J. ACM 28(1), 114-133 (1981)
8) Dassow, J.: On the number of accepting states of finite automata. J. Autom., Lang. Comb. 21(1-2), 55-67 (2016)
9) Drewes, F., Holzer, M., Jakobi, S., van der Merwe, B.: Tight bounds for cut-operations on deterministic finite automata. Fundam. Inform. 155(1-2), 89-110 (2017)
10) Fellah, A., Jürgensen, H., Yu, S.: Constructions for alternating finite automata. Int. J. Comput. Math. 35(1-4), 117-132 (1990)
11) Holzer, M., Kutrib, M.: Nondeterministic descriptional complexity of regular languages. Internat. J. Found. Comput. Sci. 14, 1087-1102 (2003)
12) Maslov, A.N.: Estimates of the number of states of finite automata. Soviet Math. Doklady 11, 1373-1375 (1970)
13) Rabin, M.O., Scott, D.: Finite automata and their decision problems. IBM Journal of Research & Development 3, 114-125 (1959)
14) Yu, S., Zhuang, Q., Salomaa, K.: The state complexities of some basic operations on regular languages. Theoret. Comput. Sci. 125(2), 315-328 (1994)

Keywords: regular languages, finite automata, descriptional complexity, regular operations, subclasses of convex languages

Tutor:	RNDr. Galina Jirásková, CSc.
Department:	FMFI.KAMŠ - Department of Applied Mathematics and Statistics
Head of department:	prof. RNDr. Marek Fila, DrSc.
Assigned:	18.08.2015
Approved:	18.08.2015 prof. RNDr. Anatolij Dvurečenskij, DrSc. Guarantor of Study Programme

....

Student Tutor

71429395

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

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

Názov: Descriptional complexity of formal systems
Popisná zložitosť formálnych systémov

Anotácia: Študujeme stavovú zložitosť zreťazenia za predpokladu, že oba jazyky sú akceptované deterministickými konečnými automatmi s viacerými koncovými stavmi, a ako výsledok dostávame presnú hodnotu zložitosti tejto operácie na alternujúcich konečných automatoch. Úplne vyriešime problém magických čísel pre operáciu strojového zreťazenia a všetky veľkosti abecedy tým, že získame nedosiahnuteľné (magické) hodnoty v unárnom prípade a ukážeme, že všetky možné hodnoty sú dosiahnuteľné v prípade binárnej abecedy, respektíve akejkoľvek abecedy veľkosti aspoň dva. Popíšeme možné rozsahy akceptačných stavových zložitostí pre niekoľko regulárnych operácií. Dostaneme presnú nedeterministickú stavovú zložitosť mocniny a pozitívneho uzáveru vo všetkých uvažovaných podtriedach konvexných jazykov. Uvažujeme popisnú zložitosť operátora forever na šiestich rôznych modeloch konečných automatov a v 32 z 36 prípadov dostávame presný prevod pre túto kombinovanú operáciu.

Cieľ:

1) Zhrnúť súčasný stav problematiky v oblasti popisnej zložitosti formálnych systémov.
2) Skúmať zložitosť operácie zreťazenia na deterministických a alternujúcich konečných automatoch.
3) Nájsť rozsah možných stavových zložitostí jazykov získaných aplikáciou operácie strojového zreťazenia.
4) Nájsť rozsah možných akceptačných stavových zložitostí pre viaceré regulárne operácie.
5) Získať nedeterministickú stavovú zložitosť mocniny a pozitívneho uzáveru na podtriedach konvexných jazykov.
6) Skúmať popisnú zložitosť operátora forever.

Literatúra:

1) Birget, J.-C.: Intersection and union of regular languages and state complexity. Inform. Process. Lett. 43(4), 185-190 (1992)
2) Birget, J.-C.: The state complexity of the forever operator and its connection with temporal logic. Inform. Process. Lett. 58(4) (1996) 185-188.
3) Brzozowski, J.A., Jirásková, G., Li, B.: Quotient complexity of ideal languages. Theoret. Comput. Sci. 470, 36-52 (2013)

4) Brzozowski, J., Jirásková, G., Li, B., Smith, J.: Quotient complexity of bifix-, factor-, and subword-free regular languages. Acta Cybernetica 21(4), 507-527 (2014)
5) Brzozowski, J.A., Jirásková, G., Zou, C.: Quotient complexity of closed languages. Theory Comput. Syst. 54(2), 277-292 (2014)
6) Brzozowski, J.A., Shallit, J., Xu, Z.: Decision problems for convex languages. Inform. And Comput. 209(3), 353-367 (2011)
7) Chandra, A.K., Kozen, D., Stockmeyer, L.J.: Alternation. J. ACM 28(1), 114-133 (1981)
8) Dassow, J.: On the number of accepting states of finite automata. J. Autom., Lang. Comb. 21(1-2), 55-67 (2016)
9) Drewes, F., Holzer, M., Jakobi, S., van der Merwe, B.: Tight bounds for cut-operations on deterministic finite automata. Fundam. Inform. 155(1-2), 89-110 (2017)
10) Fellah, A., Jürgensen, H., Yu, S.: Constructions for alternating finite automata. Int. J. Comput. Math. 35(1-4), 117-132 (1990)
11) Holzer, M., Kutrib, M.: Nondeterministic descriptional complexity of regular languages. Internat. J. Found. Comput. Sci. 14, 1087-1102 (2003)
12) Maslov, A.N.: Estimates of the number of states of finite automata. Soviet Math. Doklady 11, 1373-1375 (1970)
13) Rabin, M.O., Scott, D.: Finite automata and their decision problems. IBM Journal of Research & Development 3, 114-125 (1959)
14) Yu, S., Zhuang, Q., Salomaa, K.: The state complexities of some basic operations on regular languages. Theoret. Comput. Sci. 125(2), 315-328 (1994)

Kľúčové slová: regulárne jazyky, konečné automaty, popisná zložitosť, regulárne operácie, podtriedy konvexných jazykov

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

Dátum zadania: 18.08.2015

Dátum schválenia: 18.08.2015 prof. RNDr. Anatolij Dvurečenskij, DrSc.
garant študijného programu

.....
študent

.....
školiteľ

Declaration

I here declare that I have written this dissertation thesis without any prohibited assistance of third parties and without using aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such.

This dissertation thesis was conducted from 2015 to 2019 under the supervision of RNDr. Galina Jirásková, CSc. during PhD study at the Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava, Slovak Republic. Some parts of it were conducted during a research visit from September 2017 to January 2018 under the supervision of Prof. Dr. rer. nat. Markus Holzer at the Institut für Informatik, Universität Giessen, Germany, funded by the DAAD short-term grant ID 57314022.

Bratislava, 2019 .....

Ing. Michal Hospodár

Acknowledgments

I thank my supervisor Galina Jirásková for all her guidance, help, and patience with me throughout the four years spent together. She was a role model for my future scientific career and a positive example of openness and altruism within and beyond the scientific community.

I also thank my colleagues Peter Mlynárčik, Matúš Palmovský, and Ivana Krajňáková for their friendship and all the time spent together during our studies, discussions, and traveling.

I thank Markus Holzer for his guidance in writing two of my papers on which two chapters of this thesis were based. Next I thank Juraj Šebej and Jozef Jirásek Jr. for their help on border values in proofs of theorems in Chapter 5. Also many thanks to all reviewers who helped me to improve the presentation of my papers.

Last, but not least, I thank also to my parents, brothers, and all friends for support and tolerance.

Abstract

We determine the state complexity of concatenation on languages represented by deterministic finite automata with more than one final state. We provide ternary witness languages for an arbitrary number of final states in both automata, and binary witnesses in the case when the first automaton has at least two non-final states. We use our binary witnesses to prove that the upper bound $2^m + n + 1$ on the complexity of concatenation on alternating finite automata is tight, which solves an open problem from the literature.

We show that every value from one up to the known upper bound can be attained by the state complexity of a language resulting from the cut operation on two binary languages. In the unary case, we prove that some values are unattainable (magic). For all attainable values, we provide unary witnesses. This solves completely the so-called magic number problem for the cut operation for every alphabet size.

For the operations of intersection, symmetric difference, right and left quotients, reversal, permutation on binary finite languages, and cut, we describe the ranges of possible accepting state complexities. Except for intersection and permutation, this range is equal to all non-negative or to all positive integers. For intersection, we get the range from one up to mn. In the case of permutation, only the values zero and one are unattainable.

We show that the upper bound mn on the nondeterministic state complexity of intersection on subword-free languages is asymptotically tight for infinitely many pairs m and n in the binary case. We also provide a binary left ideal witness for reversal. This solves two open problems from the literature. We prove that the nondeterministic state complexity of the k-th power is kn in the classes of closed and convex languages, and it is $k(n-1)+1$ in the classes of free and ideal languages. Moreover, we show that the nondeterministic complexity of positive closure of every factor-closed or subword-closed language is one, and it is n in all the remaining subclasses of convex languages. All our witness languages are defined over a unary or a binary alphabet which is always optimal.

We examine the complexity of the forever operator assuming that the operand is represented by one of these six models of finite automata: complete and partial deterministic finite automata, nondeterministic finite automata with a unique or multiple initial states, alternating, and Boolean automata. The resulting language is required to be represented again by an automaton of one of these six models. In 32 of 36 cases, we get exact trade-offs, and most of our witnesses are defined over a small fixed alphabet.

Keywords: regular languages, finite automata, descriptional complexity, regular operations, concatenation, cut operation, forever operator, subclasses of convex languages

Abstrakt

Určíme stavovú zložitosť zreťazenia na jazykoch reprezentovaných deterministickými konečnými automatmi s viac ako jedným koncovým stavom. Poskytneme ternárne dosvedčujúce jazyky pre ľubovoľný počet koncových stavov v oboch automatoch a binárne v prípade, že prvý automat má aspoň dva nekoncové stavy. Naše binárne jazyky použijeme na získanie presnej hodnoty $2^m + n + 1$ vyjadrujúcej zložitosť zreťazenia na alternujúcich konečných automatoch, čo rieši otvorený problém z literatúry.

Dokážeme, že každá hodnota až po známu hornú hranicu môže byť dosiahnutá ako stavová zložitosť jazyka získaného operáciou strojového zreťazenia na dvoch binárnych jazykoch. V unárnom prípade ukážeme, že niektoré hodnoty sú nedosiahnuteľné (magické). Každú dosiahnuteľnú hodnotu dosvedčíme unárnymi jazykmi. To úplne rieši takzvaný problém magických čísel pre operáciu strojového zreťazenia pre každú veľkosť abecedy.

Pre prienik, symetrický rozdiel, pravý a ľavý kvocient, zrkadlový obraz, permutáciu na binárnych konečných jazykoch a strojové zreťazenie popíšeme rozsahy možných akceptačných stavových zložitostí. S výnimkou prieniku a permutácie je tento rozsah rovný všetkým nezáporným alebo všetkým kladným celým číslam. Pre prienik dostaneme rozsah od jednej po mn. Pre permutáciu sú jediné nedosiahnuteľné hodnoty nula a jedna.

Ukážeme, že horná hranica mn pre nedeterministickú stavovú zložitosť prieniku na bezpodslovových jazykoch je asymptoticky tesná pre nekonečne veľa dvojíc m a n v binárnom prípade. Taktiež poskytneme binárny ľavo ideálny dosvedčujúci jazyk pre zrkadlový obraz. Tým riešime dva otvorené problémy z literatúry. Dokážeme, že nedeterministická stavová zložitosť k-tej mocniny je kn na triedach uzavretých a konvexných jazykov a $k(n-1)+1$ na ostatných uvažovaných podtriedach. Navyše ukážeme, že nedeterministická zložitosť pozitívneho uzáveru každého faktorovo alebo podslovovo uzavretého jazyka je jedna a pre všetky ostatné podtriedy konvexných jazykov je n. Všetky naše dosvedčujúce jazyky sú definované na unárnej alebo binárnej abecede, ktorá je vždy optimálna.

Skúmame zložitosť operátora "navždy" za predpokladu, že operand je reprezentovaný jedným zo šiestich typov konečných automatov: úplné a čiastočné deterministické konečné automaty, nedeterministické automaty s jedným alebo s viacerými koncovými stavmi, alternujúce a booleovské automaty. Požadujeme, aby výsledný jazyk bol reprezentovaný opäť automatom jedného z týchto typov. V 32 z 36 prípadov dostaneme presný prevod a väčšina našich dosvedčujúcich jazykov je definovaná nad malou konštantnou abecedou.

Kľúčové slová: regulárne jazyky, konečné automaty, popisná zložitosť, regulárne operácie, zretazenie, strojové zretazenie, operátor forever, podtriedy konvexných jazykov

Contents

Introduction

Descriptional complexity measures the costs of description of formal languages by different formal systems such as deterministic and nondeterministic automata, grammars, and regular expressions. Some of these systems may provide a more succinct representation of a language than some others. For example, the language consisting of all strings having an a in the n-th position from the end is accepted by a nondeterministic finite automaton with $n+1$ states, while every deterministic finite automaton for this language requires at least 2^n states.

In this thesis, we study descriptional complexity in the class of regular languages and some of its subclasses. Although regular languages are the simplest languages in the Chomsky hierarchy, some challenging problems are still open. Let us mention the question of how many states are sufficient and necessary in the worst case for two-way deterministic finite automata to simulate two-way nondeterministic finite automata. The importance of this question is given by its connection to the well-known open question whether or not NLOGSPACE=DLOGSPACE [3, 84].

A natural complexity measure for regular languages is the number of states in the minimal deterministic finite automaton (DFA) for the given language. This number, called the state complexity [4, 5], is characteristic for every regular language, and regular languages form an infinite hierarchy with respect to this complexity measure.

If we consider a regular operation, for example, intersection, then we may ask, how many states are sufficient and necessary in the worst case to accept the intersection of two languages, the first of them accepted by an m-state DFA, and the second one by an n-state DFA. The product construction [83] results in an mn-state DFA for intersection, and provides an upper bound on the state complexity of this operation. On the other hand, every DFA for the intersection of languages that count the number of a's modulo m and the number of b's modulo n has at least mn states. Hence the upper bound mn is tight, and we say that the state complexity of intersection is mn.

In 1970, Maslov [74] examined the state complexity of union, concatenation, star, and some other regular operations. However, his paper remained unnoticed until 2005 [64], and the systematic study of the state complexity of operations began in 1994 with the paper by Yu, Zhuang, and Salomaa [94]. They provided the state complexity of concatenation and star, taking into account the possibility that the minimal DFAs for arguments may have more than one final state. Moreover, they included also the results for unary languages, that is, for the languages defined over a one-letter alphabet.

In 1959, Rabin and Scott [83] introduced nondeterministic finite automata (NFAs) and provided an algorithm known as the subset construction which shows that every n-state NFA can be simulated by a DFA with 2^n states. The representation of regular languages by NFAs gives another complexity measure, called nondeterministic state complexity, and defined as the smallest number of states in any NFA for the given language. Birget [4, 5] provided a lower-bound method, known as the fooling set method, for the number of states in NFAs, and Holzer and Kutrib [40] investigated the nondeterministic state complexity of basic regular operations. Some results of [40] were improved by Jirásková [57] where binary witnesses for reversal and complementation can be found.

Alternating finite automata (AFAs) were introduced by Chandra, Kozen, and Stockmeyer [21]. Fellah, Jürgensen, and Yu [32] considered AFAs as a special case of Boolean finite automata introduced by Brzozowski and Leiss [12] in which the transition function maps a state and an input symbol to a Boolean function with variables in the state set. While in BFAs an arbitrary Boolean function may be initial, AFAs have a single initial state. It is known that a language is accepted by an n-state AFA if and only if its reversal is accepted by a DFA with 2^n states, half of them final [32, 59]. This simple observation allows us to give the tightness of the upper bound $2^m + n + 1$ from [32] and solve an open problem stated in 1990 by Fellah et al. Using the same observation, the tightness of all other upper bounds from [32] has been proven in [45, 59].

In 2000, Iwama et al. [53] asked whether all values between n and 2^n can be attained by the state complexity of a language accepted by a minimal, with respect to the number of states, n-state NFA. The unattainable values were called magic numbers in [54], and the problem of finding all attainable complexities is called the magic number problem. The magic number problem was later considered also for language operations such as union and intersection by Hricko et al. [50, 51], complementation by Szabari [90, 60], reversal by Šebej [87], star by Palmovský et al. [65] and Čevorová [17], and concatenation by Jirásková et al. [66].

The smallest number of accepting states in any DFA for a language is also a complexity measure. It is called accepting state complexity and has been introduced in 2016 by Dassow [25]. He provided the ranges of accepting state complexities of languages resulting from complementation, union, concatenation, difference, and star. All the resulting ranges are given by all non-negative (or all positive) integers. This search for ranges of accepting state complexities can be also considered as a variant of the magic number problem. We solve this problem for several other regular operations.

The nondeterministic accepting state complexity also can be considered, however, as shown by Dassow, here the hierarchy of regular languages collapses to three levels: the nondeterministic accepting state complexity of every regular language may be 0, 1, or 2.

Some operations on formal languages are inspired by applications in computer systems. The cut operation, which describes the implementation of "concatenation" on UNIX text processors for regular expression matching, was formally defined by Berglund et al. [2]. This operation is regular, and its state complexity was obtained by Drewes et al. [29]. In this thesis, we investigate the magic number problem for the cut operation, and we provide a complete solution for this problem.

The state complexity of operations can be smaller if we assume that the operands belong to a specific subclass of regular languages. Already the class of unary languages, considered in [94], can be viewed as a subclass which decreases the state complexity of operations. The class of finite languages was studied by Câmpeanu et al. [16], co-finite languages were examined by Bassino et al. [1], and prefix-free languages, motivated by prefix codes, were investigated by Han, Salomaa, and Wood [37].

In 2010, Brzozowski [8] examined the state complexity in the classes of convex languages, and later, with co-authors, in their subclasses: free [14], ideal [10], and closed languages [11]. Nondeterministic state complexity on all these classes was investigated by Mlynárčik et al. in a series of papers [62, 77, 46, 47], which were summarized in his dissertation thesis [78]. Two open problems from [78] are solved in this thesis. Moreover, we consider another two operations: the k-th power and positive closure, taking into account the results by Čevorová [18, 19] on the state complexity of the second power, square, on free, ideal, and closed languages.

When we consider the (nondeterministic) state complexity or accepting state complexity, the arguments and the results of an operation are described using the same automata model. However, Birget [6] provided upper and lower bounds on the complexity of the forever operator in case when the argument and resulting language may be described by different automata models, namely, deterministic, nondeterministic, and alternating au-

tomata. We continue his research by improving some of his results and considering three more automata models.

The thesis is organized as follows. After giving basic definitions, upper and lower bound methods, and the state-of-the-art in the next three chapters, we study the concatenation operation on languages represented by deterministic finite automata with more than one final state. Our motivation comes from an open problem stated by Fellah, Jürgensen, and Yu [32] concerning the tightness of the upper bound $2^m + n + 1$ on the complexity of concatenation on alternating finite automata. To get a witness for AFAs, we need to describe DFAs with half of their states final that are hard for concatenation on DFAs. To this aim, we consider DFAs with an arbitrary number of final states and we are able to describe ternary m- and n-state witness DFAs with k and ℓ final states, respectively, for arbitrary values of m, n, k, ℓ. We also provide binary witnesses under assumption that $k \leq m-2$. We use these binary witness DFAs to describe binary witnesses for concatenation on AFAs, and we prove that the upper bound $2^m + n + 1$ is tight. This solves the open problem from [32].

Chapter 5 is devoted to the cut operation. We ask which values from one up to the known upper bound can be obtained as the state complexity of a language resulting from the cut of languages represented by minimal DFAs. In the unary case, we are able to get the magic status of every possible value in the range. We show that some values are always magic (unattainable). For all the remaining values, we describe the corresponding unary witness automata. In the binary case, or in the case of an arbitrary alphabet of size at least two, we show that no values are magic, that is, for an arbitrary number in the corresponding range we can describe minimal binary DFAs such that the state complexity of their cut gives this number. This solves the magic number problem for the cut operation for an arbitrary alphabet size.

We consider a similar problem with respect to a different complexity measure, called accepting state complexity, in Chapter 6. For the operations of symmetric difference, left and right quotients, and cut, we get contiguous ranges $\{0\} \cup \mathbb{N}$ of possible accepting state complexities, while the resulting range for intersection is $\{0, 1, \ldots, mn\}$, for reversal it is $\mathbb{N}$, and for permutation on binary finite languages it is $\mathbb{N} \setminus \{1\}$.

In Chapter 7 we represent languages by nondeterministic finite automata. First, we show that the upper bound mn on the nondeterministic state complexity of intersection on subword-free languages is asymptotically tight for infinite number of pairs m and n in the binary case. This provides a positive answer to a conjecture stated by Mlynárčik [78]. Then we describe a binary left ideal witness for reversal. This improves the result

of [78] where a ternary left ideal witness for reversal is provided. Finally, we get tight upper bounds on the nondeterministic state complexity of power and positive closure on the classes of prefix-, suffix-, factor-, and subword-free, -closed, and -convex languages, and right, left, two-sided, and all-sided ideal languages. All our witnesses are defined over a unary or binary alphabet. Moreover, whenever we use a binary alphabet, it is always optimal in the sense that the corresponding upper bound cannot be met in the unary case.

The description complexity of the forever operator, defined by $L \mapsto (\Sigma^* L^{\complement})^{\complement}$, is examined in Chapter 8. Here we represent the source language by one of six models of finite automata: complete or partial deterministic, nondeterministic with a unique or multiple initial states, Boolean, and alternating. We are asking, how many states are sufficient and necessary in the worst case for an automaton in one of these six models to accept the resulting language. In 32 of 36 cases, we get the exact trade-offs with witnesses that are often defined over an optimal alphabet. Among the most interesting results is the NFA-to-DFA trade-off for the forever operator given by Dedekind number which counts the number of antichains of subsets of a finite set.

Chapters 4 and 8 only summarize our results which have been published in two papers in scientific journals. These papers are provided in Appendices [A] and [B] at the end of this thesis.

Chapter 1

Preliminaries

In this section, we give some basic definitions and preliminary results. For details, the reader may refer to [43, 89, 92].

Let Σ be a finite non-empty alphabet of symbols. Then Σ^* denotes the set of strings over Σ including the empty string ε. The cardinality of a finite set A is denoted by $|A|$, and its power-set by 2^A. The length of a string w is denoted by $|w|$, and the number of occurrences of a symbol a in the string w is denoted by $|w|_a$. The set of positive integers is denoted $\mathbb{N}$ and the set $\{i, i+1, \ldots, j\}$ is denoted by $[i, j]$ if i and j are integers.

A *language* is any subset of Σ^*. For a language L over an alphabet Σ, the complement of L is the language $L^{\complement} = \Sigma^* \setminus L$. The *intersection* of languages K and L is the language $K \cap L = \{w \mid w \in K \text{ and } w \in L\}$. The *union* of languages K and L is the language $K \cup L = \{w \mid w \in K \text{ or } w \in L\}$. The *symmetric difference* of languages K and L is the language $K \oplus L = (K \cap L^{\complement}) \cup (L \cap K^{\complement})$. The *concatenation* of languages K and L is the language $KL = \{uv \mid u \in K \text{ and } v \in L\}$. The *right quotient* of a language K by a language L is the language $KL^{-1} = \{\, w \mid \text{there exists a string } x \in L \text{ such that } wx \in K \,\}$. The *left quotient* of a language K by a language L is the language $L^{-1}K = \{\, w \mid \text{there exists a string } x \in L \text{ such that } xw \in K \,\}$. The *cut* of languages K and L is the language $K \,!\, L = \{\, uv \mid u \in K,\, v \in L, \text{ and } uv' \notin K \text{ for every non-empty prefix } v' \text{ of } v \,\}$. The *k-th power* of a language L is the language $L^k = LL^{k-1}$, where $L^0 = \{\varepsilon\}$. The *star* of a language L is the language $L^* = \bigcup_{i \geq 0} L^i$. The *positive closure* of a language L is the language $L^* = \bigcup_{i \geq 1} L^i$. The *permutation* of a language L is the language $\mathrm{per}(L) = \bigcup_{w \in L} \{u \in \Sigma^* \mid \psi(u) = \psi(w)\}$ where $\psi(w) = (|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_{|\Sigma|}})$. The reversal of a string is defined as $\varepsilon^R = \varepsilon$ and $(wa)^R = aw^R$ for each symbol a and string w. The reversal of a language L is the language $L^R = \{w^R \mid w \in L\}$.

1.1 Deterministic and Nondeterministic Automata

A *nondeterministic finite automaton* is a 5-tuple $N = (Q, \Sigma, \cdot, I, F)$, where Q is a finite set of states, Σ is a finite alphabet, $\cdot : Q \times \Sigma \to 2^Q$ is the transition function which is extended to the domain $2^Q \times \Sigma^*$ in the natural way, $I \subseteq Q$ is the set of initial states, and $F \subseteq Q$ is the set of final (or accepting) states. The *language accepted by* N is the set $L(N) = \{w \in \Sigma^* \mid I \cdot w \cap F \neq \emptyset\}$.

If $|I| \geq 2$, we say that A is a *nondeterministic finite automaton with nondeterministic choice of initial state* (so we use the abbreviation NNFA, cf. [92]). Otherwise, if $|I| = 1$, we say that A is a *nondeterministic finite automaton with a unique initial state* (NFA). In this case, we write $A = (Q, \Sigma, \delta, s, F)$ instead of $A = (Q, \Sigma, \delta, \{s\}, F)$. Notice that every NFA is an NNFA.

For a symbol a and states p and q, we say that (p, a, q) is a transition in the automaton N if $q \in p \cdot a$, and for a string w, we write $p \xrightarrow{w} q$ if $q \in p \cdot w$. In Chapters 5 and 6, we denote the transition function by δ, and we write $\delta(p, w)$ instead of $p \cdot w$. We also say that the state q has an *in-transition* on symbol a, and the state p has an *out-transition* on symbol a. An automaton is *non-returning* if each its initial state does not have any in-transitions, and it is *non-exiting* if each its final state does not have any out-transitions. If p is an initial state, we sometimes write $\to p$.

The reverse of an NNFA $N = (Q, \Sigma, \cdot, I, F)$ is the NNFA N^R obtained from N by swapping the roles of initial and final states, and by reversing all the transitions. Formally, we have $N^R = (Q, \Sigma, \cdot^R, F, I)$, where $q \cdot^R a = \{p \in Q \mid q \in p \cdot a\}$ for every p, q in Q and every a in Σ. It is well known that the NNFA N^R accepts $L(N)^R$.

A state q is *reachable* if there exists a string w such that $q \in s \cdot w$ for an initial state s. Let $A = (Q, \Sigma, \cdot, I, F)$ be an NNFA and $S, T \subseteq Q$. We say that S is *reachable* in A if there exists a string w in Σ^* such that $S = I \cdot w$. Next, we say that T is *co-reachable* in A if T is reachable in A^R.

A state q of an NNFA A is called a *dead state* if no string is accepted by A from q, that is, if $q \cdot w \cap F = \emptyset$ for every string w. An NNFA A is a *trim* NNFA if each its state q is reachable, and, moreover, no state of A is dead.

An NFA $A = (Q, \Sigma, \cdot, s, F)$ is a *partial deterministic finite automaton* (pDFA) if for each state p and each input symbol a, the set $p \cdot a$ has at most one element. In such a case, we write $p \cdot a = q$ instead of $p \cdot a = \{q\}$. If $|p \cdot a| = 1$ for each p and a, then A is a *complete deterministic finite automaton* (DFA). Notice that every DFA is a pDFA. We usually write "partial DFA" instead of "pDFA".

Two states in a DFA are *distinguishable* if there exists a string that is accepted from one state and rejected from the other state. If two states are not distinguishable, they are equivalent. Two automata are *equivalent* if they accept the same language. A DFA A is *minimal* if there is no equivalent DFA which has less states than A. Analogously we define minimal pDFAs, NFAs, and NNFAs. It is well known that every regular language has a unique, up to isomorphism, minimal DFA. Next, a DFA is minimal if and only if all its states are reachable from the initial state, and all its states are pairwise distinguishable.

We call a state q of an NNFA *sink state* if it has a loop on each input symbol, that is, $q \cdot a = \{q\}$ for each a in Σ. Notice that every minimal pDFA has no non-final sink states, and every minimal DFA has at most one non-final sink state. It follows that the number of states in the minimal DFA and its equivalent minimal pDFA differs by at most one, and the number of accepting states in the minimal DFA and minimal pDFA is the same.

Every NNFA $N = (Q, \Sigma, \cdot, I, F)$ can be converted into an equivalent DFA

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

remind that $\cdot$ is extended to the domain $2^Q \times \Sigma$ [83]. The DFA $\mathcal{D}(N)$ is called the *subset automaton* of the NNFA N. The subset automaton may not be minimal since some of its states may be unreachable or equivalent to other states.

The *state complexity* of a regular language L, $\mathrm{sc}(L)$, is the smallest number of states in any DFA for L, that is, the number of states in the minimal DFA for L. The *nondeterministic state complexity* of a regular language L, $\mathrm{nsc}(L)$, is the smallest number of states in any NFA for L, that is, the number of states in a minimal NFA for L.

The *state complexity of a k-ary operation* $\circ$ is the function from $\mathbb{N}^k$ to $\mathbb{N}$ given by

$$(n_1, n_2, \ldots, n_k) \mapsto \max\{\mathrm{sc}(\circ(L_1, L_2, \ldots, L_k)) \mid \mathrm{sc}(L_i) \leq n_i \text{ for } i = 1, 2, \ldots, k\}.$$

The nondeterministic state complexity of an operation is defined in an analogous way. When we consider the complexity of an operation on a subclass $\mathcal{C}$ of regular languages, then we moreover require that each language L_i is in $\mathcal{C}$.

To prove the minimality of NNFAs, we use a *fooling set* lower-bound technique from [4, 5, 34, 52, 48] which is described in Chapter 2.

Definition 1.1. *A set of pairs $\mathcal{F} = \{(x_i, y_i) \mid 1 \leq i \leq n\}$ is called a fooling set for a language L if*

(F1) *for each i, we have $x_i y_i \in L$, that is, each pair concatenates to a string in L,*

(F2) *if $i \neq j$, then $x_i y_j \notin L$ or $x_j y_i \notin L$, that is, at least one mismatched concatenation is not in L.*

If languages K and L are accepted by NFAs $A = (\{0,1,\ldots,m-1\}, \Sigma, \cdot_A, 0, F_A)$ and $B = (\{0,1,\ldots,n-1\}, \Sigma, \cdot_B, 0, F_B)$, respectively, then the language $K \cap L$ is accepted by the *product automaton* [83]

$$A \times B = (\{0,1,\ldots,m-1\} \times \{0,1,\ldots,n-1\}, \Sigma, \cdot, (0,0), F_A \times F_B),$$

where

$$(p,q) \cdot a = (p \cdot_A a) \times (q \cdot_B a).$$

By the *row* r $(0 \leq r \leq m-1)$, we denote the set of states $\{(r,j) \mid 0 \leq j \leq n-1\}$, and by the *column* s $(0 \leq s \leq n-1)$, we denote the set of states $\{(i,s) \mid 0 \leq i \leq m-1\}$.

1.2 Boolean and Alternating Automata

In this section, we give some basic definitions and notations for Boolean and alternating finite automata. For details, we refer the reader to [12, 32, 59, 71, 72, 89].

A *Boolean finite automaton* (BFA, cf. [12]) is a quintuple $A = (Q, \Sigma, \delta, g_s, F)$, where Q is a finite non-empty set of states such that $Q = \{q_1, \ldots, q_n\}$, 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 is 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

(1) $\delta(g, \varepsilon) = g$;

(2) if $g = g(q_1, \ldots, q_n)$, then $\delta(g, a) = g(\delta(q_1, a), \ldots, \delta(q_n, a))$;

(3) $\delta(g, wa) = \delta(\delta(g, w), a)$.

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 of strings $L(A) = \{w \in \Sigma^* \mid \delta(g_s, w)(f) = 1\}$.

A Boolean finite automaton is called *alternating* (AFA, cf. [32]) if the initial function is a projection $g(q_1, \ldots, q_n) = q_i$. An alternating finite automaton A is an NFA if $\delta(q_k, a)$ are of the form $\bigvee_{i \in I} q_i$. If $\delta(q_k, a)$ are of the form q_i, then the automaton A is a DFA.

δ	a	b
q_1	$q_1 \vee q_2$	q_1
q_2	q_2	$\overline{q_1} \wedge q_2$

Table 1.1: The transition function of the alternating finite automaton A_1.

For an example, consider AFA $A_1 = (\{q_1, q_2\}, \{a, b\}, \delta, q_1, \{q_2\})$, where transition function δ is given in Table 1.1. We have

$$\delta(q_1, ab) = \delta(\delta(q_1, a), b) = \delta(q_1 \vee q_2, b) = q_1 \vee (\overline{q_1} \wedge q_2) = q_1 \vee q_2.$$

To determine whether $ab \in L(A_1)$, we evaluate $\delta(q_1, ab)$ at the finality vector $f = (0, 1)$. We obtain 1, hence $ab \in L(A_1)$.

1.3 Subclasses of Convex Languages

If $u, v, w, x \in \Sigma^*$ and $w = uxv$, then u is a *prefix* of w, x is a *factor* of w, and v is a *suffix* of w. Both u and v are also factors of w. If $w = u_0 v_1 u_1 \cdots v_n u_n$, where $u_i, v_i \in \Sigma^*$, then $v_1 v_2 \cdots v_n$ is a *subword* of w. For example, let $w = abbacb$. Strings $abac, bbb, bc$ are subwords of w, but string aca is not a subword of w. Every factor of w is also a subword of w. A prefix v (suffix, factor, subword) of w is *proper* if $v \neq w$.

A language L is *prefix-free* if $w \in L$ implies that no proper prefix of w is in L; it is *prefix-closed* if $w \in L$ implies that each prefix of w is in L; and it is *prefix-convex* if $u, w \in L$ and u is a prefix of w imply that each string v such that u is a prefix of v and v is a prefix of w is in L. Suffix-, factor-, and subword-free, -closed, and -convex languages are defined analogously.

A language L is a right (respectively, left, two-sided, all sided) *ideal* if $L = L\Sigma^*$ (respectively, $L = \Sigma^* L, L = \Sigma^* L \Sigma^*, L = L \sqcup \Sigma^*$).

We say that a regular language is a *free language* if it is either prefix-free, or suffix-free, or factor-free, or subword-free. Let us emphasize that we do not consider star-free, or union-free, or any other free languages in this thesis. In an analoguous way, we use the notions of *closed languages*, *convex languages*, and *ideal languages*.

Notice that the classes of prefix-free, prefix-closed, and right ideal languages are subclasses of prefix-convex languages, and similar inclusions hold also for suffix-, factor-, and subword-convex languages. Next, in the unary case, all four subclasses of free languages coincide; similarly for ideal, closed, and convex languages.

Chapter 2

Upper and Lower Bound Methods

The state complexity of a k-ary regular operation $\circ$ is defined as the function from $\mathbb{N}^k$ to $\mathbb{N}$ given by

$$(n_1, n_2, \ldots, n_k) \mapsto \max\{\mathrm{sc}(\circ(L_1, L_2, \ldots, L_k)) \mid \mathrm{sc}(L_i) \leq n_i \text{ for } 1 \leq i \leq k\}.$$

To find the state complexity of a binary operation $\circ$, we need to find a function $f(m, n)$ such that

(1) for all integers m and n, and **all languages** K and L with $\mathrm{sc}(K) \leq m$ and $\mathrm{sc}(L) \leq n$, we have $\mathrm{sc}(K \circ L) \leq f(m, n)$;

(2) for all integers m and n, **there exist languages** K and L with $\mathrm{sc}(K) \leq m$, $\mathrm{sc}(L) \leq n$, and $\mathrm{sc}(K \circ L) = f(m, n)$.

In case (1), we prove that $f(m, n)$ is an **upper bound**, and in case (2), we prove that $f(m, n)$ is a **lower bound** on the state complexity of the operation $\circ$, and the languages with $\mathrm{sc}(K \circ L) = f(m, n)$ are called *witness languages*.

Sometimes the upper and lower bound may be different. If the upper and lower bounds coincide, then we say that $f(m, n)$ is a *tight upper bound*. In this thesis, our upper and lower bounds coincide in most cases.

To get an upper bound $f(m, n)$, we need to provide a construction of a DFA for the language resulting from the operation with at most $f(m, n)$ states. To get lower bounds, we need to describe languages accepted by m- and n-state DFAs such that every DFA for the resulting language requires at least $f(m, n)$ states.

2.1 Lower Bound Methods for DFAs

To get lower bounds on the state complexity, we usually count the number of reachable and distinguishable states i a subset automaton. To get reachability of a lot of subsets, we sometimes use the following two observations.

Proposition 2.1. *In the subset automaton of the* NFA *shown in Figure 2.1, each subset containing the state* 0 *is reachable from the initial subset* $\{0\}$.

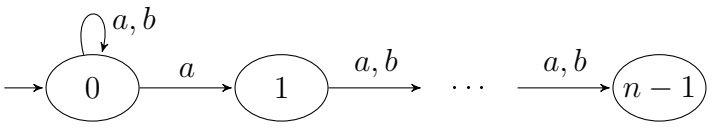


Figure 2.1: The NFA used in Proposition 2.1.

Proof. The proof is by induction on the size of subsets. The subset $\{0\}$ is the initial subset of the subset automaton. Each subset $\{0\} \cup S$ of size $k+1$, where $1 \leq k \leq n-1$, is reached from the subset $\{s - \min S \mid s \in S\}$ of size k by the string $ab^{\min S - 1}$. Notice that the proof works for arbitrary transitions on a, b in the state $n-1$. □

Proposition 2.2. *In the subset automaton of the NFA shown in Figure 2.2, each subset is reachable from the set* $\{0, 1, \ldots, n-1\}$.

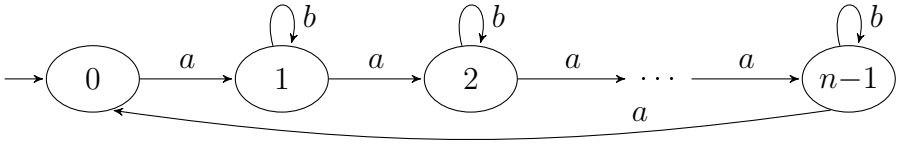


Figure 2.2: The NFA used in Proposition 2.2.

Proof. Notice that we can shift every subset cyclically by one by reading a, and we can eliminate the state 0 from every subset containing 0 by reading b. Formally, if $i \in S$, then the set $S \setminus i$ is reached from S by reading $a^{n-i}ba^i$. Hence every subset is reachable from the set $\{0, 1, \ldots, n-1\}$. □

To prove distinguishability, we use the concepts of uniquely distinguishable and uniquely reachable states, cf. [15]. We say that a state q of a nondeterministic finite automaton $N = (Q, \Sigma, \cdot, I, F)$ is uniquely distinguishable if there exists a string w which is accepted by N from and only from the state q, that is, if we have $p \cdot w \cap F \neq \emptyset$ if and only if $p = q$.

Proposition 2.3. *If two subsets of the subset automaton of an* NFA *differ in a uniquely distinguishable state, then the two subsets are distinguishable in the subset automaton.*

Proof. Let D be the subset automaton of an NFA N. Let S and T be two subsets of the subset automaton D. Let q be a uniquely distinguishable state of the NFA N such that, without loss of generality, $q \in S \setminus T$. Then there is a string w which is accepted by N from and only from q. It follows that w is accepted by D from S and rejected from T. Hence S and T are distinguishable in D. □

We say that a transition (p, a, q) is a unique in-transition in an NFA N if there is no state r with $r \neq p$ such that (r, a, q) is a transition in N. We say that a state q is uniquely reachable from a state p if there is a sequence of unique in-transitions (q_{i-1}, a_i, q_i) for $i = 1, 2, \ldots, k$ such that $q_0 = p$ and $q_k = q$.

Proposition 2.4. *Let a uniquely distinguishable state q be uniquely reachable from a state p. Then the state p is uniquely distinguishable.*

Proof. Let a string w be accepted by an NFA N from and only from a state q. If (p, a, q) is a unique in-transition, going to the state q by symbol a, then the string aw is accepted by the NFA N from and only from the state p. Now the claim can be proved by induction. □

Proposition 2.5. *If each state of an* NFA *N is uniquely distinguishable, then all the states of the subset automaton $\mathcal{D}(N)$ are pairwise distinguishable.*

Proof. Let two subsets S and T differ in a state q. Since q is uniquely distinguishable, there is a string w_q which is accepted by N from q and rejected from every other state. It follows that w_q distinguishes S and T. □

Using the notions of co-reachable subsets, we can reformulate the previous result as follows.

Proposition 2.6. *If for each state q of an* NFA *N, the singleton set $\{q\}$ is co-reachable, then all the states of the subset automaton $\mathcal{D}(N)$ are pairwise distinguishable.* □

2.2 Lower Bound Methods for NFAs

To get a lower bound on the number of states in an NNFA, the fooling set method appears to be very helpful. First, let us recall the definition of a fooling set.

Definition 2.7. *A set of pairs*

$$\mathcal{F} = \{(x_i, y_i) \mid 1 \leq i \leq n\}$$

is called a fooling set for a language L if

(F1) *for each i, we have $x_i y_i \in L$, that is, each pair concatenates to a string in L,*

(F2) *if $i \neq j$, then $x_i y_j \notin L$ or $x_j y_i \notin L$, that is, at least one mismatched concatenation is not in L.*

The next observation shows that the size of a fooling set for L provides a lower bound on the size of an NNFA for L.

Lemma 2.8 ([4, Lemma 1], **Fooling-Set Lemma).** *Let $\mathcal{F}$ be a fooling set for a language L. Then every* NNFA *for L has at least $|\mathcal{F}|$ states.*

Proof. Let N be an NNFA for L. Since $x_i y_i \in L$ for each i, we may fix an accepting computation in N on each $x_i y_i$. Let p_i be the state on this accepting computation which is reached after reading x_i. Let $i \neq j$. Then in N we have the following two accepting computations:

$$\rightarrow q_i \xrightarrow{x_i} p_i \xrightarrow{y_i} f_i$$
$$\rightarrow q_j \xrightarrow{x_j} p_j \xrightarrow{y_j} f_j$$

for some initial states q_i and q_j, and some final states f_i and f_j. It follows that $p_i \neq p_j$ because otherwise both mismatched concatenations would be in L. Hence the states $p_1, \ldots, p_n$ must be pairwise distinct, and the lemma follows. □

Let us emphasize that the size of a fooling set provides a lower bound on the size of NNFAs, that is, nondeterministic finite automata with possibly multiple initial states. Sometimes, for example in the case of union and reversal, an NNFA for the resulting language may be smaller by one state then its equivalent minimal NFA. In such a case we cannot get the tightness of an upper bound using the fooling-set lemma. Instead, the following modification of the fooling set lemma can be useful.

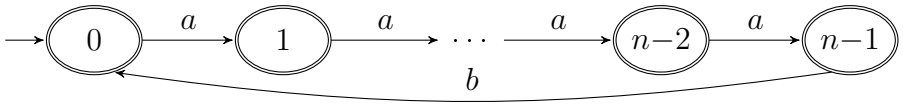


Figure 2.3: An NFA for L such that every NFA for L^R has at least $n+1$ states.

Lemma 2.9 ([61, Lemma 4], **AB-Fooling-Set Lemma).** *Let $\mathcal{A}$ and $\mathcal{B}$ be disjoint sets of pairs of strings and let u and v be strings such that $\mathcal{A}\cup\mathcal{B}$, $\mathcal{A}\cup\{(\varepsilon,u)\}$, and $\mathcal{B}\cup\{(\varepsilon,v)\}$ are fooling sets for a language L. Then every* NFA *(with a unique initial state) for L has at least $|\mathcal{A}|+|\mathcal{B}|+1$ states.*

Proof. Let $\mathcal{A}=\{(x_i,y_i) \mid 1\leq i\leq k\}$ and $\mathcal{B}=\{(x_i,y_i) \mid k+1\leq i\leq k+\ell\}$. The fact that $\mathcal{A}\cup\mathcal{B}$ is a fooling set implies that there are pairwise distinct states $p_1,\ldots,p_{k+\ell}$ as shown in the proof above. Next, after reading the empty string ε, the NFA is in its unique initial state q_0. Since $\mathcal{A}\cup\{(\varepsilon,u)\}$ is a fooling set, the state q_0 must be different from the states $p_1,\ldots,p_k$. Since $\mathcal{B}\cup\{(\varepsilon,v)\}$ is a fooling set, the state q_0 must be different from the states $p_{k+1},\ldots,p_{k+\ell}$. This proves the lemma. □

Example 2.10. *Let L be the language accepted by the partial* DFA *from Figure 2.3. Let us show that the minimal* NFA *for L^R has $n+1$ states. Since L^R is accepted by an n-state* NNFA*, we cannot use Lemma 2.8 (Fooling-Set Lemma). However, we can successfully use Lemma 2.9 (AB-Fooling-Set Lemma) with*

$\mathcal{A}=\{(ba^i,a^{n-1-i}) \mid 1\leq i\leq n-1\}$,
$\mathcal{B}=\{(b,a^{n-1})\}$,
$u=a^{n-1}$, and $v=b$. □

To avoid tedious descriptions of fooling sets, the following two observations can be successfully used.

Lemma 2.11 ($\{q\}$-Lemma). *Let for each state q of an* NNFA *N, the singleton set $\{q\}$ be reachable and co-reachable. Then the* NNFA *N is minimal.*

Proof. Let $N=(Q,\Sigma,\cdot,I,F)$. Since $\{q\}$ is reachable, there is a string x_q with $\{q\}=I\cdot x_q$. Since $\{q\}$ is co-reachable, there is a string y_q which is accepted by N from and only from the state q. It follows that $\{(x_q,y_q) \mid q\in Q\}$ is a fooling set for $L(N)$. By Lemma 2.8, every NNFA for $L(N)$ has at least $|Q|$ states. Hence N is a minimal NNFA. □

Example 2.12. *Let $L=\{a^m\}^*\cup\{b^n\}^*$. Then L is accepted by the $m+n$-state* NNFA *with two initial states and two disjoint cycles on a and b of length m and n, respectively. In this* NNFA*, all singleton sets are reachable and co-reachable, so this* NNFA *is minimal. To prove that a minimal* NFA *for L requires $m+n+1$ states, we have to use Lemma 2.9.* □

Lemma 2.13 (Trim-Lemma). *Let N be a trim partial* DFA. *If N^R is a partial* DFA, *then N is a minimal* NNFA.

Proof. If N and N^R satisfy the conditions of this lemma, then for each state q of N, the singleton set $\{q\}$ is reachable and co-reachable, and the result follows from Lemma 2.11 ($\{q\}$-Lemma). □

Example 2.14. *Let $L = \{a^{n-1}\}$. Then L is accepted by the partial* DFA *from Figure 2.4. By Lemma 2.13 (Trim-Lemma), this* DFA *is in fact a minimal* NNFA *for L.* □

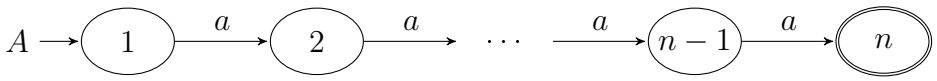


Figure 2.4: A partial DFA for $L = \{a^{n-1}\}$.

The next lemma provides another possibility how to avoid describing fooling sets and get a lower bound on the size of nondeterministic finite automata.

Lemma 2.15 (Greater-Smaller Lemma). *Let $n \geq 2$. Let $N = (\{1, 2, \ldots, n\}, \Sigma, \cdot, I, F)$ be an* NNFA *and $\{(X_i, Y_i) \mid i = 1, 2, \ldots, n\}$ be a set of pairs of subsets of the state set of N such that for each i in $\{1, 2, \ldots, n\}$,*

(1) X_i is reachable and Y_i is co-reachable in N,

(2) $i \in X_i \cap Y_i$, and

(3) $X_i \subseteq \{i, i+1, \ldots, n\}$ and $Y_i \subseteq \{1, 2, \ldots, i\}$.

Then N is a minimal NNFA.

Proof. Since X_i is reachable, there is a string x_i such that $I \cdot x_i = X_i$. Since Y_i is co-reachable, there is a string y_i which is accepted by N from every state in Y_i and rejected from every other state. Since $X_i \cap Y_i = \{i\}$, the string $x_i y_i$ is in $L(N)$. Let $i > j$. Then $X_i \cap Y_j = \emptyset$, so $x_i y_j$ is not in $L(N)$. Thus the set $\{(x_i, y_i) \mid i = 1, 2, \ldots, n\}$ is a fooling set for $L(N)$. Hence N is a minimal NNFA by Lemma 2.8 (Fooling-Set Lemma). □

With a symmetric argument as for Lemma 2.15, we can prove also the following result.

Lemma 2.16 (Smaller-Greater Lemma). *Let $n \geq 2$. Let $N = (\{1, 2, \ldots, n\}, \Sigma, \cdot, I, F)$ be an* NNFA *and $\{(X_i, Y_i) \mid i = 1, 2, \ldots, n\}$ be a set of pairs of subsets of the state set of N such that for each i in $\{1, 2, \ldots, n\}$,*

(1) X_i is reachable and Y_i is co-reachable in A,

(2) $i \in X_i \cap Y_i$, and

(3) $X_i \subseteq \{1, 2, \ldots, i\}$ and $Y_i \subseteq \{i, i+1, \ldots, n\}$.

Then N is a minimal NNFA. □

Notice that $\{q\}$-lemma is a special case of both the above lemmas. We will use Lemma 2.15 in Chapter 7 to prove the minimality of the result of the k-th power on binary subword-closed languages.

2.3 Upper and Lower Methods for AFAs and BFAs

It follows from [32] and [59] that a language L is accepted by an n-state AFA if and only if the language L^R is accepted by a DFA with 2^n states and 2^{n-1} final states. As this is a crucial observation for Chapter 8, we restate these results and provide proof ideas.

Lemma 2.17 (cf. [32, Theorem 4.1, Corollary 4.2] and [59, Lemma 1])**.** *Let L be a language accepted by an n-state* AFA*. Then the language L^R is accepted by a* DFA *of 2^n states, of which 2^{n-1} are final.*

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

- for every $b = (b_1, \ldots, b_n) \in \{0,1\}^n$ and every $a \in \Sigma$,
  $\delta'(b, a) = \{b' \in \{0,1\}^n \mid \delta(q_i, a)(b') = b_i \text{ for } i = 1, \ldots, n\}$;
- $I = \{(b_1, \ldots, b_n) \in \{0,1\}^n \mid b_1 = 1\}$;
- $f = (b_1, \ldots, b_n) \in \{0,1\}^n$ with $b_i = 1$ if and only if $q_i \in F$.

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

In the next corollary, $\mathrm{asc}(L)$ denotes the smallest number of states in any AFA for L.

Corollary 2.18. *For every regular language L,* $\mathrm{asc}(L) \geq \lceil \log(\mathrm{sc}(L^R)) \rceil$. □

Lemma 2.19 (cf. [59, 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* AFA*.*

Proof Idea. Consider 2^n-state NNFA A^R for L which has 2^{n-1} initial states and exactly one final state. Let the state set Q of A^R be $\{0, 1, \ldots, 2^n - 1\}$ with the set of initial states $\{2^{n-1}, \ldots, 2^n - 1\}$ and a final state k. 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.

Define an n-state AFA $A' = (Q', \Sigma, \delta', q_1, F')$, where

- $Q' = \{q_1, \ldots, q_n\}$,
- $F' = \{q_\ell \mid \mathrm{bin}(k)_\ell = 1\}$, and
- $(\delta'(q_1, a), \ldots, \delta'(q_n, a))(\mathrm{bin}(i)) = \mathrm{bin}(j)$ where $i \in \delta(j, a)$ for each i in Q and a in Σ.

Then $L(A') = L(A^R)$, hence L is accepted by an n-state AFA. □

The results of the two lemmas above are summarized in the following corollary.

Corollary 2.20. *A language L is accepted by an n-state* AFA *if and only if its reversal L^R is accepted by a* DFA *of 2^n states, of which 2^{n-1} are final.* □

A similar result holds also for Boolean finite automata, however, here the number of final states does not matter.

Theorem 2.21. *A language L is accepted by an n-state* BFA *if and only if its reversal L^R is accepted by a* DFA *of 2^n states.* □

Simple observations given in Theorem 2.21 and Corollary 2.20 provide both upper and lower bound methods for alternating and Boolean automata. We use them to transform a problem on AFAs and BFAs to a corresponding problem on DFAs for reversals. Since reversal commutes with all operations under consideration, we are able to go back from the solution on DFAs to a solution on AFAs or BFAs. In the case of alternating finite automata, we should be able to find DFA languages that are hard for an operation and moreover have half of their states final. This is the motivation for Chapter 4 where we examine the complexity of concatenation on DFAs with more final states.

Chapter 3

Descriptional Complexity – State-of-the-Art

In 1959, Rabin and Scott [83] defined the *product automaton* and proved that the intersection of two regular languages is also regular using this automata construction. Moreover, this construction provides the upper bound mn on the state complexity of intersection. They also defined nondeterministic finite automata (NFAs) and described an algorithm known as the "subset construction" which shows that every n-state NFA can be simulated by a 2^n-state deterministic finite automaton (DFA). Binary witnesses meeting this upper bound were described in 1962 by Yershov [91], in 1963 Lupanov [73], and in 1971 by Moore [79] and Meyer and Fischer [75]. These witnesses are shown in Figures 3.1 to 3.4.

Rabin and Scott provided a proof that also reversal is a regular operation since it is accepted by the reverse automaton which can be constructed from every NFA. In 1966, Mirkin [76] pointed out that the ternary witness for determinization described by Lupanov [73] is the reverse of a DFA, which proved the optimality of the upper bound 2^n on the state complexity of reversal. A binary witness meeting the upper bound 2^n for reversal, shown in Figure 3.5 (above), was given in 1981 by Leiss [72]; notice that every even state is final. A binary witness for reversal provided by Šebej in 2012 [67] and showin in Figure 3.6 has a unique final state and moreover is deterministic union-free.

The construction of an NFA for concatenation, given by Rabin and Scott [83], proves that also the concatenation operation preserves regularity of languages. The upper bound $m2^n - 2^{n-1}$ was shown already in 1970 by Maslov [74] who also provided binary witness languages. If the first automaton has k final states, then the upper bound is $m2^n - k2^{n-1}$, as was shown by Yu, Zhuang, and Salomaa in 1994 [94]. This paper was the cornerstone of a systematic research of state complexity of regular operations.

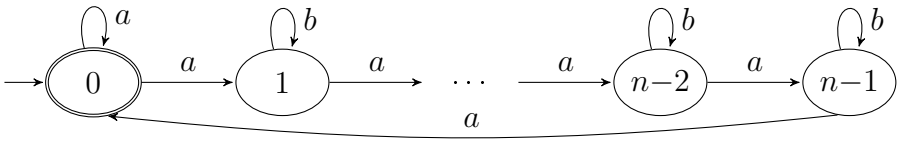


Figure 3.1: Yershov's binary witness for determinization from [91].

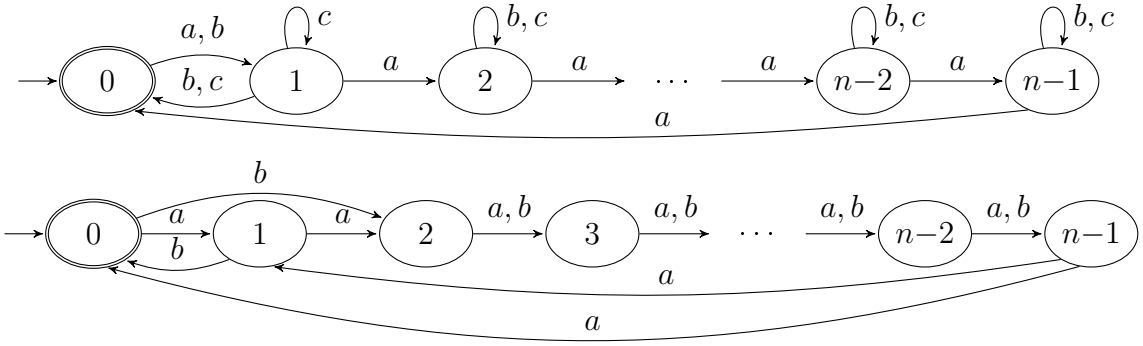


Figure 3.2: Lupanov's ternary and binary witnesses for determinization from [73]. The ternary witness is the reverse of a DFA, which is thus a witness for reversal [76].

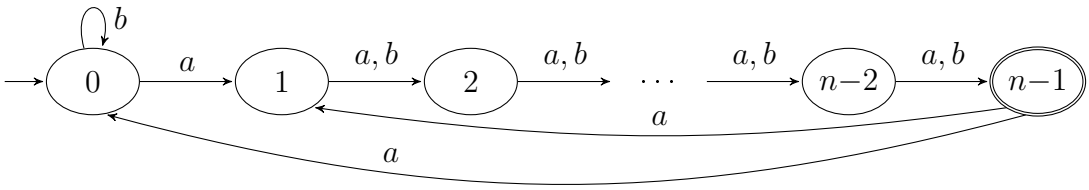


Figure 3.3: Moore's binary witness for determinization from [79].

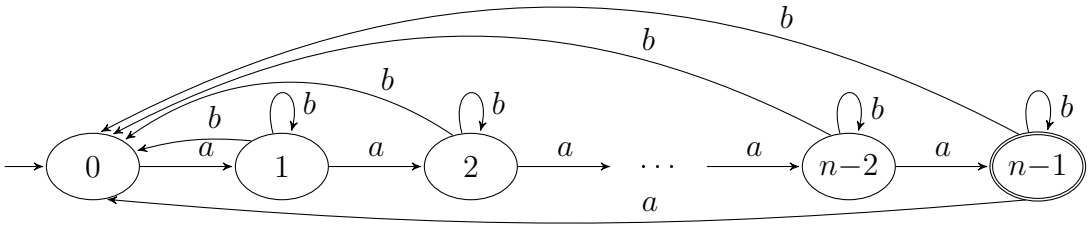


Figure 3.4: Binary witness for determinization by Meyer and Fischer [75].

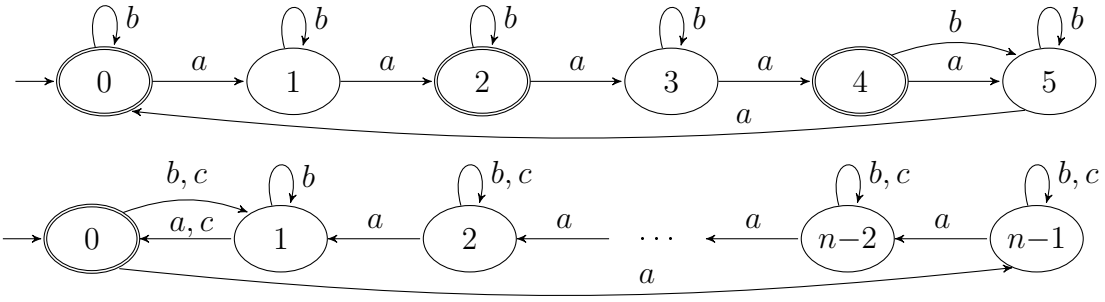


Figure 3.5: Binary and ternary witnesses for reversal by Leiss [72]; for the binary witness with half states final, an example with $n = 6$ is displayed.

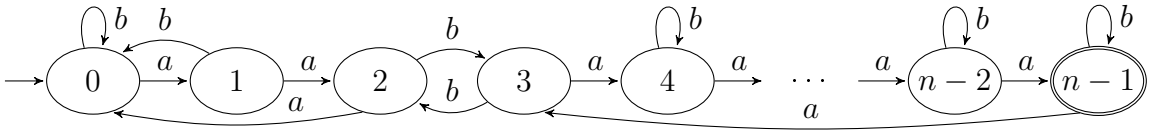


Figure 3.6: Šebej's binary witness for reversal with a unique final state from [67].

In [94], a ternary witness for concatenation was shown, and the binary case was left open. The authors did not know about the paper by Maslov [74] which was discovered in 2005 in [64].

Rabin and Scott [83] also considered the operation of Kleene closure, called *star* in this thesis. They provided a construction of an NFA for the closure of a language which

sc	Results in [74]	$\lvert\Sigma\rvert$	Results in [94]	$\lvert\Sigma\rvert$
$K \cap L$			mn	2
$K \cup L$	mn	2	mn	2
KL	$(m-1)2^n + 2^{n-1}$	2	$m2^n - k2^{n-1}$	3
unary case			mn; $\gcd(m,n) = 1$	1
L^*	$(3/4)2^n$	2	$2^{n-1} + 2^{n-1-k}$	2
unary case			$(n-1)^2 + 1$	1
L^R	2^n (Mirkin, Lupanov)	3	2^n (Leiss)	3

Table 3.1: The state complexity of operations from Maslov [74] and Yu, Zhuang, Salomaa [94]; k is the number of states in the first automaton in the case of concatenation and it is the number of final states which are not initial in the case of star.

gave an upper bound $(3/4)2^n$ for the size of DFA accepting this operation. Maslov [74] provided a binary witness meeting this upper bound.

If we require that the languages belong to certain subclass of regular languages, then the complexity of some operations may be smaller. Yu et al. [94] considered also the state complexity of operations on unary languages. They have shown that the state complexity of star on unary languages is $(n-1)^2+1$, unlike on binary languages where it is $(3/4)2^n$. In 1999, Nicaud [80] provided the tail-loop structure of minimal unary DFAs. Unary languages were investigated in more detail by Pighizzini and Shallit in 2002 [82]. The complexity of operation on other subclasses of regular languages were also considered, for example, finite languages by Câmpeanu et al. in 1999 [16] and co-finite language by Bassino et al. in 2010 [1]. Jirásková examined union-free languages with Masopust [61] and with Nagy [63]. The class of star-free languages was considered by Brzozowski and Liu [13]. The class of non-returning languages was investigated by Eom, Han, and Jirásková in 2016 [30].

The subclasses of languages from Section 1.3 are defined by partial orders on strings, such as prefix, suffix, factor, and subword. State complexity of operations in the class of prefix-free regular languages was investigated by Han, Salomaa, and Wood in 2006 [37]. In 2009, Han and Salomaa examined the state complexity of operations on suffix-free languages [35]. In these papers, the following was observed.

Fact 3.1 ([35, 37])**.**
(a) A minimal DFA *recognizes a prefix-free language if and only if it is non-exiting.*
(b) If a minimal DFA *recognizes a suffix-free language, then it is non-returning.* □

In 2011, Cmorik provided a sufficient condition for a DFA to accept a suffix-free language.

Lemma 3.2 ([23, Lemma 1])**.** *Let A be a* DFA *such that*
(1) A is non-returning,
(2) A has a unique final state, and
(3) each state of A has at most one in-transition on every input symbol.
Then $L(A)$ is suffix-free. □

In 2010, Brzozowski [9] proposed the notion of the *quotient complexity* which has the same value as the state complexity for every regular language. State/quotient complexity of operations were considered by Brzozowski et al. in 2013 on ideal languages [10], and in 2014 on bifix-, factor-, and subword-free languages [14] and closed languages [11]. In these papers, the square operation, which is defined as the second power of a language, was not

considered. This operation was investigated by Čevorová on free, ideal, and prefix-closed languages in 2015 [18] and on suffix-, factor-, and subword-closed languages in 2016 [19].

It is known that in the unary case, all four subclasses of free languages coincide; similarly for ideal, closed, and convex languages. The following straightforward observation shows the structure of unary languages belonging to these classes.

Fact 3.3. *(a) A unary free language is either empty or consists of one string, so it is of the form* $\{a^\ell\}$ *for some non-negative integer* ℓ*.*

(b) A unary ideal language is either empty or of the form $\{a^i \mid i \geq \ell\}$ *for some non-negative integer* ℓ*.*

(c) A unary closed language is either empty or universal or of the form $\{a^i \mid 0 \leq i \leq \ell\}$ *for some non-negative integer* ℓ*.*

(d) A unary convex language is either unary ideal or of the form $\{a^i \mid k \leq i \leq \ell\}$ *for some non-negative integers* k, ℓ*.* □

We can determine the descriptional complexity of unary finite languages. Let L be a unary finite language with the longest string of length ℓ. Then $\mathrm{nsc}(L) = \ell + 1$ since $\{(a^i, a^{\ell-i}) \mid 0 \leq i \leq \ell\}$ is a fooling set for L (with longer mismatched concatenations), and $\mathrm{sc}(L) = \ell + 2$ since we have to add the dead state.

The *nondeterministic state complexity* of a language L is the smallest number of states in any NFA for L. Although it was examined already in 1992 by Birget [4], the systematic research of nondeterministic state complexity began with the paper by Holzer and Kutrib in 2003 [40]. Birget in [4] and [5] proposed the *fooling set method* to prove lower bounds on the number of states in nondeterministic finite automata. It was modified to the *AB-fooling set method* [61] which also works for NFAs with a unique initial state. Some results of [40] were improved by Jirásková in 2005 [57] who provided binary witnesses for the already tight upper bounds $n + 1$ for reversal and 2^n for complementation.

Nondeterministic state complexity of operations on subclasses of regular languages was investigated in 2009 by Han, Salomaa, and Wood for prefix-free [38], in 2010 for suffix-free by Han and Salomaa [36], in 2011 for union-free by Jirásková and Masopust [61], and in 2012 for star-free languages by Holzer, Kutrib, and Meckel [42]. Jirásková and Mlynárčik [62] improved some results from [36, 38]. In the further papers by Mlynárčik [77, 46, 47], nondeterministic state complexity of six basic operations on free, ideal, closed, and convex languages was investigated, and tight upper bounds were obtained for 94 of 96 cases.

The cut operation, inspired by UNIX text processors, was formally defined in 2013 in a paper by Berglund, Björklund, Drewes, van der Merwe, and Watson [2]. The paper [29]

from 2017 by Drewes, Holzer, Jakobi, and van der Merwe provided tight upper bounds on the state complexity of the cut operation.

Iwama, Kambayashi, and Takaki in 2000 [53] asked whether all values from n up to 2^n can be obtained as the state complexity of a language accepted by a minimal n-state NFA. Iwama, Matsuura, and Paterson [54] called the unattainable numbers *magic*. It was shown in 2007 by Geffert [33] that in the unary case, there is a lot of magic numbers. However, no specific one was given. On the other hand, no number is magic if we have an alphabet of size three, as was shown in 2011 by Jirásková [58]. The problem is still open for a binary alphabet, although many numbers were identified as non-magic, and computations show that no magic numbers exist up to $n = 17$. The magic number problem on subclasses of regular languages was examined by Holzer, Jakobi, and Kutrib in 2012 [39].

The magic number problem was originally considered for determinization of NFAs. It can be generalized also to language operations. Hricko in his master's thesis [50] showed that every number from 1 up to mn can be obtained as the state complexity of union or intersection of languages accepted by minimal DFAs of size m and n. Similar result for NFAs is present in the dissertation thesis of Szabari [90] where also magic number problems for determinization and complementation are considered. For complementation, all values from $\log n$ to 2^n are attained using an alphabet of size five [60].

Čevorová in 2013 [17] examined the magic number problem for star on unary languages, and contrary to the result of Geffert [33], who only prove the existence of magic numbers for unary determinization but did not specify any of them, she was able to find two specific contiguous ranges of magic numbers of length n in the range from 1 to $(n-1)^2+1$. Nevertheless, there are still quadratically many numbers with unknown magic status, and the smallest magic number is not known. Jirásková, Palmovský, and Šebej [65] have shown that every number from 1 up to $(3/4)2^n$ is attainable as the state complexity of star of an n-state DFA language. Similarly, every number from $\log n$ up to 2^n is non-magic for reversal, as shown by Šebej [87]. Jirásková, Szabari, and Šebej in 2017 [68] have shown that the full range is attainable also for concatenation. However, all these three results used a linearly growing alphabet. The magic number problem for star, reversal, and concatenation on a fixed alphabet is still open. Results on the magic number problems are shown in Table 3.2.

Accepting state complexity was introduced by Dassow in 2016 [25]. He considered the infinite hierarchy of regular languages with respect to the minimal number of accepting states in DFAs. He asked, which values can be obtained as an accepting state complexity of languages resulting from a regular operation with accepting state complexity of operands

Operation	Range	Magic numbers	$\|\Sigma\|$	Source
NFA-to-DFA conversion	$[n, 2^n]$	no	3	[58]
DFA union, intersection	$[1, mn]$	no	2	[50]
NFA union	$[1, m+n+1]$	no	2	[90]
NFA complementation	$[\log n, 2^n]$	no	5	[60]
DFA reversal	$[\log n, 2^n]$	no	$2n$	[87]
DFA star	$[1, (3/4)2^n]$	no	$2n$	[65]
DFA concatenation	$[1, m2^n - 2^{n-1}]$	no	$2n$	[66]

Table 3.2: The known results on ranges of (nondeterministic) state complexities for some operations.

given as a parameter. Finding these numbers is also a variant of the magic number problem. His results are summarized in Table 3.3.

In 1980, Brzozowski and Leiss [12] introduced Boolean finite automata (BFAs). In 1981, Chandra, Kozen, and Stockmeyer [21], considered a similar notion, alternating finite automata (AFAs). In 1990, Fellah, Jürgensen, and Yu [32] defined alternating finite automata as Boolean automata with one initial state. They provided automata constructions for complementation, intersection, union, star, and concatenation. This gave the upper bounds that are displayed in the first column of Table 3.4.

Operation	Range	Magic numbers	$\|\Sigma\|$
complementation	$\mathbb{N} \cup \{0 \mid n = 1\}$	no	1
union	$\mathbb{N}$	no	1
concatenation	$\mathbb{N}$	no	1
difference	$\{0\} \cup \mathbb{N}$	no	1
star	$\mathbb{N}$	no	1

Table 3.3: Results obtained in [25]. The languages K and L have accepting state complexity m and n, respectively, for $m, n \geq 1$. The range indicates the attainable accepting state complexities of the operation under consideration and the status of the magic number problem refers to whether or not there exist magic numbers in the given range.

	AFA	DFA	$\|\Sigma\|$	NFA	$\|\Sigma\|$
complementation	n	n	1	2^n	2
intersection	$\leq m+n+1$	mn	2	mn	2
union	$\leq m+n+1$	mn	2	$m+n+1$	2
concatenation	$\leq 2^m+n+1$	$m2^n-2^{n-1}$	2	$m+n$	2
reversal	$\leq 2^n+1$	2^n	2	$n+1$	2
star	$\leq 2^n+1$	$\frac{3}{4}2^n$	2	$n+1$	1
left quotient	$\leq m+1$	2^m-1	2	$m+1$	2
right quotient	$\leq 2^m+1$	m	1	m	1

Table 3.4: The complexity of operations on languages represented by AFAs, DFAs, NFAs. The results for DFAs are from [67, 74, 94], the results for NFAs are from [40, 57], and the results for AFAs are from [32] and [59].

It is known [32, Theorem 4.1, Corollary 4.2] and [59, Lemma 1, Lemma 2] that a language L is accepted by an n-state AFA if and only if its reversal L^R is accepted by a 2^n-state DFA with 2^{n-1} states final. Hence to get a lower bound for some operation on AFAs, we need witness languages represented by DFAs with half of states final that are hard for this operation on DFAs. Moreover, this operation must commute with reversal. The first column of Table 3.4 shows known upper bounds on complexity of considered operations on languages represented by AFAs. The second and third column show known results on complexity of these operations for DFAs and NFAs.

In 1996, Birget [6] examined the descriptional complexity of the forever operation defined as $L \mapsto (\Sigma^* L^{\complement})^{\complement}$. He considered different complexity measures based on representation of languages by DFAs, NFAs, and AFAs and obtained several trade-offs between these models. His results are provided in Table 3.5.

$L \setminus (\Sigma^* L^{\complement})^{\complement}$	DFA	$\|\Sigma\|$	NFA	$\|\Sigma\|$	AFA	$\|\Sigma\|$
DFA	2^{n-1}	2	2^{n-1}	3	$\leq n+1$	
NFA			$2^{n-1} \leq \cdot \leq 2^{n+1}+1$	3	$\leq n+1$	
AFA			$\leq 2^{n+1}+1$		$\leq n+1$	

Table 3.5: Results of Birget [6] on the descriptional complexity of the forever operator.

Chapter 4

Concatenation on Deterministic and Alternating Finite Automata

Concatenation is a binary operation on formal languages K and L defined as

$$KL = \{uv \mid u \in K \text{ and } v \in L\}.$$

This operation is regular, which means that if K and L are regular languages, than KL is also a regular language. Let $A = (Q_A, \Sigma, \cdot_A, s_A, F_A)$ be a DFA for K with $|Q_A| = m$ and $|F_A| = k$. Next, let $B = (Q_B, \Sigma, \cdot_B, s_B, F_B)$ be a DFA for L with $|Q_B| = n$ such that Q_A and Q_B are disjoint. We construct an NNFA N for KL from A and B as follows: For each state q in Q_A, each final state f in F_A, and each input symbol a, we add a transition (q, a, s_B) whenever there is a transition (q, a, f) in the DFA A. The set of initial states of N is $\{s_A\}$ if $s_A \notin F_A$ and $\{s_A, s_B\}$ otherwise, and the set of final states of N is F_B. Figure 4.1 shows a schematic drawing of the NNFA N.

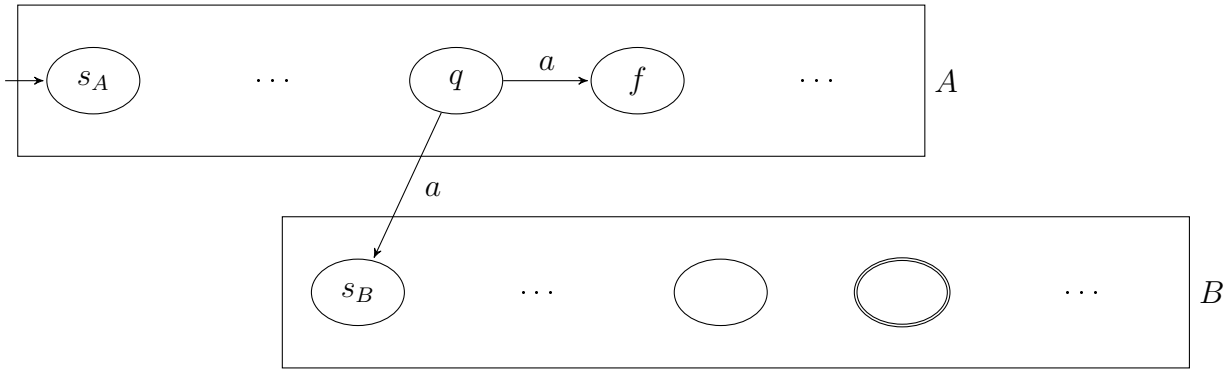


Figure 4.1: A schematic drawing of the NNFA N for concatenation of $L(A)$ and $L(B)$.

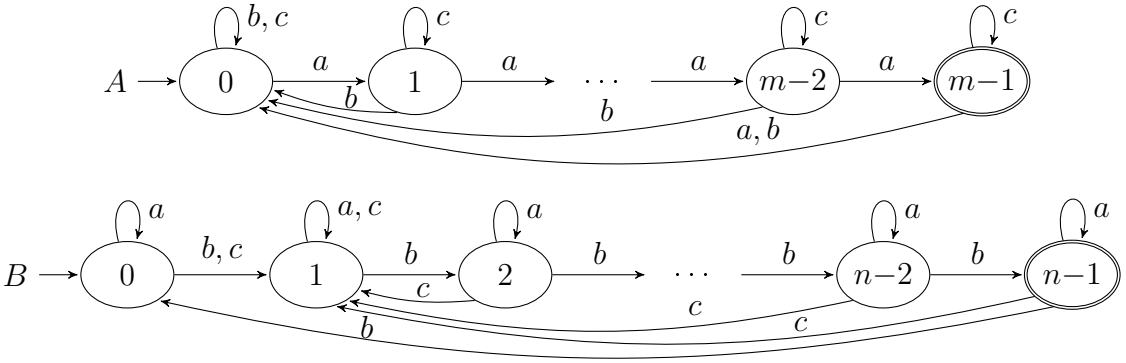


Figure 4.2: Ternary witnesses of Yu et al. [94] meeting the upper bound $m2^n - 2^{n-1}$.

To prove the upper bound for the state complexity of concatenation, we need to determinize the NFA N. In the subset automaton $\mathcal{D}(N)$, only subsets containing exactly one state in Q_A are reachable since A is deterministic and complete. Moreover, each subset containing a state in F_A and not containing s_B is unreachable. This gives the upper bound $m2^n - k2^{n-1}$ on the size of the minimal DFA for KL, as was shown by Yu, Zhuang, and Salomaa in 1994 [94, Theorem 2.3]. This bound is maximal if $k = 1$, and the lower bound $m2^n - 2^{n-1}$ was proven to be tight in [94, Theorem 2.1] using ternary witnesses shown in Figure 4.2. The authors of [94] left the binary case open, not knowing about the paper by Maslov from 1970 [74] where the tight upper bound $m2^n - 2^{n-1}$ was proven using binary witnesses shown in Figure 4.3. Before the paper [74] was discovered in 2005, tightness of the bound $m2^n - 2^{n-1}$ on binary languages was shown by Jirásková in 2005 [57, Theorem 1] using witnesses shown in Figure 4.4.

In this chapter, we study the concatenation operation on languages represented by DFAs with more final states. We are motivated by the complexity of concatenation on alternating finite automata (AFAs). In 1990, Fellah, Jürgensen, and Yu [32] provided

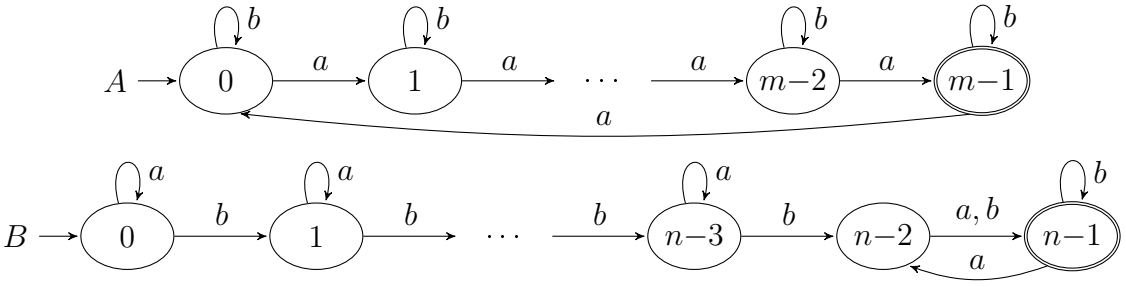


Figure 4.3: Binary witnesses of Maslov [74] meeting the upper bound $m2^n - 2^{n-1}$ if $n \geq 3$.

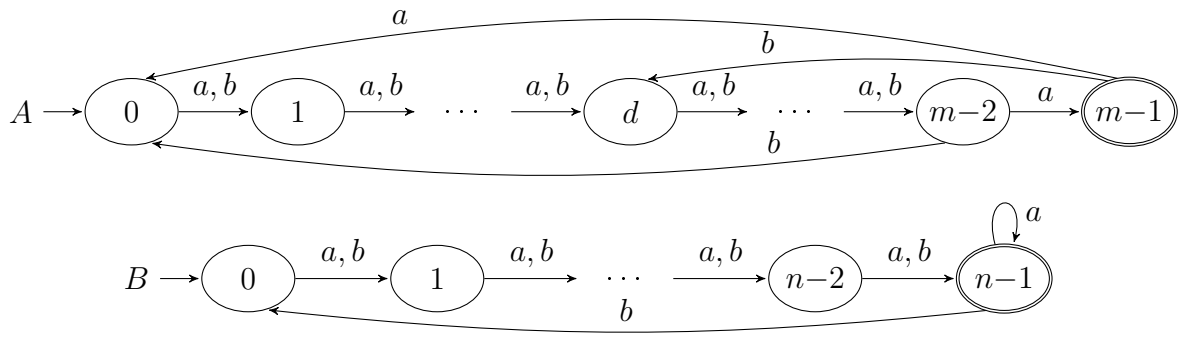


Figure 4.4: Jirásková's binary witnesses from [57, Theorem 1] meeting the tight upper bound $m2^n - 2^{n-1}$ for concatenation; we have $d = (m - n + 1) \bmod (m - 1)$.

constructions for several operations on AFAs, and they provide the upper bound 2^m+n+1 for concatenation of languages accepted by m-state and n-state AFAs [32, Theorem 9.3]. They conjectured that this bound is tight. We have mentioned in Corollary 2.20 that every language L is accepted by an n-state AFA if and only if its reversal L^R is accepted by a DFA with 2^n states and 2^{n-1} final states. Next, we have $(KL)^R = L^R K^R$. It follows that to prove lower bound on the complexity of concatenation on AFAs, it is enough to find two languages accepted by DFAs with 2^n and 2^m states and half of them final that are hard for concatenation on DFAs. Jirásek, Jirásková, and Szabari in 2005 [56] provided binary witnesses shown in Figure 4.5 meeting the tight upper bound $m2^n - k2^{n-1}$ for every k with $1 \leq k \leq m - 1$. Jirásková in 2012 [59] used these languages to prove the lower bound $2^m + n$ for concatenation on AFAs. However, she did not realize that the witnesses from [56] do not work if the second DFA has more than one final state. Moreover, there is an error in the proof of Theorem 1 in [56]. The aim of this chapter is to fix this error and to show tightness of the upper bound $2^m + n + 1$ for concatenation on AFAs. We also provide some other results concerning concatenation on DFAs with more

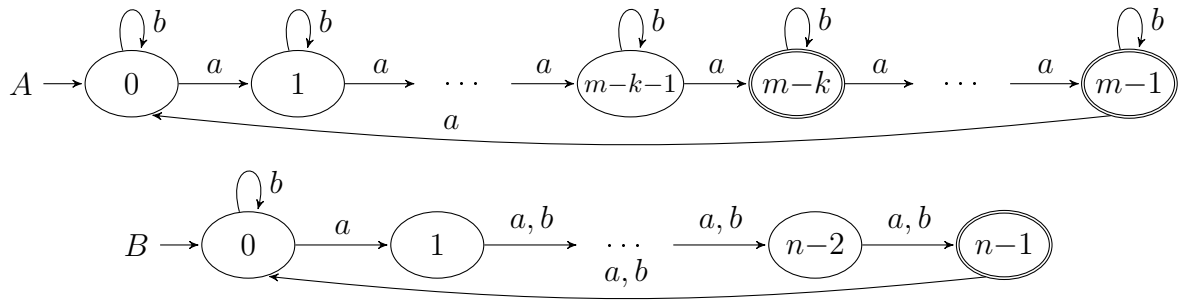


Figure 4.5: Binary witnesses of Jirásek et al. [56] meeting the upper bound $m2^n - k2^{n-1}$.

final states.
This chapter is based on the published paper which can be found in Appendix [A] at the end of this thesis:

> Hospodár, M., Jirásková, G.: The complexity of concatenation on deterministic and alternating finite automata. *RAIRO – Theoretical Informatics and Applications* 52 (2018), pp. 153–168.

We inspect three worst-case examples from the literature [74, 94, 56], and modify them by making k states in the first automaton final.

In [A, Lemma 4.3], we show that if we take the ternary witnesses of Yu et al. from Figure 4.2 and make the last k states in A final, then the upper bound $m2^n - k2^{n-1}$ is tight whenever $m, n \geq 2$ and $1 \leq k \leq m-1$.

In [A, Lemma 4.4], we show that also Maslov's binary witnesses from Figure 4.3, modified to have the last k states final in A with $1 \leq k \leq m-1$, meet the upper bound $m2^n - k2^{n-1}$ if $m \geq 2$ and $n \geq 3$.

In [A, Lemma 4.5], we provide a correct proof of [56, Theorem 1], showing that the languages accepted by DFAs shown in Figure 4.5 meet the upper bound $m2^n - k2^{n-1}$ on the complexity of concatenation for every m, n, k with $n \geq 2$.

Our next goal is to describe, for all m, n, k, ℓ with $n \geq 2$, two DFAs of m and n states, and k and ℓ final states, respectively, meeting the upper bound $m2^n - k2^{n-1}$ on the complexity of the concatenation of their languages. We try to modify the witness automata in all cases, by making the second half of their states final. The upper bound in such a case is $3m \cdot 2^{n-2}$. Table 4.1 shows that none of the three witnesses presented in papers [56, 74, 94] meets this bound. Even making two states final in DFA B results in a complexity of concatenation less that the upper bound in all three cases. Therefore we present new pairs of witness languages.

	Upper Bound			Maslov 1970 [74]			Yu et al. 1994 [94]			Jirásek et al. 2005 [56]		
	2	4	6	2	4	6	2	4	6	2	4	6
2	6	24	96	5	4	18	6	14	27	6	22	84
4	12	48	192	10	5	35	12	28	54	12	42	156
6	18	72	288	15	6	52	18	42	81	18	63	225

Table 4.1: State complexity of concatenation of $L(A)$ and $L(B)$ if the witness languages from [56, 74, 94] have the second half of their states final; in rows we have m, in columns n.

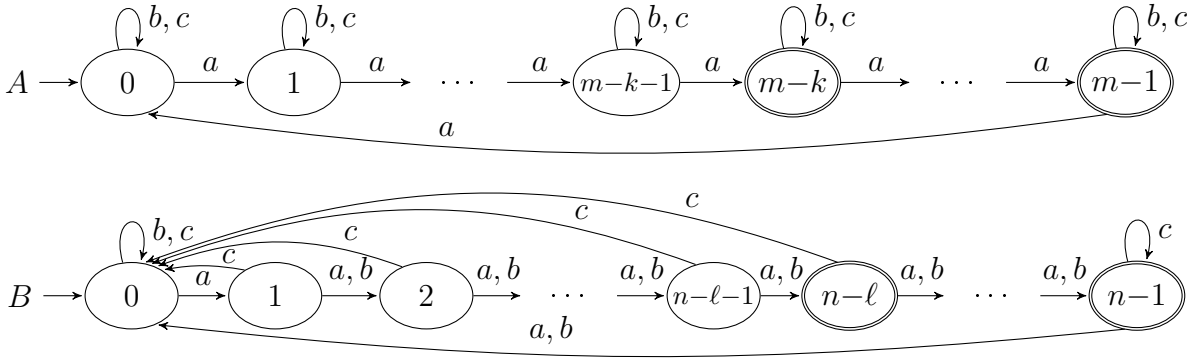


Figure 4.6: Ternary witnesses from [A, Theorem 4.6], meeting the bound $m2^n - k2^{n-1}$.

To cover all possible values of m, n, k, ℓ of the numbers of states and of final states in the DFAs A and B, we modified the witness from [56, Theorem 1] by defining transitions on a new symbol c, and thus get the ternary witnesses shown in Figure 4.6; the symbol c performs an identity on $n-1$ and leads each other state to 0. The modified ternary automata meet the upper bound $m2^n - k2^{n-1}$ for concatenation of their languages; the proof is in [A, Theorem 4.6].

We might ask whether there are binary languages with more final states in B meeting this bound. We provide a positive answer in [A, Theorem 4.7] using witness languages accepted by DFAs from Figure 4.7. However, we require $k \leq m-2$ here, that is, the first DFA must have at least two non-final states. Moreover, we assume that $m \geq 3$ and $n \geq 4$. The DFA B is the same as in [28].

Next we turn our attention to the concatenation of binary languages represented by an m-state DFA with $m-1$ final states and an n-state DFA with more than one final state. In the general case, the upper bound is $(m+1)2^{n-1}$. Theorem 5.1 of [A] provides a lower bound $(m+1)2^{n-1}-1$ which is smaller just by one. Our computations show that

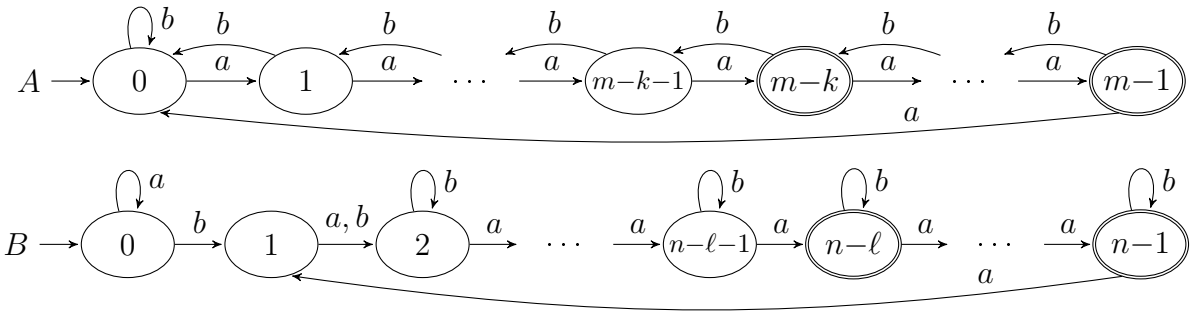


Figure 4.7: Binary witnesses from [A, Theorem 4.7] meeting the bound $m2^n - k2^{n-1}$.

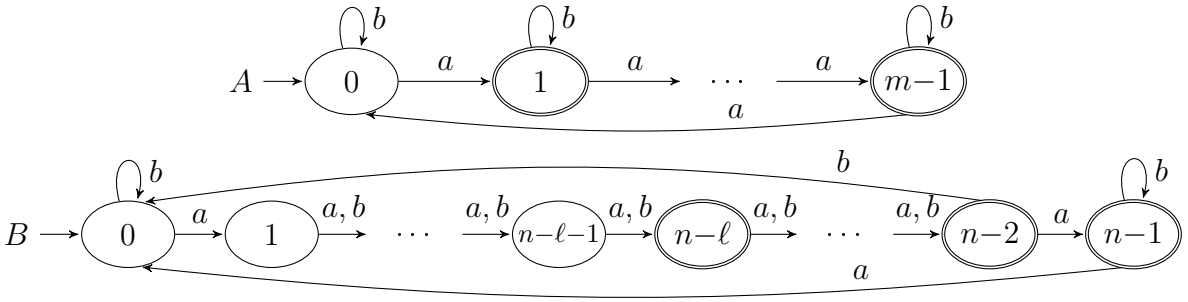


Figure 4.8: The binary DFAs from [A, Theorem 5.1] meeting the bound $(m+1)2^{n-1}-1$.

no pair of binary languages meets the bound $(m+1)2^{n-1}$ in the case of $m, n \leq 4$. The witness languages are accepted by DFAs shown in Figure 4.8. The DFA B is the same as the witness DFA for square in [70] in the case $k = n-1$.

Using witness languages from [A, Theorem 4.7], we get the main result of this chapter. Recall that $\mathrm{sc}(L)$ denotes the smallest number of states in any DFA for L, and $\mathrm{asc}(L)$ denotes the smallest number of states in any AFA for L.

Theorem 4.1 ([A, Lemma 6.4], Concatenation on AFAs: Lower Bound). *There exist binary languages K and L accepted by an m-state and n-state* AFA*, respectively, such that every* AFA *for KL has at least $2^m + n + 1$ states, where $m, n \geq 2$.*

Proof. Let L^R and K^R be the binary languages accepted by the minimal DFAs A and B from Figure 4.7 with 2^n and 2^m states, respectively, both having half of states final. Then, by Lemma 2.19, the languages K and L are accepted by an m-state and n-state AFA, respectively. Since L^R and K^R are witnesses for concatenation on DFAs, we get

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

and therefore $\mathrm{asc}(KL) \geq \lceil \log(2^{n-1} \cdot 2^{2^m}(1+1/2)) \rceil = 2^m + n$ by Corollary 2.18.

Our next aim is to show that $\mathrm{asc}(KL) \geq 2^m + n + 1$. Suppose for a contradiction that KL is accepted by an AFA of $2^m + n$ states. Then, by Lemma 2.17, $(KL)^R$ is accepted by a 2^{2^m+n}-state DFA with 2^{2^m+n-1} final states. It follows that the minimal DFA for $(KL)^R$ has at most 2^{2^m+n-1} final states. However, the number of final states in the minimal DFA for $(KL)^R$ is

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

which is more than 2^{2^m+n-1}, a contradiction. It follows that $\mathrm{asc}(KL) \geq 2^m + n + 1$. □

The lower bound in the theorem above meets the upper bound from [32, Theorem 9.3], which solves the open problem stated by Fellah, Jürgensen, and Yu in 1990 [32].

Chapter 5

Range of State Complexities for the Cut Operation

The cut operation is a machine implementation of "concatenation" on UNIX text processors which behaves greedy-like in its left term of concatenation. Formally, the cut operation on languages K and L, denoted by $K \,!\, L$, is defined as

$$K \,!\, L = \{\, uv \mid u \in K,\, v \in L, \text{ and } uv' \notin K \text{ for every non-empty prefix } v' \text{ of } v \,\}.$$

This operation preserves regularity, as shown by Berglund et al. [2]. The state complexity of the cut operation on DFAs was obtained by Drewes et al. [29, Theorem 3.1] and it is given by the following function from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$:

$$f(m,n) = \begin{cases} m, & \text{if } n = 1; \\ (m-1)n + m, & \text{if } n \geq 2. \end{cases} \tag{5.1}$$

In the unary case, the state complexity of cut is given by the function

$$f_1(m,n) = \begin{cases} 1, & \text{if } m = 1; \\ m, & \text{if } m \geq 2 \text{ and } n = 1; \\ 2m-1, & \text{if } m,n \geq 2 \text{ and } m \geq n; \\ m+n-2, & \text{if } m,n \geq 2 \text{ and } m < n; \end{cases} \tag{5.2}$$

as proven in [29, Theorem 3.2]. Notice that the state complexity of the cut operation is only linearly growing with both parameters, while the state complexity $m2^n - 2^{n-1}$ of concatenation is growing linearly with m and exponentially with n.

In this chapter, we show for every value from 1 up to $f_1(m,n)$ whether or not it can be attained by the state complexity of the cut of two *unary* languages accepted by minimal DFAs with m and n states. We show that only complexities up to $2m-1$ and between n and $m+n-2$ can be attained, while complexities from $2m$ up to $n-1$ turn out to be magic in the unary case. To get these results, the tail-loop structure of minimal unary DFAs is very valuable in the proofs.

On the other hand, we show that the entire range of complexities, up to the known upper bound $f(m,n)$, can be produced by the cut operation on minimal DFAs with m and n states, respectively, in case when the input alphabet consists of at least two symbols. The proof of this result resembles some ideas used by Hricko [50] and Szabari [90] for the magic number problem of the intersection and union operations on DFAs.

To the best of our knowledge, this is the first operation where for every alphabet size, every value in the range of possible complexities is known to be either attainable or not, and not all values are attainable in the unary case. Hence, the magic number problem for the cut operation is completely solved in this chapter. The results of this chapter are based on the conference paper:

> Holzer, M., Hospodár, M.: The Range of state complexities of languages resulting from the cut operation. In: Martín-Vide, C., Okhotin, A., Shapira, D. (eds.): *Proc. 13th International Conference on Language and Automata Theory and Applications, LATA 2019, Saint Petersburg, Russia, March 26–29, 2019.* Lecture Notes in Computer Science, vol. 11417, Springer, pp. 190–202 (2019).

Since we are interested in the state complexity of languages resulting from the cut operation, we briefly recall the construction of a DFA for the cut operation. For two languages accepted by DFAs A and B with m and n states, respectively, we can construct the *cut automaton* $A\,!\,B$ as follows. The cut automaton has states in a grid like the product automaton $A \times B$, plus one additional column which we denote by $\perp$. The column $\perp$ corresponds to the situation that we have not read a string in $L(A)$ yet. When we reach a final state in A for the first time, we leave the column $\perp$ and enter the product part of $A\,!\,B$ in the corresponding state. Unless we reach a final state of A again, the transitions in $A\,!\,B$ are the same as in the product automaton $A \times B$. When we reach a final state in A again, we reset to a state in the column for the initial state of B. This corresponds to the situation that we have read a longer string in $L(A)$, and therefore we need to reset the computation of B to its initial state. The final states of $A\,!\,B$ are all states in columns corresponding to the final states of B.

Formally, let $A = (Q_A, \Sigma, \delta_A, s_A, F_A)$ and $B = (Q_B, \Sigma, \delta_B, s_B, F_B)$ be two DFAs. Let $\perp \notin Q_B$. Define the cut automaton

$$A\,!\,B = (Q := (Q_A \times \{\perp\}) \cup (Q_A \times Q_B), \Sigma, \delta, s, Q_A \times F_B)$$

where $s = (s_A, \perp)$ if $\varepsilon \notin L(A)$ and $s = (s_A, s_B)$ otherwise, and for each state (p, q) in Q and each input symbol a in Σ we have

$$\delta((p, \perp), a) = \begin{cases} (\delta_A(p, a), \perp), & \text{if } \delta_A(p, a) \notin F_A; \\ (\delta_A(p, a), s_B), & \text{otherwise;} \end{cases}$$

and

$$\delta((p, q), a) = \begin{cases} (\delta_A(p, a), \delta_B(q, a)), & \text{if } \delta_A(p, a) \notin F_A; \\ (\delta_A(p, a), s_B), & \text{otherwise.} \end{cases}$$

Then $L(A\,!\,B) = L(A)\,!\,L(B)$. Figures 5.1, 5.3, 5.2, and 5.4 show the schematic drawings of A and B, as well as the cut automaton $A\,!\,B$. The DFA A is drawn in the column in the left part of each figure, and the DFA B is drawn in the row in the top part of each figure.

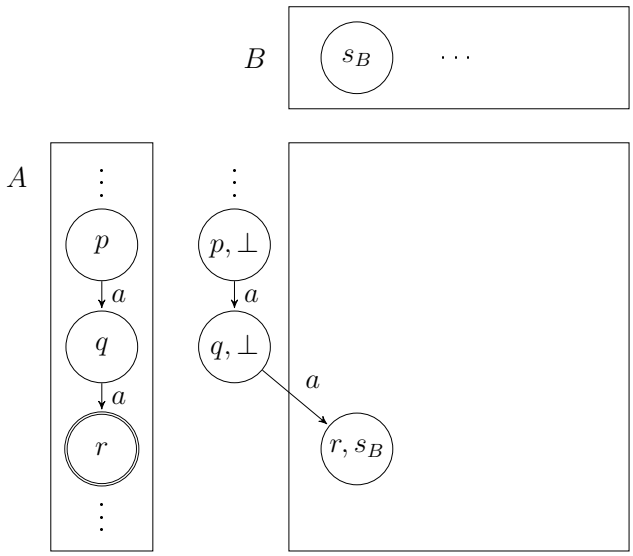


Figure 5.1: The DFAs A and B and the cut automaton $A\,!\,B$. Here, $r = \delta_A(q, a)$ is the first final state in a computation in the DFA A. Since r is a final state of A, no state except for (r, s_B) is reachable in row r of the cut automaton $A\,!\,B$. The transitions in column $\perp$, are the same as the corresponding transitions in A going to a non-final state.

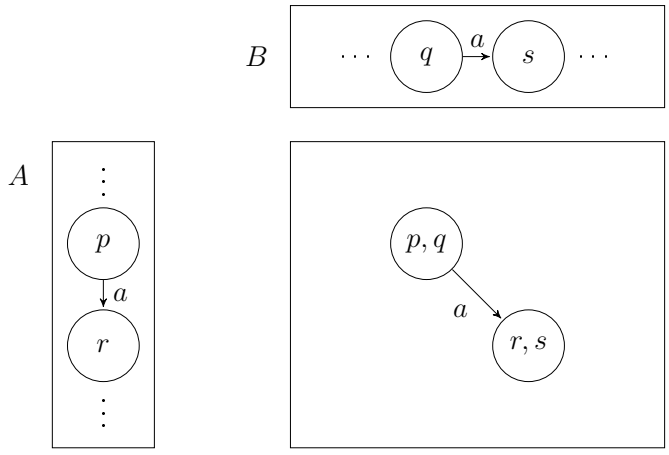


Figure 5.2: The DFAs A and B and the cut automaton $A\,!\,B$. We have $r = \delta_A(p, a)$, $s = \delta_B(q, a)$, and r is non-final in A, hence we proceed like in the product automaton $A \times B$.

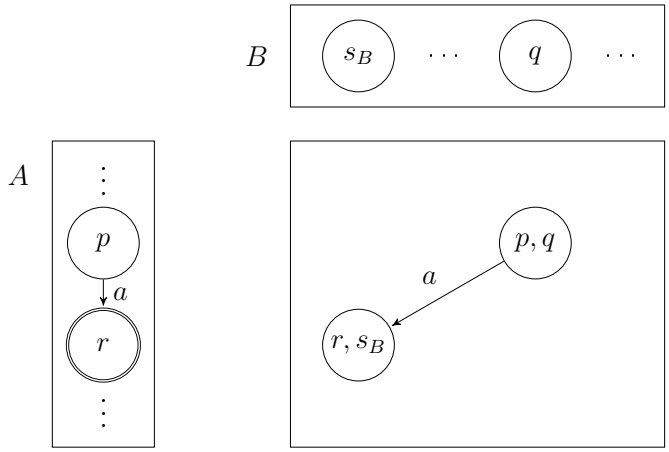


Figure 5.3: The DFAs A and B and the cut automaton $A\,!\,B$. We have $r = \delta_A(p, a)$, and the state r is final in A, hence we reset the computation in $A\,!\,B$ to the state (r, s_B).

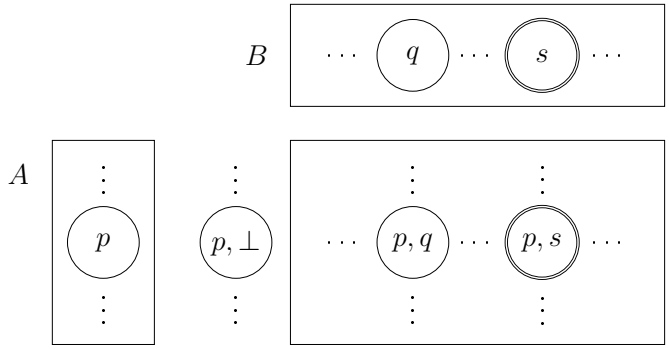


Figure 5.4: The DFAs A and B and the cut automaton $A\,!\,B$. Since the state s is final in B, each state (p, s) is final in $A\,!\,B$.

For better understanding of the cut operation, we present the following example.

Example 5.1. *Consider* DFA*s* A *and* B *shown in Figure 5.5 (left) and (top). Since the initial state of* A *is non-final, the initial state of the cut automaton* $A\,!\,B$ *is* $(0, \bot)$.

(1) After reading a*, the* DFA A *is in a non-final state, therefore in* $A\,!\,B$ *there is the transition from* $(0, \bot)$ *to* $(1, \bot)$ *on* a*. After reading the second* a*, the* DFA A *is in the final state* 2*, therefore the cut automaton enters its product part in the state* $(2, 0)$*. Similarly, after reading* aba*, the cut automaton enters its product part in the state* $(4, 0)$.

(2) In the product part, $A\,!\,B$ *behaves like the ordinary product automaton* $A \times B$ *provided that the current transition sends the first component to a non-final state of* A*, see, for example, the transitions from* $(0, 0)$ *to* $(1, 0)$ *on* a *or from* $(0, 0)$ *to* $(0, 1)$ *on* b.

(3) Otherwise, if the current transition sends the first component to the final state 2 *or* 4*, the computation of the cut automaton is reset to the corresponding state* $(2, 0)$ *or* $(4, 0)$*, respectively, in column* 0.

(4) The set of final states of $A\,!\,B$ *consists of all states in columns* 1 *and* 3.

The cut automaton is not minimal: if p *is a final state of* A*, then all states in row* p *except for* $(p, 0)$ *are unreachable. In our example, also the state* $(1, 1)$ *is unreachable.*

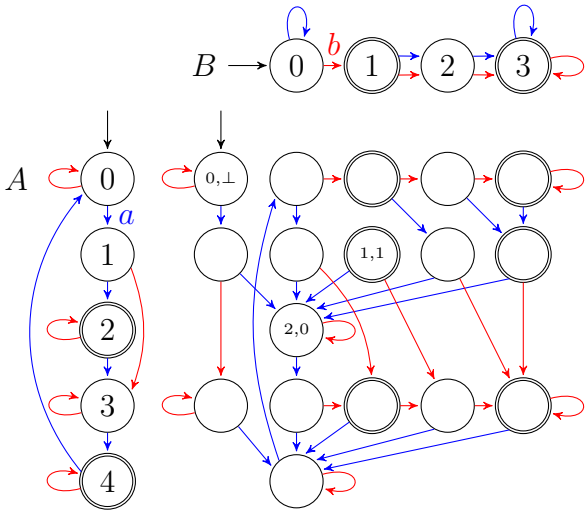


Figure 5.5: An example of DFAs A and B with $m = 5$ and $n = 4$ and their cut automaton $A\,!\,B$; notice that the state $(1, 1)$ is unreachable.

In what follows, all our figures display only the reachable states of cut automata.

To help the reader to become even more familiar with the cut operation, we prove the following two observations. The first one considers the case when the first language is a right ideal.

Proposition 5.2. *Let K be a right ideal over Σ, that is, a language satisfying $K = K\Sigma^*$. Then*

$$K \,!\, L = \begin{cases} K, & \text{if } \varepsilon \in L; \\ \emptyset, & \text{otherwise.} \end{cases}$$

Proof. By definition of the cut operation, we have

$$K \,!\, L = \{uv \mid u \in K,\ v \in L, \text{ and } uv' \notin K \text{ for every non-empty prefix } v' \text{ of } v\}.$$

If $\varepsilon \in L$, then $K \subseteq K \,!\, L$ since for every w in K we have $w = w \cdot \varepsilon$ with $\varepsilon \in L$ and the set of non-empty prefixes of ε is empty. Conversely, we have $K \,!\, L \subseteq KL \subseteq K\Sigma^* = K$ since K is a right ideal.

Now let $\varepsilon \notin L$. If $K = \emptyset$ or $L = \emptyset$, then $KL = \emptyset$, so $K \,!\, L = \emptyset$ by definition. Otherwise, let $u \in K$ and v be a non-empty string in L. Since K is a right ideal, $uv' \in K$ for every non-empty prefix v' of v. Thus no string uv can be in $K \,!\, L$, and the proposition follows. □

Our second observation shows that if the second language is universal, the cut operation acts like ordinary concatenation.

Proposition 5.3. *Let K be a language over an alphabet Σ. Then $K \,!\, \Sigma^* = K\Sigma^*$.*

Proof. We have $K \,!\, \Sigma^* \subseteq K\Sigma^*$ by definition. Let $w \in K\Sigma^*$. Then $w = uv$ for some $u \in K$ and $v \in \Sigma^*$. Let v' be the longest non-empty prefix of v such that $uv' \in K$. Then we have $w = uv'v''$ and $uv' \cdot v'' \in K \,!\, \Sigma^*$. Hence $K\Sigma^* \subseteq K \,!\, \Sigma^*$, and the stated claim follows. □

5.1 The Cut Operation on Unary Regular Languages

When working with unary DFAs, we use the notational convention proposed by Nicaud in [80]. Every unary DFA consists of a *tail* path, which starts from the initial state, followed by a cycle of one or more states, called the *loop*. Therefore a unary DFA is totally determined by the number of states, the loop number given by the length of the tail path, and the set of final states.

Let $A = (Q, \{a\}, \delta, s, F)$ be a unary DFA with $|Q| = n$. We can identify the states of A with integers from $[0, n-1]$ via $q \mapsto \min\{\, i \mid \delta(s, a^i) = q \,\}$; recall that $[i, j]$ denotes the set $\{i, i+1, \ldots, j\}$. In particular, the initial state s is mapped to 0. Let $\ell = \delta(s, a^n)$. Then the unary DFA A with n states, loop number ℓ with $0 \leq \ell \leq n-1$, and set of final states F with $F \subseteq [0, n-1]$ is referred to as $A = (n, \ell, F)$. The following characterization of minimal unary DFAs is known.

Lemma 5.4 ([80, Lemma 1])**.** *A unary* DFA $A = (n, \ell, F)$ *is minimal if and only if*

1. *its loop is minimal, and*
2. *if $\ell \neq 0$, then states $n-1$ and $\ell-1$ do not have the same finality, that is, exactly one of them is final; see Figure 5.6 for an illustration.*

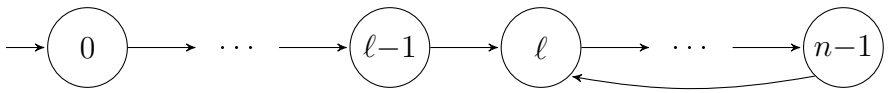


Figure 5.6: The structure of a minimal unary DFA; states $\ell - 1$ and $n - 1$ do not have the same finality.

Now we are ready for our first result on the state complexity with respect to the cut operation of unary regular languages represented by DFAs. In a series of lemmas we consider the state complexity α of the resulting language in increasing order of α. The first interval we are going to discuss is $[1, m]$.

Lemma 5.5. *Let $m, n \geq 1$ and $1 \leq \alpha \leq m$. There exist a minimal unary m-state* DFA A *and a minimal unary n-state* DFA B *such that the minimal* DFA *for $L(A) \,!\, L(B)$ has α states.*

Proof. The proof has five cases:

(1) Let $m = 1$, so we must have $\alpha = 1$. Let A be the one-state DFA accepting the empty language and B be the minimal n-state DFA for $a^{n-1}a^*$. Then $L(A) \,!\, L(B) = \emptyset$ which is accepted by a minimal one-state DFA.

(2) Let $m \geq 2$ and $n = 1$. Let A be the minimal m-state DFA for $a^{\alpha-1}(a^m)^*$ and B be the one-state DFA for a^*. The reachable part of the cut automaton $A \,!\, B$ consists of the tail of non-final states $(i, \perp)$ with $0 \leq i \leq \alpha - 2$ and the loop of final states $(i, 0)$ with $0 \leq i \leq m - 1$; see Figure 5.7 for an illustration. Since all the final states are equivalent, the minimal DFA for $L(A) \,!\, L(B)$ has α states.

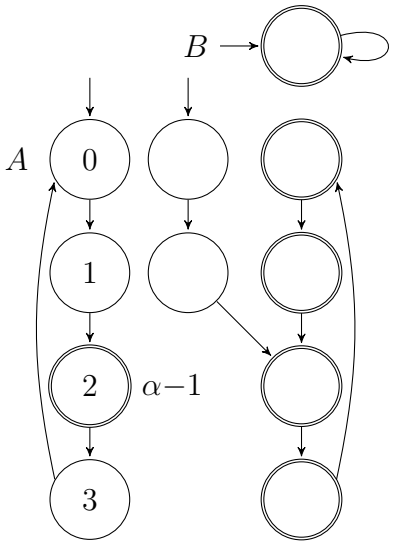


Figure 5.7: An example of the DFAs A and B and the cut automaton with $m = 4$, $n = 1$, and $\alpha = 3$ for case 2 of Lemma 5.5.

(3) Let $m, n \geq 2$ and $\alpha = 1$. Consider the unary languages $a^{m-1}a^*$ and $a^{n-1}a^*$ accepted by minimal DFAs A and B of m and n states, respectively. Then the reachable part of the cut automaton consists of the tail of non-final states $(i, \perp)$ with $1 \leq i \leq m-2$ and the loop consisting of a single non-final state $(m-1, 0)$; see Figure 5.8 and notice that 0 is a non-final state in B. Hence $L(A)\,!\,L(B)$ is the empty language accepted by a one-state DFA.

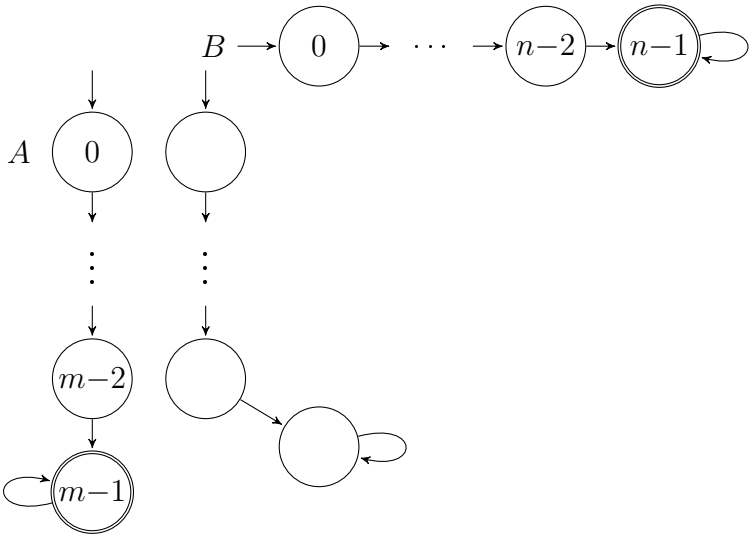


Figure 5.8: The DFAs A and B and the cut automaton for case 3 of Lemma 5.5.

(4) Let $m \geq 2$, $n = 2$, and $2 \leq \alpha \leq m$. Consider the unary languages K and L defined as follows. If $m - \alpha$ is even, then let $K = \{ a^{\alpha-2}, a^{m-2} \}$ and $L = a(aa)^*$, otherwise, let $K = \{ a^{\alpha-1}, a^{m-2} \}$ and $L = (aa)^*$. The minimal DFAs for K and L have m and 2 states, respectively, see Figure 5.9 for an illustration of an example with $m = 8$. We have $K \,!\, L = a^{\alpha-1}(aa)^*$, which is accepted by a minimal α-state DFA.

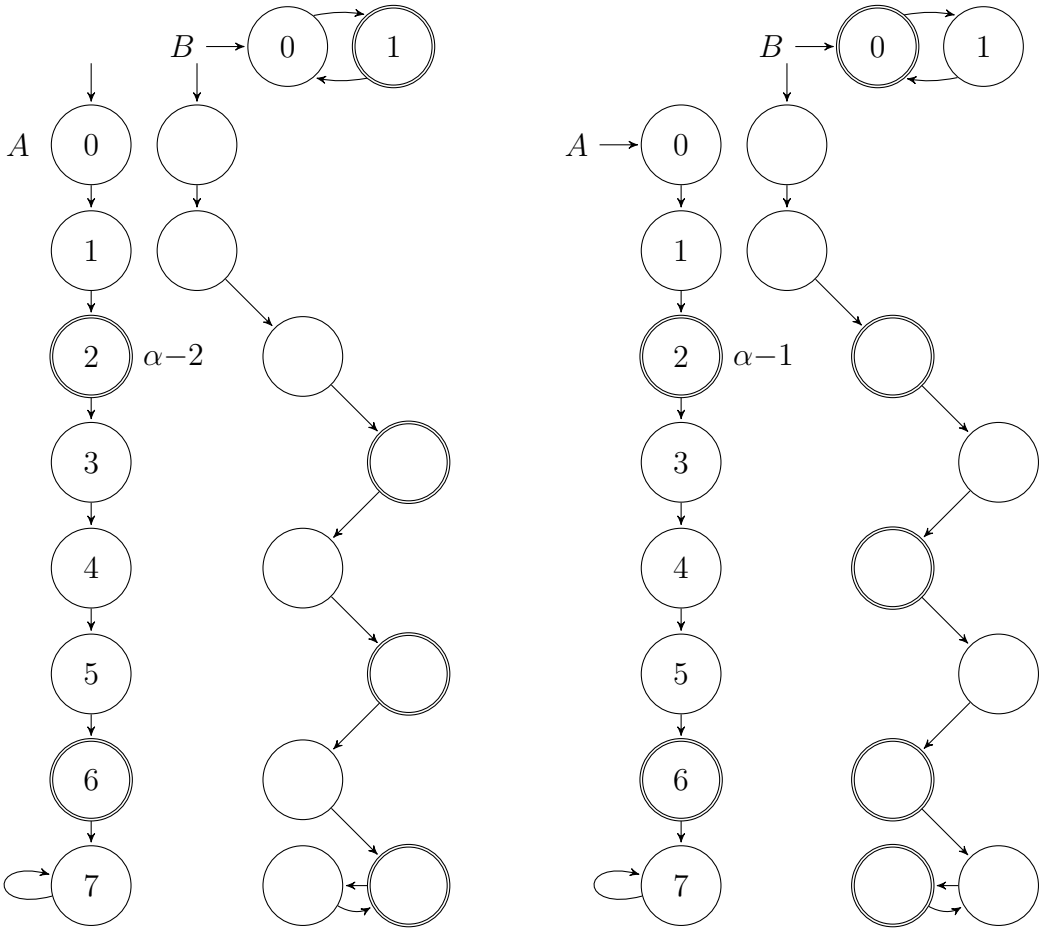


Figure 5.9: The minimal DFAs for K and L with $m = 8$ and $n = 2$ for case 4 of Lemma 5.5. In the left figure, we have $\alpha = 4$, so $m - \alpha$ is even, and in the right figure, we have $\alpha = 3$, so $m - \alpha$ is odd. The minimal DFA for $K \,!\, L$ has α states in both subcases.

(5) Let $m \geq 2$, $n \geq 3$, and $2 \leq \alpha \leq m$. Consider the DFAs $A = (m, \alpha-2, [\alpha-1, m-1])$ and $B = (n, n-1, [0, n-2])$. By Lemma 5.4, the DFAs A and B are minimal. The reachable part of the cut automaton consists of the tail of $\alpha - 1$ non-final states and of the loop of $m - \alpha + 2$ final states, as illustrated in Figure 5.10. Hence the minimal DFA $(\alpha, \alpha - 1, \{\alpha - 1\})$ for $L(A) \,!\, L(B)$ has α states. □

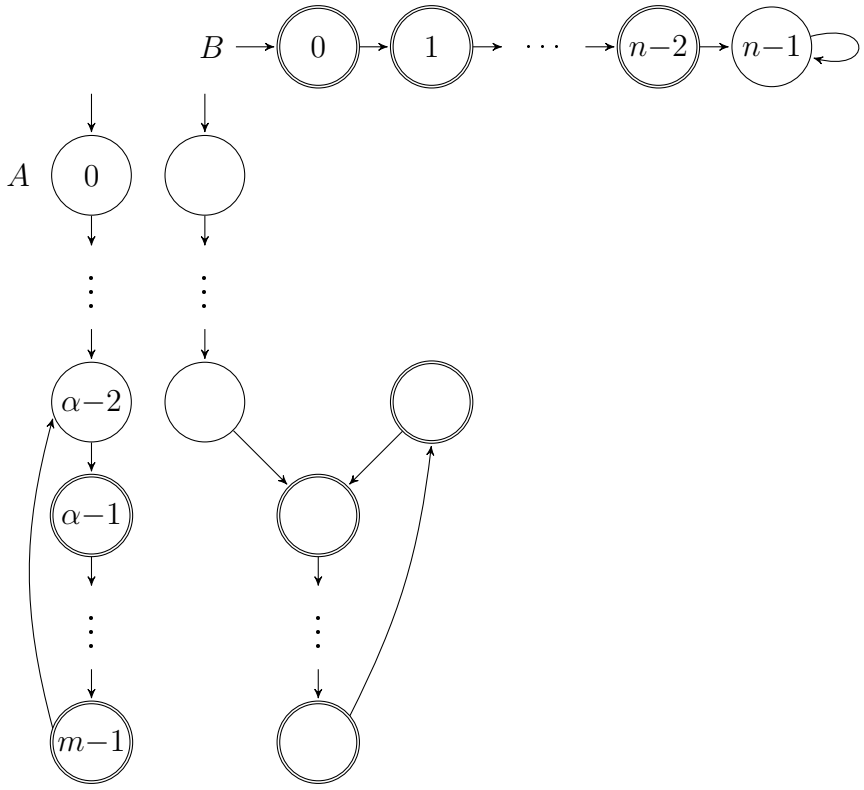


Figure 5.10: The DFAs A and B and the cut automaton for case 5 of Lemma 5.5.

Our next interval is $[m+1, 2m-1]$; cf. $f_1(m,n)$ defined by (2) on page 35.

Lemma 5.6. *Let $m, n \geq 2$ and $m+1 \leq \alpha \leq 2m-1$. There exist a minimal unary m-state* DFA *A and a minimal unary n-state* DFA *B such that the minimal* DFA *for $L(A)\,!\,L(B)$ has α states.*

Proof. We have $\alpha = m + \beta$ for some integer β with $1 \leq \beta \leq m-1$. Consider the unary DFA $A = (m, 0, \{\beta\})$. Define the unary DFA B as follows:

$$B = \begin{cases} (n, 0, \{m-1\}), & \text{if } m < n; \\ (n, n-1, \{n-1\}), & \text{otherwise.} \end{cases}$$

By Lemma 5.4, the DFAs A and B are minimal. If $m < n$, then $L(A)\,!\,L(B)$ is accepted by the DFA $(\alpha, \beta, \{\alpha - 1\})$, see Figure 5.11 for $m = 5$, $n = 6$, and $\alpha = 7$; otherwise, it is accepted by the DFA $(\alpha, \beta, [n + \beta - 1, \alpha - 1])$, see Figure 5.12 for $m = 6$, $n = 5$, and $\alpha = 10$. The resulting DFA is minimal by Lemma 5.4. □

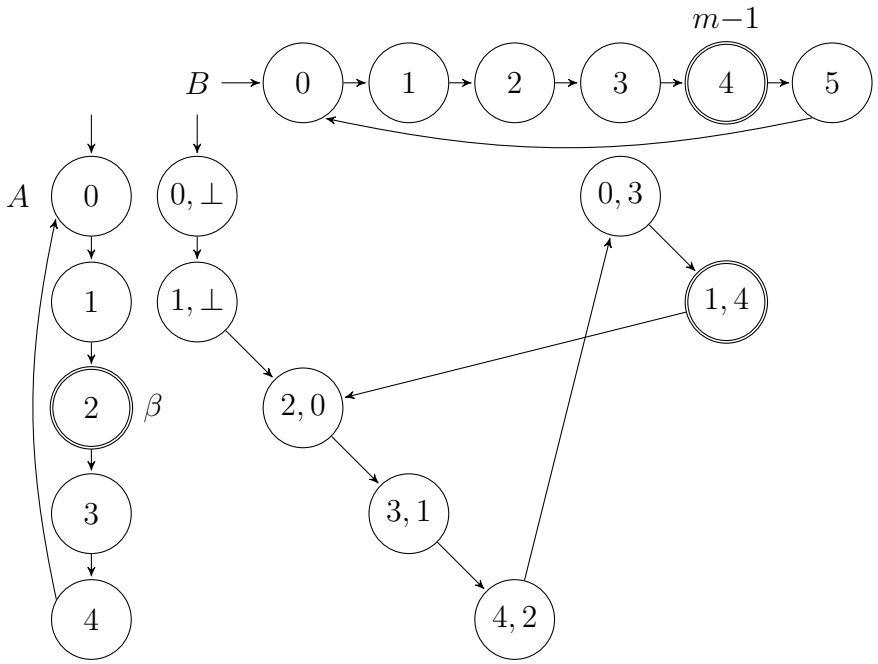


Figure 5.11: An example of the DFAs A and B and the cut automaton for Lemma 5.6, case $m < n$; we have $m = 5$, $n = 6$, and $\alpha = 7$, so $\beta = 2$.

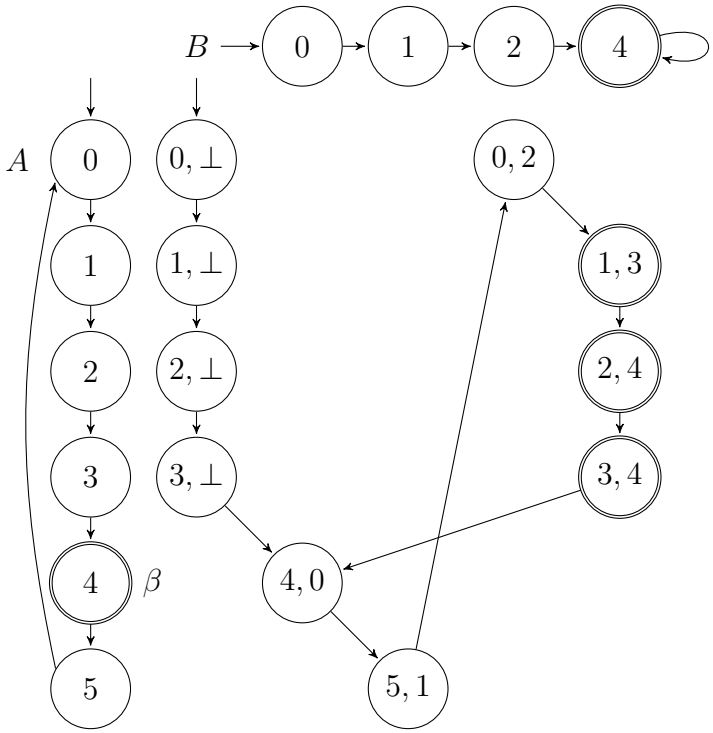


Figure 5.12: An example of the DFAs A and B and the cut automaton for Lemma 5.6, case $m \geq n$; we have $m = 6$, $n = 4$, and $\alpha = 10$, so $\beta = 4$.

The last interval we are considering in this series of lemmas is $[n, m+n-2]$.

Lemma 5.7. *Let $m, n \geq 2$, $\alpha \geq m$, and $n \leq \alpha \leq m+n-2$. There exist a minimal unary m-state* DFA *A and a minimal unary n-state* DFA *B such that the minimal* DFA *for $L(A) \,!\, L(B)$ has α states.*

Proof. Consider the DFAs $A = (m, m-1, \{m-2\})$ and $B = (n, 0, \{\alpha - m + 1\})$ which are minimal by Lemma 5.4; notice that $1 \leq \alpha - m + 1 \leq n-1$. The DFAs A and B and their cut automaton are illustrated in Figure 5.13 for case $m = 4$, $n = 5$, and $\alpha = 6$.

In the cut automaton $A \,!\, B$, the states $(m-2, 0)$ and $(m-1, 0)$ are non-final and both of them are sent to $(m-1, 1)$ on a, hence they are equivalent. Next, for each i with $1 \leq i \leq n - \alpha + m - 2$, the states $(m-2-i, \bot)$ and $(m-1, n-i)$ are equivalent as well; notice that $n - \alpha + m - 2 \geq 0$ since $\alpha \leq m+n-2$. To get the minimal DFA for the cut, we redirect the out-transition from the state $(m-1, \alpha-m+1)$ to the state $(\alpha-n, \bot)$ if $\alpha \leq m+n-3$ and to the state $(\alpha-n, 0)$ if $\alpha = m+n-2$. Since $m-1+(\alpha-m+1) = \alpha$, the resulting minimal DFA for $L(A) \,!\, L(B)$ has α states. □

For certain values of m and n, the intervals stated in the previous lemmas may not be contiguous. For instance, if $m = 2$ and $n = 5$, then the intervals from Lemmas 5.5, 5.6, and 5.7 cover $\{1, 2, 3, 5\}$. Hence the value 4 from the interval $[2m, n-1]$ is missing. In fact, we show that whenever this interval is non-empty, these values cannot be obtained by an application of the cut operation on minimal DFAs with m and n states.

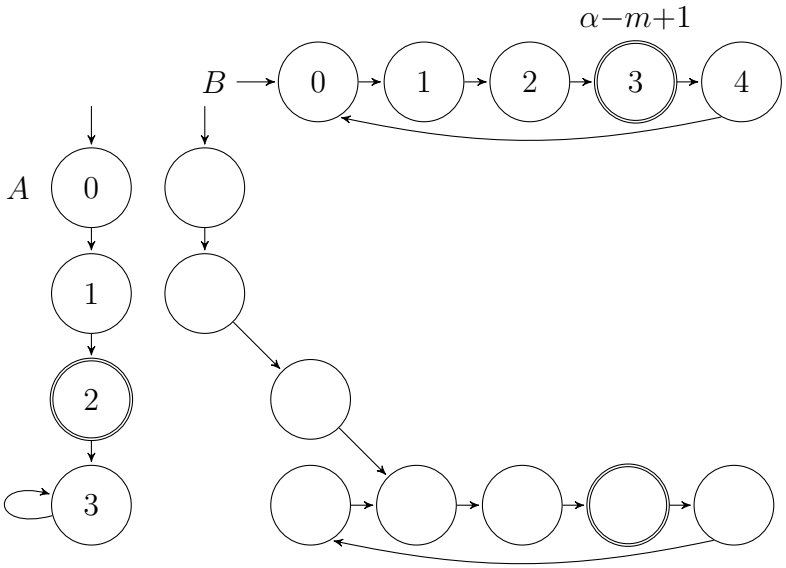


Figure 5.13: An example for the DFAs A and B and the cut automaton for Lemma 5.7; we have $m = 4$, $n = 5$, and $\alpha = 6$.

Lemma 5.8. *Let $m, n \geq 2$ be numbers satisfying $2m \leq n-1$. Then for every α such that $2m \leq \alpha \leq n-1$, there exist* no *minimal unary m-state* DFA *A and minimal unary n-state* DFA *B such that the minimal* DFA *for $L(A)\,!\,L(B)$ has α states.*

Proof. We discuss two cases depending on whether $L(A)$ is infinite or finite.

If $L(A)$ is infinite, then A must have a final state in its loop. Denote the size of the loop in A by ℓ and the smallest final state in the loop of A by j. Consider the cut automaton $A\,!\,B$. Notice that its initial state is sent to the state $(j, 0)$ by the string a^j. Next, the state $(j, 0)$ is sent to itself by the string a^ℓ. It follows that $A\,!\,B$ is equivalent to a DFA $(j + \ell, j, F)$ for some set $F \subseteq [0, j + \ell - 1]$. Since $j \leq m - 1$ and $\ell \leq m$, the DFA for $L(A)\,!\,L(B)$ has at most $2m - 1$ states.

If $L(A)$ is finite, then A has a loop in the non-final state $m - 1$ and the state $m - 2$ is final. Let $A = (m, m - 1, F)$ and $B = (n, \ell, F')$ be minimal unary DFAs for some sets $F \subseteq [0, m - 1]$ and $F' \subseteq [0, n - 1]$. It follows that in the cut automaton $A\,!\,B$, the state $(m - 2, 0)$ and the states $(m - 1, j)$ with $1 \leq j \leq n - 1$ are reachable. Two distinct states $(m - 1, j)$ and $(m - 1, j')$ are distinguishable by the same string as the states j and j' in B, and the state $(m - 2, 0)$ and a state $(m-1, j)$ are distinguishable by the same string as the states 0 and j are distinguishable in B. It follows that the cut automaton has at least n reachable and pairwise distinguishable states, and the theorem follows. □

Now let us summarize the results of this section; recall that by (2) on page 35, the state complexity of the cut operation on unary languages is given by the function

$$f_1(m, n) = \begin{cases} 1, & \text{if } m = 1; \\ m, & \text{if } m \geq 2 \text{ and } n = 1; \\ 2m - 1, & \text{if } m, n \geq 2 \text{ and } m \geq n; \\ m + n - 2, & \text{if } m, n \geq 2 \text{ and } m < n. \end{cases}$$

Theorem 5.9 (Unary Case). *For every $m, n, \alpha \geq 1$ such that*

(a) $\alpha = 1$ if $m = 1$,

(b) $1 \leq \alpha \leq m$ if $m \geq 2$ and $n = 1$, or

(c) $1 \leq \alpha \leq 2m - 1$ or $n \leq \alpha \leq m + n - 2$ if $m, n \geq 2$,

there exist a minimal unary m-state DFA *A and a minimal unary n-state* DFA *B such that the minimal* DFA *for $L(A)\,!\,L(B)$ has α states. If $m, n \geq 2$ and $2m \leq \alpha \leq n - 1$, then there do not exist minimal unary m-state and n-state* DFA*s A and B such that the minimal* DFA *for $L(A)\,!\,L(B)$ has α states.* □

5.2 The Cut Operation on Binary Regular Languages

Next we consider the range of state complexities of languages resulting from the cut operation on regular languages over an arbitrary alphabet. The aim of this section is to show that the entire range of complexities up to the known upper bound can be produced in this case, even for languages over a binary alphabet. First we show that the numbers in $[1, m+n-2]$ are attainable in the binary case. The values in $[1, 2m-1]$ as well as the cases of $m = 1$ and $n = 1$ are covered by Theorem 5.9 since adding dummy input symbols or duplicating them does not change the state complexity.

Lemma 5.10. *Let $m, n \geq 2$ and $2m \leq \alpha \leq m+n-2$. There exist a minimal binary m-state* DFA *A and a minimal binary n-state* DFA *B such that the minimal* DFA *for $L(A) \,!\, L(B)$ has α states.*

Proof. Notice that in this case we must have $m \leq n-2$, so $m < n$. Consider the binary DFA $A = ([0, m-1], \{a, b\}, \delta_A, 0, \{m-1\})$, where

$$\delta_A(i, a) = (i+1) \bmod m \quad \text{and} \quad \delta_A(i, b) = \begin{cases} (i+1) \bmod m, & \text{if } i \neq m-2, \\ m-2, & \text{otherwise.} \end{cases}$$

Next, consider the binary DFA $B = ([0, n-1], \{a, b\}, \delta_B, 0, \{m-1\})$, where

$$\delta_B(j, a) = (j+1) \bmod n \quad \text{and} \quad \delta_B(j, b) \qquad = \begin{cases} (j+1) \bmod n, & \text{if } j \neq \alpha - m, \\ m-1, & \text{otherwise.} \end{cases}$$

Both automata A and B are depicted in Figure 5.14.

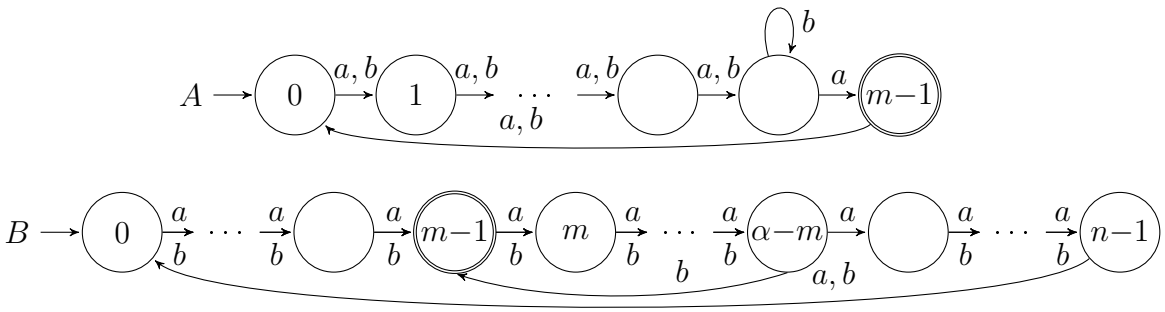


Figure 5.14: The DFAs A (top) and B (bottom) for the case $2m \leq \alpha \leq m+n-2$.

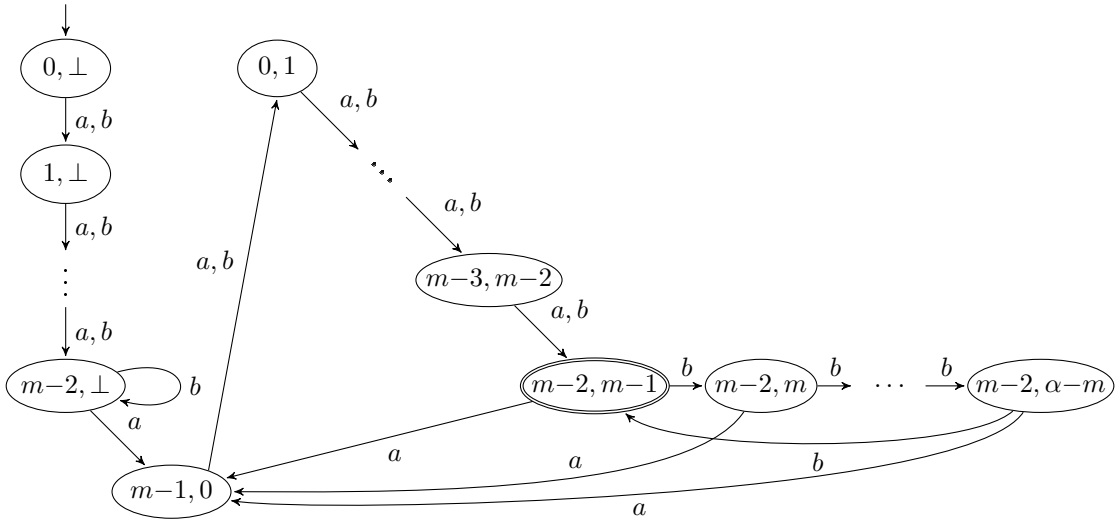


Figure 5.15: The cut automaton for the DFAs in Figure 5.14.

In the cut automaton $A\,!\,B$ we consider the following sets of states:

$$\mathcal{R}_1 = \{\,(i,\bot) \mid 0 \le i \le m-2\,\} \cup \{(m-1,0)\} \cup \{\,(i,i+1) \mid 0 \le i \le m-3\,\},$$
$$\mathcal{R}_2 = \{\,(m-2,j) \mid m-1 \le j \le \alpha-m\,\}.$$

Each state in $\mathcal{R}_1 \cup \{(m-2,m-1)\}$ is reached from $(0,\bot)$ by a string in a^*, and each state in $\mathcal{R}_2$ is reached from $(m-2,m-1)$ by a string in b^*. Figure 5.15 shows that no other state is reachable in the cut automaton.

To prove distinguishability, notice that two distinct states in $\mathcal{R}_1$ are distinguishable by a string in a^* and two distinct states in $\mathcal{R}_2$ are distinguishable by a string in b^*. The states $(i,\bot)$ in $\mathcal{R}_1$ are distinguishable from each state in $\mathcal{R}_2$ by a string in b^*. Every other state in $\mathcal{R}_1$ is distinguishable from each state in $\mathcal{R}_2$ by a string in a^*. Since $|\mathcal{R}_1 \cup \mathcal{R}_2| = \alpha$, our proof is complete. □

Since the state complexity of the cut operation in general is higher than in the case of unary languages, we have to consider also the remaining interval $[m+n-1,(m-1)n+m]$. This is done in the following steps, cf. [90, Subsection 4.2.1]:

1. First we show that some values of α, corresponding to the number of states of the cut automaton in the first r rows and the first s columns, see Figure 5.16, are attainable, namely $\alpha = 1+(r-1)n+(m-r)s$ for some r,s with $2 \le r \le m$ and $1 \le s \le n$.

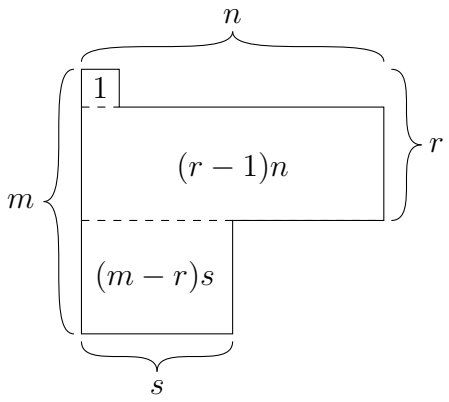


Figure 5.16: A schematic drawing of the reachable part of $A \,!\, B$ for the first task.

2. Then we show that all the remaining values of α up to $(m-1)n+1$ are attainable.

3. Finally, we show that all the values of α in $[(m-1)n+2, (m-1)n+m]$ are attainable.

Let us start with the first task.

Lemma 5.11. *Let $m, n \geq 2$ and let r, s be any integers such that $2 \leq r \leq m$ and $1 \leq s \leq n$. Then there exist a minimal binary m-state* DFA $A_{r,s}$ *and a minimal binary n-state* DFA $B_{r,s}$ *such that the minimal* DFA *for $L(A_{r,s}) \,!\, L(B_{r,s})$ has exactly $1+(r-1)n+(m-r)s$ states.*

Proof. Our aim is to define the binary DFAs $A_{r,s} = ([0, m-1], \{a, b\}, \delta_A, 0, \{0\})$ and $B_{r,s} = ([0, n-1], \{a, b\}, \delta_B, 0, \{n-1\})$ in such a way that in the DFA $A_{r,s} \,!\, B_{r,s}$ the states in the following set would be reachable and pairwise distinguishable:

$$\mathcal{R} = \{(0,0)\} \cup \{\, (i,j) \mid 1 \leq i \leq r-1 \text{ and } 0 \leq j \leq n-1 \,\} \\ \cup \{\, (i,j) \mid r \leq i \leq m-1 \text{ and } 0 \leq j \leq s-1 \}.$$

Moreover, we have to guarantee that no other state of the cut automaton is reachable. Because $|\mathcal{R}| = 1 + (r-1)n + (m-r)s$, the DFAs $A_{r,s}$ and $B_{r,s}$ will be the desired DFAs. To this aim, we define δ_A and δ_B as follows:

$$\delta_A(i,a) = (i+1) \bmod m \quad \text{and} \quad \delta_A(i,b) = \begin{cases} i, & \text{if } i \leq r-1; \\ r-1, & \text{if } i \geq r; \end{cases}$$

and

$$\delta_B(j,b) = (j+1) \bmod n \quad \text{and} \quad \delta_B(j,a) = \begin{cases} j, & \text{if } j \leq s-1; \\ s-1, & \text{if } j \geq s. \end{cases}$$

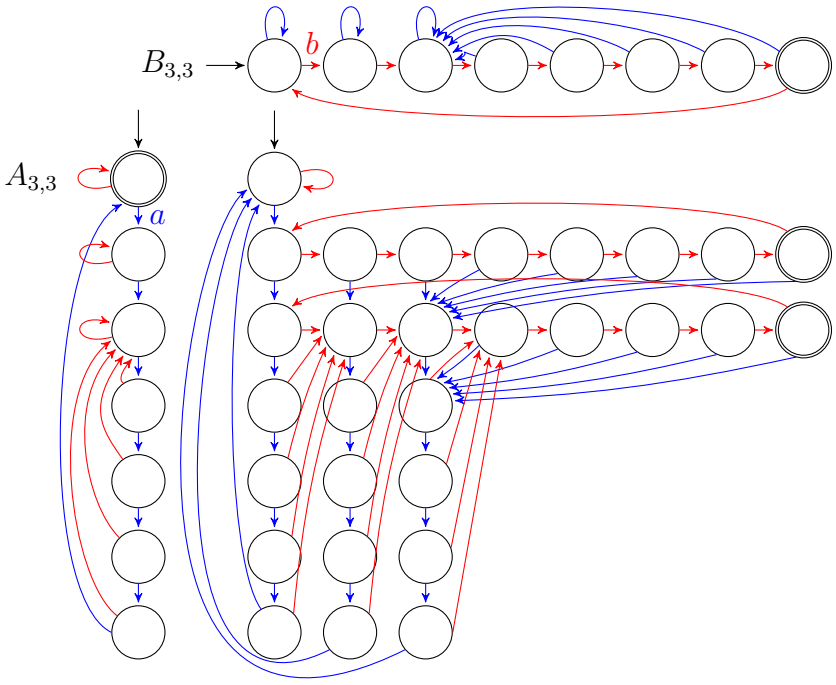


Figure 5.17: The DFAs $A_{3,3}$ (left) and $B_{3,3}$ (top) with $m = 7$ and $n = 8$ and the cut automaton $A_{3,3} ! B_{3,3}$.

Figure 5.17 shows DFAs $A_{3,3}$ (left) and $B_{3,3}$ (top) with $m = 6$ and $n = 5$, and the corresponding cut automaton.

In the cut automaton $A_{r,s} ! B_{r,s}$, the state $(0, 0)$ is the initial state, each state (i, j) in $\mathcal{R}$ with $i \leq r - 1$ is reached from $(0, 0)$ by $a^i b^j$, and each state (i, j) in $\mathcal{R} \setminus \{(0, 0)\}$ with $j \leq s - 1$ is reached from $(0, 0)$ by $ab^j a^{i-1}$. To show that no other state is reachable, notice that each state (i, j) in $\mathcal{R}$ goes on a to a state (i', j') where $j' \leq s - 1$, and it goes on b to a state (i'', j'') where $i'' \leq r - 1$. Since both resulting states are in $\mathcal{R}$, no other state is reachable in the cut automaton.

It remains to prove the distinguishability of states in $\mathcal{R}$. The state $(0, 0)$ and any other state in $\mathcal{R}$ are distinguishable by a string in b^*. Two states in different columns are distinguishable by a string in b^* since exactly one of them can be moved to the column $n - 1$ containing the final states of the cut automaton. Two states in different rows are distinguishable by a string in a^* since exactly one of them can be moved to the state $(0, 0)$. This proves distinguishability and concludes the proof. □

In the above lemma we obtained the values $\alpha_{r,s}$ in $[m + n - 1, (m - 1)n + 1]$ that correspond to the number of states in the first r rows and the first s columns of the

cut automaton. We can add one more row and get the value $\alpha_{r+1,s}$ using an analogous construction for automata $A_{r+1,s}$ and $B_{r+1,s}$. Similarly, we can add one more column. Nevertheless, we still need to get the values between $\alpha_{r,s}$ and $\alpha_{r+1,s}$ or between $\alpha_{r,s}$ and $\alpha_{r,s+1}$, respectively. Since we have $\alpha_{r+1,s} - \alpha_{r,s} = n - s$ and $\alpha_{r,s+1} - \alpha_{r,s} = m - r$, we need to obtain the complexities $\alpha_{r,s} + t$ with $1 \leq t \leq \min\{n - s - 1, m - r - 1\}$. The next lemma produces these complexities.

Lemma 5.12. *Let $m, n \geq 2$ and let r, s be any integers with $2 \leq r \leq m$ and $1 \leq s \leq n$. Moreover, let t satisfy $1 \leq t \leq \min\{n - s - 1, m - r - 1\}$. Then there exist a minimal binary m-state* DFA *$A_{r,s,t}$ and a minimal binary n-state* DFA *$B_{r,s,t}$ such that the minimal* DFA *for the language $L(A_{r,s,t}) \,!\, L(B_{r,s,t})$ has exactly $1 + (r-1)n + (m-r)s + t$ states.*

Proof. Let $\alpha_{r,s} = 1 + (r-1)n + (m-r)s$. Then in the cut automaton $A_{r,s} \,!\, B_{r,s}$ described in the previous proof, exactly $\alpha_{r,s}$ states are reachable and distinguishable. Our aim is to modify both automata in such a way that the resulting cut automaton has t more reachable states. To achieve this goal, we modify DFAs $A_{r,s}$ and $B_{r,s}$ as follows.

In $A_{r,s}$ we replace each transition $(r + i, b, r - 1)$ by $(r + i, b, r + i - 1)$, if $2 \leq i \leq t$ and i is even. Since $i \leq t \leq (m - r) - 1$, we have $r + i \leq m - 1$. In $B_{r,s}$ we replace each transition $(s + i, a, s - 1)$ by the transition $(s + i, a, s + i - 1)$, if $1 \leq i \leq t$ and i is odd. Since $i \leq t \leq (n - s) - 1$, we have $s + i \leq n - 1$. Denote the resulting DFAs by $A_{r,s,t}$ and $B_{r,s,t}$, respectively; see Figure 5.18 for an illustration in case $m = 7$, $n = 8$, $r = 3$, $s = 3$, and $t = 3$.

Consider the cut automaton $A_{r,s,t} \,!\, B_{r,s,t}$. Let $\mathcal{R}$ be the same set as in the previous proof. Then each state (i, j) in $\mathcal{R}$ with $i \leq r - 1$ is reached from $(0, 0)$ by $a^i b^j$, and each state (i, j) in $\mathcal{R} \setminus \{(0, 0)\}$ with $j \leq s - 1$ is reached from $(0, 0)$ by $ab^j a^{i-1}$. Next, if i is odd, then each state $q_i = (r, s + i - 1)$ is reached from $(r - 1, s + i)$ by a, and otherwise, each state $q_i = (r + i - 1, s)$ is reached from $(r + i, s - 1)$ by b.

Now, let us show that no other state is reachable. Notice that each state in $\mathcal{R}$ goes either to a state in $\mathcal{R}$ or to a state in $\{q_1, q_2, \ldots, q_t\}$ on a and b; each state $(r - 1, s + i)$ with i even goes to $(r, s - 1)$ on a, and each state $(r + i, s - 1)$ with $i = 0$ or i odd goes to $(r - 1, s)$ on b. Next, each state q_i with i odd goes to the state $(r + 1, s - 1)$ on a and to a state in row $r - 1$ on b. Finally, each state q_i with i even goes to a state in column $s - 1$ on a and to the state $(r - 1, s + 1)$ on b. Since all the resulting states are in $\mathcal{R} \cup \{q_1, q_2, \ldots, q_t\}$, no other state is reachable in the cut automaton.

The proof of distinguishability is exactly the same as in Lemma 5.11. □

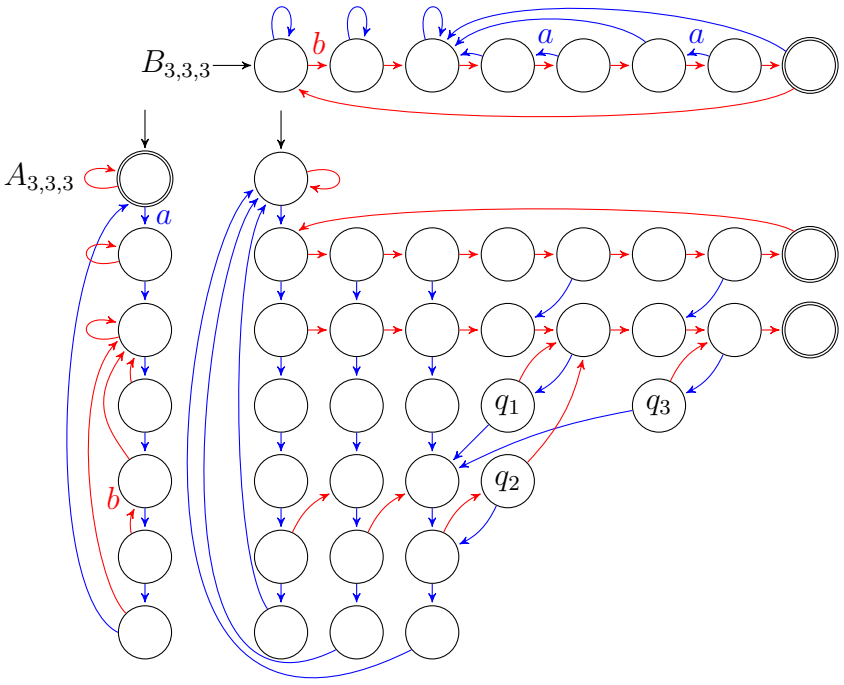


Figure 5.18: The DFAs $A_{3,3,3}$ (left) and $B_{3,3,3}$ (top) with $m = 7$ and $n = 8$ and the cut automaton $A_{3,3,3} ! B_{3,3,3}$ where some transitions are not displayed.

In Lemmas 5.11 and 5.12, we have produced all values in the range from $m + n - 1$ up to $(m - 1)n + 1$. It remains to show that the complexities from $(m - 1)n + 2$ up to $(m - 1)n + m$ are attainable as well.

Lemma 5.13. *Let $m, n \geq 2$ and $(m-1)n+2 \leq \alpha \leq (m-1)n+m$. There exist a minimal binary m-state* DFA *A and a minimal binary n-state* DFA *B such that the minimal* DFA *for $L(A) ! L(B)$ has exactly α states.*

Proof. We have

$$\alpha = (m - 1)n + 1 + \beta$$

for some β with $1 \leq \beta \leq m - 1$. Let A be a minimal m-state DFA over $\{a, b\}$ that accepts the strings in which the number of a's modulo m is β. Let B be a minimal n-state DFA over $\{a, b\}$ that accepts the strings in which the number of b's modulo n is $n - 1$. Figure 5.19 shows the DFAs A and B with $m = 6$, $n = 5$, and $\beta = 3$, and the corresponding cut automaton with some transitions not shown.

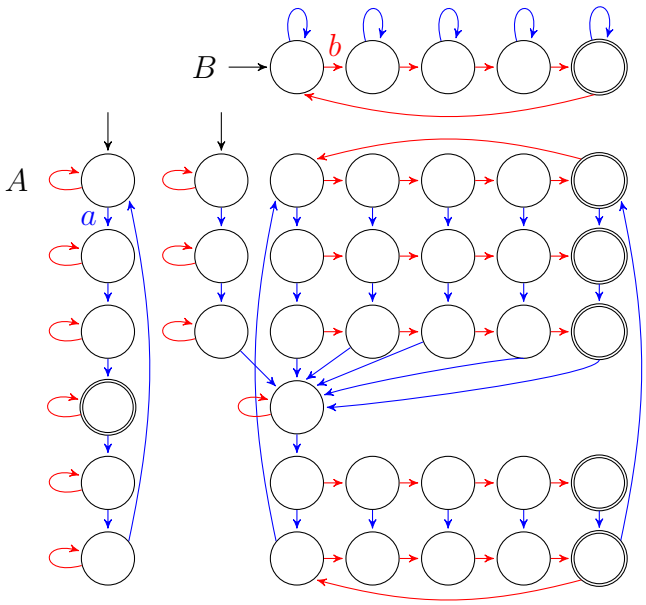


Figure 5.19: The DFAs A and B with $m = 6$, $n = 5$, and $\beta = 3$ and the cut automaton $A\,!\,B$; the transitions $(m-1, j) \xrightarrow{a} (0, j)$ with $1 \leq j \leq n-2$ and $(i, n-1) \xrightarrow{b} (i, 0)$ with $1 \leq i \leq m-2$ are not shown.

Consider the cut automaton $A\,!\,B$. Denote

$$\mathcal{R}_1 = \{\, (i, \perp) \mid i \in [0, \beta - 1] \,\} \cup \{(\beta, 0)\},$$

and

$$\mathcal{R}_2 = \{\, (i, j) \mid i \in [0, \beta - 1] \cup [\beta + 1, m - 1] \text{ and } j \in [0, n - 1] \,\}.$$

Notice that each state $(i, \perp)$ in $\mathcal{R}_1$ is reachable from the initial state $(0, \perp)$ by a^i, and each state $(i, 0)$ is reachable by a^{m+i}. Each state (i, j) in $\mathcal{R}_2$ is reached from $(0, 0)$ by $a^i b^j$. Since the state β is a final state in A, it follows from the construction of the cut automaton that no state $(i, \perp)$ with $i \geq \beta$ and no state (β, j) with $j \geq 1$ is reachable.

To prove distinguishability, let p and q be two different states in $\mathcal{R}_1 \cup \mathcal{R}_2$. If $p \in \mathcal{R}_1$ and $q \in \mathcal{R}_2$, then p is a non-final state with a loop on b, while a string in b^* is accepted from q. If both states p and q are in $\mathcal{R}_1$, then a string in a^* leads one of them to the state $((\beta + 1) \bmod m, 0)$ in $\mathcal{R}_2$, while it leads the second one to a state in $\mathcal{R}_1$, and the resulting states are distinguishable as shown above. Finally, let p and q be two states in $\mathcal{R}_2$. If they are in different columns, then a string in b^* distinguishes them. If they are in different rows, then a string in a^* leads one of them to the state $(\beta, 0)$ in $\mathcal{R}_1$, and it leads the second one to a state in $\mathcal{R}_2$. □

The next theorem summarizes the results of this section; recall that by (1) on page 35, the state complexity of the cut operation is given by the function

$$f(m,n) = \begin{cases} m, & \text{if } n = 1; \\ (m-1)n + m, & \text{if } n \geq 2; \end{cases}$$

Theorem 5.14 (General Case). *Let $m, n \geq 1$ and $f(m,n)$ be the state complexity of the cut operation. For each α such that $1 \leq \alpha \leq f(m,n)$, there exist a minimal binary m-state* DFA *A and a minimal binary n-state* DFA *B such that the minimal* DFA *for $L(A)\,!\,L(B)$ has α states.* □

Observe that this theorem solves the magic number problem for the cut operation for every alphabets of size at least two by duplicating input symbols.

5.3 Conclusions

We examined the state complexity of languages resulting from the cut operation on minimal DFAs with m and n states. We showed that the range of state complexities of languages resulting from the cut operation is contiguous from one up to the known upper bound for every alphabet of size at least two.

Our results in the unary case are different. We proved that no value from $2m$ up to $n-1$ is attainable by the state complexity of the cut of two unary languages represented by minimal deterministic finite automata with m and n states. All the remaining values up to the known upper bound are attainable.

This means that the problem of finding all attainable state complexities for the cut operation is completely solved for every size of alphabet. To the best of our knowledge, the cut operation is the first operation where this is the case.

Chapter 6

Ranges of Accepting State Complexities

The minimal number of accepting states in any DFA for a language can be considered as a complexity measure. This complexity measure was introduced by Dassow in 2016 [25]. Formally, the *accepting state complexity* of a language L, $\mathrm{asc}(L)$, is defined by

$$\mathrm{asc}(L) = \min\{n \mid L \text{ is accepted by a DFA with } n \text{ accepting states}\}.$$

The major contribution of [25] is the investigation of the accepting state complexity of languages resulting from the operations of complementation, union, concatenation, difference, and Kleene star. This can be seen as a variant of the magic number problem, in which we count the number of accepting states instead of the number of all states.

In this chapter, we continue the research concerning the accepting state complexity. We provide an answer to the open problem from [25] concerning the nondeterministic accepting state complexity. We also show that the range of accepting state complexities of languages resulting from the intersection operation is contiguous from 0 to mn, which solves the second open problem from [25]. Then we consider the operations of symmetric difference, right and left quotients, reversal, and permutation on binary finite languages. All of these operations turn out to have an infinite range of accepting state complexities with no magic numbers. The results of this chapter, except for symmetric difference on unary finite languages and the cut operation, are based on the published paper

> Michal Hospodár, Markus Holzer: The ranges of accepting state complexities of languages resulting from some operations. In: Câmpeanu, Cezar (ed.) *Proc. 23rd International Conference on Implementation and Application of Automata, CIAA 2018, Charlottetown, PE, Canada, July 30 – August 2, 2018.* Lecture Notes in Computer Science, vol. 10977, pp. 198–210. Springer (2018). ISBN: 978-3-319-94811-9.

The following result from [25] shows that the accepting state complexity of a language L is equal to the number of accepting states in the minimal DFA for L.

Theorem 6.1 ([25, Theorem 1])**.** *Let L be a language accepted by a minimal* DFA *A. Then the number of accepting states of A is equal to* $\mathrm{asc}(L)$. □

Since omitting a dead state does not change the language and does not change the number of accepting states, we use minimal partial DFAs to describe languages. Moreover, we only need to prove the distinguishability of accepting states.

Recall that for $\mathrm{c} \in \{\mathrm{sc}, \mathrm{asc}\}$, a k-ary regular operation $\circ$ on languages, and non-negative integers $n_1, n_2, \ldots, n_k$, we have

$$g_\circ^{\mathrm{c}}(n_1, n_2, \ldots, n_k) = \{\mathrm{c}(\circ(L_1, L_2, \ldots, L_k)) \mid \mathrm{c}(L_i) = n_i \text{ for } i = 1, 2, \ldots, k\},$$

that is, $g_\circ^{\mathrm{c}}(n_1, n_2, \ldots, n_k)$ is the set of all integers α such that there exist k regular languages $L_1, L_2, \ldots, L_k$ with $\mathrm{c}(L_i) = n_i$, for $i = 1, 2, \ldots, k$, and $\mathrm{c}(\circ(L_1, L_2, \ldots, L_k)) = \alpha$. In case we only consider unary (finite, respectively) languages $L_1, L_2, \ldots, L_k$ we write $g_\circ^{\mathrm{c},u}$ ($g_\circ^{\mathrm{c},f}$, respectively) instead. In order to explain the notation $g_\circ^{\mathrm{asc}}(n_1, n_2, \ldots, n_k)$, we give a small example.

Example 6.2. *Complementation is a unary operation denoted $L^{\complement} := \Sigma^* \setminus L$. We have*

$$g_{\complement}^{\mathrm{sc}}(n) = \{n\}, \quad \text{if } n \geq 1.$$

On the other hand, in [25, Theorem 4] the following behavior was proven:

$$g_{\complement}^{\mathrm{asc}}(n) = \begin{cases} \{1\}, & \text{if } n = 0; \\ \{0\} \cup \mathbb{N}, & \text{if } n = 1; \\ \mathbb{N}, & \text{if } n \geq 2. \end{cases}$$

Moreover, all values in these ranges can be attained by unary languages, so we have

$$g_{\complement}^{\mathrm{sc},u}(n) = g_{\complement}^{\mathrm{sc}}(n) \quad \text{and} \quad g_{\complement}^{\mathrm{asc},u}(n) = g_{\complement}^{\mathrm{asc}}(n).$$ □

The *nondeterministic accepting state complexity* of a language L, denoted by $\mathrm{nasc}(L)$, refers to the minimal number of accepting states in any NFA for the language L. It was shown in [25, Theorem 3] that for every non-empty regular language L, we have $\mathrm{nasc}(L) = 1$ if $\varepsilon \notin L$, but $\mathrm{nasc}(L) \leq 2$ if $\varepsilon \in L$. Thus, the nondeterministic accepting state complexity is not very interesting. Nevertheless, it was left open to give a sufficient and necessary condition for a language L such that $\mathrm{nasc}(L) = 1$ and $\varepsilon \in L$. The next proposition solves this problem.

Proposition 6.3. *A language L satisfies $\varepsilon \in L$ and $\operatorname{nasc}(L) = 1$ if and only if $L = L^*$.*

Proof. If $\varepsilon \in L$ and $\operatorname{nasc}(L) = 1$, then there exists an NFA for L in which the single accepting state is the initial state. Therefore $L = L^*$.

Conversely, let $A = (Q, \Sigma, \delta, s, F)$ be an NFA accepting the language L. If $L = L^*$, then $\varepsilon \in L$, so the initial state s of A is accepting. For every accepting state q_f in $F \setminus \{s\}$ and every transition (q, a, q_f), we add the transition (q, a, s) to A and make the state q_f rejecting. Since $L = L^*$, the resulting automaton, which has exactly one accepting state, accepts L. It follows that $\operatorname{nasc}(L) = 1$. □

6.1 Intersection and Symmetric Difference

In this section, we examine the accepting state complexity of languages resulting from the intersection operation, solving an open problem stated in [25].

Recall that for two DFAs A and B, the intersection $L(A) \cap L(B)$ is recognized by the *product automaton* $A \times B$. If A and B have m and n states, respectively, then $A \times B$ has mn states.

In [93] it was shown that this upper bound is necessary in the worst case, that is, it can be reached by two appropriately chosen DFAs with m and n states. Moreover, in [51] it was shown that for every α with $1 \leq \alpha \leq mn$ there exist minimal binary m-state and n-state DFAs such that the intersection of the languages described by these automata is accepted by a minimal DFA with exactly α states. Thus $g_{\cap}^{\mathrm{sc}}(m, n) = [1, mn]$.

Now consider the accepting state complexity of languages resulting from the intersection operation. The next theorem solves an open problem stated in [25] by showing that for every α with $0 \leq \alpha \leq mn$, there exist minimal binary DFAs A and B with m and n accepting states, respectively, such that the accepting state complexity of $L(A) \cap L(B)$ is exactly α.

Theorem 6.4. *We have $g_{\cap}^{\mathrm{asc}}(m, n) = [0, mn]$. All values in this range are attained by binary languages.*

Proof. Let $0 \leq \alpha \leq mn$. We describe minimal partial DFAs A and B with m and n accepting states, respectively, such that $\operatorname{asc}(L(A) \cap L(B)) = \alpha$. Notice that α can be expressed as $\alpha = kn + \ell$, for some integers k and ℓ with $0 \leq k \leq m$ and $0 \leq \ell \leq n - 1$.

Define the partial DFA $A = ([1, m+1], \{a, b\}, \delta_A, m+1, [1, m])$, where

$$\delta_A(i, a) = i - 1, \text{ if } 2 \leq i \leq m+1; \qquad \delta_A(i, b) = \begin{cases} i, & \text{if } 1 \leq i \leq m; \\ k+1, & \text{if } i = m+1. \end{cases}$$

Next, define the partial DFA $B = ([0, n+1], \{a, b\}, \delta_B, n+1, [1, n])$, where

$$\delta_B(j, a) = n, \text{ if } j = 0; \qquad \delta_B(j, b) = \begin{cases} j-1, & \text{if } 1 \le j \le n; \\ \ell, & \text{if } j = n+1. \end{cases}$$

The partial DFAs A and B are shown in Figure 6.1. Both DFAs are minimal.

The product automaton $A \times B$ has the initial state $(m+1, n+1)$ and the following transitions shown in Figure 6.2:

$\to (m+1, n+1) \xrightarrow{b} (k+1, \ell) \xrightarrow{b} (k+1, \ell-1) \xrightarrow{b} \ldots \xrightarrow{b} (k+1, 1) \xrightarrow{b} (k+1, 0)$,

$(k+1, 0) \xrightarrow{a} (k, n) \xrightarrow{b} (k, n-1) \xrightarrow{b} \ldots \xrightarrow{b} (k, 1) \xrightarrow{b} (k, 0)$,

$(k, 0) \xrightarrow{a} (k-1, n) \xrightarrow{b} (k-1, n-1) \xrightarrow{b} \cdots \xrightarrow{b} (k-1, 1) \xrightarrow{b} (k-1, 0)$, etc., and

$(2, 0) \xrightarrow{a} (1, n) \xrightarrow{b} (1, n-1) \xrightarrow{b} \ldots \xrightarrow{b} (1, 1) \xrightarrow{b} (1, 0)$.

No other transitions are used in $A \times B$. It follows that $L(A \times B)$ is a finite language with the longest string $b^{\ell+1}(ab^n)^{k-1}ab^{n-1}$. Hence every NFA accepting the language $L(A \times B)$ has at least $k(n+1)+\ell+1$ states. Thus $A \times B$ with the state $(1, 0)$ omitted is a minimal NFA for $L(A) \cap L(B)$. Next, since $A \times B$ is deterministic, it is a minimal partial DFA, so every pair of states of $A \times B$ is distinguishable. Note that the states (i, j) with $1 \le i \le k$ and $1 \le j \le n$, and $(k+1, j)$ with $1 \le j \le \ell$, are reachable and accepting in $A \times B$. It follows that we have $kn + \ell$ reachable and pairwise distinguishable accepting states. Thus $\text{asc}(L(A) \cap L(B)) = kn + \ell = \alpha$, and the theorem follows. □

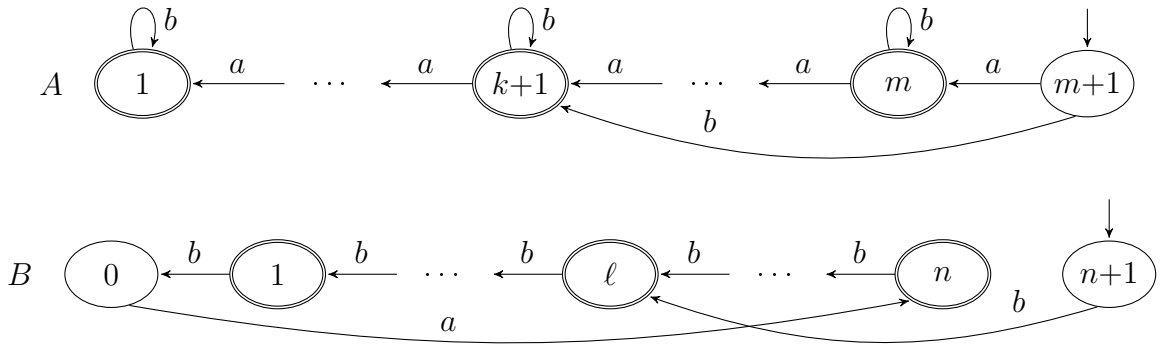


Figure 6.1: The binary witness DFAs A (top) and B (bottom) for intersection meeting the complexity $\alpha = kn + \ell$ with $0 \le k \le m$ and $0 \le \ell \le n$.

Notice that witnesses in the previous proof were defined over a binary alphabet. The range of accepting state complexities of languages resulting from the intersection operation in the unary case remains open.

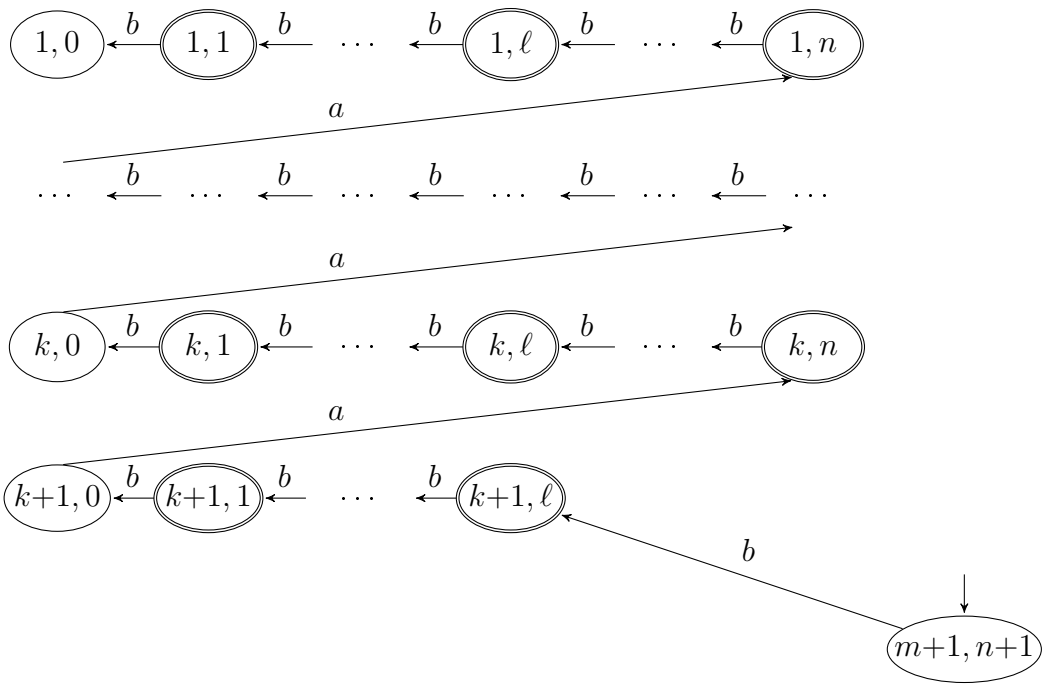


Figure 6.2: The reachable part of the product automaton $A \times B$ of DFAs A and B from Figure 6.1 with $kn + \ell$ accepting states.

The symmetric difference ($\oplus$) of two languages accepted by DFAs A and B can also be obtained by a product automaton $A \times B$, in which the set of accepting states consists of those pairs in which exactly one component is accepting in the corresponding DFA. Thus, for the ordinary state complexity, the upper bound is mn which was shown to be tight by Brzozowski [9]. To the best of our knowledge, the magic number problem for state complexity of the symmetric difference operation was not investigated so far. In our investigation of accepting state complexity of unary languages, we use the following fact.

Fact 6.5. *Let L be a unary finite language. Then* $\mathrm{asc}(L) = |L|$. □

The next lemma shows that every positive integer can be obtained as the accepting state complexity of symmetric difference of non-empty unary languages.

Lemma 6.6. *Let $m, n, \alpha \geq 1$. There exist minimal unary* DFA*s A and B with m and n accepting states, respectively, such that the minimal* DFA *for $L(A) \oplus L(B)$ has α accepting states.*

Proof. Since symmetric difference is commutative, we may assume that $m \leq n$. Let A and B be the minimal DFAs with m and n accepting states, respectively, shown in Fig-

ure 6.3. Then

$$\begin{aligned} L(A) &= \{a^i \mid 0 \le i \le m-2 \text{ or } i \ge m\}, \\ L(B) &= \{a^i \mid 0 \le i \le m-2 \text{ or } m \le i \le n-1 \text{ or } i \ge n+\alpha\}, \text{and} \\ L(A) \oplus L(B) &= \{\, a^i \mid n \le i \le n+\alpha-1 \,\}. \end{aligned}$$

The symmetric difference $L(A) \oplus L(B)$ is accepted by a minimal partial DFA C with α accepting states shown in Figure 6.3. □

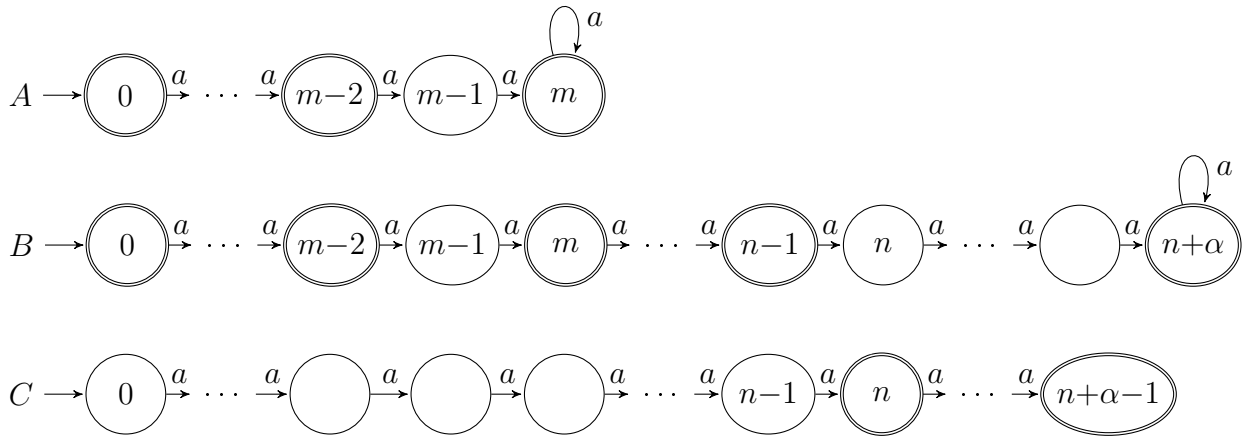


Figure 6.3: The witness DFAs A and B and the partial DFA C for $L(A) \oplus L(B)$.

Now we are ready to describe the behavior of the accepting state complexity measure with respect to the symmetric difference operation.

Theorem 6.7. *We have*

$$g_{\oplus}^{\mathrm{asc},u}(m,n) = g_{\oplus}^{\mathrm{asc}}(m,n) = \begin{cases} \{n\}, & \text{if } m = 0; \\ \{m\}, & \text{if } n = 0; \\ \{0\} \cup \mathbb{N}, & \text{if } m,n \ge 1 \text{ and } m = n; \\ \mathbb{N}, & \text{if } m,n \ge 1 \text{ and } m \ne n. \end{cases}$$

Proof. The only language with accepting state complexity 0 is the empty language $\emptyset$. Since $\emptyset \oplus L = L$ and $K \oplus \emptyset = K$, the first two cases follow. If $m, n \ge 1$ and $m = n$, then we can set $K = L$ to get 0, or use Lemma 6.6 to get an arbitrary positive integer. If $m \ne n$, then $K \ne L$ and $K \oplus L \ne \emptyset$, so $\mathrm{asc}(K \oplus L) \ne 0$. Finally, $g_{\oplus}^{\mathrm{asc}}(m,n) = g_{\oplus}^{\mathrm{asc},u}(m,n)$ since all our witnesses are unary languages. □

In the proof of Lemma 6.6, the provided witness languages were both infinite languages producing a finite language as their symmetric difference. Because for symmetric difference we have $K \oplus L = M$ if and only if $K \oplus M = L$, we can rephrase the result of Lemma 6.6 by exchanging the roles of one witness language and the resulting language. This results in the next corollary.

Corollary 6.8. *Let $m, n, \alpha \geq 1$. There exist minimal unary* DFA*s A and B with m and n accepting states, respectively, such that $L(A)$ is finite, $L(B)$ is infinite, and the minimal* DFA *for $L(A) \oplus L(B)$ has α accepting states.* □

However, if both involved language are finite, we find the following result:

Theorem 6.9. *Let $m \geq n$. Then $g_{\oplus}^{\mathrm{asc},u,f}(m,n) = \{m-n, m-n+2, \ldots, m+n\}$.*

Proof. Let K and L be unary finite languages with $\mathrm{asc}(K) = m$ and $\mathrm{asc}(L) = n$. By Fact 6.5, we have $|K| = m$ and $|L| = n$. It follows that $|K \oplus L| = m + n - 2|K \cap L|$. Since $\mathrm{asc}(K \oplus L) = |K \oplus L|$, we have that $m+n-\mathrm{asc}(K \oplus L) = 2|K \cap L|$, which must be even and between 0 and $2n$. The value of $|K \cap L|$ can be every integer between 0 and n. It follows that all and only even numbers between 0 and $2n$ can be the value of $2|K \cap L|$, and the theorem follows. □

In the previous theorem, we have found some magic numbers for accepting state complexity with respect to symmetric difference in the unary finite case. The case if both languages are finite sets over a general alphabet is left open. We conjecture that $g_{\oplus}^{\mathrm{asc},f}(m,n) = \mathbb{N}$, hence no number is magic.

6.2 Right and Left Quotients

The *right quotient* of a language K by a language L is defined as follows:

$$KL^{-1} = \{\, w \mid \text{there exists a string } x \in L \text{ such that } wx \in K \,\}.$$

The DFA accepting KL^{-1} is the same as the DFA accepting K except that the set of accepting states is different. To be more precise, let $A = (Q, \Sigma, \delta, s, F)$ be the DFA accepting K, then $B = (Q, \Sigma, \delta, s, \{\, q \mid \exists\, x \in L : \delta(q,x) \in F \,\})$ accepts the language KL^{-1}. Thus, for an m-state DFA, the upper bound for the state complexity of the right quotient with respect to any language is m, which is known to be tight by Yu et al. [94].

Similarly one defines the *left quotient* of K by L as

$$L^{-1}K = \{\, w \mid \text{there exists a string } x \in L \text{ such that } xw \in K \,\}.$$

It was proven that for an m-state DFA language K, the state complexity of the left quotient of K by any language L is at most $2^m - 1$. Again, this bound is tight [94].

When considering unary languages K and L, the right and left quotient coincide, that is, $KL^{-1} = L^{-1}K$. Thus, in the unary case, the state complexity of left quotient is bounded by the state complexity of K. To the best of our knowledge, the magic number problem for state complexity of the quotient operations was not investigated so far. In this section, we consider the magic number problem for accepting state complexity with respect to the quotient operations.

Lemma 6.10. *Let $m, n \geq 1$ and $\alpha \geq 0$. There exist minimal unary* DFA*s A and B with m and n accepting states, respectively, such that the minimal* DFA *for $L(A)L(B)^{-1}$ has α accepting states.*

Proof. We consider two cases:

(1) Let $0 \leq \alpha \leq n-1$. Define the languages $K = \{\, a^i \mid 0 \leq i \leq m-2 \text{ or } i = m+\alpha \,\}$ and $L = \{\, a^i \mid m+1 \leq i \leq m+n \,\}$. By Fact 6.5, the language K (L, respectively) is accepted by a minimal DFA with m (n, respectively) accepting states. Next, we have $KL^{-1} = \{\, a^i \mid 0 \leq i \leq \alpha - 1 \,\}$, whose minimal DFA has α accepting states. Observe that this case also covers $\alpha = 0$ where KL^{-1} becomes empty.

(2) Now let $\alpha \geq n$. Let K be the same language as in the first case and define the unary language $L = \{\, a^i \mid m \leq i \leq m+n-2 \text{ or } i \geq m+n \,\}$. By Fact 6.5, the language K (L, respectively) is accepted by a minimal DFA with m (n, respectively) accepting states. Next $KL^{-1} = \{\, a^i \mid 0 \leq i \leq \alpha - n \text{ or } \alpha - n + 2 \leq i \leq \alpha \,\}$, whose minimal DFA has α accepting states. □

In the next theorem, we denote the right quotient by rq and the left quotient by lq.

Theorem 6.11. *We have*

$$g_{rq}^{\mathrm{asc}}(m,n) = g_{lq}^{\mathrm{asc}}(m,n) = \begin{cases} \{0\}, & \text{if } m = 0 \text{ or } n = 0; \\ \{0\} \cup \mathbb{N}, & \text{if } m, n \geq 1. \end{cases}$$

All values in these ranges are attained by unary finite languages.

Proof. If $m = 0$ or $n = 0$, then at least one of the involved languages for the right quotient operation is the empty set. Because $\emptyset L^{-1} = K\emptyset^{-1} = \emptyset$ for every languages K and L, we obtain our first result. The remaining case follows from Lemma 6.10; notice that these constructions can work with an arbitrary alphabet since we can duplicate the letters and we have already obtained all possible values. Moreover, because the left quotient of unary languages coincides with the right quotient, the theorem follows. □

6.3 Reversal

To get an NNFA accepting the reverse of a language L accepted by a DFA $A = (Q, \Sigma, \delta, s, F)$, it is enough to reverse all transitions and swap the role of the initial and accepting states. This results in an NNFA A^R that accepts the language L^R.

Before we consider the accepting state complexity with respect to the reversal operation, we take a closer look on the subset automaton $\mathcal{D}(A^R)$. Observe that the state s is the single accepting state of the NNFA A^R. Therefore the accepting subsets of the corresponding subset automaton $\mathcal{D}(A^R)$ are those containing the initial state s of A. Moreover, if A is a DFA without unreachable states, then the subset automaton $\mathcal{D}(A^R)$ does not have equivalent states [67, Proposition 3]. Now we are ready to consider accepting state complexity with respect to reversal in general.

Lemma 6.12. *Let $n, \alpha \geq 1$. There exists a minimal binary* DFA *A with n accepting states such that the minimal* DFA *for $L(A^R)$ has α accepting states.*

Proof. Let $A = ([1, \alpha + n], \{a, b\}, \delta, 1, F)$ where $F = [\alpha + 1, \alpha + n]$ and

$$\delta(i, a) = \begin{cases} i, & \text{if } i = 1 \text{ or } i = \alpha + 1; \\ i - 1, & \text{otherwise,} \end{cases}$$

$$\delta(i, b) = \begin{cases} \alpha + n, & \text{if } i = 1; \\ \alpha, & \text{if } i = \alpha + 1. \end{cases}$$

The DFA A is shown in Figure 6.4. Two rejecting states are distinguished by a string in a^*b and two accepting states by a string in $a^*ba^{\alpha-1}b$. Hence A is minimal.

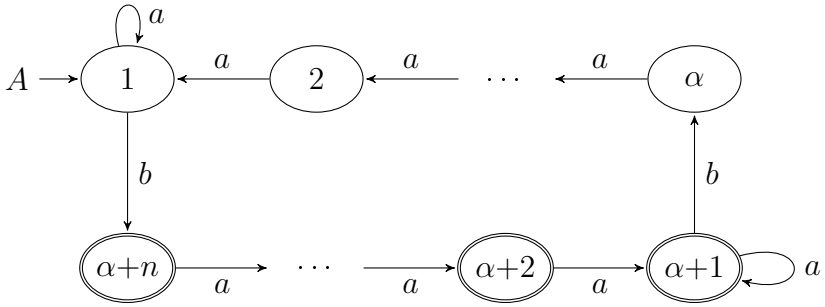


Figure 6.4: The witness DFA A for the reversal operation with $n, \alpha \geq 1$.

We construct the NNFA $A^R = ([1, \alpha + n], \{a, b\}, \delta^R, F, \{1\})$ from the DFA A by reversing all the transitions, and by swapping the roles of the initial and accepting states.

The subset automaton $\mathcal{D}(A^R)$ has the initial state F and the following transitions shown in Figure 6.5:

$$\to F \xrightarrow{a} F \xrightarrow{b} \{1\} \xrightarrow{a} [1,2] \xrightarrow{a} [1,3] \xrightarrow{a} \cdots \xrightarrow{a} [1,\alpha] \xrightarrow{a} [1,\alpha] \xrightarrow{b} \{\alpha+1\}$$

and

$$\{\alpha+1\} \xrightarrow{a} [\alpha+1,\alpha+2] \xrightarrow{a} [\alpha+1,\alpha+3] \xrightarrow{a} \cdots \xrightarrow{a} [\alpha+1,\alpha+n-1] \xrightarrow{a} F.$$

Since every other transition from these reachable states goes to the empty set, no more states are reachable. Since only the subsets containing 1 are accepting, there are exactly α reachable accepting subsets. By [67, Proposition 3], the subset automaton $\mathcal{D}(A^R)$ does not have equivalent states, and the theorem follows. □

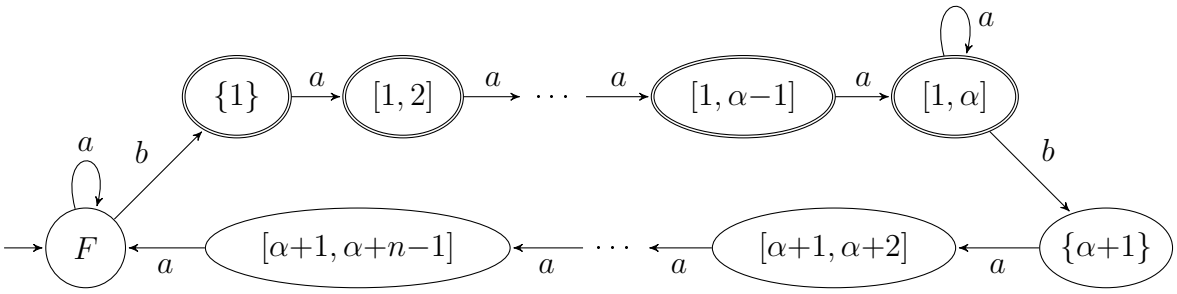


Figure 6.5: The reachable part of DFA $\mathcal{D}(A^R)$. Notice that $[\alpha+1,\alpha+n-1] = F\setminus\{\alpha+n\}$.

Taking into account that the only language with accepting state complexity 0 is the empty language, and for non-empty languages Lemma 6.12 applies, we get the following range of accepting state complexities of languages resulting from reversal. Moreover, since the reverse of a unary language is the same language, we immediately get the result for reversal of unary regular languages, too.

Theorem 6.13. *We have*

$$g_R^{\mathrm{asc}}(n) = \begin{cases} \{0\}, & \text{if } n = 0; \\ \mathbb{N}, & \text{if } n \geq 1. \end{cases}$$

All values are attained by binary languages. Next, we have $g_R^{\mathrm{asc},u}(n) = \{n\}$ *if* $n \geq 0$. □

6.4 Permutation on Finite Languages

Let $\Sigma = \{a_1, a_2, \ldots, a_{|\Sigma|}\}$ and w be a string over the alphabet Σ. The *Parikh vector* of w is defined as $\psi(w) = (|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_{|\Sigma|}})$. The *permutation* of a language L is defined as $\mathrm{per}(L) = \bigcup_{w \in L} \{u \in \Sigma^* \mid \psi(u) = \psi(w)\}$. It is known that the permutation operation does not preserve regularity on infinite languages. For example, the language $\mathrm{per}(\{ab\}^*) = \{\, w \in \{a,b\}^* \mid |w|_a = |w|_b \,\}$ is *not* regular. On the other hand, permutation of a finite language is always finite, and every finite language is regular, so permutation is a regularity preserving operation on finite languages.

Since every unary language is equal to its permutation, we may consider the descriptional complexity with respect to permutation on *binary* finite languages. The ordinary state complexity of permutation on binary finite languages was considered by Cho et al. in [22] where an upper bound of $\frac{n^2-n+2}{2}$ was shown. To the best of our knowledge, the magic number problem for state complexity of permutation on (binary) finite languages was not considered so far. For the accepting state complexity we prove the following three lemmas.

Lemma 6.14. *Let $1 \leq n \leq \alpha$. There exists a minimal binary* DFA *A with n accepting states such that the minimal* DFA *for* $\mathrm{per}(L(A))$ *has α accepting states.*

Proof. Consider the partial DFA $A = ([q_0, q_{\alpha-n}] \cup [1, n], \{a, b\}, \delta, q_0, [1, n])$, where

$\delta(q_i, a) = 1$, if $0 \leq i \leq \alpha - n$,

$\delta(q_i, b) = q_{i+1}$, if $0 \leq i \leq \alpha - n - 1$, and

$\delta(i, b) = i + 1$, if $1 \leq i \leq n - 1$.

The minimal partial DFA A is shown in Figure 6.6 (left). Every string w in $L(A)$ has $|w|_a = 1$ and $0 \leq |w|_b \leq \alpha - 1$. Hence $\mathrm{per}(L(A))$ is accepted by the minimal partial DFA B shown in Figure 6.6 (right). Since B has α accepting states, the theorem follows. □

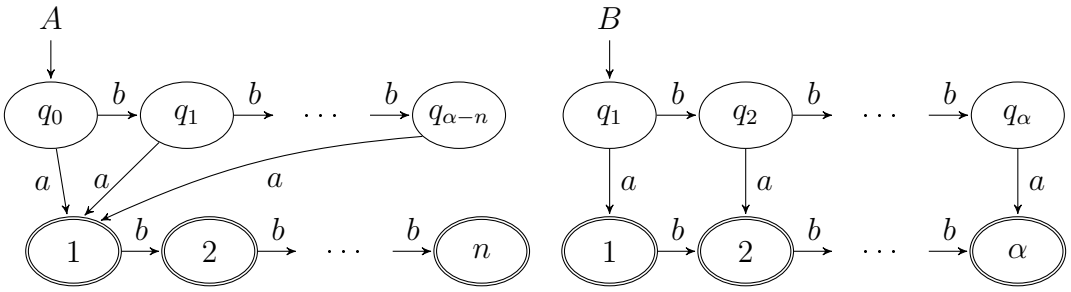


Figure 6.6: The witness partial DFA A for permutation in case $1 \leq n \leq \alpha$ (left) and the minimal partial DFA B for $\mathrm{per}(L(A))$ (right).

Lemma 6.15. *Let L be a finite language accepted by a minimal* DFA *with at least two accepting states. Then every* DFA *for* $\mathrm{per}(L)$ *has at least two accepting states.*

Proof. Assume to the contrary that $\mathrm{per}(L)$ is accepted by a DFA with one accepting state. It follows by [26, Lemma 1] that $\mathrm{per}(L)$ is prefix-free. Hence L is prefix-free, which is a contradiction with L having accepting state complexity more than one. □

Lemma 6.16. *Let $2 \leq \alpha \leq n-1$. There exists a minimal binary* DFA *A with n accepting states such that the minimal* DFA *for* $\mathrm{per}(L(A))$ *has α accepting states.*

Proof. Since $2 \leq \alpha \leq n-1$, we can choose m with $m \geq 1$ such that

$$m + 1 \leq n - (\alpha - 1) \leq 2^m.$$

We first describe a minimal binary DFA A with $2^m + (\alpha - 1)$ accepting states such that the minimal DFA for $\mathrm{per}(L(A))$ has α accepting states, and then we modify the construction to get n accepting states in the witness language.

The idea for the construction is as follows: for a string w in $\{a,b\}^m$, let x_w refer to the string $b^{|w|_a}a^{|w|_b}$ of length m. Then define the finite language

$$L = \{\, wx_w \in \{a,b\}^* \mid |w| = m \,\} \cup \{wx_w w^R y \in \{a,b\}^* \mid |w| = m \text{ and } 0 \leq |y| \leq \alpha - 2\}.$$

The DFA for L is illustrated in Figure 6.7; note that only the states in level $2m$ and in levels from $3m$ up to $3m + \alpha - 2$ are accepting. Figure 6.8 shows the DFA A in the case of $n = 10$, $m = 3$, and $\alpha = 3$.

We show that L has accepting state complexity $2^m + (\alpha - 1)$. First, consider two strings of the form ux_u and vx_v with $|u| = |v| = m$ and $u \neq v$. Note that both ux_u and vx_v are in L. Then the Myhill-Nerode equivalence classes $[ux_u]$ and $[vx_v]$ are different because $ux_u u^R \in L$, but $vx_v u^R \notin L$. Hence the strings of length $2m$ induce 2^m different equivalence classes. Next, consider two strings $wx_w w^R y$ and $wx_w w^R z$ in L for $w \in \{a,b\}^m$ and $y, z \in \{a,b\}^*$ with $0 \leq |y|, |z| \leq \alpha - 2$. Without loss of generality, we may assume that $|y| < |z|$ (in case $|y| = |z|$ both strings $wx_w w^R y$ and $wx_w w^R z$ would belong to the same equivalence class). Then $wx_w w^R y a^{\alpha-2-|y|} \in L$, but $wx_w w^R z a^{\alpha-2-|y|} \notin L$ because the string is too long. This gives additional $\alpha - 1$ equivalence classes. It remains to show that all the 2^m equivalence classes are different to the $\alpha - 1$ equivalence classes. Thus, consider two strings wx_w and $wx_w w^R y$ in L, for some $w \in \{a,b\}^m$ and $y \in \{a,b\}^*$ with $0 \leq |y| \leq \alpha - 2$. Then $wx_w w^R a^{\alpha-2} \in L$ and $wx_w w^R y w^R a^{\alpha-2} \notin L$. Hence, the language L has the desired accepting state complexity.

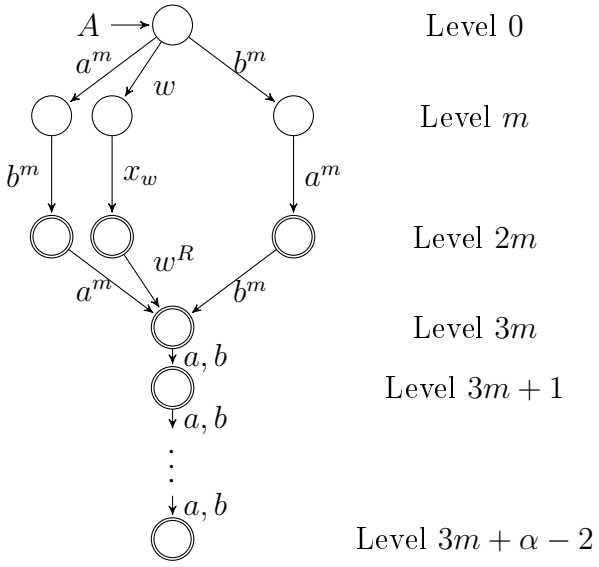


Figure 6.7: A schematic drawing of a DFA A with $2^m + (\alpha - 1)$ accepting states; only the states in level $2m$ and in levels from $3m$ up to $3m + \alpha - 2$ are accepting.

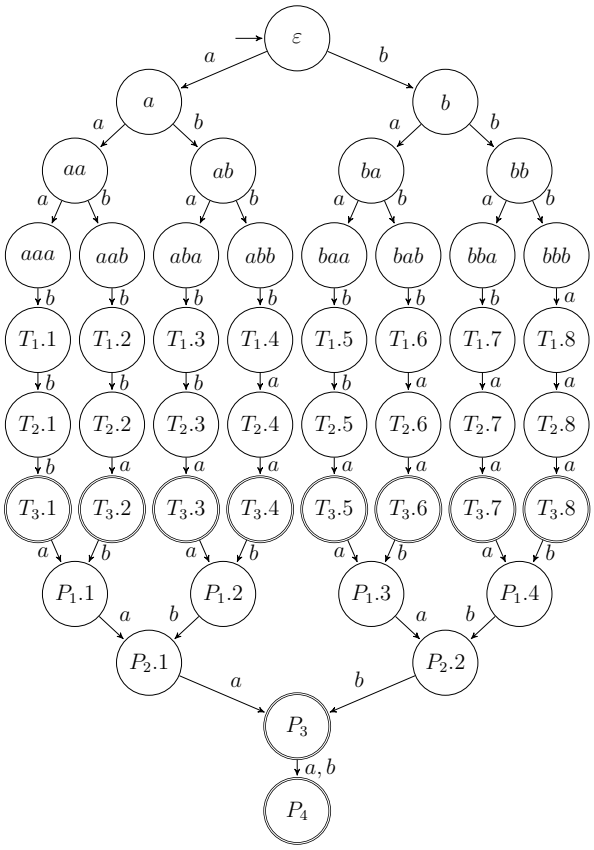


Figure 6.8: An example of the DFA A with $n = 10$, $m = 3$, and $\alpha = 3$.

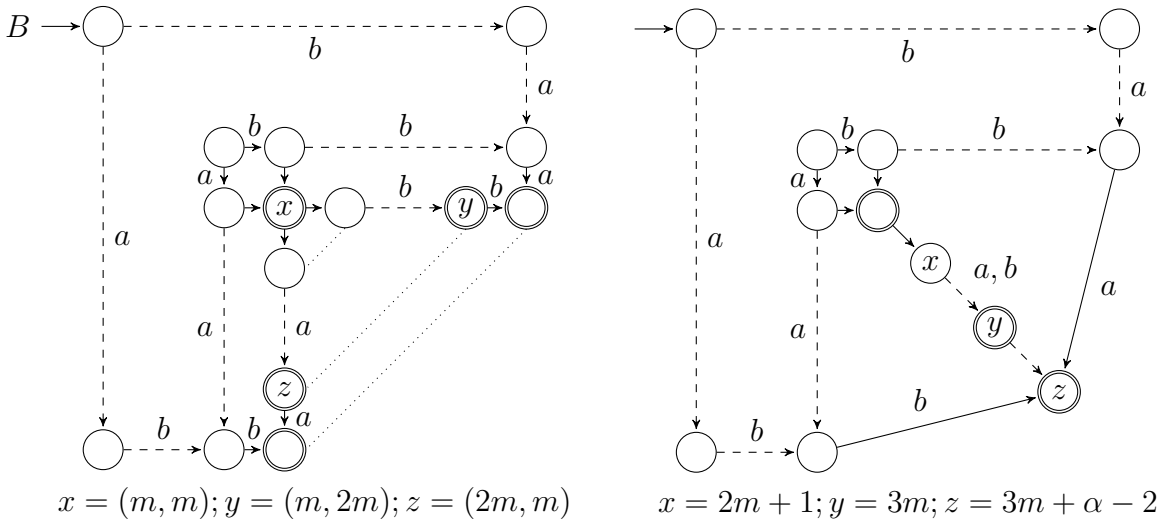


Figure 6.9: A schematic drawing of the grid-like DFA B (left) for per(L) and its minimal equivalent DFA (right) obtained from B by merging the states connected by dotted lines.

The automaton B accepting per(L) is constructed according to [22, Lemma 3.1]. It has a grid-like structure with a truncated lower right where each state (i, j) can be identified with the Parikh vector (i, j) satisfying $0 \leq i, j \leq 2m + (\alpha - 2)$ and $i + j \leq 3m + (\alpha - 2)$. A schematic drawing is given in Figure 6.9 (left). The states in B corresponding to a Parikh vector of a string in L are marked accepting. Since every string wx_w with $w \in \{a, b\}^m$ has the Parikh vector (m, m), the state (m, m) is accepting—see the accepting state x in the schematic drawing in Figure 6.9 (left). The strings of the form wx_ww^R with $w \in \{a, b\}^m$, whose Parikh vectors lie in the set $\{\, (m+i, 2m-i) \mid 0 \leq i \leq m \,\}$, induce the topmost anti-diagonal of accepting states. This anti-diagonal is followed by $\alpha - 2$ further anti-diagonals of accepting states since every string wx_ww^Ry is in L for every string y of length at most $\alpha - 2$. Again, see Figure 6.9 (left). The DFA B is not minimal because all states in a fixed anti-diagonal are equivalent. A schematic drawing of the minimal DFA accepting the permutation of the finite language L is shown in Figure 6.9 (right). Since every anti-diagonal in B corresponds to an accepting state in the minimal DFA, and there are $\alpha - 1$ anti-diagonals in B, the minimal DFA for per(L) has α accepting states.

In order to decrease the accepting state complexity of L to n, it is enough to remove all strings with prefix wx_w for some strings w in $\{a, b\}^m$. Let L' refer to the resulting language. In order to keep the construction working as described above, we must ensure that all accepting states in the topmost anti-diagonal can be reached. This requirement is fulfilled if the Parikh vectors of all strings wx_ww^R with $wx_w \in L'$ form the

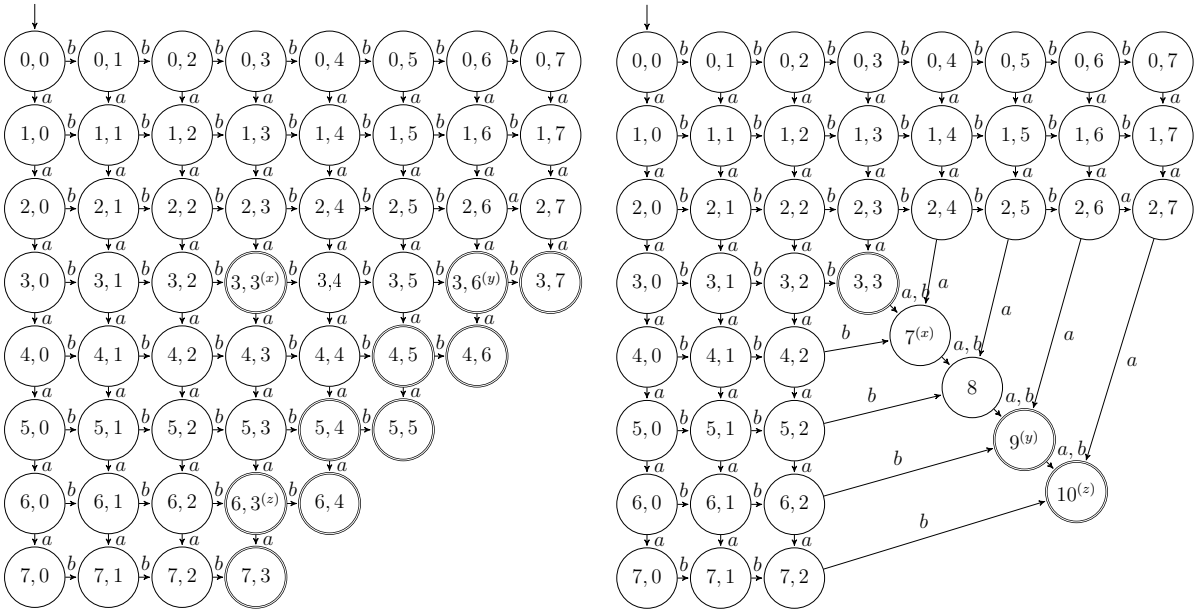


Figure 6.10: An example of the DFA B and its minimal equivalent DFA with $n = 10$, $m = 3$, and $\alpha = 3$.

set $\{(m+i, 2m-i) \mid 0 \leq i \leq m\}$. To this aim, consider the following set of strings:

$$W = \{a^i b^{m-i} \mid 0 \leq i \leq m\}.$$

The set of Parikh vectors of $wx_w w^r$ for each w in W is equal to

$$\{(m+i, 2m-i) \mid 0 \leq i \leq m\}.$$

Hence, as long as we do not remove any of the strings with prefix in W from L, the obtained language fulfills the desired property. On the other hand, whenever all strings with prefix w for a fixed w in $\{a, b\}^* \setminus W$ are removed, the accepting state complexity is decreased by one because the equivalence class $[wx_w]$ is eliminated from the language L'. Iterating this process eliminates further equivalence classes of this form up to the point where only the equivalence classes $[wx_w]$ with $w \in W$ from the original 2^m equivalence classes remain. Thus, there are $m+1$ equivalence classes of this form left in L'. Therefore, the original language L can be modified such that the accepting state complexity of L' varies between $m + 1 + (\alpha - 1)$ and $2^m + (\alpha - 1)$.

Thus, at least $m+1$ strings of the form wx_w for $w \in \{a, b\}^*$ must belong to L'. Finally, this allows us to set the accepting state complexity of L' to n by choosing the parameter m appropriately, which proves the original statement. □

Taking into account Lemmas 6.14, 6.15, and 6.16, we get the following result.

Theorem 6.17. *We have*

$$g_{\mathrm{per}}^{\mathrm{asc}}(n) = g_{\mathrm{per}}^{\mathrm{asc},f}(n) = \begin{cases} \{0\}, & \text{if } n = 0; \\ \mathbb{N}, & \text{if } n = 1; \\ \mathbb{N} \setminus \{1\}, & \text{if } n \geq 2. \end{cases}$$

For unary regular languages, we have $g_{\mathrm{per}}^{\mathrm{asc},u}(n) = \{n\}$ *if* $n \geq 0$.

Proof. If $\mathrm{asc}(L) = 0$, then L is empty, so $\mathrm{per}(L)$ is also empty. The case of $n = 1$ is covered by Lemma 6.14. By Lemma 6.15, we cannot get $\mathrm{asc}(\mathrm{per}(L)) = 1$ if $\mathrm{asc}(L) \geq 2$. All the remaining positive values are covered either by Lemma 6.16 if $\alpha \leq n - 1$ or by Lemma 6.14 if $\alpha \geq n$. □

6.5 Cut Operation

In this section, we examine the accepting state complexity with respect to the cut operation which has been considered in Chapter 5, where the construction of the cut automaton was described. The next lemma deals with non-empty languages.

Lemma 6.18. *Let* $m, n \geq 1$. *Then for every* α *with* $\alpha \geq 0$ *there exists a minimal unary* DFA A *with* m *accepting states and a minimal unary* DFA B *with* n *accepting states such that the minimal* DFA *for* $L(A) \,!\, L(B)$ *has* α *accepting states.*

Proof. We consider three cases.

(1) Let $\alpha = 0$. Define minimal unary DFAs A and B with m and n accepting states, respectively, using Nicaud's notation, as follows:

$$A = (m + 1, m, [0, m - 2] \cup \{m\}),$$
$$B = (n + 3, n + 2, [2, n + 1]).$$

In the cut automaton $A \,!\, B$, only states in columns 0 and 1 are reachable. All of them are rejecting. Hence $L(A) \,!\, L(B) = \emptyset$.

(2) Let $1 \leq \alpha < n$. Define minimal unary DFAs A and B with m and n accepting states, respectively, as follows:

$$A = (m + \alpha, 0, [0, m - 1]),$$
$$B = (n + 2, n + 1, [1, n - 1] \cup \{n + 1\}).$$

Construct the DFA $A\,!\,B$ as shown in Figure 6.11. Since 0 is accepting in A, the initial state of $A\,!\,B$ is $(0,0)$. Each state $(i,0)$ with $1 \leq i \leq m-1$ is reachable by the string a^i since the state i is accepting in A. Moreover, the state $(i,0)$ is rejecting since the state 0 in B is rejecting. Each state $(m{-}1{+}j,\ j)$ with $1 \leq j \leq \alpha$ is reachable by the string a^{m-1+j} since the state $m-1+j$ is rejecting in A. Moreover, the state $(m-1+j,\ j)$ is accepting since the state j in B is accepting. The state $(m-1+\alpha,\ \alpha)$ goes to $(0,0)$ on a. Hence no other state is reachable in $A\,!\,B$. Next, the number of reachable accepting states is α, and all of them are pairwise distinguishable.

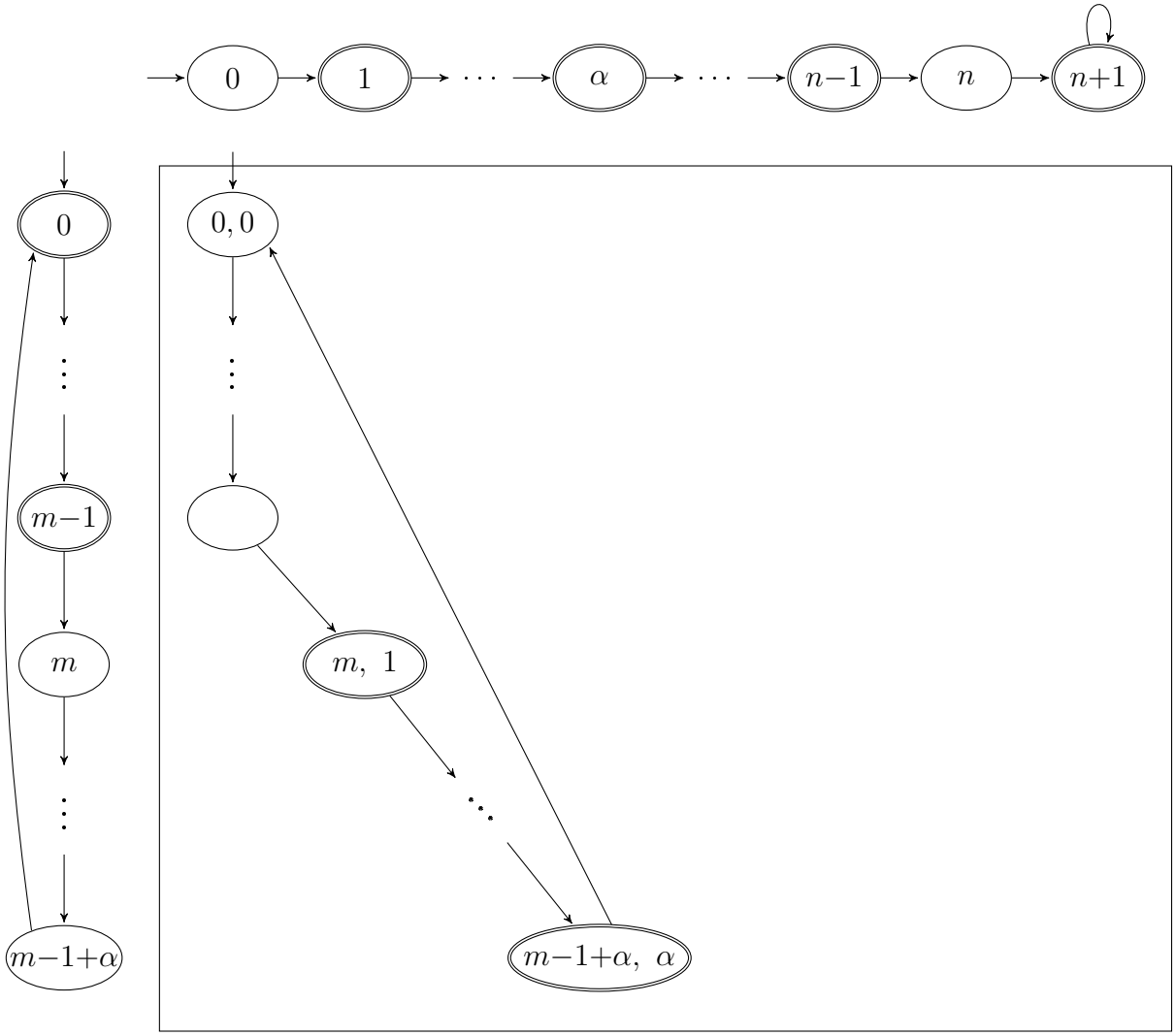


Figure 6.11: The witness DFAs A and B and the result of the cut operation; case $\alpha < n$.

(3) Let $\alpha \geq n$. Define minimal unary DFAs A and B with m and n accepting states, respectively, as follows:

$$A = (m + \alpha + 1, 0, [0, m-1])$$

and

$$B = (n + 2, n + 1, [1, n-1] \cup \{n+1\}).$$

Construct the cut automaton $A\,!\,B$ as shown in Figure 6.12. Since 0 is accepting in A, the initial state of $A\,!\,B$ is $(0, 0)$. Each state $(i, 0)$ with $1 \leq i \leq m-1$ is reachable and rejecting as in case (2). Each state $(m-1+j, j)$ with $1 \leq j \leq n+1$ is reachable by the string a^{m-1+j} since the state $m-1+j$ is rejecting in A. Moreover, all of them but $(m-1+n, n)$ are accepting. The states $(m+k, n+1)$ with $n \leq k \leq \alpha$ are reachable by the string a^{m+k} since the state $m+k$ is rejecting in A and they are accepting since $n+1$ is accepting in B. The state $(m+\alpha, n+1)$ goes to $(0, 0)$ on a. Hence no other state is reachable in $A\,!\,B$.

Next, the number of reachable accepting states is α, and all of them are pairwise distinguishable if $m \geq 2$. If $m = 1$, then we modify the DFA B to have one more rejecting state before the accepting sink state $n+2$. This results in α reachable and pairwise distinguishable accepting states in the cut automaton: we have either a group of rejecting states followed by a group of accepting states in a cycle (if $n = 1$), or we have one rejecting state followed by a group of $n-1$ accepting states, two rejecting states, and $\alpha - n + 1$ accepting states in a cycle (if $n \geq 2$). □

Since we have $\emptyset\,!\,L = K\,!\,\emptyset = \emptyset$ for every language K and L, by the previous lemma, we conclude with the following result stating that every non-negative integer can be attained by the accepting state complexity of the cut of two unary languages unless at least one of them is empty.

Theorem 6.19. *We have*

$$g_{!}^{\mathrm{asc},u}(m,n) = g_{!}^{\mathrm{asc}}(m,n) = \begin{cases} \{0\}, & \text{if } m = 0 \text{ or } n = 0; \\ \{0\} \cup \mathbb{N}, & \text{otherwise.} \end{cases}$$

□

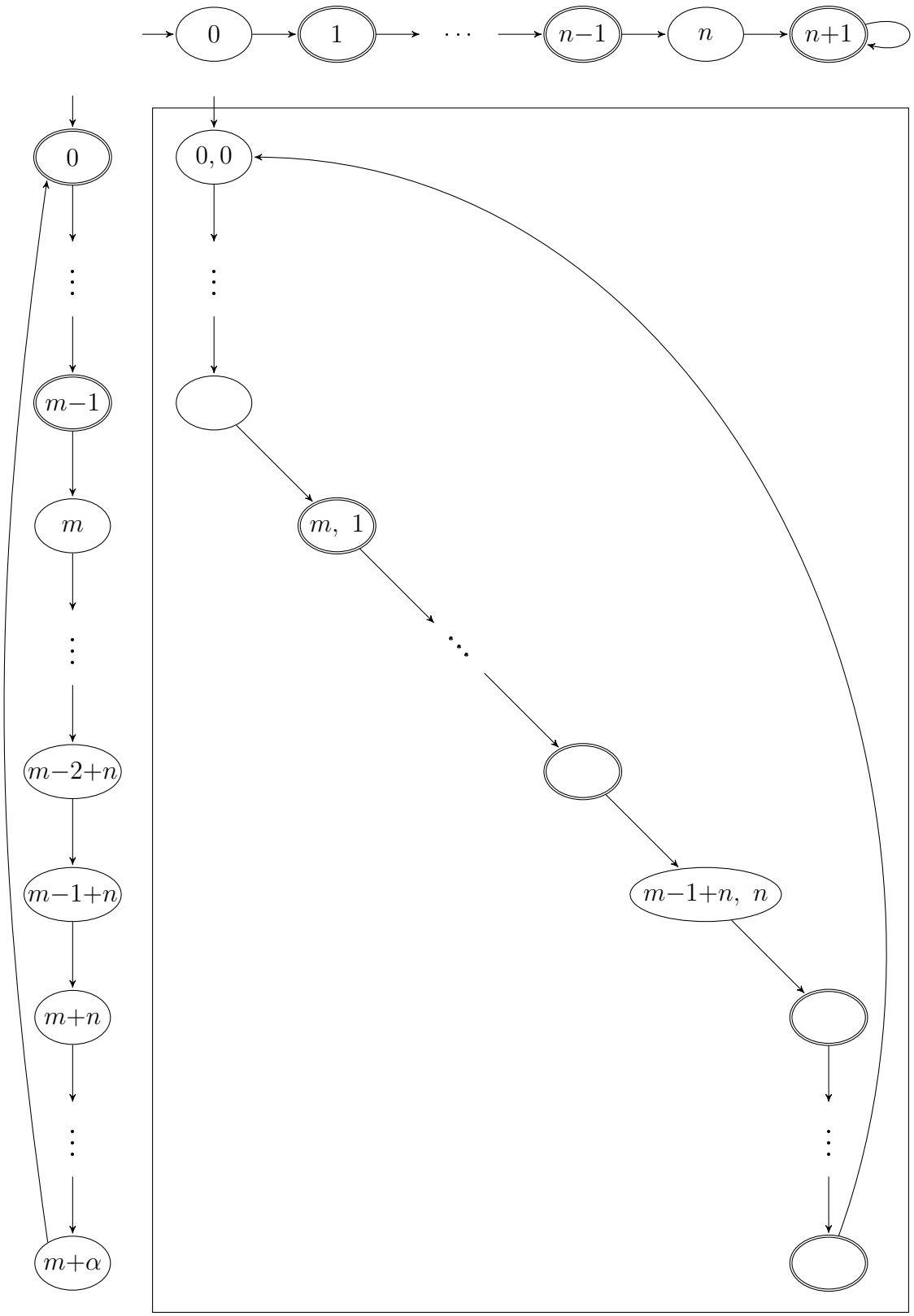


Figure 6.12: The witness DFAs A and B and the result of the cut operation; case $\alpha \geq n$.

6.6 Conclusions

Continuing the work of Dassow [25], we investigated the accepting state complexity of languages resulting from unary and binary operations on regular languages. In the cases of symmetric difference, right and left quotient, reversal, and cut operation, we were able to provide witness for every natural numbers m, n reaching arbitrary complexity α as the result. In the case of permutation on finite languages, every natural number is attainable except one if the argument has complexity at least two. On the other hand, the upper bound for intersection is mn and every number in the range is attainable. This and the result for reversal answer the questions stated in [25]. Results by Dassow are summarized in Table 6.1 (left) which is the same as Table 3.3 from chapter Known results. The results obtained in this chapter are summarized in Table 6.1 (right).

The table also displays the size of alphabet used to describe witnesses. Notice that for reversal and permutation, the unary case is not interesting since the result of the operation is the same language. We did not consider unary case for intersection.

In the case of symmetric difference on *unary finite* languages, the range of possible accepting state complexities is not contiguous, so magic numbers exist, and this is the only case that we have found some magic numbers in this chapter.

Operation	Range	M.	$\lvert\Sigma\rvert$
$L^{\complement}$	$\mathbb{N} \cup \{0 \mid n = 1\}$	no	1
$K \cup L$	$\mathbb{N}$	no	1
KL	$\mathbb{N}$	no	1
$K \setminus L$	$\{0\} \cup \mathbb{N}$	no	1
L^*	$\mathbb{N}$	no	1
$K \cap L$	$[0, mn]$	?	

Operation	Range	M.	$\lvert\Sigma\rvert$
$K \cap L$	$[0, mn]$	no	2
$K \oplus L$	$\{0\} \cup \mathbb{N}$	no	1
KL^{-1}	$\{0\} \cup \mathbb{N}$	no	1
$L^{-1}K$	$\{0\} \cup \mathbb{N}$	no	1
L^R	$\mathbb{N}$	no	2
$\mathrm{per}(L)$	$\mathbb{N} \setminus \{\, 1 \mid n \geq 2 \,\}$	no	2
$K \,!\, L$	$\{0\} \cup \mathbb{N}$	no	1

Table 6.1: Results from [25] (left) and results of this chapter (right). Let $\mathrm{asc}(K) = m$ and $\mathrm{asc}(L) = n$ for $m, n \geq 1$. Then the range indicates the attainable accepting state complexities of the operation under consideration and the status of the magic number problem (M.) refers to whether there exist magic numbers in the given range or not.

Chapter 7

Nondeterministic State Complexity in Subclasses of Convex Languages

In this chapter we examine the descriptional complexity of operations on subclasses of convex regular languages. The complexity of an operation on a subclass can be smaller than on the class of regular languages. This was the motivation for Brzozowski et al. [14, 10, 11] who considered the complexity of operations on free, ideal, and closed languages represented by DFAs and for Mlynárčik et al. [46, 47, 62, 77, 78] who studied the nondeterministic state complexity of operations on the same classes of languages.

Here we continue this research. First, we solve two open problems from [78] concerning the nondeterministic state complexity of intersection on binary subword-free languages and reversal on binary left ideal languages. Next, we examine the nondeterministic state complexity of power and positive closure on the classes of prefix-, suffix-, factor-, and subword-free, closed, and convex, and right, left, two-sided, and all-sided ideal languages. For both operations we get the exact complexity in all considered classes, and all our witnesses are defined over an optimal binary or unary alphabet. This chapter is based on the following two papers:

Hospodár, M., Jirásková, G., Mlynárčik, P.: Nondeterministic complexity in subclasses of convex languages. *Theoretical Computer Science* (to appear).
DOI: 10.1016/j.tcs.2018.12.027

Hospodár, M., Palmovský, M.: Nondeterministic complexity of power and positive closure on subclasses of convex languages. In: Freund, R. et al. (eds.): *Tenth Workshop on Non-Classical Models of Automata and Applications, NCMA 2018, Košice, Slovakia, August 21–22, 2018. Short Papers.* pp. 35–44 (2018).

7.1 Intersection on Binary Subword-Free Languages

The nondeterministic state complexity of intersection on subword-free languages is $(m-2)(n-2)+2$ and it is met by languages defined over an alphabet of size $m+n-5$ [49, Theorem 19(2)]. In this section, we show that there exist two binary languages, both accepted by n-state NFAs, such that the nondeterministic state complexity of their intersection is at least $n^2/6$. This gives a positive answer to [78, Conjecture 4.9].

First, we provide a sufficient condition for a partial DFA to accept a subword-free language.

Lemma 7.1. *Let $A = (\{0,1,\ldots,n-1\}, \Sigma, \cdot, 0, \{n-1\})$ be an n-state partial* DFA *such that for all states i and j and each input symbol σ in Σ, we have*

(1) $i \cdot \sigma > i$ whenever $i \cdot \sigma$ is defined;

(2) if $i < j$, then $i \cdot \sigma < j \cdot \sigma$ whenever $i \cdot \sigma$ and $j \cdot \sigma$ are defined.

Then $L(A)$ is subword-free.

Proof. By induction on $|w|$, the properties (1) and (2) can be proved for an arbitrary non-empty string w. Since A is a partial DFA, every string w in $L(A)$ has exactly one computation in A. Consider a string

$$w = v_0 u_1 v_1 u_2 \cdots v_{k-1} u_k v_k$$

and its proper subword $v = v_0 v_1 \cdots v_k$ with $|v_0|, |v_k| \geq 0$ and $|u_i|, |v_i| > 0$ if $1 \leq i \leq k-1$. Assume that both w and v are in L. This means that both w and v have accepting computations in A:

$$0 \xrightarrow{v_0} p_1 \xrightarrow{u_1} q_1 \xrightarrow{v_1} p_2 \xrightarrow{u_2} q_2 \xrightarrow{v_2} \cdots \xrightarrow{u_k} q_k \xrightarrow{v_k} p_{k+1} = n-1,$$

$$0 \xrightarrow{v_0} p_1 \xrightarrow{v_1} p'_2 \xrightarrow{v_2} \cdots \xrightarrow{v_k} p'_{k+1} = n-1.$$

Since u_1 is non-empty, $p_1 < q_1$, so $p'_2 = p_1 \cdot v_1 < q_1 \cdot v_1 = p_2$, and by induction, we have $p'_{k+1} < p_{k+1} = n-1$, a contradiction. Hence $L(A)$ is subword-free. □

Now we are able to get the main result of this section which shows that the upper bound mn for intersection is asymptotically tight for infinitely many pairs of numbers m and n in the class of binary subword-free languages. The binary alphabet cannot be decreased here since the nondeterministic state complexity of intersection on unary subword-free languages is $\min\{m,n\}$; recall that a unary language is subword-free if and only if it consists of at most one string.

Theorem 7.2. *Let $n \geq 4$. There exist binary subword-free languages K and L accepted by n-state* NFA*s such that every* NFA *for $K \cap L$ has at least $n^2/6$ states.*

Proof. Let $k = \lfloor (n-1)/3 \rfloor$. Let K be the binary language accepted by the partial DFA $A = (\{0, 1, \ldots, n-1\}, \{a, b\}, \cdot, 0, \{n-1\})$, with transitions defined as follows:

$$i \cdot a = i + 1, \quad \text{if } 0 \leq i \leq n-2; \qquad i \cdot b = \begin{cases} i+2, & \text{if } 0 \leq i \leq 2k-2 \text{ and } i \text{ is even;} \\ i+1, & \text{if } n-k-1 \leq i \leq n-2; \end{cases}$$

where all the remaining transitions are undefined; Figure 7.1 (left) shows the partial DFA A in the case of $n = 13$. By Lemma 7.1, the language K is subword-free. Let L be the language accepted by the partial DFA B obtained from A^R by renaming each state i to $n-1-i$ $(0 \leq i \leq n-1)$; see Figure 7.1 (top). We have $L = K^R$, so L is subword-free as well.

Consider the product automaton $A \times B$ for $K \cap L$. Let us show that for each state in the main diagonal, as well as for some states in the k diagonals below the main one, the corresponding singleton set is reachable as well as co-reachable in $A \times B$; notice that each diagonal (but the main one) has three elements less than the diagonal just above it. To this aim, notice that for $j = 0, 1, \ldots, k$ and $i = 0, 1, \ldots, n-1-3j$, we have

$$\{(0,0)\} \xrightarrow{b^j} \{(2j, j)\} \xrightarrow{a^i} \{(2j+i, j+i)\}$$

in $A \times B$, while in $(A \times B)^R$, we have

$$\{(n-1, n-1)\} \xrightarrow{b^j} \{(n-1-j, n-1-2j)\} \xrightarrow{a^{n-1-3j-i}} \{(2j+i, j+i)\}.$$

So the total number of singleton sets that are reachable and co-reachable is

$$\sum_{j=0}^{k} n - 3j = (k+1) \cdot \frac{n + n - 3k}{2} \geq \frac{n}{3} \cdot \frac{2n - (n-1)}{2} = \frac{n}{3} \cdot \frac{n+1}{2} \geq \frac{n^2}{6}.$$

This proves the theorem. □

7.2 Reversal on Binary Left Ideal Languages

The nondeterministic state complexity of reversal on left ideal languages is $n + 1$ with a ternary witness [78, Theorem 6.9(b)]. Here we show that the upper bound can be met by a binary left ideal language. The binary alphabet is optimal here since the reversal of every unary language is the same language.

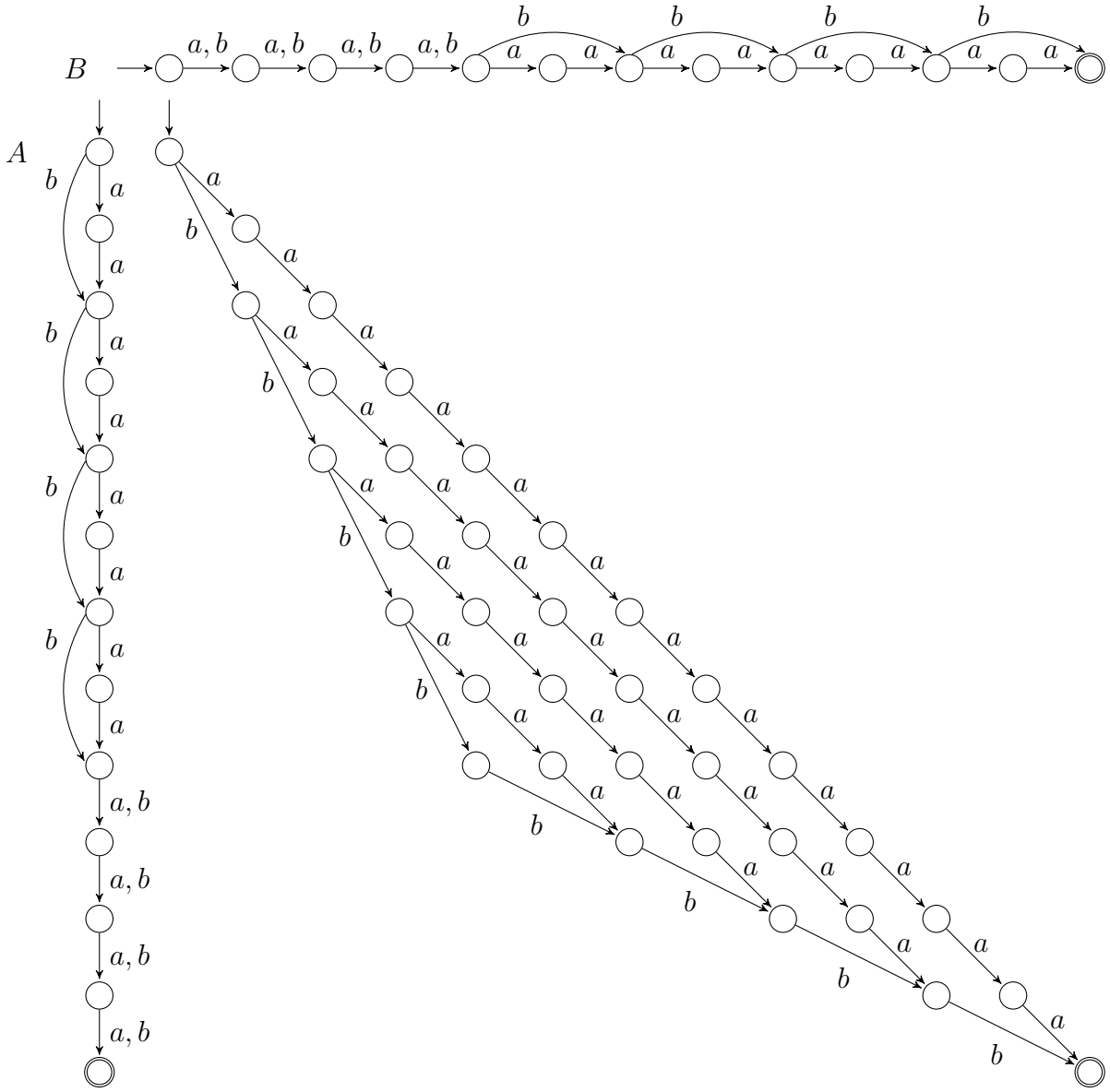


Figure 7.1: The partial DFAs A (left) and B (top) and some transitions in the partial DFA $A \times B$ for $K \cap L$ in the case of $n = 13$, so $k = 4$.

Theorem 7.3. *Let $n \geq 5$. There exists a binary left ideal language L accepted by an n-state* NFA *such that every* NFA *for L^R has at least $n + 1$ states.*

Proof. Let L be the left ideal accepted by the n-state NFA shown in Figure 7.2. Let

$$\mathcal{A} = \{(a^{n-3}, a^{n-4}b)\} \cup \{(a^i, a^{n-4-i}b) \mid 1 \leq i \leq n-4\} \cup \{(a^{n-4}b, \varepsilon)\},$$
$$\mathcal{B} = \{(bb, ba), (b, a)\},$$
$$u = ba, \text{ and}$$
$$v = a^{n-4}b.$$

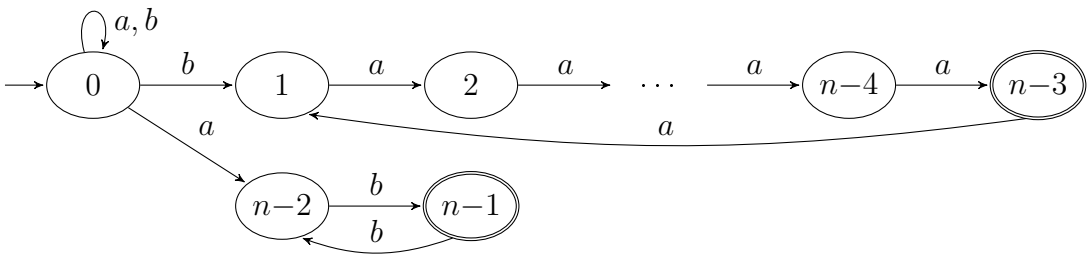


Figure 7.2: The binary left ideal witness NFA A for reversal meeting the bound $n+1$.

The sets $\mathcal{A}$ and $\mathcal{B}$ are disjoint. Next, we have $\{a^{2n-7}b, a^{n-4}b, b^3a, ba\} \subseteq L^R$, so all pairs in $\mathcal{A} \cup \mathcal{B}$, as well as the pairs (ε, u) and (ε, v) concatenate in a string in L^R. On the other hand, for each two distinct pairs in $\mathcal{A} \cup \mathcal{B}$, a mismatched concatenation results either in a string a^kb with $k \neq (n-4) \bmod (n-3)$, or in a string in a^*, or in a string in b^*, or in a string beginning with bba. No such string is in L^R. Hence $\mathcal{A} \cup \mathcal{B}$ is a fooling set for L^R. Next, $a^{n-3} \cdot ba$ is not in L^R, and ε concatenated with the second component of each pair in $\mathcal{A}$ but $a^{n-4}b$ results in a string which is not in L^R. Hence $\mathcal{A} \cup \{(\varepsilon, u)\}$ is a fooling set for L^R. Finally, $bb \cdot a^{n-4}b$ and $\varepsilon \cdot a$ are not in L^R, so $\mathcal{B} \cup \{(\varepsilon, v)\}$ is a fooling set for L^R. By Lemma 2.9, every NFA for L^R has at least $n+1$ states. □

7.3 Power on Convex Languages

In this section, we examine the nondeterministic state complexity of the k-th power and positive closure on subclasses of convex languages. In particular, we consider the classes of prefix-, suffix-, factor-, and subword-free, -closed, and -convex languages, and the classes of right, left, two-sided, and all-sided ideal languages. For both of these operations, we get a tight upper bound in each of the considered classes. All our witnesses are defined over optimal binary or unary alphabets.

Let us recall that if a language is prefix-free, then every minimal NFA for it is non-exiting [38] and if a language is suffix-free, then every minimal NFA for it is non-returning [36]. Next, if a language is a right (left) ideal, then it is accepted by a minimal NFA such that its unique final (initial) state has a loop on each symbol and no other out-transitions (in-transitions) [49, Proposition 12]. We use these automata characterizations to get upper bounds in the considered classes. To get lower bounds, we use the fooling set method given by Lemma 2.8 or, in the case of binary subword-closed languages, its simplification given by Lemma 2.15 (Greater-Smaller Lemma).

The nondeterministic state complexity of the k-th power on regular languages is kn [27, Theorem 3]. We first show that in the classes of free and ideal languages, the upper bound is smaller than kn.

Lemma 7.4 (Power on Free and Ideal Languages: Upper Bounds). *Let $k, n \geq 1$. Let L be a prefix-free or suffix-free, or right or left ideal language accepted by an* NFA *with n states. Then L^k is accepted by an* NFA *with at most $k(n-1)+1$ states.*

Proof. We may assume that a minimal NFA for a prefix-free language L is non-exiting and has a unique final state. To get an NFA for L^k, we take k copies of a minimal NFA for L and we merge the final state in the j-th copy with the initial state in the $(j+1)$-th copy. The initial state of the resulting NFA is the initial state in the first copy, and its unique final state is the final state in the k-th copy.

Now consider right ideal languages. We may assume that a minimal NFA for a right ideal language L has a loop on each symbol in its unique final state which has no other out-transitions. The construction of an NFA for L^k is the same as for prefix-free languages.

If L is suffix-free, then we may assume that a minimal NFA for L is non-returning. To get an NFA for L^k, we take k copies of a minimal NFA for L. For each symbol a and every final state p in the j-th copy with $1 \leq j \leq k-1$, we make the state p non-final and add the transitions (p, a, q) whenever there is a transition on a to q from the initial state in the $(j+1)$-th copy. Next, we omit the unreachable initial state of the $(j+1)$-th copy.

If L is left ideal, we may assume that a minimal NFA for L has a loop on each symbol in its initial state which has no other in-transitions. The construction of an NFA for L^k is the same as for suffix-free languages except that we add a loop on each symbol on p.

In all four cases, we get an NFA for L^k with $k(n-1)+1$ states. □

Lemma 7.5 (Power: Lower Bounds). *Let $k \geq 1$ and $n \geq 2$.*

(a) There exists a unary subword-free language L accepted by an n-state NFA *such that every* NFA *for L^k has at least $k(n-1)+1$ states.*

(b) There exists a unary all-sided ideal language L accepted by an n-state NFA *such that every* NFA *for L^k has at least $k(n-1)+1$ states.*

(c) There exists a binary subword-closed language L accepted by an n-state NFA *such that every* NFA *for L^k has at least kn states.*

Proof. (a) Let $L = \{a^{n-1}\}$, which is accepted by an n-state NFA. We have $L^k = \{a^{k(n-1)}\}$ and the set $\{(a^i, a^{k(n-1)-i}) \mid 0 \leq i \leq k(n-1)\}$ is a fooling set for L^k. By Lemma 2.8, every NFA for L^k has at least $k(n-1)+1$ states.

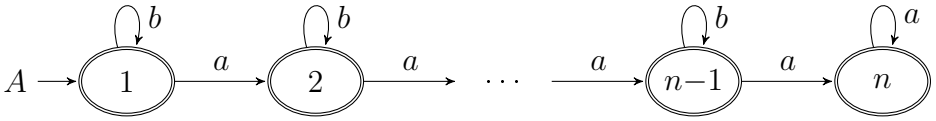


Figure 7.3: Binary subword-closed witness for the k-th power.

(b) Consider the language $L = \{a^i \mid i \geq n-1\}$ which is accepted by an n-state NFA. We have $L^k = \{a^i \mid i \geq k(n-1)\}$ and the same set as above is a fooling set for L^k.

(c) Let L be the language accepted by the partial DFA A shown in Figure 7.3. We have

$$L = \{w \in \{a,b\}^* \mid w = uv \text{ with } |u|_a \leq n-2 \text{ and } v \in a^*\}.$$

It follows that every subword of every string w in L is in L as well. Hence L is subword-closed.

First, construct an NNFA N for L^k in a standard way: take k copies of the NFA A, add the transition on a from every state in the i-th copy to the initial state of the $(i+1)$-th copy, and add the transition on b from every state, but the last one, in the i-th copy to the initial state of the $(i+1)$-th copy $(1 \leq i \leq k-1)$. The set of initial states of N consists of the initial states in all copies of A. The set of final states of N consists of the states in the k-th copy of A. Apply the subset construction to N to get the subset automaton $\mathcal{D}(N)$. The reachable states of $\mathcal{D}(N)$ have the same structure as shown in Figure 7.4. By applying the subset construction to N, and taking into account only the reachable non-empty subsets, we get, after renaming the states, the partial DFA D with kn states shown in Figure 7.4.

Let us prove that D is a minimal NFA for L^k. For $i = 1, 2, \ldots, kn$, let $X_i = \{i\}$ and $Y_i = \{1, 2, \ldots, i\}$. Each X_i is reachable since D is deterministic and does not have unreachable states. Next, let us show that each Y_i is co-reachable in D. Notice that the set of initial states of D^R is $\{1, 2, \ldots, kn\}$ and each Y_{i-1} is reachable in D^R from Y_i either by b or by a. Moreover, $i \in X_i \cap Y_i$, $X_i \subseteq \{i, i+1, \ldots, kn\}$, and $Y_i \subseteq \{1, 2, \ldots, i\}$, so the sets X_i and Y_i satisfy the conditions of Lemma 2.15 (Greater-Smaller Lemma). Hence we have $\mathrm{nsc}(L^k) \leq kn$. □

In the lemma above, we must have $n \geq 2$ since for every positive integer k, the k-th power of a language accepted by a 1-state NFA is the same language. The next theorem shows that two symbols are necessary to meet the bound kn for the k-th power on classes of closed languages. Notice that in the unary case, the classes of prefix-, suffix-, factor-, and subword-convex languages coincide.

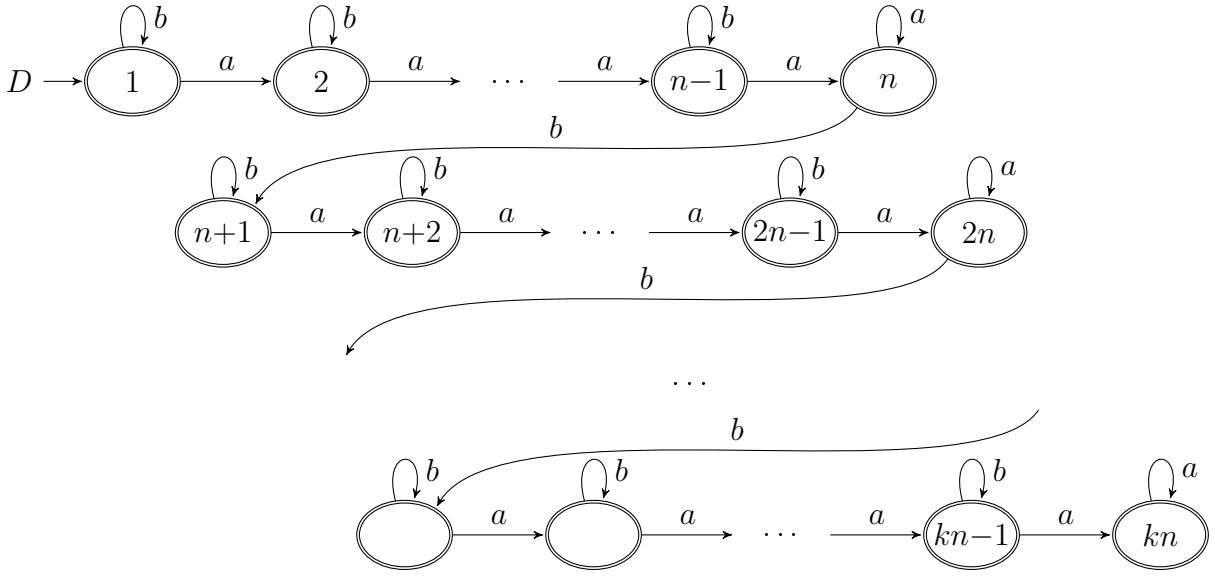


Figure 7.4: A partial DFA D for $L(A)^k$ where A is shown in Figure 7.3.

Lemma 7.6 (Power on Unary Convex Languages). *Let $k \geq 1$ and $n \geq 1$. Let L be a unary convex language accepted by an n-state* NFA. *Then L^k is accepted by an* NFA *with $k(n-1)+1$ states. There exists a unary closed, so convex, language L accepted by an n-state* NFA *such that every* NFA *for L^k has at least $k(n-1)+1$ states.*

Proof. If L is infinite, then $L = \{a^i \mid i \geq \ell\}$ with $\ell \leq n-1$. Hence $L^k = \{a^i \mid i \geq k\ell\}$ and $\mathrm{nsc}(L^k) = k\ell + 1 \leq k(n-1)+1$. If L is finite, then the length of the longest string in L is at most $n-1$, so the length of the longest string in L^k is at most $k(n-1)$. In both cases, the language L^k is accepted by an NFA with $k(n-1)+1$ states. For the lower bound, consider the unary subword-closed language $L = \{a^i \mid 0 \leq i \leq n-1\}$. Then $L^k = \{a^i \mid 0 \leq i \leq k(n-1)\}$ and $\mathrm{nsc}(L^k) = k(n-1)+1$. □

The next theorem summarizes the results obtained in the three lemmas above.

Theorem 7.7 (Power on Convex NFA Languages). *Let k and n be positive integers with $n \geq 2$. The nondeterministic state complexity of the k-th power is:*

(a) ***kn*** *on prefix-, suffix-, factor-, and subword-closed and -convex languages with a binary subword-closed witness, and a binary alphabet is optimal here;*

(b) ***k(n − 1) + 1*** *on prefix-, suffix-, factor-, and subword-free languages with a unary free witness, on right, left, two-sided, and all-sided ideals with a unary all-sided ideal witness, and on unary closed and convex languages with a closed witness.* □

7.4 Positive Closure on Convex Languages

Now we consider the operation of positive closure. The upper bound on nondeterministic state complexity of positive closure on regular languages is n [40, Theorem 9] since we can get an NFA for L^+ from an NFA for L by adding the transition (q, a, s) whenever there is a transition (q, a, f) for a final state f. Our first observation shows that the positive closure of every factor-closed language is of complexity one.

Lemma 7.8 (Positive Closure on Factor-Closed Languages). *Let L be a factor-closed language. Then* $\mathrm{nsc}(L^+) = 1$.

Proof. Let Γ be the set of symbols present in at least one string of L. Then $L \subseteq \Gamma^*$, and since L is factor-closed, $\Gamma \cup \{\varepsilon\} \subseteq L$. It follows that $L^+ = \Gamma^*$, so $\mathrm{nsc}(L^+) = 1$ □

Since every subword-closed language is also factor-closed, the nondeterministic state complexity of positive closure of every subword-closed language is one. For other subclasses, the regular upper bound n holds, and the next theorem proves its tightness.

Lemma 7.9 (Positive Closure: Lower Bounds). *Let $n \geq 1$. There exists*

(a) a unary subword-free (so, subword-convex) language L

(b) a unary all-sided ideal language L

(c) a binary prefix-closed language L

(d) a binary suffix-closed language L

accepted by an n-state NFA *such that every* NFA *for L^+ has at least n states.*

Proof. (a) Consider the language $L = \{a^{n-1}\}$, which is accepted by an n-state NFA. We have $L^+ = \{a^{k(n-1)} \mid k \geq 1\}$ and the set $\{(a^i, a^{n-1-i}) \mid 0 \leq i \leq n-1\}$ is a fooling set for L^+ of size n. By Lemma 2.8, every NFA for L^+ has at least n states.

(b) Let $L = \{a^i \mid i \geq n-1\}$, which is accepted by an n-state NFA. We have $L^+ = L$ and the same set as above is a fooling set for L^+ of size n.

(c) Let L be the language accepted by the NFA shown in Figure 7.5. Notice that each state of this NFA is final, hence L is prefix-closed.

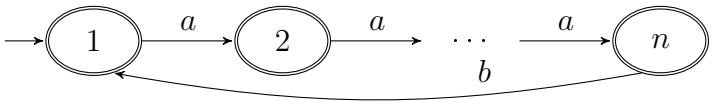


Figure 7.5: A binary prefix-closed witness for positive closure meeting the bound n.

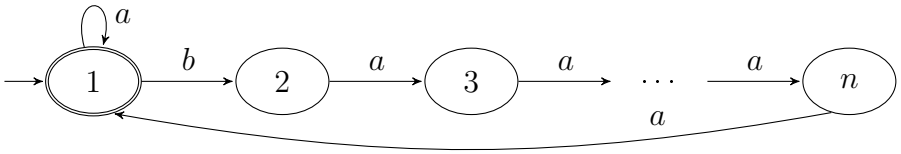


Figure 7.6: A binary suffix-closed witness for positive closure meeting the bound n.

Consider the set of pairs of strings

$$\mathcal{F} = \{(a^i, a^{n-1-i}b) \mid 0 \leq i \leq n-1\}$$

of size n. We have $a^i a^{n-1-i}b = a^{n-1}b$. Since the string $a^{n-1}b$ is in L, it is in L^+. Let $0 \leq i < j \leq n-1$. Then $a^i a^{n-1-j}b$ is not in L^+. Hence the set $\mathcal{F}$ is a fooling set for L^+ of size n.

(d) Let L be the language accepted by the NFA shown in Figure 7.6. Notice that only strings beginning with a are accepted from non-initial states, and for every aw accepted from a non-initial state we have $w \in L$, hence L is suffix-closed. Since the initial state is a unique final state, we have $L = L^+$. Consider the set of pairs of strings

$$\mathcal{F} = \{(ba^i, a^{n-1-i}) \mid 0 \leq i \leq n-1\}$$

of size n. We have $ba^i a^{n-1-i} = ba^{n-1}$, which is in L^+. Let $0 \leq i < j \leq n-1$. Then $ba^i a^{n-1-j}$ is not in L^+. Hence the set $\mathcal{F}$ is a fooling set for L^+ of size n. □

Notice that two symbols are necessary to meet the bound n for positive closure on classes of prefix-closed and suffix-closed languages since every unary prefix-closed or suffix-closed language is subword-closed, and hence its positive closure is of complexity one by Lemma 7.8. The results of the two lemmas above are summarized in the next theorem.

Theorem 7.10 (Positive Closure on Convex NFA Languages). *Let n be a positive integer. The nondeterministic state complexity of positive closure is:*

(a) ***n*** *on prefix-, suffix-, factor, and subword-free and -convex languages and on right, left, two-sided, and all-sided ideal languages with unary witnesses, and on prefix- and suffix-closed languages with binary witnesses,*

(b) **1** *on factor-closed, subword-closed, and unary closed languages.* □

7.5 Conclusions

We proved that the upper bound mn on the nondeterministic state complexity of intersection on subword-free languages is asymptotically tight in the binary case if $m = n$. This solves the open problem stated in [78, Conjecture 4.9].

Then we provided a binary left ideal witness for reversal which improved a result from [78, Theorem 6.9(b)] where a ternary witness was given.

Table 7.1 shows our results on nondeterministic state complexity of power and positive closure on subclasses of convex languages. It also displays the sizes of alphabets used to describe witnesses. Whenever we used a binary or unary alphabet, it was always optimal. Notice that in the case of unary regular languages, the exact complexity of the k-th power is not known.

Class \ Operation	L^k	$\lvert\Sigma\rvert$	L^+	$\lvert\Sigma\rvert$
prefix-, suffix-, factor-, subword-free	$k(n-1)+1$	1	n	1
right, left, two-sided, all-sided ideal	$k(n-1)+1$	1	n	1
prefix-closed, suffix-closed	kn	2	n	2
factor-closed, subword-closed	kn	2	1	1
unary closed	$k(n-1)+1$		1	
prefix-, suffix-, factor-, subword-convex	kn	2	n	1
unary convex	$k(n-1)+1$		n	
regular	kn	2 [27]	n	1 [40]
unary regular	$k(n-1)+1 \leq \cdot \leq kn$	[27]	n	[40]

Table 7.1: Nondeterministic state complexity of the k-th power and positive closure.

Chapter 8

Descriptional Complexity of the Forever Operation

Formal languages are described by various kinds of formal systems. We can compare them by their computational power, or from the descriptional complexity point of view. For example, the description of languages by NFAs is exponentially succinct compared to DFAs. This trade-off results from the subset construction presented by Rabin and Scott in 1959 [83]. Similarly, the trade-off from Boolean finite automata (BFAs) to DFAs is 2^{2^n}, as shown by Leiss in 1981 [72], and the trade-off from alternating finite automata (AFAs) to NFAs is $2^n + 1$, as shown by Jirásková in 2012 [59].

The descriptional complexity of combined operations was investigated by Salomaa, Salomaa, and Yu in 2007 [85, 86] and by more than dozen other papers including Eom and Palmovský in 2016 [31] and Jirásek and Jirásková in 2018 [55]. Sometimes, the state complexity of the combined operation is the same as the composition of the state complexities of individual operations. This is the case of star of intersection. On the other hand, star-complement-star [55] has a double exponential upper bound from composition, but a smaller exact complexity in $2^{\Theta(n \log n)}$.

In this chapter, we investigate the descriptional complexity of the combined operation $(\Sigma^* L^{\complement})^{\complement}$ where L is a language over an alphabet Σ and $L^{\complement}$ stands for the complement of the language L. Jean-Éric Pin asked the following question: Let L be a regular language over Σ recognized by an NFA or a DFA with n states. How many states are sufficient and necessary in the worst case for an NFA (DFA) to recognize the language $(\Sigma^* L^{\complement})^{\complement}$? Jean-Camille Birget in 1996 [6] provided the exact trade-off 2^{n-1} from DFAs to NFAs, and lower and upper bounds 2^{n-1} and $2^{n+1} + 1$, respectively, for the nondeterministic state complexity of $(\Sigma^* L^{\complement})^{\complement}$.

The motivation of Pin's question came from the word model of Propositional Temporal Logic [24]. The set of all models of a formula φ over Σ is a regular language $L(\varphi)$ over Σ. Some of the temporal operators used in this logic are $\circ$ ("next") and $\diamond$ ("eventually", or "at some moment in the future"); there is also the negation operator $-$. A natural dual to the "eventually" operator is the "forever" (or, "always in the future") operator $\Box$, defined to be $-\diamond-$ ("not eventually not"). Formulas and their models are related as follows:

$L(\overline{\varphi}) = L(\varphi)^{\complement}$,

$L(\circ\varphi) = \Sigma L(\varphi)$,

$L(\diamond\varphi) = \Sigma^* L(\varphi)$.

Thus $L(\Box\varphi) = L(\overline{\diamond\overline{\varphi}}) = (\Sigma^* L^{\complement})^{\complement}$. Hence in [6], Birget studied the "forever" operator.

We continue this research by investigating the complexity of the forever operator for different models of finite automata. We consider complete (DFAs) and partial deterministic finite automata (pDFAs), nondeterministic automata with a single (NFAs) or multiple initial states (NNFAs), and Boolean automata with a single initial state, called *alternating* finite automata (AFAs) in [6, 32], or with an initial function (BFAs) [12]. Similarly as Pin, we ask the following question: If a language L is represented by an n-state automaton of some model, how many states are sufficient and necessary in the worst case for an automaton of some other model to accept $(\Sigma^* L^{\complement})^{\complement}$? The answer is present in the published paper which can be found in Appendix [B] at the end of this thesis:

> Hospodár, M., Jirásková, G., Mlynárčik, P.: Descriptional complexity of the forever operator. *International Journal of Foundations of Computer Science* 30(1), pp. 115–134 (2019). DOI: 10.1142/S0129054119400069.

We study all the possible 36 trade-offs, and except for four cases, we always get tight upper bounds. In particular, we prove that the upper bound on the nondeterministic state complexity of $(\Sigma^* L^{\complement})^{\complement}$ is 2^{n-1}, which improves Birget's upper bound $2^{n+1}+1$ and meets his lower bound for DFA-to-NFA trade-off. The most interesting result of [B] is the tight upper bound on the NFA-to-DFA trade-off given by the Dedekind number $\mathrm{M}(n-1)$.

In [B, Lemma 6], some properties of the forever operator are provided. These properties, also with the properties of finite automata from [B, Lemma 5], are used to prove upper and lower bounds on some of the 36 possible trade-offs. However, we do not need to prove each trade-off separately since bounds on some trade-offs follow from bounds on some other trade-offs. This is shown for upper bounds in Table 8.1 using an arrow from one cell to another. A circle denotes the proven upper bound. Similarly, Table 8.2 shows the corresponding arrows for lower bounds; here, a dashed circle denotes the same lower bound proven using a smaller alphabet.

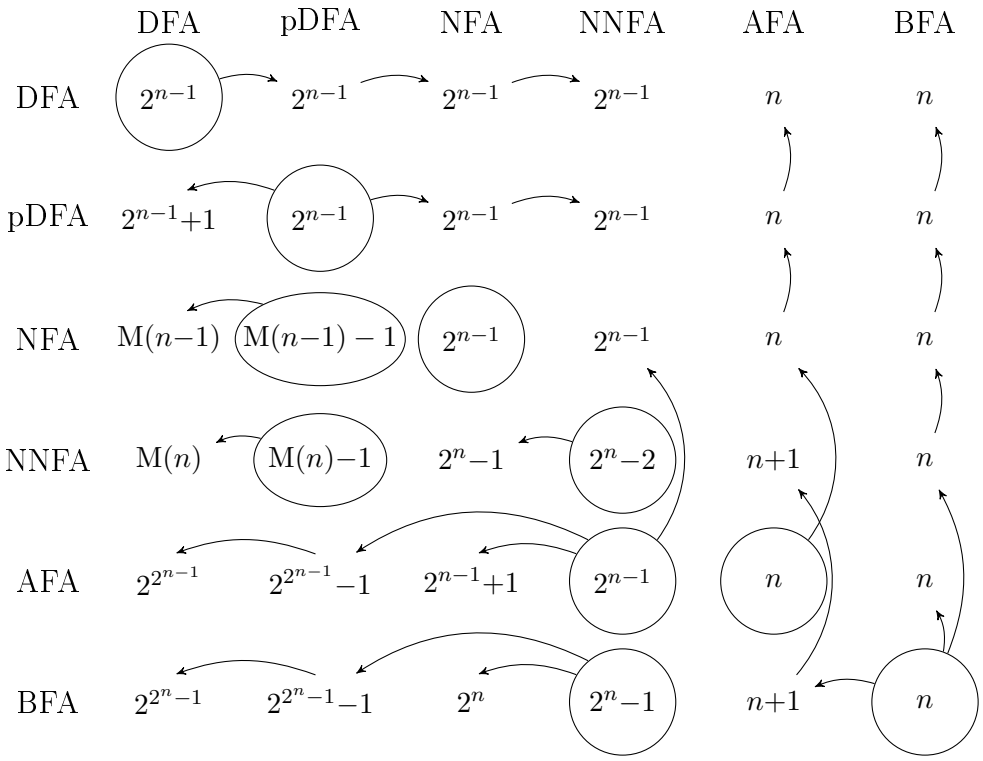


Table 8.1: Upper bounds on the complexity of $(\Sigma^* L^{\complement})^{\complement}$.

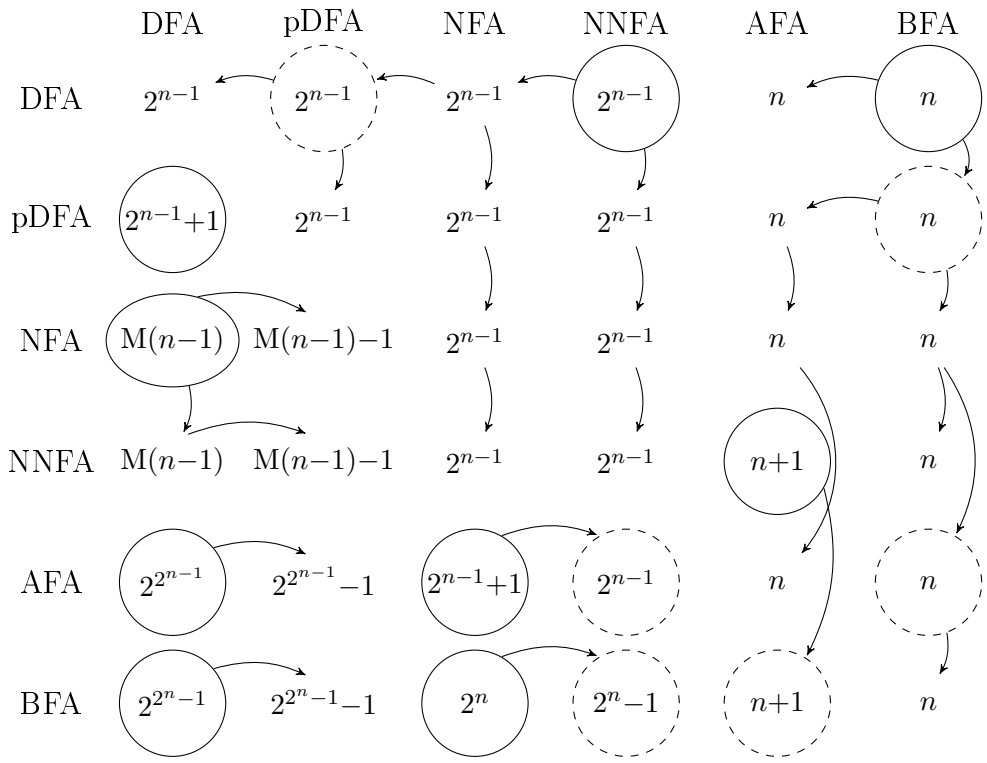


Table 8.2: Lower bounds on the complexity of $(\Sigma^* L^{\complement})^{\complement}$.

$L\backslash(\Sigma^* L^{\complement})^{\complement}$	DFA	$\lvert\Sigma\rvert$	pDFA	$\lvert\Sigma\rvert$	NFA	$\lvert\Sigma\rvert$	NNFA	$\lvert\Sigma\rvert$	AFA	$\lvert\Sigma\rvert$	BFA	$\lvert\Sigma\rvert$
DFA	2^{n-1}	2	2^{n-1}	2	2^{n-1}	3	2^{n-1}	3	n	3	n	3
pDFA	$2^{n-1}+1$	4	2^{n-1}	2	2^{n-1}	3	2^{n-1}	3	n	2	n	2
NFA	$\mathrm{M}(n-1)$	2^{n+1}	$\mathrm{M}(n-1)-1$	2^{n+1}	2^{n-1}	3	2^{n-1}	3	n	2	n	2
NNFA	$\geq \mathrm{M}(n-1)$ $\leq \mathrm{M}(n)$	2^{n+1}	$\geq \mathrm{M}(n-1)-1$ $\leq \mathrm{M}(n)-1$	2^{n+1}	$\geq 2^{n-1}$ $\leq 2^n-1$	3	$\geq 2^{n-1}$ $\leq 2^n-2$	3	$n+1$	2	n	2
AFA	$2^{2^{n-1}}$	2	$2^{2^{n-1}}-1$	2	$2^{n-1}+1$	2	2^{n-1}	1	n	1	n	1
BFA	2^{2^n-1}	2	$2^{2^n-1}-1$	2	2^n	2	2^n-1	1	$n+1$	1	n	1

Table 8.3: The complexity of $(\Sigma^* L^{\complement})^{\complement}$ for various types of finite automata. The DFA-NFA and DFA-NNFA trade-offs are from [6].

$L\backslash(\Sigma^* L^{\complement})^{\complement}$	DFA	pDFA	NFA	NNFA	AFA	BFA
DFA	n	$n-1$	$n-1$	$n-1$	$\lceil \log n \rceil + 1$	$\lceil \log n \rceil$
pDFA	$n+1$	n	n	n	$\lceil \log n \rceil + 1$	$\lceil \log(n+1) \rceil$
NFA	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\lceil \log(\mathrm{D}(n)) \rceil + 1$	$\lceil \log(\mathrm{D}(n)) \rceil$
NNFA	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\lceil \log(\mathrm{D}(n)) \rceil + 1$	$\lceil \log(\mathrm{D}(n)) \rceil$
AFA	$2^{n-1}+1$	2^{n-1}	2^{n-1}	2^{n-1}	n	n
BFA	2^n	2^n-1	2^n-1	2^n-1	$n+1$	n

Table 8.4: The upper bounds on the complexity of the forever operator in the unary case. We have $\mathrm{D}(n) = \mathrm{F}(n-1) + n^2 - 2 \in 2^{O(\sqrt{n \log n})}$ and $\lceil \log(\mathrm{D}(n)) \rceil < n$.

Results from Table 8.1 and 8.2 are summarized in Table 8.3 which also shows the size of alphabet used to describe witness languages for the trade-offs. Table 8.4 displays the upper bound in the unary case and shows that whenever we use binary alphabet, it is optimal in the sense that the upper bound cannot be met by any unary language. Whenever we use an alphabet of size more than two, we do not know whether it is optimal. We conjecture that the upper bounds in the cases NFA-to-{DFA, pDFA} can be met by a constant alphabet. We would also like to achieve tightness in the cases NNFA-to-{DFA, pDFA, NFA, NNFA} where upper and lower bounds are not the same. Our computations show that the upper bound $\mathrm{M}(n)$ for the NNFA-to-DFA trade-off is not tight.

Chapter 9

Conclusions and Future Work

In this thesis, we investigated the descriptional complexity in the class of regular languages and in some of its subclasses. In particular, we considered the state complexity of regular operations on languages represented by deterministic, nondeterministic, alternating, and Boolean finite automata. We also considered another complexity measure called accepting state complexity.

In Chapter 4, we described ternary DFAs with an arbitrary number of final states meeting the upper bound on the state complexity of concatenation. We also provided binary witnesses assuming that the first automaton has at least two non-final states. We used these binary witnesses to describe binary alternating finite automata with m and n states meeting the upper bound $2^m + n + 1$ on the alternating state complexity of concatenation. This gave a solution to an open problem stated in 1990 by Fellah, Jürgensen, and Yu [32].

We provided a complete solution of the magic number problem for the cut operation in Chapter 5 by describing binary witnesses for each number from one up to the known upper bound, and by determining the magic status of each value in the corresponding range in the unary case.

Chapter 6 was devoted to the accepting state complexity. We proved that the range of possible accepting state complexities for intersection is the set $\{0, 1, \ldots, mn\}$ with binary witnesses, which solved an open problem stated by Dassow [25]. We also showed that the corresponding range for symmetric difference and left and right quotients is given by the set of all non-negative integers with unary witnesses, for reversal it is given by the set of all positive integers with binary witnesses, and for permutation on finite languages the range is given by the set of all integers greater than one with binary witnesses.

In Chapter 7 we investigated the nondeterministic state complexity of the operations of intersection, reversal, k-th power, and positive closure in the classes of prefix-, suffix-, factor-, ad subword-free, -closed, and -convex languages, and right, left, two-sided, and all-sided ideal languages. We first described binary subword-free languages accepted by n-state NFAs such that their intersection requires an NFA with at least $n^2/6$ states. This provided a positive answer to a conjecture stated by Mlynárčik [78] and showed that the upper bound mn is asymptotically tight in the binary case whenever $m = n$. Then we provided a binary left ideal witness for reversal, which improved a ternary solution given by Mlynárčik et al. [46]. Finally, we obtained the nondeterministic state complexity of the k-th power and positive closure in all considered subclasses. Our witnesses were defined over a binary or unary alphabet. Whenever we used a binary alphabet, it was always optimal in the sense that the corresponding upper bounds cannot be met by unary languages.

We examined the descriptional complexity of the forever operator $L \mapsto (\Sigma^* L^{\complement})^{\complement}$ in Chapter 8. We represented a language L by an automaton of one of the following six models: complete and partial deterministic finite automata, nondeterministic finite automata with a unique or multiple initial states, alternating, and Boolean automata. We required the result of the forever operator to be represented again by one of these six models. This gave us 36 possible cases. For 32 of them, we obtained the exact trade-offs. Except for four cases, our witness languages were defined over a small fixed alphabet which was often optimal. By showing that the nondeterministic state complexity of this operator is 2^{n-1}, we improved a result by Birget [6]. The most interesting result of this chapter is the NFA-to-DFA trade-off which is given by the Dedekind number $\mathrm{M}(n-1)$.

Some problems remained open in this thesis and we conclude this chapter with their list.

Open Problem 1 (Concatenation on Binary DFAs with One Non-Final State in the First DFA, cf. [A, Theorem 5.1]). Let A be a DFA with m states and $m-1$ final states and B be a DFA with n states and at least two final states Let $L(A)$ and $L(B)$ be binary. What is the state complexity of concatenation $L(A)L(B)$ in the worst case? We have an upper bound $(m+1)2^{n-1}$ and a lower bound smaller by one.

Open Problem 2 (Concatenation on Binary AFAs with One State in the Second AFA). Let A and B be AFAs with m and 1 states, respectively. Let $L(A)$ and $L(B)$ be binary. What is the alternating state complexity of concatenation $L(A)L(B)$ in the worst case? We have an upper bound $2^m + 2$. This problem is open since the DFA B in [A, Theorem 4.7] requires 4 states.

Open Problem 3 (**Concatenation on Unary AFAs, cf. [A, Theorem 6.7]**). Let A and B be AFAs with m and n states, respectively. Let $L(A)$ and $L(B)$ be unary. What is the alternating state complexity of $L(A)L(B)$ in the worst case? We have an upper bound $m+n+1$ and a lower bound $m+n-1$.

Open Problem 4 (**Descriptional Complexity of the Forever Operator, cf. [B]**). Let L be a language accepted by an n-state NNFA. What number of states is necessary in the worst case to accept $(\Sigma^*(L^{\complement}))^{\complement}$ by a DFA, a pDFA, an NFA, or an NNFA? We have upper and lower bounds on all four trade-offs.

Open Problem 5 (**State Complexity of Cut with More Final states, cf. [A]).** Let A be a DFA with m states and k final states and B be a DFA with n states and ℓ final states. What is the state complexity of cut $L(A)\,!\,L(B)$ in the worst case? We have an upper bound and reachability, but more final states in B make distinguishability complicated.

Open Problem 6 (**Magic Number Problem for Cut with More Final States, cf. [A]).** Let A be a minimal DFA with m states and k final states and B be a minimal DFA with n states and ℓ final states. What is the set of all numbers attainable as the state complexity of cut $L(A)\,!\,L(B)$?

Open Problem 7 (**Ranges of State Complexities for Operations).** Let K and L be languages accepted by minimal DFAs with m and n states, respectively. What is the set of all numbers attainable by the state complexity of the symmetric difference $K \oplus L$, right quotient KL^{-1}, left quotient $L^{-1}K$, or permutation of L? These operations were considered for ranges of accepting state complexities in Chapter 6.

Open Problem 8 (**Ranges of Complexities for Unary Operations on Unary Languages, cf. [17, 20]).** Let A be a unary automaton with n states. What is the set of all numbers attainable as the complexity of $L(A)^{\circ}$? If $\circ$ is square and A is an NFA, we have an upper bound $2n$ and conjecture that no number is magic. We even do not know whether the upper bound is $2n$ or $2n-1$. If $\circ$ is the Kleene closure and A is a DFA, we have an upper bound $(n-1)^2+1$ and we have a set of known magic numbers of size $O(n)$. However, we do not know the smallest magic number and there are still $O(n^2)$ numbers with unknown magic status.

Bibliography

[1] Frédérique Bassino, Laura Giambruno, and Cyril Nicaud. "Complexity of Operations on Cofinite Languages". In: *LATIN 2010: Theoretical Informatics, 9th Latin American Symposium, Oaxaca, Mexico, April 19-23, 2010. Proceedings*. Ed. by Alejandro López-Ortiz. Vol. 6034. Lecture Notes in Computer Science. Springer, 2010, pp. 222–233. DOI: 10.1007/978-3-642-12200-2_21.

[2] Martin Berglund, Henrik Björklund, Frank Drewes, Brink van der Merwe, and Bruce Watson. "Cuts in regular expressions". In: *Developments in Language Theory - 17th International Conference, DLT 2013, Marne-la-Vallée, France, June 18-21, 2013. Proceedings*. Ed. by Marie-Pierre Béal and Olivier Carton. Vol. 7907. Lecture Notes in Computer Science. Springer, 2013, pp. 70–81. DOI: 10.1007/978-3-642-38771-5_8.

[3] Piotr Berman and Andrzej Lingas. "On the complexity of regular languages in terms of finite automata". In: *Technical Report 304,* Polish Academy of Sciences (1977), pp. 185–190.

[4] Jean-Camille Birget. "Intersection and Union of Regular Languages and State Complexity". In: *Inform. Process. Lett.* 43.4 (1992), pp. 185–190. DOI: 10.1016/0020-0190(92)90198-5.

[5] Jean-Camille Birget. "Partial Orders on Words, Minimal Elements of Regular Languages and State Complexity". In: *Theoret. Comput. Sci.* 119.2 (1993), pp. 267–291. DOI: 10.1016/0304-3975(93)90160-U.

[6] Jean-Camille Birget. "The State Complexity of $\overline{\Sigma^*\overline{L}}$ and its Connection with Temporal Logic". In: *Inform. Process. Lett.* 58.4 (1996), pp. 185–188. DOI: 10.1016/0020-0190(96)00044-0.

[7] Henning Bordihn et al., eds. *Eighth Workshop on Non-Classical Models of Automata and Applications, NCMA 2016, Debrecen, Hungary, August 29-30, 2016. Proceed-*

ings. Vol. 321. books@ocg.at. Österreichische Computer Gesellschaft, 2016. ISBN: 978-3-903035-10-2.

[8] Janusz A. Brzozowski. "Complexity in Convex Languages". In: *Language and Automata Theory and Applications, 4th International Conference, LATA 2010, Trier, Germany, May 24-28, 2010. Proceedings*. Ed. by Adrian-Horia Dediu, Henning Fernau, and Carlos Martín-Vide. Vol. 6031. Lecture Notes in Computer Science. Springer, 2010, pp. 1–15. DOI: 10.1007/978-3-642-13089-2_1.

[9] Janusz A. Brzozowski. "Quotient Complexity of Regular Languages". In: *Journal of Automata, Languages and Combinatorics* 15.1/2 (2010), pp. 71–89. DOI: 10.25596/jalc-2010-071.

[10] Janusz A. Brzozowski, Galina Jirásková, and Baiyu Li. "Quotient complexity of ideal languages". In: *Theoret. Comput. Sci.* 470 (2013), pp. 36–52. DOI: 10.1016/j.tcs.2012.10.055.

[11] Janusz A. Brzozowski, Galina Jirásková, and Chenglong Zou. "Quotient Complexity of Closed Languages". In: *Theory Comput. Syst.* 54.2 (2014), pp. 277–292. DOI: 10.1007/s00224-013-9515-7.

[12] Janusz A. Brzozowski and Ernst L. Leiss. "On Equations for Regular Languages, Finite Automata, and Sequential Networks". In: *Theoret. Comput. Sci.* 10 (1980), pp. 19–35. DOI: 10.1016/0304-3975(80)90069-9.

[13] Janusz A. Brzozowski and Bo Liu. "Quotient complexity of star-free languages". In: *Internat. J. Found. Comput. Sci.* 23.6 (2012), pp. 1261–1276. DOI: 10.1142/S0129054112400515.

[14] Janusz Brzozowski, Galina Jirásková, Baiyu Li, and Joshua Smith. "Quotient Complexity of bifix-, factor-, and subword-free Regular Languages". In: *Acta Cybernetica* 21.4 (2014), pp. 507–527. DOI: 10.14232/actacyb.21.4.2014.1.

[15] Janusz A. Brzozowski, Galina Jirásková, Bo Liu, Aayush Rajasekaran, and Marek Szykula. "On the State Complexity of the Shuffle of Regular Languages". In: *DCFS 2016*. Ed. by Cezar Câmpeanu, Florin Manea, and Jeffrey Shallit. Vol. 9777. Lecture Notes in Computer Science. Springer, 2016, pp. 73–86. DOI: 10.1007/978-3-319-41114-9_6.

[16] Cezar Câmpeanu et al. "State Complexity of Basic Operations on Finite Languages". In: *Automata Implementation, 4th International Workshop on Implementing Automata, WIA'99, Potsdam, Germany, July 17-19, 1999, Revised Papers*. Ed. by Oliver Boldt and Helmut Jürgensen. Vol. 2214. Lecture Notes in Computer Science. Springer, 1999, pp. 60–70. DOI: 10.1007/3-540-45526-4_6.

[17] Kristína Čevorová. "Kleene star on unary regular languages". In: *DCFS 2013*. Ed. by Helmut Jürgensen and Rogério Reis. Vol. 8031. Lecture Notes in Computer Science. Springer, 2013, pp. 277–288. DOI: 10.1007/978-3-642-39310-5_26.

[18] Kristína Čevorová. "Square on Ideal, Closed and Free Languages". In: *DCFS 2015*. Ed. by Jeffrey Shallit and Alexander Okhotin. Vol. 9118. Lecture Notes in Computer Science. Springer, 2015, pp. 70–80. DOI: 10.1007/978-3-319-19225-3_6.

[19] Kristína Čevorová. "Square on closed languages". In: *NCMA 2016*. Ed. by Henning Bordihn et al. Vol. 321. books@ocg.at. Österreichische Computer Gesellschaft, 2016, pp. 121–130. ISBN: 978-3-903035-10-2.

[20] Kristína Čevorová, Galina Jirásková, and Ivana Krajňáková. "On the square of regular languages". In: *CIAA 2014*. Ed. by Markus Holzer and Martin Kutrib. Vol. 8587. Lecture Notes in Computer Science. Springer, 2014, pp. 136–147. DOI: 10.1007/978-3-319-08846-4_10.

[21] Ashok K. Chandra, Dexter Kozen, and Larry J. Stockmeyer. "Alternation". In: *J. ACM* 28.1 (1981), pp. 114–133. DOI: 10.1145/322234.322243.

[22] Da-Jung Cho, Daniel Goč, Yo-Sub Han, Sang-Ki Ko, Alexandros Palioudakis, and Kai Salomaa. "State complexity of permutation on finite languages over a binary alphabet". In: *Theoret. Comput. Sci.* 682 (2017), pp. 67–78.

[23] Roland Cmorik and Galina Jirásková. "Basic Operations on Binary Suffix-Free Languages". In: *Mathematical and Engineering Methods in Computer Science - 7th International Doctoral Workshop, MEMICS 2011, Lednice, Czech Republic, October 14-16, 2011, Revised Selected Papers*. Ed. by Zdeněk Kotásek et al. Vol. 7119. Lecture Notes in Computer Science. Springer, 2011, pp. 94–102. DOI: 10.1007/978-3-642-25929-6_9.

[24] Joëlle Cohen, Dominique Perrin, and Jean-Éric Pin. "On the Expressive Power of Temporal Logic". In: *J. Comput. Syst. Sci.* 46.3 (1993), pp. 271–294. DOI: 10.1016/0022-0000(93)90005-H.

[25] Jürgen Dassow. "On the Number of Accepting States of Finite Automata". In: *Journal of Automata, Languages and Combinatorics* 21(1–2) (2016), pp. 55–67. DOI: 10.25596/jalc-2016-055.

[26] Jürgen Dassow. "Descriptional complexity and operations – Two non-classical cases". In: *Descriptional Complexity of Formal Systems - 19th IFIP WG 1.02 International Conference, DCFS 2017, Milano, Italy, July 3-5, 2017, Proceedings*. Ed. by Giovanni Pighizzini and Cezar Câmpeanu. Vol. 10316. Lecture Notes in Computer Science. Springer, 2017, pp. 33–44. DOI: 10.1007/978-3-319-60252-3_3.

[27] Michael Domaratzki and Alexander Okhotin. "State complexity of power". In: *Theoret. Comput. Sci.* 410.24-25 (2009), pp. 2377–2392. DOI: 10.1016/j.tcs.2009.02.025.

[28] Erik Dorčák. "Concatenation of regular languages and state complexity". ŠVK thesis. PF UPJŠ, Košice, 2015.

[29] Frank Drewes, Markus Holzer, Sebastian Jakobi, and Brink van der Merwe. "Tight bounds for cut-operations on deterministic finite automata". In: *Fundam. Inform.* 155.1-2 (2017), pp. 89–110. DOI: 10.3233/FI-2017-1577.

[30] Hae-Sung Eom, Yo-Sub Han, and Galina Jirásková. "State Complexity of Basic Operations on Non-Returning Regular Languages". In: *Fundam. Inform.* 144.2 (2016), pp. 161–182. DOI: 10.3233/FI-2016-1326.

[31] Hae-Sung Eom and Matúš Palmovský. "Combined operations on prefix-free and suffix-free languages". In: *NCMA 2016*. Ed. by Henning Bordihn et al. Vol. 321. books@ocg.at. Österreichische Computer Gesellschaft, 2016, pp. 147–162. ISBN: 978-3-903035-10-2.

[32] Abdelaziz Fellah, Helmut Jürgensen, and Sheng Yu. "Constructions for alternating finite automata". In: *Intern. J. Computer Math.* 35.1-4 (1990), pp. 117–132. DOI: 10.1080/00207169008803893.

[33] Viliam Geffert. "Magic numbers in the state hierarchy of finite automata". In: *Inform. Comput.* 205.11 (2007), pp. 1652–1670. DOI: 10.1016/j.ic.2007.07.001.

[34] Ian Glaister and Jeffrey Shallit. "A Lower Bound Technique for the Size of Nondeterministic Finite Automata". In: *Inform. Process. Lett.* 59.2 (1996), pp. 75–77. DOI: 10.1016/0020-0190(96)00095-6.

[35] Yo-Sub Han and Kai Salomaa. "State complexity of basic operations on suffix-free regular languages". In: *Theoret. Comput. Sci.* 410.27-29 (2009), pp. 2537–2548. DOI: 10.1016/j.tcs.2008.12.054.

[36] Yo-Sub Han and Kai Salomaa. "Nondeterministic State Complexity for Suffix-Free Regular Languages". In: *Proceedings Twelfth Annual Workshop on Descriptional Complexity of Formal Systems, DCFS 2010, Saskatoon, Canada, 8-10th August 2010*. Ed. by Ian McQuillan and Giovanni Pighizzini. Vol. 31. EPTCS. 2010, pp. 189–196. DOI: 10.4204/EPTCS.31.21.

[37] Yo-Sub Han, Kai Salomaa, and Derick Wood. "State Complexity of Prefix-Free Regular Languages". In: *8th International Workshop on Descriptional Complexity of Formal Systems - DCFS 2006, Las Cruces, New Mexico, USA, June 21 - 23, 2006. Proceedings*. Ed. by Hing Leung and Giovanni Pighizzini. New Mexico State University, Las Cruces, New Mexico, USA, 2006, pp. 165–176.

[38] Yo-Sub Han, Kai Salomaa, and Derick Wood. "Nondeterministic State Complexity of Basic Operations for Prefix-Free Regular Languages". In: *Fundam. Inform.* 90.1-2 (2009), pp. 93–106. DOI: 10.3233/FI-2009-0008.

[39] Markus Holzer, Sebastian Jakobi, and Martin Kutrib. "The Magic Number Problem for Subregular Language Families". In: *Internat. J. Found. Comput. Sci.* 23.1 (2012), pp. 115–131. DOI: 10.1142/S0129054112400084.

[40] Markus Holzer and Martin Kutrib. "Nondeterministic descriptional complexity of regular languages". In: *Internat. J. Found. Comput. Sci.* 14.6 (2003), pp. 1087–1102. DOI: 10.1142/S0129054103002199.

[41] Markus Holzer and Martin Kutrib, eds. *Implementation and Application of Automata - 19th International Conference, CIAA 2014, Giessen, Germany, July 30 - August 2, 2014. Proceedings*. Vol. 8587. Lecture Notes in Computer Science. Springer, 2014. DOI: 10.1007/978-3-319-08846-4.

[42] Markus Holzer, Martin Kutrib, and Katja Meckel. "Nondeterministic state complexity of star-free languages". In: *Theoret. Comput. Sci.* 450 (2012), pp. 68–80. DOI: 10.1016/j.tcs.2012.04.028.

[43] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979. ISBN: 0-201-02988-X.

[44] Michal Hospodár and Galina Jirásková. "Concatenation on deterministic and alternating automata". In: *NCMA 2016*. Ed. by Henning Bordihn et al. Vol. 321. books@ocg.at. Österreichische Computer Gesellschaft, 2016, pp. 179–194. ISBN: 978-3-903035-10-2.

[45] Michal Hospodár, Galina Jirásková, and Ivana Krajňáková. "Operations on Boolean and Alternating Finite Automata". In: *Computer Science - Theory and Applications - 13th International Computer Science Symposium in Russia, CSR 2018, Moscow, Russia, June 6-10, 2018, Proceedings*. Ed. by Fedor V. Fomin and Vladimir V. Podolskii. Vol. 10846. Lecture Notes in Computer Science. Springer, 2018, pp. 181–193. DOI: 10.1007/978-3-319-90530-3_16.

[46] Michal Hospodár, Galina Jirásková, and Peter Mlynárčik. "Nondeterministic Complexity of Operations on Closed and Ideal Languages". In: *Implementation and Application of Automata - 21st International Conference, CIAA 2016, Seoul, South Korea, July 19-22, 2016, Proceedings*. Ed. by Yo-Sub Han and Kai Salomaa. Vol. 9705. Lecture Notes in Computer Science. Springer, 2016, pp. 125–137. DOI: 10.1007/978-3-319-40946-7_11.

[47] Michal Hospodár, Galina Jirásková, and Peter Mlynárčik. "Nondeterministic Complexity of Operations on Free and Convex Languages". In: *Implementation and Application of Automata - 22nd International Conference, CIAA 2017, Marne-la-Vallée, France, June 27-30, 2017, Proceedings*. Ed. by Arnaud Carayol and Cyril Nicaud. Vol. 10329. Lecture Notes in Computer Science. Springer, 2017, pp. 138–150. DOI: 10.1007/978-3-319-60134-2_12.

[48] Michal Hospodár, Galina Jirásková, and Peter Mlynárčik. "A Survey on Fooling Sets as Effective Tools for Lower Bounds on Nondeterministic Complexity". In: *Adventures Between Lower Bounds and Higher Altitudes - Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*. Ed. by Hans-Joachim Böckenhauer, Dennis Komm, and Walter Unger. Vol. 11011. Lecture Notes in Computer Science. Springer, 2018, pp. 17–32. DOI: 10.1007/978-3-319-98355-4_2.

[49] Michal Hospodár, Galina Jirásková, and Peter Mlynárčik. "Nondeterministic complexity in subclasses of convex languages". In: *Theoret. Comput. Sci.* (2019). DOI: 10.1016/j.tcs.2018.12.027.

[50] Marek Hricko. "Finite automata, regular languages, and state complexity". Master's thesis. PF UPJŠ, Košice, 2005.

[51] Marek Hricko, Galina Jirásková, and Alexander Szabari. "Union and intersection of regular languages and descriptional complexity". In: *7th International Workshop on Descriptional Complexity of Formal Systems - DCFS 2005, Como, Italy, June 30 - July 2, 2005. Proceedings*. Ed. by Carlo Mereghetti et al. Università degli Studi di Milano, 2005, pp. 170–181.

[52] Juraj Hromkovič. *Communication Complexity and Parallel Computing*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 1997. DOI: 10.1007/978-3-662-03442-2.

[53] Kazuo Iwama, Yahiko Kambayashi, and Kazuya Takaki. "Tight bounds on the number of states of DFAs that are equivalent to n-state NFAs". In: *Theoret. Comput. Sci.* 237.1-2 (2000), pp. 485–494. DOI: 10.1016/S0304-3975(00)00029-3.

[54] Kazuo Iwama, Akihiro Matsuura, and Mike Paterson. "A family of NFA's which need $2^n-\alpha$ deterministic states". In: *Mathematical Foundations of Computer Science 2000, 25th International Symposium, MFCS 2000, Bratislava, Slovakia, August 28 - September 1, 2000, Proceedings*. Ed. by Mogens Nielsen and Branislav Rovan. Vol. 1893. Lecture Notes in Computer Science. Springer, 2000, pp. 436–445. DOI: 10.1007/3-540-44612-5_39.

[55] Jozef Jirásek and Galina Jirásková. "The Exact Complexity of Star-Complement-Star". In: *Implementation and Application of Automata - 23rd International Conference, CIAA 2018, Charlottetown, PE, Canada, July 30 - August 2, 2018, Proceedings*. Ed. by Cezar Câmpeanu. Vol. 10977. Lecture Notes in Computer Science. Springer, 2018, pp. 223–235. DOI: 10.1007/978-3-319-94812-6_19.

[56] Jozef Jirásek, Galina Jirásková, and Alexander Szabari. "State complexity of concatenation and complementation". In: *Internat. J. Found. Comput. Sci.* 16.3 (2005), pp. 511–529. DOI: 10.1142/S0129054105003133.

[57] Galina Jirásková. "State complexity of some operations on binary regular languages". In: *Theoret. Comput. Sci.* 330.2 (2005), pp. 287–298. DOI: 10.1016/j.tcs.2004.04.011.

[58] Galina Jirásková. "Magic numbers and ternary alphabet". In: *Internat. J. Found. Comput. Sci.* 22.2 (2011), pp. 331–344. DOI: 10.1142/S0129054111008076.

[59] Galina Jirásková. "Descriptional Complexity of Operations on Alternating and Boolean Automata". In: *CSR 2012*. Ed. by Edward A. Hirsch et al. Vol. 7353. Lecture Notes in Computer Science. Springer, 2012, pp. 196–204. DOI: 10.1007/978-3-642-30642-6_19.

[60] Galina Jirásková. "The Ranges of State Complexities for Complement, Star, and Reversal of Regular Languages". In: *Internat. J. Found. Comput. Sci.* 25.1 (2014), p. 101. DOI: 10.1142/S0129054114500063.

[61] Galina Jirásková and Tomáš Masopust. "Complexity in union-free regular languages". In: *Internat. J. Found. Comput. Sci.* 22.7 (2011), pp. 1639–1653. DOI: 10.1142/S0129054111008933.

[62] Galina Jirásková and Peter Mlynárčik. "Complement on Prefix-Free, Suffix-Free, and Non-Returning NFA Languages". In: *Descriptional Complexity of Formal Systems - 16th International Workshop, DCFS 2014, Turku, Finland, August 5-8, 2014. Proceedings*. Ed. by Helmut Jürgensen, Juhani Karhumäki, and Alexander Okhotin. Vol. 8614. Lecture Notes in Computer Science. Springer, 2014, pp. 222–233. DOI: 10.1007/978-3-319-09704-6_20.

[63] Galina Jirásková and Benedek Nagy. "On Union-Free and Deterministic Union-Free Languages". In: *Theoretical Computer Science - 7th IFIP TC 1/WG 2.2 International Conference, TCS 2012, Amsterdam, The Netherlands, September 26-28, 2012. Proceedings*. Ed. by Jos C. M. Baeten, Thomas Ball, and Frank S. de Boer. Vol. 7604. Lecture Notes in Computer Science. Springer, 2012, pp. 179–192. DOI: 10.1007/978-3-642-33475-7_13.

[64] Galina Jirásková and Alexander Okhotin. "State complexity of cyclic shift". In: *7th International Workshop on Descriptional Complexity of Formal Systems - DCFS 2005, Como, Italy, June 30 - July 2, 2005. Proceedings*. Ed. by Carlo Mereghetti et al. Università degli Studi di Milano, Milan, Italy, 2005, pp. 182–193.

[65] Galina Jirásková, Matúš Palmovský, and Juraj Šebej. "Kleene closure on regular and prefix-free languages". In: *CIAA 2014*. Ed. by Markus Holzer and Martin Kutrib. Vol. 8587. Lecture Notes in Computer Science. Springer, 2014, pp. 226–237. DOI: 10.1007/978-3-319-08846-4_17.

[66] Galina Jirásková, Alexander Szabari, and Juraj Šebej. "The Complexity of Languages Resulting from the Concatenation Operation". In: *Journal of Automata, Languages and Combinatorics* 22.1-3 (2017), pp. 123–143. DOI: 10.25596/jalc-2017-123.

[67] Galina Jirásková and Juraj Šebej. "Reversal of binary regular languages". In: *Theoret. Comput. Sci.* 449 (2012), pp. 85–92. DOI: 10.1016/j.tcs.2012.05.008.

[68] Jozef Jirásek Jr., Matúš Palmovský, and Juraj Šebej. "Kuratowski Algebras Generated by Factor-, Subword-, and Suffix-Free Languages". In: *DCFS 2017*. Ed. by Giovanni Pighizzini and Cezar Câmpeanu. Vol. 10316. Lecture Notes in Computer Science. Springer, 2017, pp. 189–201. DOI: 10.1007/978-3-319-60252-3_15.

[69] Helmut Jürgensen and Rogério Reis, eds. *Descriptional Complexity of Formal Systems - 15th International Workshop, DCFS 2013, London, ON, Canada, July 22-25, 2013. Proceedings*. Vol. 8031. Lecture Notes in Computer Science. Springer, 2013. DOI: 10.1007/978-3-642-39310-5.

[70] Ivana Krajňáková and Galina Jirásková. "Square on Deterministic, Alternating, and Boolean Finite Automata". In: *DCFS 2017*. Ed. by Giovanni Pighizzini and Cezar Câmpeanu. Vol. 10316. Lecture Notes in Computer Science. Springer, 2017, pp. 214–225. DOI: 10.1007/978-3-319-60252-3_17.

[71] Ernst L. Leiss. "On Generalized Language Equations". In: *Theoret. Comput. Sci.* 14 (1981), pp. 63–77. DOI: 10.1016/0304-3975(81)90005-0.

[72] Ernst L. Leiss. "Succint Representation of Regular Languages by Boolean Automata". In: *Theoret. Comput. Sci.* 13.3 (1981), pp. 323–330. DOI: 10.1016/S0304-3975(81)80005-9.

[73] Oleg B. Lupanov. "A comparison of two types of finite automata". In: *Problemy Kibernetiki* 9 (1963). pp. 321–326 (in Russian). German translation: Über den Vergleich zweier Typen endlicher Quellen. Probleme der Kybernetik 6, pp. 328–335 (1966).

[74] A. N. Maslov. "Estimates of the number of states of finite automata". In: *Dokl. Akad. Nauk SSSR* 194.6 (1970). pp. 1266–1268 (in Russian). English translation: Soviet Math. Doklady 11, pp. 1373–1375 (1970).

[75] Albert R. Meyer and Michael J. Fischer. "Economy of description by automata, grammars, and formal systems". In: *Proceedings of the 12th Annual Symposium on Switching and Automata Theory*. IEEE Computer Society Press, 1971, pp. 188–191. DOI: 10.1109/T-C.1971.223108.

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

[77] Peter Mlynárčik. "Complement on Free and Ideal Languages". In: *DCFS 2015*. Ed. by Jeffrey Shallit and Alexander Okhotin. Vol. 9118. Lecture Notes in Computer Science. Springer, 2015, pp. 185–196. DOI: 10.1007/978-3-319-19225-3_16.

[78] Peter Mlynárčik. "Nondeterministic state complexity in subregular classes". Dissertation thesis. FMFI UK, Bratislava, 2017. URL: http://im.saske.sk/~jiraskov/students/phd_thesis_mlynarcik.pdf.

[79] Frank R. Moore. "On the bounds for state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way finite automata". In: *IEEE Transaction on Computing* C-20 (1971), pp. 1211–1219. DOI: 10.1109/T-C.1971.223108.

[80] Cyril Nicaud. "Average State Complexity of Operations on Unary Automata". In: *Mathematical Foundations of Computer Science 1999, 24th International Symposium, MFCS'99, Szklarska Poręba, Poland, September 6-10, 1999, Proceedings*. Ed. by Miroslaw Kutylowski, Leszek Pacholski, and Tomasz Wierzbicki. Vol. 1672. Lecture Notes in Computer Science. Springer, 1999, pp. 231–240. DOI: 10.1007/3-540-48340-3_21.

[81] Giovanni Pighizzini and Cezar Câmpeanu, eds. *Descriptional Complexity of Formal Systems - 19th IFIP WG 1.02 International Conference, DCFS 2017, Milano, Italy, July 3-5, 2017, Proceedings*. Vol. 10316. Lecture Notes in Computer Science. Springer, 2017. DOI: 10.1007/978-3-319-60252-3.

[82] Giovanni Pighizzini and Jeffrey Shallit. "Unary language operations, state complexity and Jacobsthal's function". In: *Internat. J. Found. Comput. Sci.* 13.1 (2002), pp. 145–159. DOI: 10.1142/S012905410200100X.

[83] Michael O. Rabin and Dana Scott. "Finite Automata and Their Decision Problems". In: *IBM J. Res. Dev.* 3 (1959), pp. 114–125. DOI: 10.1147/rd.32.0114.

[84] William J. Sakoda and Michael Sipser. "Nondeterminism and the Size of Two Way Finite Automata". In: *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*. Ed. by Richard J. Lipton et al. ACM, 1978, pp. 275–286. DOI: 10.1145/800133.804357.

[85] Arto Salomaa, Kai Salomaa, and Sheng Yu. "State complexity of combined operations". In: *Theoret. Comput. Sci.* 383.2-3 (2007), pp. 140–152. DOI: 10.1016/j.tcs.2007.04.015.

[86] Kai Salomaa and Sheng Yu. "On the State Complexity of Combined Operations and their Estimation". In: *Internat. J. Found. Comput. Sci.* 18.4 (2007), pp. 683–698. DOI: 10.1142/S0129054107004917.

[87] Juraj Šebej. "Reversal on regular languages and descriptional complexity". In: *Descriptional Complexity of Formal Systems - 15th International Workshop, DCFS 2013, London, ON, Canada, July 22-25, 2013. Proceedings*. Ed. by Helmut Jürgensen and Rogério Reis. Vol. 8031. Lecture Notes in Computer Science. Springer, 2013, pp. 265–276. DOI: 10.1007/978-3-642-39310-5_25.

[88] Jeffrey Shallit and Alexander Okhotin, eds. *Descriptional Complexity of Formal Systems - 17th International Workshop, DCFS 2015, Waterloo, ON, Canada, June 25-27, 2015. Proceedings*. Vol. 9118. Lecture Notes in Computer Science. Springer, 2015. DOI: 10.1007/978-3-319-19225-3.

[89] Michael Sipser. *Introduction to the theory of computation*. PWS Publishing Company, 1997. ISBN: 978-0-534-94728-6.

[90] Alexander Szabari. "Descriptional complexity of regular languages". Dissertation thesis. MÚ SAV, Košice, 2010. URL: http://ekos.esperanto.sk/sano.pdf.

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

[92] Sheng Yu. "Chapter 2: Regular Languages". In: *Handbook of Formal Languages (1)*. Ed. by Grzegorz Rozenberg and Arto Salomaa. Berlin, Heidelberg: Springer, 1997, pp. 41–110. DOI: 10.1007/978-3-642-59136-5_2.

[93] Sheng Yu and Qingyu Zhuang. "On the state complexity of intersection of regular languages". In: *SIGACT News* 22.3 (1991), pp. 52–54.

[94] Sheng Yu, Qingyu Zhuang, and Kai Salomaa. "The state complexities of some basic operations on regular languages". In: *Theoret. Comput. Sci.* 125.2 (1994), pp. 315–328. DOI: 10.1016/0304-3975(92)00011-F.

Appendix

The list of my published papers

Journal papers:

[A] Michal Hospodár, Galina Jirásková: The complexity of concatenation on deterministic and alternating finite automata. *RAIRO – Theoretical Informatics and Applications* 52, pp. 153–168 (2018). ISSN: 0988-3754, DOI: 10.1051/ita/2018011

[B] Michal Hospodár, Galina Jirásková, Peter Mlynárčik: Descriptional complexity of the forever operator. *International Journal of Foundations of Computer Science* 30(1), pp. 115–134 (2019). ISSN: 0129-0541, DOI: 10.1142/S0129054119400069

[C] Michal Hospodár, Galina Jirásková, Peter Mlynárčik: Nondeterministic complexity in subclasses of convex languages. *Theoretical Computer Science* (2019, to appear). ISSN: 0304-3975, DOI: 10.1016/j.tcs.2018.12.027

Conference papers and festschrifts:

[D] Michal Hospodár, Galina Jirásková, Peter Mlynárčik: Nondeterministic Complexity of Operations on Closed and Ideal Languages. In: Yo-Sub Han, Kai Salomaa (eds): *Implementation and Application of Automata – 21st International Conference, CIAA 2016, Seoul, South Korea, July 19–22, 2016, Proceedings.* Lecture Notes in Computer Science, vol. 9705, Springer, pp. 125–137 (2016). ISBN: 978-3-319-40945-0, DOI: 10.1007/978-3-319-40946-7_11

[E] Michal Hospodár, Galina Jirásková: Concatenation on deterministic and alternating automata. In: Henning Bordihn, Rudolf Freund, Benedek Nagy, and György Vaszil (eds.): *Eighth Workshop on Non-Classical Models of Automata and Applications, NCMA 2016, Debrecen, Hungary, August 29–30, 2016. Proceedings.* books@ocg.at,

vol. 321, Wien: Österreichische Computer Gesellschaft, pp. 179–194 (2016). ISBN: 978-3-903035-10-2

[F] Michal Hospodár: Complexity of unary union-free and unary star-free languages. In: Henning Bordihn, Rudolf Freund, Benedek Nagy, and György Vaszil (eds.): *Eighth Workshop on Non-Classical Models of Automata and Applications, NCMA 2016, Debrecen, Hungary, August 29–30, 2016. Short Papers.* Wien: Institut für Computersprachen TU Wien, pp. 15–24 (2016). ISBN: 978-3-200-04725-9

[G] Michal Hospodár, Galina Jirásková, Peter Mlynárčik: Nondeterministic Complexity of Operations on Free and Convex Languages. In: Arnaud Carayol, Cyril Nicaud (eds.): *Implementation and Application of Automata – 22nd International Conference, CIAA 2017, Marne-la-Vallée, France, June 27–30, 2017, Proceedings.* Lecture Notes in Computer Science, vol. 10329, Springer, pp. 138–150 (2017). ISBN: 978-3-319-60133-5, DOI: 10.1007/978-3-319-60134-2_12

[H] Michal Hospodár, Galina Jirásková, Peter Mlynárčik: On the Descriptive Complexity of $\overline{\Sigma^*\overline{L}}$. In: Émilie Charlier, Julien Leroy, Michel Rigo (eds.): *Developments in Language Theory – 21st International Conference, DLT 2017, Liège, Belgium, August 7–11, 2017, Proceedings.* Lecture Notes in Computer Science, vol. 10396, Springer, pp. 222–234 (2017). ISBN: 978-3-319-62808-0, DOI: 10.1007/978-3-319-62809-7_16

[I] Michal Hospodár, Galina Jirásková, Peter Mlynárčik: A Survey on Fooling Sets as Effective Tools for Lower Bounds on Nondeterministic Complexity. In: Hans-Joachim Böckenhauer, Dennis Komm, Walter Unger (eds.): *Adventures Between Lower Bounds and Higher Altitudes – Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday.* Lecture Notes in Computer Science, vol. 11011, Springer, pp. 17–32 (2018). ISBN: 978-3-319-98354-7, DOI: 10.1007/978-3-319-98355-4_2

[J] 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, Springer, pp. 181–193 (2018). ISBN: 978-3-319-90529-7, DOI: 10.1007/978-3-319-90530-3_16

[K] Michal Hospodár, Markus Holzer: The Ranges of Accepting State Complexities of Languages Resulting From Some Operations. In: Cezar Câmpeanu (ed.): *Implementation and Application of Automata – 23rd International Conference, CIAA 2018,*

Charlottetown, PE, Canada, July 30 – August 2, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10977, Springer, pp. 198–210 (2018). ISBN: 978-3-319-94811-9, DOI: 10.1007/978-3-319-94812-6_17

[L] Markus Holzer, Michal Hospodár: Complexity of languages resulting from the cut operation in the unary case. In: Rudolf Freund, Michal Hospodár, Galina Jirásková, and Giovanni Pighizzini (eds.): *Tenth Workshop on Non-Classical Models of Automata and Applications, NCMA 2018, Košice, Slovakia, August 21–22, 2018. Short Papers.* Wien, pp. 25–34 (2018).

[M] Michal Hospodár, Matúš Palmovský: Nondeterministic complexity of power and positive closure on subclasses of convex languages. In: Rudolf Freund, Michal Hospodár, Galina Jirásková, and Giovanni Pighizzini (eds.): *Tenth Workshop on Non-Classical Models of Automata and Applications, NCMA 2018, Košice, Slovakia, August 21–22, 2018. Short Papers.* Wien, pp. 35–44 (2018).

[N] Markus Holzer, Michal Hospodár: The Range of State Complexities of Languages Resulting from the Cut Operation. In: Carlos Martín-Vide, Alexander Okhotin, Dana Shapira (eds.): *Language and Automata Theory and Applications – 13th International Conference, LATA 2019, Saint Petersburg, Russia, March 26–29, 2019, Proceedings.* Lecture Notes in Computer Science, vol. 11417, Springer, pp. 190–202 (2019). ISBN: 978-3-030-13434-1, DOI: 10.1007/978-3-030-13435-8_14

Abstracts:

[O] Michal Hospodár: The Story of Concatenation. In: Ján Buša, Jozef Doboš (eds.): *18. Konferencia košických matematikov, Herľany, 20.–22. apríla 2017.* Košice: JSMF, PF UPJŠ, FEI TUKE, pp. 20–21 (2017). ISBN 978-80-553-3146-1
`http://people.tuke.sk/jan.busa/JSMF/Herlany2017BOA.pdf`

[P] Markus Holzer, Michal Hospodár: The Range of State Complexities of Languages Resulting from the Cut Operation. In: *DLT's Satellite workshop in Kyoto, In honor of Masami Ito's KIJU and Pál Dömösi's 75th birthday. ABSTRACTS OF THE TALKS.* Kyoto Sangyo University, September 5-7, 2018, p. 12.
`https://satellitedlt.sciencesconf.org/data/pages/Proceedings_3.pdf`

[Q] Michal Hospodár, Matúš Palmovský: Nondeterministic Complexity of Power and Positive Closure on Subclasses of Convex Languages. In: *DLT's Satellite workshop in Kyoto, ABSTRACTS OF THE TALKS.* 2018, p. 14.

The list of given talks

1. Concatenation on deterministic and alternating automata. *8th International Workshop on Non-Classical Models of Automata and Applications, NCMA 2016.* Debrecen, Hungary, August 30, 2016.

2. Complexity of unary union-free and unary star-free languages. *8th International Workshop on Non-Classical Models of Automata and Applications, NCMA 2016.* Debrecen, Hungary, August 30, 2016.

3. The story of concatenation.

   (a) *18. Konferencia košických matematikov.* Herľany, Slovakia, April 20, 2017.

   (b) *Second Workshop on Černý's Conjecture and Optimization Problems on Finite Automata.* Opava, Czech Republic, May 17, 2017.

   (c) *Dissertation exam.* Bratislava, Slovakia, May 25, 2017.

4. On the descriptive complexity of $\overline{\Sigma^*\overline{L}}$. *21st International Conference on Developments in Language Theory, DLT 2017.* Liège, Belgium, August 8, 2017.

5. Operations on Boolean and alternating finite automata. *13th International Computer Science Symposium in Russia, CSR 2018.* Moscow, Russia, June 7, 2018.

6. Nondeterministic complexity of L^k and L^+ on subclasses of convex languages.

   (a) *10th International Workshop on Non-Classical Models of Automata and Applications, NCMA 2018.* Košice, Slovakia, August 22, 2018.

   (b) *DLT's Satellite Workshop in Kyoto.* Kyoto, Japan, September 6, 2018.

7. Complexity of languages resulting from the cut operation in the unary case. *10th International Workshop on Non-Classical Models of Automata and Applications, NCMA 2018.* Košice, Slovakia, August 22, 2018.

8. The range of state complexities of languages resulting from the cut operation.

   (a) *DLT's Satellite Workshop in Kyoto.* Kyoto, Japan, September 6, 2018.

   (b) *Second Workshop on Efficient Algorithms, Automata and Data Structures, EAADS 2018.* Košice, Slovakia, December 4, 2018.

   (c) *13th International Conference on Language and Automata Theory and Applications, LATA 2019.* Saint Petersburg, Russia, March 27, 2019.

Appendix [A]

Michal Hospodár, Galina Jirásková:
The complexity of concatenation on deterministic and alternating finite automata.
RAIRO - Theoretical Informatics and Applications, Vol. 52, pp. 153–168 (2018).
ISSN: 0988-3754, DOI: 10.1051/ita/2018011

RAIRO-Theor. Inf. Appl. 52 (2018) 153–168
https://doi.org/10.1051/ita/2018011

RAIRO - Theoretical Informatics and Applications
www.rairo-ita.org

THE COMPLEXITY OF CONCATENATION ON DETERMINISTIC AND ALTERNATING FINITE AUTOMATA[☆]

Michal Hospodár[*] and Galina Jirásková

Abstract. We study the state complexity of the concatenation operation on regular languages represented by deterministic and alternating finite automata. For deterministic automata, we show that the upper bound $m2^n - k2^{n-1}$ on the state complexity of concatenation can be met by ternary languages, the first of which is accepted by an m-state DFA with k final states, and the second one by an n-state DFA with ℓ final states for arbitrary integers m, n, k, ℓ with $1 \leq k \leq m-1$ and $1 \leq \ell \leq n-1$. In the case of $k \leq m-2$, we are able to provide appropriate binary witnesses. In the case of $k = m-1$ and $\ell \geq 2$, we provide a lower bound which is smaller than the upper bound just by one. We use our binary witnesses for concatenation on deterministic automata to describe binary languages meeting the upper bound $2^m + n + 1$ for the concatenation on alternating finite automata. This solves an open problem stated by Fellah *et al.* [*Int. J. Comput. Math.* **35** (1990) 117–132].

Mathematics Subject Classification. 68Q19, 68Q45.

Accepted November 21, 2018.

1. Introduction

Concatenation is a binary operation on formal languages defined as $KL = \{uv \mid u \in K \text{ and } v \in L\}$. It is known that if a language K is accepted by an m-state deterministic finite automaton (DFA) and L is accepted by an n-state DFA, then the concatenation KL is accepted by a DFA of at most $m2^n - 2^{n-1}$ states [10].

Ternary languages meeting this upper bound were described by Yu *et al.* [14]. Maslov [10] proposed binary witnesses for concatenation, but he did not provide any proof. The tightness of this upper bound in the binary case was proven in [6].

However, if the minimal DFA recognizing the first language has more than one final state, then the upper bound $m2^n - 2^{n-1}$ on the state complexity of concatenation cannot be met; here, the state complexity of a regular language is the number of states in the minimal DFA for the language, and the state complexity of a regular operation is the number of states that are sufficient and necessary in the worst case for a DFA to recognize the language resulting from the operation considered as a function of the state complexities of the operands. Yu *et al.* [14] showed that the state complexity of concatenation is at most $m2^n - k2^{n-1}$, where k is the number of final states in the minimal DFA for the first language. The binary languages meeting this upper bound were described for each k with $1 \leq k \leq m-1$ in Theorem 1 of [5], but there are some errors in the proof

[☆] Research supported by VEGA grant 2/0084/15 and grant APVV-15-0091. This work was conducted as a part of PhD study of the first author at Comenius University in Bratislava.

Keywords and phrases: Regular languages, finite automata, concatenation, state complexity.

Mathematical Institute, Slovak Academy of Sciences, Grešákova 6, 040 01 Košice, Slovakia.

[*] Corresponding author: hosmich@gmail.com

Article published by EDP Sciences

of this theorem, and one of our aims is to fix them. We also show that the witnesses from [10, 14] meet the upper bound $m2^n - k2^{n-1}$ if we make the k last states final in the DFA for the first language.

Then we study the complexity of concatenation also in the case where the second automaton has more than one final state. Our motivation comes from the paper by Fellah *et al.* [3], where the authors consider the concatenation operation on languages represented by alternating finite automata (AFA), and get an upper bound $2^m + n + 1$. They also write: *"We conjecture that this number of states is actually necessary in the worst case, but have no proof."*

It is known ([3], Thm 4.1, Cor. 4.2) and ([7], Lem. 1, Lem. 2) that a language L is accepted by an n-state AFA if and only if its reversal L^R is accepted by a 2^n-state DFA with 2^{n-1} states final. Hence to get a lower bound for concatenation on AFAs, we need two languages represented by DFAs with half of states final that are hard for concatenation on DFAs.

We first inspect the witnesses from [5, 10, 14] and show that none of them meets the upper bound $m2^n - k2^{n-1}$ if the second automaton has more then one final state. Then we describe ternary languages meeting this bound for all m, n, k, ℓ, where m and k is the number of states and the number of final states in the minimal DFA for the first language, and n and ℓ is the number of states and the number of final states in the minimal DFA for the second language. Then, in the case of $k \leq m - 2$, that is, if the first automaton has at least two non-final states, we describe appropriate binary languages. Finally, we consider the case of $k = m - 1$ and $\ell \geq 2$ over a binary alphabet. In such a case, the upper bound is $(m + 1)2^{n-1}$, and we provide languages meeting the bound $(m + 1)2^{n-1} - 1$. We strongly conjecture that this lower bound is tight, but have no proof.

We use the binary witnesses for the concatenation on DFAs to define binary languages K and L accepted by an m-state and n-state AFA, respectively, such that the minimal AFA for KL requires $2^m + n + 1$ states. This proves that the upper bound $2^m + n + 1$ from [3] is tight, and solves the open problem stated in Theorem 9.3 of [3].

2. Preliminaries

In this section, we give some basic definitions and preliminary results. For details and all unexplained notions, the reader may refer to [4, 12, 13].

Let Σ be a finite alphabet of symbols. Then Σ^* denotes the set of strings over Σ including the empty string 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 cardinality of a finite set A is denoted by $|A|$, and its power-set by 2^A. We define an operator $\ominus$ as follows: If $i, j \in \{0, 1, \ldots, n-1\}$, then $j \ominus i = (j - i) \bmod n$, and if $S \subseteq \{0, 1, \ldots, n-1\}$, then $S \ominus i = \{j \ominus i \mid j \in S\}$.

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

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

The reversal L^R of a language L is defined as $L^R = \{w^R \mid w \in L\}$, where w^R is the mirror image of the string w. For every finite automaton $N = (Q, \Sigma, \cdot, I, F)$ we can construct the automaton $N^R = (Q, \Sigma, \cdot^R, F, I)$ where $p \in q \cdot^R a$ iff $q \in p \cdot a$ for every p, q in Q and every a in Σ. Then $L(N^R) = (L(N))^R$.

Every NFA $N = (Q, \Sigma, \cdot, I, F)$ can be converted into an equivalent DFA $D = (2^Q, \Sigma, \cdot', I, F')$ where $F' = \{S \in 2^Q \mid S \cap F \neq \emptyset\}$, and for every set S in 2^Q and every symbol a, we have $S \cdot' a = S \cdot a$ [11]. The DFA D is called the *subset automaton* of the NFA N. The subset automaton may not be minimal since some of its states may be unreachable or equivalent to other states.

We say that a state q of an NFA $N = (Q, \Sigma, \cdot, I, F)$ is *uniquely distinguishable* if there is a string w which is accepted by N from and only from the state q, that is, if we have $p \cdot w \cap F \neq \emptyset$ iff $p = q$.

Proposition 2.1. *If two subsets of states of an* NFA *differ in a uniquely distinguishable state, then the two subsets are distinguishable in the subset automaton .*

Proof. Let S and T be two subsets of states of an NFA N. Let q be a uniquely distinguishable state of N such that, without loss of generality, $q \in S \setminus T$. Then there is a string w which is accepted by N from and only from q. It follows that w is accepted by the subset automaton of N from S and rejected from T. Hence S and T are distinguishable in the subset automaton of N. □

We say that a transition (p, a, q) is a *unique in-transition* in an NFA N if there is no state r with $r \neq p$ such that (r, a, q) is a transition in N. We say that a state q is *uniquely reachable* from a state p if there is a sequence of unique in-transitions (q_{i-1}, a_i, q_i) for $i = 1, 2, \ldots, k$ such that $k \geq 1$, $q_0 = p$, and $q_k = q$.

Proposition 2.2. *Let a uniquely distinguishable state q be uniquely reachable from a state p. Then the state p is uniquely distinguishable.*

Proof. Let a string w be accepted by an NFA N from and only from a state q. If (p, a, q) is a unique in-transition, then the string aw is accepted by N from and only from the state p. Now the claim follows by induction. □

3. Construction of NFA for concatenation

Let K and L be accepted by minimal DFAs A and B, respectively. Without loss of generality, we may assume that the state set of A is $\{q_0, q_1, \ldots, q_{m-1}\}$ with the initial state q_0, and the state set of B is $\{0, 1, \ldots, n-1\}$ with the initial state 0. Moreover, we denote the transition function in both A and B by $\cdot$; there is no room for confusion since A and B have distinct state sets. We first recall the construction of an NFA for the concatenation of languages K and L.

Construction 3.1. (DFA A *and* DFA $B \to$ NFA N *for* $L(A)L(B)$).
Let $A = (\{q_0, q_1, \ldots, q_{m-1}\}, \Sigma, \cdot, q_0, F_A)$ *and* $B = (\{0, 1, \ldots, n-1\}, \Sigma, \cdot, 0, F_B)$ *be* DFA*s. We construct* NFA $N = (\{q_0, q_1, \ldots, q_{m-1}\} \cup \{0, 1, \ldots, n-1\}, \Sigma, \cdot, I, F_B)$ *from* DFA*s* A *and* B *as follows:*

- *for each* a *in* Σ *and each state* q_i *of* A*, if* $q_i \cdot a \in F_A$*, then add the transition* $(q_i, a, 0)$*;*
- *the set* I *of initial states of* N *is* $\{q_0\}$ *if* $q_0 \notin F_A$*, and it is* $\{q_0, 0\}$ *otherwise;*
- *the set of final states of* N *is* F_B*.*

Using Construction 3.1, we get an upper bound on the state complexity of concatenation. Notice that the bound depends on the number of final states in the DFA for the first language.

Proposition 3.2 (**Concatenation: Upper Bound if $|F_A| = k$**). *Let A be an m-state* DFA *with k final states and let B be an n-state* DFA*. Then we have* $\mathrm{sc}(L(A)L(B)) \leq m2^n - k2^{n-1}$.

Proof. Consider DFAs $A = (\{q_0, q_1, \ldots, q_{m-1}\}, \Sigma, \cdot, q_0, F_A)$, where $|F_A| = k$, and $B = (\{0, 1, \ldots, n-1\}, \Sigma, \cdot, 0, F_B)$. Construct an NFA N for $L(A)L(B)$ as in the construction above, and consider the corresponding subset automaton D. Since A is deterministic and complete, each reachable subset in D is of the form $\{q_i\} \cup S$, where $S \subseteq \{0, 1, \ldots, n-1\}$. Moreover, if q_i is a final state of A, then $0 \in S$ since the NFA N has the transition $(q, a, 0)$ whenever a state q of A goes to a final state q_i on a symbol a. If q_0 is final, then D starts in $\{q_0, 0\}$. It follows that no subset containing a final state of A and not containing state 0 is reachable in D. Hence the subset automaton D has at most $m2^n - k2^{n-1}$ reachable subsets. □

Since $m2^n - k2^{n-1}$ is maximal if $k = 1$, we get the following upper bound on the state complexity of concatenation [10, 14].

Corollary 3.3 (**Concatenation: Upper Bound**). *Let A and B be an m-state and n-state* DFA*, respectively. Then* $\mathrm{sc}(L(A)L(B)) \leq m2^n - 2^{n-1}$. □

4. Ternary and binary witness languages

Motivated by the open problem from [3] concerning the tightness of the upper bound $2^m + n + 1$ for concatenation on alternating automata, we study the state complexity of the concatenation of languages represented by deterministic finite automata that have more than one final state. Let us start with the following observation in which we assume that the state complexity of the second language is one.

Observation 4.1. *Let $m \geq 1$ and $1 \leq k \leq m$. Let A be an m-state* DFA *with k final states and B be a 1-state* DFA*, both over an alphabet Σ. Then $\mathrm{sc}(L(A)L(B)) \leq m - k + 1$, and the bound is tight if $|\Sigma| \geq 1$.*

Proof. If a complete DFA B has one state, then either $L(B) = \emptyset$ or $L(B) = \Sigma^*$. Since $L(A)\,\emptyset = \emptyset$, and hence $\mathrm{sc}(L(A)\,\emptyset) = 1$, we assume that $L(B) = \Sigma^*$. We construct the DFA for $L(A)L(B)$ from A as follows: for every final state p and every a in Σ, we replace the transition (p, a, q) by the transition (p, a, p). The resulting automaton is deterministic and complete, has m states and k final states. All the final states are equivalent since every string is accepted from any of them. Thus we can merge all final states into a single final state. This gives the upper bound $m - k + 1$.

To prove tightness, let us consider the unary deterministic finite automaton $A = (\{0, 1, \ldots, m-1\}, \{a\}, \cdot, 0, \{q \mid m - k \leq q \leq m - 1\})$, where $q \cdot a = q + 1 \bmod m$ for $q = 0, 1, \ldots, m - 1$. For each final state p, we remove all the transitions going from p, and add the transition (p, a, p) to get a DFA for $L(A)\,\Sigma^*$. Then we merge all final states into a single final state. The resulting minimal automaton accepts the language $a^{m-k}a^*$ and has $m - k + 1$ states. □

In what follows, we assume that the state complexity of the second language is at least two. We inspect three worst-case examples from the literature, and modify them by making some states in the first automaton final. To simplify the proofs, we use the property of all these witnesses that the letter a performs the permutation $q_i \cdot a = q_{(i+1) \bmod m}$ in A and a permutation in B. If these two conditions are satisfied, then we get the following observation.

Lemma 4.2. *Let $A = (\{q_0, q_1, \ldots, q_{m-1}\}, \Sigma, \cdot, q_0, \{q_i \mid m - k \leq i \leq m - 1\})$ and $B = (Q_B, \Sigma, \cdot, 0, F_B)$, where $Q_B = \{0, 1, \ldots, n - 1\}$, be* DFA*s. Assume that there is a symbol a in Σ such that $q_i \cdot a = q_{(i+1) \bmod m}$ and the symbol a performs a permutation on Q_B. Let N be an* NFA *for $L(A)L(B)$ from Construction 3.1. Then in the subset automaton of N, we have*

1. *For each subset S of Q_B with $0 \in S$, the set $\{q_{m-k}\} \cup S$ is reachable from a set $\{q_{m-k-1}\} \cup S'$, where $S' \subseteq Q_B$ and $|S'| = |S| - 1$;*
2. *For each subset S of Q_B and each $i = 1, 2, \ldots, m - k - 1$, the set $\{q_i\} \cup S$ is reachable from a set $\{q_0\} \cup S'$, where $S' \subseteq Q_B$ and $|S'| = |S|$;*
3. *Moreover, if $0 \cdot a = 0$, then for each subset S of Q_B with $0 \in S$ and for each $i = 0, 1, \ldots, m - 1$, the set $\{q_i\} \cup S$ is reachable from a set $\{q_{m-k-1}\} \cup S'$, where $S' \subseteq Q_B$ and $|S'| = |S| - 1$.*

Proof. Since a is a permutation on Q_B, we can use $q \cdot a^{-1}$ to denote the state p with $p \cdot a = q$. Next, we can extend a^{-1} to subsets of Q_B and to a^{-i} for every positive integer i.

1. Let $S' = (S \setminus \{0\}) \cdot a^{-1}$. Then $|S'| = |S| - 1$ and the set $\{q_{m-k}\} \cup S$ is reached from $\{q_{m-k-1}\} \cup S'$ by a.
2. Let $S' = S \cdot a^{-i}$ where $i = 1, 2, \ldots, m - k - 1$. Then $|S'| = |S|$ and the set $\{q_i\} \cup S$ is reached from $\{q_0\} \cup S'$ by a^i.
3. Let $S' = (S \setminus \{0\}) \cdot a^{-(k+1+i)}$ where $i = 0, 1, \ldots, m - 1$. Then $|S'| = |S| - 1$ and $\{q_i\} \cup S$ is reached from $\{q_{m-k-1}\} \cup S'$ by a^{k+1+i} since $0 \cdot a = 0$. □

Ternary witness languages meeting the upper bound $m2^n - 2^{n-1}$ for concatenation are described in Theorem 2.1 of [14]. We modify these languages by making k states final in the first DFA. Then we prove that the state complexity of the resulting concatenation meets the upper bound $m2^n - k2^{n-1}$.

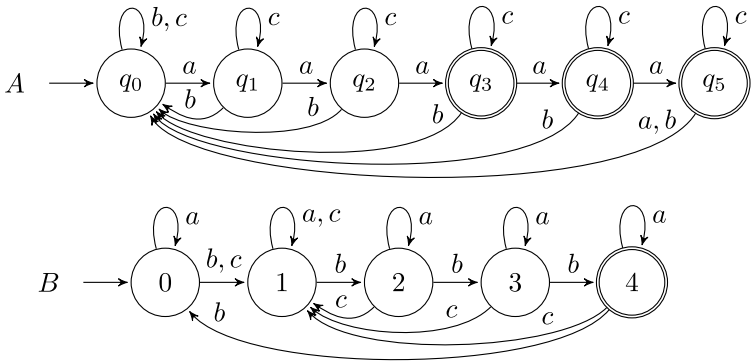


FIGURE 1. Ternary witnesses for concatenation meeting the upper bound $m2^n - k2^{n-1}$; $m = 6$, $k = 3$, and $n = 5$.

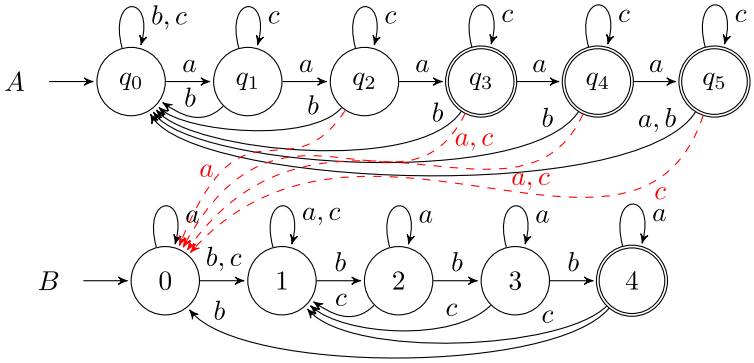


FIGURE 2. An NFA N for $L(A)L(B)$, where DFAs A and B are shown in Figure 1.

Lemma 4.3 (Ternary Witness Automata with $|F_A| = k$ and $|F_B| = 1$). *Let $m, n \geq 2$ and $1 \leq k \leq m-1$. There exist a ternary m-state* DFA *A with k final states and a ternary n-state* DFA *B such that* $\mathrm{sc}(L(A)L(B)) = m2^n - k2^{n-1}$.

Proof. Define an m-state DFA $A = (\{q_0, q_1, \ldots, q_{m-1}\}, \{a, b, c\}, \cdot, q_0, F_A)$, where $F_A = \{q_i \mid m-k \leq i \leq m-1\}$ and for each i in $\{0, 1, \ldots, m-1\}$, we have

$$q_i \cdot a = q_{(i+1) \bmod m}, \qquad q_i \cdot b = q_0, \qquad \text{and} \qquad q_i \cdot c = q_i.$$

Define an n-state DFA $B = (Q_B, \{a, b, c\}, \cdot, 0, \{n-1\})$, where $Q_B = \{0, 1, \ldots, n-1\}$ and for each j in Q_B, we have

$$j \cdot a = j, \qquad j \cdot b = (j+1) \bmod n, \qquad \text{and} \qquad j \cdot c = 1.$$

The DFAs A and B, where $m = 6, k = 3$, and $n = 5$, are shown in Figure 1.

Construct an NFA N for $L(A)L(B)$ from DFAs A and B by adding transitions $(q_{i-1}, a, 0)$ and $(q_i, c, 0)$ for each i with $m-k \leq i \leq m-1$; the initial state of N is q_0, and the set of final states is $\{n-1\}$. Figure 2 shows the NFA N resulting from DFAs A and B from Figure 1. Let $\mathcal{R}$ be the following family of $m2^n - k2^{n-1}$ subsets of states of the NFA N:

$$\mathcal{R} = \big\{\{q_i\} \cup S \mid 0 \le i \le m-k-1 \text{ and } S \subseteq Q_B\big\} \cup \big\{\{q_i\} \cup S \mid m-k \le i \le m-1,\ S \subseteq Q_B \text{ and } 0 \in S\big\}.$$

To prove the lemma, we only need to show that each subset in $\mathcal{R}$ is reachable in the subset automaton of N, and that all these subsets are pairwise distinguishable.

We first prove reachability. The proof is by induction on $|\{q_i\} \cup S|$. The basis, $|\{q_i\} \cup S| = 1$, holds true since $\{q_0\}$ is the initial subset of the subset automaton, and it goes to the subset $\{q_i\}$ by a^i if $1 \le i \le m-k-1$. Let $1 \le t \le n$, and assume that each subset in $\mathcal{R}$ of size t is reachable. Notice that the symbol a performs the permutation $q_i \cdot a = q_{(i+1) \bmod m}$ on states of A and a permutation on states of B and moreover $0 \cdot a = 0$. By Lemma 4.2 case 3, each set $\{q_i\} \cup S$ of size $t+1$, where $m-k \le i \le m-1$ and $S \subseteq Q_B$ with $0 \in S$, can be reached from a set of size t. Next, by Lemma 4.2 case 2, each set $\{q_i\} \cup S$ of size $t+1$ where $1 \le i \le m-k-1$ is reached from a set $\{q_0\} \cup S'$ of size $t+1$. Hence it is enough to show the reachability of sets $\{q_0\} \cup S$ for every subset S of Q_B such that $|\{q_0\} \cup S| = t+1$. We have

$$\{q_{m-1}\} \cup (S \ominus \min S) \cdot a^{-1} \xrightarrow{a} \{q_0\} \cup (S \ominus \min S) \xrightarrow{b^{\min S}} \{q_0\} \cup S,$$

where $0 \in S \ominus \min S$ and the set $\{q_{m-1}\} \cup (S \ominus \min S)$ can be reached from a set of size t by Lemma 4.2 case 3. This proves reachability.

To prove distinguishability, let $\{q_i\} \cup S$ and $\{q_j\} \cup T$ be two distinct subsets in $\mathcal{R}$. Notice that the state $n-1$ is uniquely distinguishable in NFA N since it is a unique final state. Next, the state $n-1$ is reached from each state of Q_B in the subgraph of unique in-transitions $(t, b, t+1)$ where $0 \le t \le n-2$. It follows that each state in Q_B is uniquely distinguishable. By Proposition 2.1, if $S \ne T$, then $\{q_i\} \cup S$ and $\{q_j\} \cup T$ are distinguishable. Now let $S = T$. Then $i \ne j$, and without loss of generality, $0 \le i < j \le m-1$. There are three cases:

1. Let $i < m-k \le j$, that is, q_i is non-final and q_j is final in A. Then $0 \notin (\{q_i\} \cup S) \cdot c$, but $0 \in (\{q_j\} \cup S) \cdot c$, so after reading c, the resulting sets differ in state 0 and are distinguishable as shown above.
2. Let $m-k \le i < j$, that is, both q_i and q_j are final in A. Then we read a^{m-j} and get the sets $\{q_0\} \cup S$ and $\{q_{m-j+i}\} \cup S$ which are considered in case 1.
3. Let $i < j < m-k$, that is, both q_i and q_j are non-final in A. Then we read a^{m-k-j} and get the sets $\{q_{m-k-j+i}\} \cup S$ and $\{q_{m-k}\} \cup \{0\} \cup S$ which either differ in state 0 or are considered in case 1.

This proves distinguishability and concludes our proof. □

Yu *et al.* [14] left the binary case open. Later, a paper by Maslov [10] was found, in which the author describes binary witnesses meeting the upper bound $m2^n - 2^{n-1}$ assuming that $n \ge 3$. Let us show that his witnesses, modified to have k final states in A as shown in Figure 3 for $m = 6, k = 3$, and $n = 5$, meet the upper bound $m2^n - k2^{n-1}$ whenever $n \ge 3$.

Lemma 4.4 (Binary Witnesses with $|F_A| = k$ and $|F_B| = 1$; $n \ge 3$). *Let $m \ge 2$, $n \ge 3$, and $1 \le k \le m-1$. There exist a binary m-state* DFA *A with k final states and a binary n-state* DFA *B such that* $\mathrm{sc}(L(A)L(B)) = m2^n - k2^{n-1}$.

Proof. Define an m-state DFA $A = (\{q_0, q_1, \ldots, q_{m-1}\}, \{a, b\}, \cdot, q_0, F_A)$, where $F_A = \{q_i \mid m-k \le i \le m-1\}$ and for each i in $\{0, 1, \ldots, m-1\}$, we have $q_i \cdot a = q_{(i+1) \bmod m}$ and $q_i \cdot b = q_i$.

Define an n-state DFA $B = (\{0, 1, \ldots, n-1\}, \{a, b\}, \cdot, 0, \{n-1\})$, where for each state j of B, we have $j \cdot a = j$ if $j \le n-3, (n-2) \cdot a = n-1, (n-1) \cdot a = n-2$, and $j \cdot b = j+1$ if $j \le n-2, (n-1) \cdot b = n-1$.

The DFAs A and B, where $m = 6, k = 3$, and $n = 5$, are shown in Figure 3.

Construct an NFA N for $L(A)L(B)$ from DFAs A and B by adding transitions $(q_{i-1}, a, 0)$ and $(q_i, b, 0)$ for each i with $m-k \le i \le m-1$; the initial state of N is q_0, and the set of final states is $\{n-1\}$. Let $\mathcal{R}$ be the same family of $m2^n - k2^{n-1}$ subsets as in the previous proof. We need to show that all sets in $\mathcal{R}$ are reachable

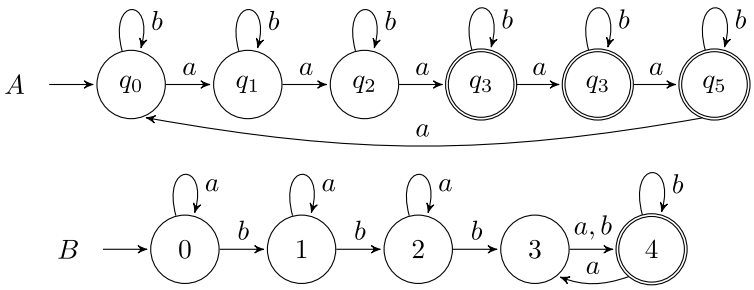


FIGURE 3. Binary witnesses for concatenation meeting the upper bound $m2^n - k2^{n-1}$ assuming that $n \geq 3$; $m = 6$, $k = 3$, $n = 5$.

and pairwise distinguishable in the subset automaton of N. The proof of reachability is exactly the same as in the proof of Lemma 4.3.

To prove distinguishability, let $\{q_i\} \cup S$ and $\{q_j\} \cup T$ be two distinct subsets in $\mathcal{R}$. Notice that the state $n-1$ is uniquely distinguishable since it is a unique final state in N. Next, the state $n-1$ is uniquely reachable from each state in $\{0, 1, \ldots, n-1\}$ through the following unique in-transitions $0 \xrightarrow{b} 1 \xrightarrow{b} \cdots \xrightarrow{b} n-2 \xrightarrow{a} n-1$. It follows that each state in $\{0, 1, \ldots, n-1\}$ is uniquely distinguishable. By Proposition 2.1, if $S \neq T$, then $\{q_i\} \cup S$ and $\{q_j\} \cup T$ are distinguishable. Now let $S = T$. Then $i \neq j$, and without loss of generality, $0 \leq i < j \leq m-1$. There are four cases:

1. Let $i < m - k \leq j$, so $0 \in S$. Then we read b and get $\{q_i\} \cup (S \cdot b)$ and $\{q_j\} \cup \{0\} \cup (S \cdot b)$, which differ in state 0 since $0 \notin S \cdot b$.
2. If $m - k \leq i < j$, then we read a^{m-j} and get $\{q_{m-j+i}\} \cup (S \cdot a^{m-j})$ and $\{q_0\} \cup (S \cdot a^{m-j})$, which are considered in case 1.
3. If $i < j < m - k$ and $0 \in S$, then we read a^{m-k-j} and get $\{q_{m-k-j+i}\} \cup (S \cdot a^{m-k-j})$ and $\{q_{m-k}\} \cup (S \cdot a^{m-k-j})$, which are considered in case 1.
4. If $i < j < m - k$ and $0 \notin S$, then we read a^{m-k-j} and get $\{q_{m-k-j+i}\} \cup (S \cdot a^{m-k-j})$ and $\{q_{m-k}\} \cup \{0\} \cup (S \cdot a^{m-k-j})$, which differ in state 0.

This concludes our proof. □

While the ternary witnesses from Lemma 4.3 require $m \geq 2$ and $n \geq 2$, the binary witnesses from Lemma 4.4 do not work if $n = 2$. In Theorem 1 from [5], binary witnesses for $m \geq 1$ and $n \geq 2$ are described. However, the proof of Theorem 1 from [5] does not work. For example, it is claimed that the set $\{q_{m-k-1}, j_2 - 1, \ldots, j_s - 1\}$ goes to $\{q_{m-k+1}, 0, j_2, \ldots, j_s\}$ by aab^{n-1}; *cf.* line -4 on page 515. In fact it goes to $\{q_{m-k+1}, 0\}$. Such an error occurs several times in the proof, namely, on line -2 on page 515, and on lines 2 and 8 on page 516. The authors overlooked that ab^{n-1} does not perform an identity on $\{0, 1, \ldots, n-1\}$, but moves this set to $\{0\}$. Here we provide a correct proof.

Lemma 4.5 (**[5], Binary Witness Automata with $|F_A| = k$ and $|F_B| = 1$**). *Let $m \geq 1$ and $n \geq 2$. Let $k = 1$ if $m = 1$, and $1 \leq k \leq m-1$ otherwise. There exist a binary m-state* DFA *A with k final states and a binary* DFA *B such that* $\mathrm{sc}(L(A)L(B)) = m2^n - k2^{n-1}$.

Proof. Define an m-state DFA $A = (\{q_0, q_1, \ldots, q_{m-1}\}, \{a, b\}, \cdot, q_0, F_A)$, where $F_A = \{q_i \mid m - k \leq i \leq m-1\}$ and for each i in $\{0, 1, \ldots, m-1\}$, we have $q_i \cdot a = q_{(i+1) \bmod m}$ and $q_i \cdot b = q_i$.

Define an n-state DFA $B = (\{0, 1, \ldots, n-1\}, \{a, b\}, \cdot, 0, \{n-1\})$, where for each state j of B, we have $j \cdot a = (j+1) \bmod n$, $0 \cdot b = 0$, and $j \cdot b = (j+1) \bmod n$ if $j \geq 1$. The DFAs A and B, where $m = 6, k = 3$, and $n = 5$, are shown in Figure 4.

First let $m = 1$, so $L(A) = \{a, b\}^*$. Construct an NFA N for $L(A)L(B)$ from the DFA B by adding the transition $(0, a, 0)$. In the subset automaton of N, the singleton set $\{0\}$ is the initial subset, and each

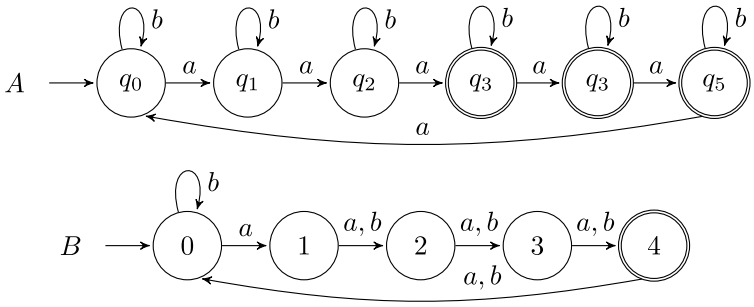


FIGURE 4. Binary witnesses meeting the bound $m2^n - k2^{n-1}$; $m = 6$, $k = 3$, and $n = 5$ [5].

subset S of size $t+1$ such that $0 \in S$ is reached from the subset $(S \setminus \{0\}) \ominus \min(S \setminus \{0\})$ of size t by the string $ab^{\min(S\setminus\{0\})-1}$. Since the state $n-1$ is uniquely distinguishable and uniquely reachable from every other state in $\{0, 1, \ldots, n-1\}$, all the states of the subset automaton of N are pairwise distinguishable by Proposition 2.1. Hence $\mathrm{sc}(L(A)L(B)) = 2^{n-1}$.

Now let $m \geq 2$. Construct an NFA N for $L(A)L(B)$ from DFAs A and B as in the Construction 3.1. Let $\mathcal{R}$ be the same family of $m2^n - k2^{n-1}$ subsets as in the proof of Lemma 4.3. Let us show that each subset $\{q_i\} \cup S$ in $\mathcal{R}$ is reachable in the subset automaton of N. The proof is by induction on $|\{q_i\} \cup S|$.
The basis, with $|\{q_i\} \cup S| \leq 2$, holds true, since we have

$$\{q_0\} \xrightarrow{a^i} \{q_i\} \quad (1 \leq i \leq m-k-1),$$
$$\{q_{m-k-1}\} \xrightarrow{a} \{q_{m-k}, 0\} \xrightarrow{(ab^n)^i} \{q_{m-k+i}, 0\} \quad (1 \leq i \leq k-1),$$
$$\{q_{m-1}, 0\} \xrightarrow{a} \{q_0, 1\} \xrightarrow{b^{j-1}} \{q_0, j\} \quad (2 \leq j \leq n-1),$$
$$\{q_0, n-1\} \xrightarrow{b} \{q_0, 0\},$$
$$\{q_0, j \ominus i\} \xrightarrow{a^i} \{q_i, j\} \quad (1 \leq i \leq m-k-1).$$

Let $1 \leq t \leq n$, and assume that each set in $\mathcal{R}$ of size t is reachable. By Lemma 4.2 case 1, every set $\{q_{m-k}\} \cup S$ in $\mathcal{R}$ of size $t+1$ is reachable from a set in $\mathcal{R}$ of size t. Now let $\{q_i\} \cup S$ be a set in $\mathcal{R}$ of size $t+1$ with $i \neq m-k$. Consider four cases:

(*i*) Let $m-k+1 \leq i \leq m-1$, so $0 \in S$. Take $S' = S \setminus \{0\}$. Then

$$\{q_{i-1}\} \cup (S' \ominus \min S') \xrightarrow{a} \{q_i\} \cup \{0\} \cup (S' \ominus (\min S' - 1)) \xrightarrow{b^{\min S'-1}} \{q_i\} \cup S;$$

notice that $0 \in S' \ominus \min S'$. This proves this case by induction on i.

(*ii*) Let $i = 0$ and $0 \notin S$. Then $\{q_{m-1}\} \cup (S \ominus \min S) \xrightarrow{ab^{\min S-1}} \{q_0\} \cup S$, where the former set is considered in case (*i*).

(*iii*) Let $i = 0$ and $0 \in S$. Take $S' = S \setminus \{0\}$. Then $\{q_{m-1}\} \cup (S' \ominus \min S') \cup \{n-1\} \xrightarrow{a} \{q_0\} \cup \{0\} \cup (S' \ominus (\min S' - 1)) \xrightarrow{b^{\min S'-1}} \{q_0\} \cup S$, where the first set is considered in case (*i*).

(*iv*) Let $1 \leq i \leq m-k-1$. Then $\{q_i\} \cup S$ is reachable by Lemma 4.2 case 2.

To prove distinguishability, let $\{q_i\} \cup S$ and $\{q_j\} \cup T$ be two distinct subsets in $\mathcal{R}$. Notice that the state $n-1$ is uniquely distinguishable since it is a unique final state of N. Next, the state $n-1$ is uniquely reachable from states in Q_B since for every $j = 0, 1, \ldots, n-2$ the transition $(j, a, j+1)$ is a unique in-transition. By Proposition 2.1, if $S \neq T$, then the sets $\{q_i\} \cup S$ and $\{q_j\} \cup T$ are distinguishable.

TABLE 1. The state complexity of concatenation if the witness languages from [5, 10, 14] have the second half of their states final; in rows we have m, in columns n.

	Upper bound			Maslov [10]			YZS [14]			JJS [5]		
	2	4	6	2	4	6	2	4	6	2	4	6
2	6	24	96	5	4	18	6	14	27	6	22	84
4	12	48	192	10	5	35	12	28	54	12	42	156
6	18	72	288	15	6	52	18	42	81	18	63	225

Now let $S = T$, so $i < j$. If $S = \emptyset$, we read a^{m-k-j} and get $\{q_{m-k-j+i}\}$ and $\{q_{m-k}, 0\}$. If $S \neq \emptyset$, we first read $a^{m-1-j}b^n$ to get $\{q_{m-1-j+i}, 0\}$ and $\{q_{m-1}, 0\}$. Now we read a. There are two sub-cases:

1. If $m - j + i \geq m - k$, then we get $\{q_{m-j+i}, 0, 1\}$ and $\{q_0, 1\}$ which are distinguishable.
2. If $m - j + i < m - k$, then we get $\{q_{m-j+i}, 1\}$ and $\{q_0, 1\}$. Then we read $a^{(m-k-1)-(m-j+i)}b^n$ and get $\{q_{m-k-1}, 0\}$ and $\{q_{j-i-k-1}, 0\}$. Finally we read a and get $\{q_{m-k}, 0, 1\}$ and $\{q_{j-i-k}, 1\}$, which are distinguishable. □

Our next goal is to describe, for all m, n, k, ℓ with $n \geq 2$, two DFAs of m and n states, and k and ℓ final states, respectively, meeting the upper bound $m2^n - k2^{n-1}$ on the complexity of the concatenation of their languages. We try to modify the witness automata in all cases, by making the second half of their states final. The upper bound in such a case is $3m \cdot 2^{n-2}$.

Table 1 shows that none of the three witnesses presented in [5, 10, 14] meets this bound. Even making two states final in DFA B, results in a complexity of concatenation less that $m2^n - 2 \cdot 2^{n-1}$ in all three cases. Therefore we present new pairs of witness languages. To cover all possible values of m, n, k, ℓ, we modified the witness from Theorem 1 of [5] by defining transitions on a new symbol c. Notice that making some states final in DFA B does not play any role in the proof of reachability. We use the new symbol c only in the proof of distinguishability.

Theorem 4.6 (Ternary Witness Languages with $|F_A| = k$ and $|F_B| = \ell$). *Let $m \geq 1$ and $n \geq 2$. Let $k = 1$ if $m = 1$ and $1 \leq k \leq m - 1$ otherwise. Let $1 \leq \ell \leq n - 1$. There exist a ternary* DFA *A with m states and k final states and a ternary* DFA *B with n states and ℓ final states such that* $\mathrm{sc}(L(A)L(B)) = m2^n - k2^{n-1}$.

Proof. Define an m-state DFA $A = (\{q_0, q_1, \ldots, q_{m-1}\}, \{a, b, c\}, \cdot, q_0, F_A)$, where $F_A = \{q_i \mid m - k \leq i \leq m - 1\}$ and for each i in $\{0, 1, \ldots, m - 1\}$, we have $q_i \cdot a = q_{(i+1) \bmod m}$, $q_i \cdot b = q_i$, and $q_i \cdot c = q_i$,

Define an n-state DFA $B = (\{0, 1, \ldots, n - 1\}, \{a, b, c\}, \cdot, 0, \{n - 1\})$, where $F_B = \{j \mid n - \ell \leq j \leq n - 1\}$ and for each state j of B,

$$
\begin{aligned}
&j \cdot a = (j + 1) \bmod n,\\
&0 \cdot b = 0, \quad j \cdot b = (j + 1) \bmod n \text{ if } j \geq 1,\\
&j \cdot c = 0 \text{ if } j \leq n - 2, \quad (n - 1) \cdot c = n - 1.
\end{aligned}
$$

The DFAs A and B, where $m = 6, k = 3, n = 5$, and $\ell = 2$ are shown in Figure 5. Notice that the transitions on a and b are the same as in Theorem 1 of [5].

Construct an NFA for $L(A)L(B)$ from DFAs A and B as described in Construction 3.1. Since the transitions on a and b are the same as in DFAs in the proof of Lemma 4.5, the proof of reachability is the same; notice that making some states final in DFA B does not play any role in the proof of reachability.

We only need to prove distinguishability. To this aim, let $\{q_i\} \cup S$ and $\{q_j\} \cup T$ be two distinct reachable subsets. Notice that the state $n - 1$ is uniquely distinguishable by the string c since we have $\ell \leq n - 1$, so state 0 is not final. Next, the state $n - 1$ is uniquely reachable from all states in Q_B through unique in-transitions on a.

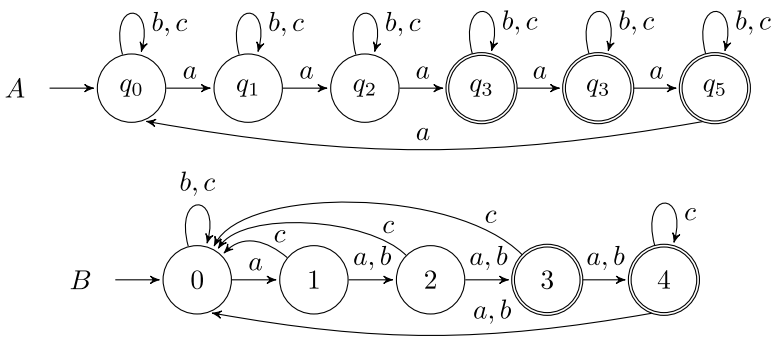


FIGURE 5. Ternary witnesses meeting the bound $m2^n - k2^{n-1}$; $m = 6$, $k = 3$, $n = 5$, and $\ell = 2$.

It follows that $\{q_i\} \cup S$ and $\{q_j\} \cup T$ are distinguishable if $S \neq T$. Now let $S = T$. In this case we continue exactly the same way as in the proof of Lemma 4.5. □

We have proven that for every number of states in A and B, except for one state in B, and every number of final states in A and B, except for none or all, there exist ternary automata meeting the upper bound $m2^n - k2^{n-1}$ for concatenation of their languages. We might ask whether there are binary languages with more final states in B meeting this bound. We provide a positive answer in the next theorem. However, notice that we require $k \leq m - 2$ here, that is, the first DFA must have at least two non-final states.

Theorem 4.7 (Binary Witness Automata with $|F_A| \leq m - 2$). *Let $m \geq 3$, $n \geq 4$, $1 \leq k \leq m - 2$, and $1 \leq \ell \leq n - 1$. There exist a binary* DFA *A with m states and k final states and a binary* DFA *B with n states and ℓ final states such that* $\mathrm{sc}(L(A)L(B)) = m2^n - k2^{n-1}$.

Proof. Define an m-state DFA $A = (Q_A, \{a, b\}, \cdot, q_0, F_A)$, where we have $Q_A = \{q_0, q_1, \ldots, q_{m-1}\}$, $F_A = \{q_i \mid m - k \leq i \leq m - 1\}$, and for each i in $\{0, 1, \ldots, m - 1\}$,

$$q_i \cdot a = q_{(i+1) \bmod m}, \quad q_0 \cdot b = q_0 \quad \text{and} \quad q_i \cdot b = q_{i-1} \text{ if } 1 \leq i \leq m - 1.$$

Define an n-state DFA $B = (Q_B, \{a, b\}, \cdot, 0, \{n - 1\})$, where $Q_B = \{0, 1, \ldots, n - 1\}$, $F_B = \{n - \ell, n - \ell + 1, \ldots, n - 1\}$ if $\ell \leq n - 2$, and $F_B = Q_B \setminus \{1\}$ if $\ell = n - 1$. For each state j of B, we have

$$\begin{aligned} &0 \cdot a = 0, && j \cdot a = j + 1 \text{ if } 1 \leq j \leq n - 2, && \text{and} && (n - 1) \cdot a = 1, \\ &0 \cdot b = 1, && 1 \cdot b = 2, && \text{and} && j \cdot b = j \text{ if } 2 \leq j \leq n - 1. \end{aligned}$$

The DFAs A and B, where $m = 6, k = 3, n = 5$, and $\ell = 2$ are shown in Figure 6; notice that the DFA B is the same as in [2]. Since $k \leq m - 2$, the states q_0 are q_1 are never final. By definition of B, state 1 is not final either.

Construct an NFA N for $L(A)L(B)$ as described in Construction 3.1. We prove that the subset automaton of N has $m2^n - k2^{n-1}$ reachable and pairwise distinguishable states. The proof of reachability is by induction on $|\{q_i\} \cup S|$.

The base, with $|\{q_i\} \cup S| = 1$, holds true since $\{q_0\} \xrightarrow{a^i} \{q_i\}$ for $1 \leq i \leq m - k - 1$. By Lemma 4.2, we need only to prove that every set $\{q_0\} \cup S$ of size $t + 1$ and $0 \notin S$ is reachable. Let $S' = ((S \ominus (\min S - 1)) \setminus \{1\}) \cup \{0\}$. Then $|S'| = |S|$ and $0 \in S'$. By Lemma 4.2 case 3, the set $\{q_0\} \cup S'$ is reachable from a set of size t. Next we have

$$\{q_0\} \cup S' \xrightarrow{b(ab)^{\min S - 1}} \{q_0\} \cup S;$$

notice that $1 \in S \ominus (\min S - 1)$ and $q_0 \xrightarrow{ab} q_0$, because $q_1 \notin F_A$ since $k \leq m - 2$. This proves reachability.

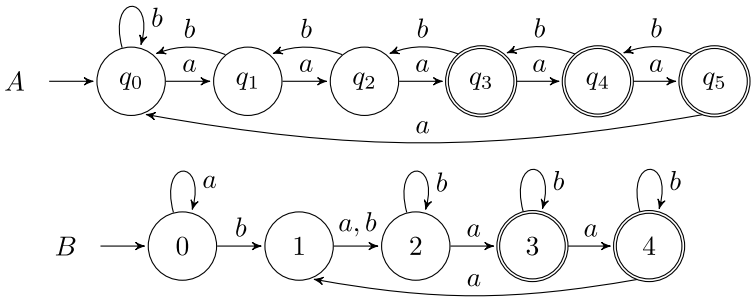


FIGURE 6. Binary witnesses meeting the bound $m2^n - k2^{n-1}$ in the case $k \leq 2$ (modified from [2]); $m = 6$, $k = 3$, $n = 5$, $\ell = 2$.

To prove distinguishability, we use the string $w = \prod_{i=0}^{n-4} a^{n-3-i}b^m a^{i+2}$. We have

$$\{2\} \cdot w = \{2\}; \quad (Q_B \setminus \{2\}) \cdot w = \{1\}; \quad Q_A \cdot w \subseteq Q_A \cup \{0, 1\}.$$

We now use these properties to prove distinguishability. Let $\{q_i\} \cup S$ and $\{q_j\} \cup T$ where $i, j \in \{0, 1, \ldots, m-1\}$ and $S, T \subseteq \{0, 1, \ldots, n-1\}$ be two reachable states of the subset automaton of N. We consider several cases:

1. If $2 \in S$ and $2 \notin T$, then

$$\{q_i\} \cup S \xrightarrow{wab^m} \{q_0\} \cup \{2, 3\} \xrightarrow{(ba)^{n-2}} \{q_1\} \cup \{1, 3\}, \text{ and}$$
$$\{q_j\} \cup T \xrightarrow{wab^m} \{q_0\} \cup \{2\} \xrightarrow{(ba)^{n-2}} \{q_1\} \cup \{1\}.$$

   If $3 \in F_B$, we have distinguished the sets. If not, we read $a^{n-\ell-3}$ and distinguish the sets since all states j with $j < n - \ell$ are non-final.
2. If $1 \leq s \leq n-1$ and $s \neq 2$, $s \in S$ and $s \notin T$, we read a^{n+1-s} to get the case 1.
3. If $0 \in S$ and $0 \notin T$, we read b to get the case 2.
4. If $S = T$ and $1 \leq i < m - k \leq j$, we read ba and get $\{q_i\} \cup S \cdot ba$ and $\{q_j\} \cup \{0\} \cup S \cdot ba$. Since $0 \notin S \cdot ba$, we get the case 3. If $0 = i < m - k \leq j$, we read ba and get $\{q_1\} \cup S \cdot ba$ and $\{q_j\} \cup \{0\} \cup S \cdot ba$. We again get the case 3.
5. If $m - k \leq i < j$, we read the string a^{m-j} and get $\{q_{m-j+i}\} \cup S \cdot a^{m-j}$ and $\{q_0\} \cup S \cdot a^{m-j}$, which is considered in the case 4.
6. If $i < j < m - k$, we read the string a^{m-k-j} and get $\{q_{m-k-j+i}\} \cup S \cdot a^{m-k-j}$ and $\{q_{m-k}\} \cup \{0\} \cup S \cdot a^{m-k-j}$. If $0 \notin S \cdot a^{m-k-j}$, we get the case 3. If $0 \in S \cdot a^{m-k-j}$, we get the case 4. □

5. BINARY CONCATENATION; $|F_A| = m - 1$ AND $2 \leq |F_B| \leq n - 1$

Now we turn our attention to the concatenation of binary languages represented by m-state DFA with $m-1$ final states and n-state DFA with more than one final state. In the general case, the upper bound is $(m+1)2^{n-1}$. The next theorem provides a lower bound that is smaller just by one. Our computations show that no pair of binary languages meets the bound $(m+1)2^{n-1}$ in the case of $m, n \leq 4$.

Theorem 5.1 (Binary Concatenation with $|F_A| = m - 1$; Lower Bound). *Let $m, n \geq 3$ and $2 \leq \ell \leq n - 1$. There exist a binary* DFA *A with m states and $m - 1$ final states and a binary* DFA *B with n states and ℓ final states such that* $\mathrm{sc}(L(A)L(B)) \geq (m+1)2^{n-1} - 1$.

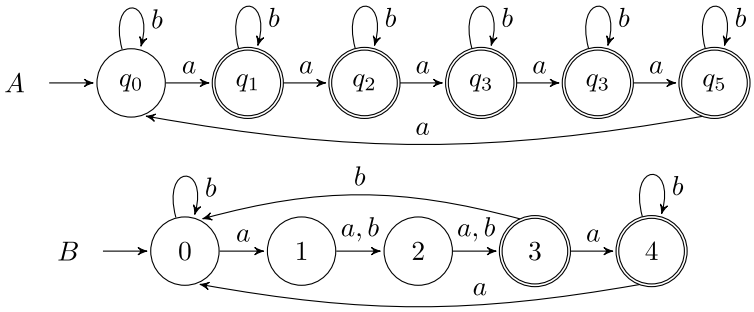


FIGURE 7. The binary DFAs meeting the bound $(m+1)2^{n-1}-1$; $m=6$, $k=5$, $n=5$, and $\ell=2$.

Proof. Define an m-state DFA $A=(\{q_0,q_1,\ldots,q_{m-1}\},\{a,b\},\cdot,q_0,F_A)$, where $F_A=\{q_1,q_2,\ldots,q_{m-1}\}$, and for each i in $\{0,1,\ldots,m-1\}$,

$$q_i\cdot a=q_{(i+1) \bmod m} \qquad \text{and} \qquad q_i\cdot b=q_i.$$

Define an n-state DFA $B=(\{0,1,\ldots,n-1\},\{a,b\},\cdot,0,F_B)$, where we have $F_B=\{j \mid n-\ell\le j\le n-1\}$, and for each j in $\{0,1,\ldots,n-1\}$,

$$\begin{aligned}&j\cdot a=(j+1) \bmod n,\\ &0\cdot b=0,\quad j\cdot b=(j+1) \bmod (n-1) \text{ if } 1\le j\le n-2,\quad (n-1)\cdot b=(n-1).\end{aligned}$$

The DFAs A and B for $m=6,k=5,n=5$, and $\ell=2$ are shown in Figure 7.

Construct an NFA N for $L(A)L(B)$ from DFAs A and B as described in Construction 3.1. We prove that the subset automaton of N has $(m+1)2^{n-1}-1$ reachable and pairwise distinguishable states. To this aim, consider the following family of $(m+1)2^{n-1}-1$ subsets:

$$\begin{aligned}\mathcal{R}=&\{\{q_0\}\cup X \mid X\subseteq\{0,1,\ldots,n-1\} \text{ and } X\neq\{n-1\}\}\ \cup\\ &\{\{q_i\}\cup X \mid 1\le i\le n-1, X\subseteq\{0,1,\ldots,n-1\}, \text{ and } 0\in X\}.\end{aligned}$$

First we prove that each set in $\mathcal{R}$ is reachable. The proof of reachability is by induction on $|q_i\cup X|$.

The basis, with $|S|\le 2$, holds true since $\{q_0\}$ is the initial subset, $\{q_0\}\xrightarrow{a}\{q_1,0\}$, and $\{q_{(i-1) \bmod m},0\}\xrightarrow{ab}\{q_i,0\}$ for $i=0,1,\ldots,m-1$.

Let $2\le t\le n$ and assume that each subset in $\mathcal{R}$ of size t is reachable. By Lemma 4.2, we only need to prove that every set $\{q_0\}\cup S$ of size $t+1$ and $0\notin S$ is reachable. To show that $\{q_0\}\cup S$ is reachable, recall that $\{q_{m-1}\}\cup(S\ominus\min S)$ is reachable by Lemma 4.2 case 3 since $0\in S\ominus\min S$. Next we have

$$\{q_{m-1}\}\cup(S\ominus\min S)\xrightarrow{a}\{q_0\}\cup(S\ominus(\min S-1))\xrightarrow{b^{\min S-1}}\{q_0\}\cup S.$$

This proves reachability.

To prove distinguishability, let $\{q_i\}\cup S$ and $\{q_j\}\cup T$ be two distinct sets in $\mathcal{R}$. In a similar way as in the proof of Lemma 4.5, we can show that for every state t, $0\le t\le n-1$, the string $a^{n-1-t}b(ab^{n-2})^{n-3}$ is accepted by NFA N only from t. It follows that the sets $\{q_i\}\cup S$ and $\{q_j\}\cup T$ are distinguishable if $S\neq T$.

If $S=T$, then we may assume that $S\neq\emptyset$ because $\{q_0\}$ is the only reachable set of size one. Thus we need to distinguish the sets $\{q_i\}\cup S$ and $\{q_j\}\cup S$, where $S\neq\emptyset$ and $i\neq j$. We first read $(ab^{n-2})^{n-2}$ to get $\{q_x,0\}$

and $\{q_y, 0\}$ with $x \neq y$; notice that both symbols a and b perform a permutation on the states of DFA A. We may assume that $x < y$. Consider two cases:

1. If $y = m-1$, we use a to get $\{q_{x+1}, 0, 1\}$ and $\{q_0, 1\}$, which differ in state 0.
2. If $y < m-1$, then we use $(ab)^{m-1-y}$ to get $\{q_{x+m-1-y}, 0\}$ and $\{q_{m-1}, 0\}$ which are considered in case 1.

This completes our proof. □

6. Concatenation on alternating finite automata

In this section, we consider the concatenation operation on alternating finite automata (AFAs) [3]. Our aim is to describe languages K and L accepted by an m-state and n-state AFA, respectively, such that the minimal AFA for the language KL requires $2^m + n + 1$ states. This solves an open problem stated by Fellah, Jürgensen, and Yu in [3], where the upper bound is proven to be the same. First, let us give some basic definitions and notations. For details, we refer the reader to [1, 3, 7–9, 12].

An *alternating finite automaton* (AFA) is a quintuple $A = (Q, \Sigma, \delta, s, F)$, 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 over the n variables $q_1, \ldots, q_n$, $s \in Q$ is the initial state, and $F \subseteq Q$ is the set of final states. For an example, consider AFA $A_1 = (\{q_1, q_2\}, \{a, b\}, \delta, q_1, \{q_2\})$, where transition function δ is given in Table 2.

Table 2. The transition function of the alternating finite automaton A_1.

δ	a	b
q_1	$q_1 \vee q_2$	q_1
q_2	q_2	$\overline{q_1} \wedge q_2$

The transition function δ is 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^*, $\delta(g, \varepsilon) = g$; 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 AFA A is the set $L(A) = \{w \in \Sigma^* \mid \delta(s, w)(f) = 1\}$.

In our example we have $\delta(q_1, ab) = \delta(\delta(q_1, a), b) = \delta(q_1 \vee q_2, b) = q_1 \vee (\overline{q_1} \wedge q_2) = q_1 \vee q_2$. To determine whether $ab \in L(A_1)$, we evaluate $\delta(q_1, ab)$ at the vector $f = (0, 1)$. We obtain 1, hence $ab \in L(A_1)$.

Recall that the state complexity of a regular language L, $\mathrm{sc}(L)$, is the smallest number of states in any DFA accepting L. Similarly, the alternating state complexity of L, $\mathrm{asc}(L)$, is the smallest number of states in any AFA for L. It follows from Theorem 4.1, Corollary 4.2 of [3] and Lemma 1, Lemma 2 of [7] that a language L is accepted by an n-state AFA if and only if L^R is accepted by a DFA with 2^n states and 2^{n-1} final states. As this is a crucial observation for this section, we restate these results and provide proof ideas.

Lemma 6.1 ([3, 7])**.** *Let L be a language accepted by an n-state* 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 $A = (\{q_1, q_2, \ldots, q_n\}, \Sigma, \delta, q_1, F)$ be an n-state AFA 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 Σ, $\delta'(u, a) = \{u' \in \{0,1\}^n \mid \delta(q_i, a)(u') = u_i \text{ for } i = 1, \ldots, n\}$;
- $S = \{(b_1, \ldots, b_n) \in \{0,1\}^n \mid b_1 = 1\}$;
- $f = (f_1, \ldots, f_n) \in \{0,1\}^n$ with $f_i = 1$ iff $q_i \in F$.

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

Corollary 6.2. *For every regular language L,* $\mathrm{asc}(L) \geq \lceil \log(\mathrm{sc}(L^R)) \rceil$.

Lemma 6.3 (*cf.* [7], Lem. 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* AFA*.*

Proof Idea. Consider 2^n-state NFA A^R for L which has 2^{n-1} initial states and exactly one final state. Let the state set Q of A^R be $\{0,1,\ldots,2^n-1\}$ with initial states $\{2^{n-1},\ldots,2^n-1\}$ and final state k. 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 $b_1b_2\cdots b_n$ is the binary notation of i on n digits with leading zeros if necessary.

Define an n-state AFA $A'=(Q',\Sigma,\delta',q_1,F')$, where $Q'=\{q_1,\ldots,q_n\}$, $F'=\{q_\ell \mid \mathrm{bin}(k)_\ell=1\}$, and 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 $L(A')=L(A^R)$. □

By Corollary 6.2, we have $\mathrm{asc}(L)\geq\lceil\log(\mathrm{sc}(L^R))\rceil$. The upper bound for concatenation on AFAs is 2^m+n+1, as proven by Fellah *et al.* Theorem 9.3 of [3]. They conjectured that this bound is tight.

In [7], the lower bound 2^m+n was proven, however, the witnesses from Theorem 1 of [5] with half of states final in both automata were used. As we mentioned above, *cf.* Table 1, these witness languages do not meet the upper bound for concatenation on DFAs. Hence the proof in Theorem 5 of [7] is not correct, so the problem is still open. Our next aim is to prove the tightness of the upper bound 2^m+n+1 for concatenation on AFAs. We might use the ternary witness from Theorem 4.6, but, as we show below, for $\mathrm{asc}(K)\geq 2$, $\mathrm{asc}(L)\geq 2$, it is sufficient to use the binary witness languages described in the proof of Theorem 4.7 to get languages that meet the upper bound 2^m+n+1 for concatenation on AFAs. The following lemma not only proves the claim in Theorem 5 of [7], but also solves the open problem mentioned above.

Lemma 6.4 (**Concatenation on AFAs: Lower Bound**)**.** *Let $m,n\geq 2$. There exist binary languages K and L accepted by an m-state and n-state* AFA*, respectively, such that* $\mathrm{asc}(KL)=2^m+n+1$.

Proof. Let L^R be the binary regular language accepted by the minimal DFA A from the proof of Theorem 4.7, with 2^n states and 2^{n-1} final states. Let K^R be the binary regular language accepted by the minimal DFA B from the proof of Theorem 4.7, with 2^m states and 2^{m-1} final states. Then, by Lemma 6.3, the languages K and L are accepted by an m-state and n-state AFA, respectively. Using Theorem 4.7, we get

$$\mathrm{sc}((KL)^R)=\mathrm{sc}(L^RK^R)=2^n\cdot 2^{2^m}-2^{n-1}\cdot 2^{2^m-1}=2^{n-1}\cdot 2^{2^m}(1+1/2).$$

By Corollary 6.2, we have $\mathrm{asc}(KL)\geq\lceil\log(2^{n-1}\cdot 2^{2^m}(1+1/2))\rceil=2^m+n$.

Our next aim is to show that $\mathrm{asc}(KL)\geq 2^m+n+1$. Suppose for a contradiction that KL is accepted by an AFA of 2^m+n states. Then $(KL)^R$ is accepted by a 2^{2^m+n}-state DFA with 2^{2^m+n-1} final states. It follows that the minimal DFA for $(KL)^R$ has at most 2^{2^m+n-1} final states. However, the minimal DFA for $(KL)^R$ has $2^n2^{2^m}-2^{n-1}2^{2^m-1}$ states, of which $2^{n-1}2^{2^m-1}+2^{n-1}2^{2^{m-1}-1}$ are non-final; notice that $\{q_i\}\cup S$ is non-final iff $i\leq 2^{n-1}-1$ and $S\subseteq\{0,1,\ldots,2^{m-1}-1\}$ or $2^{n-1}\leq i\leq 2^n-1$ and $S\subseteq\{0,1,\ldots,2^{m-1}-1\}$ with $0\in S$. Thus the number of final states in the minimal DFA for $(KL)^R$ is

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

and since $m\geq 2$, we get

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

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

We continue with examining the complexity of concatenation of unary AFA languages. Since the reverse of every unary language is the same language, we get that a unary language is accepted by an n-state AFA if and only if it is accepted by a 2^n-state DFA with 2^{n-1} final states. So in order to have languages K and L accepted by an m-state and n-state AFA, respectively, we only need to find unary languages K and L represented by a 2^m-state and 2^n-state DFA with half states final, respectively. The next lemma shows that the upper bound for binary AFAs cannot be met in the unary case. Then we provide a lower bound.

Lemma 6.5 (**Concatenation of Unary AFAs; Upper Bound**). *Let $m, n \geq 1$. Let K and L be unary languages accepted by an m-state and n-state* AFA*, respectively. Then* $\mathrm{asc}(KL) \leq m + n + 1$.

Proof. By Lemma 6.1, the unary language K is accepted by a 2^m-state DFA with 2^{m-1} final states and L is accepted by a 2^n-state DFA with 2^{n-1} final states. It follows that KL is accepted by a DFA with $2^m \cdot 2^n$ states, as is proven in Theorem 5.5 of [14]. By adding some final or non-final unreachable states, we can construct an equivalent DFA with 2^{m+n+1} states and 2^{m+n} final states. By Lemma 6.3, the language KL is accepted by an $(m + n + 1)$-state AFA. □

Lemma 6.6 (**Concatenation of Unary AFAs; Lower Bound**). *Let $m, n \geq 1$. There exist unary languages accepted by an m-state and n-state* AFA*, respectively, such that* $\mathrm{asc}(KL) \geq m + n - 1$.

Proof. We have $\gcd(2^{m-1}, 2^{n-1} + 1) = 1$.

Consider DFA $A = (\{0, 1, \ldots, 2^m - 1\}, \{a\}, \cdot, 0, \{i \mid 2^{m-1} - 1 \leq i \leq 2^m - 2\})$, where $i \cdot a = i + 1$ if $0 \leq i < 2^{m-1} - 1$, and $i \cdot a = 0$ otherwise. Thus A has 2^m states and 2^{m-1} final states, so $L(A)$ is accepted by an m-state AFA; notice that only 2^{m-1} states are reachable in A.

Next, consider DFA $B = (\{0, 1, \ldots, 2^n - 1\}, \{a\}, \cdot, 0, \{j \mid 2^{n-1} \leq j \leq 2^n - 1\})$, where $j \cdot a = j + 1$ if $0 \leq j < 2^{n-1}$, and $j \cdot a = 0$ otherwise. Similarly as above, $L(B)$ is accepted by an n-state AFA, and this time only $2^{n-1} + 1$ states are reachable in B. As shown in Theorem 5.4 of [14] $\mathrm{sc}(L(A)L(B)) = 2^{m-1} \cdot (2^{n-1} + 1) = 2^{m+n-2} + 2^{m-1}$. By Lemma 6.3, we have $\mathrm{asc}(L(A)L(B)) \geq m + n - 1$. □

As a corollary of Lemmas 6.4, 6.5, and 6.6, we state the following theorem.

Theorem 6.7 (**Concatenation on AFAs**). *Let $m, n \geq 2$. Let K and L be languages over an alphabet Σ accepted by an m-state and n-state* AFA*, respectively. Then* $\mathrm{asc}(KL) \leq 2^m + n + 1$*, and this bound is tight if* $|\Sigma| \geq 2$*. If* $|\Sigma| = 1$*, then* $\mathrm{asc}(KL) \leq m + n + 1$*. There exist unary m-state and n-state* AFA *languages meeting the bound $m + n - 1$.*

7. Conclusions

We studied the state complexity of the concatenation of languages represented by deterministic and alternating finite automata. First, we described ternary languages meeting the upper bound $m2^n - k2^{n-1}$ for all possible values of m, n, k, ℓ, where m and k is the number of states and the number of final states in the minimal DFA for the first language, and n and ℓ is the number of states and the number of final states in the minimal DFA for the second language. Then, in the case of $k \leq m - 2$, that is, if the first automaton has at least two non-final states, we described appropriate binary languages. Finally, we considered the case of $k = m - 1$ and $\ell \geq 2$ over a binary alphabet, and obtained a lower bound that is smaller than the corresponding upper bound just by one. We strongly conjecture that our lower bound is tight in this case.

We used our binary witnesses for the concatenation on DFAs to define binary languages K and L accepted by an m-state and n-state AFA, respectively, such that the minimal AFA for KL requires $2^m + n + 1$ states. This proves that the upper bound $2^m + n + 1$ from [3] is tight, and solves the open problem stated in Theorem 9.3 of [3]. We also proved that this upper bound cannot be met by unary AFA languages, where we get upper bound $m + n + 1$ and lower bound $m + n - 1$.

References

[1] J. Brzozowski and E. Leiss, On equations for regular languages, finite automata, and sequential networks. *Theoret. Comput. Sci.* **10** (1980) 19–35.

[2] E. Dorčák, *Concatenation of regular languages and state complexity*, ŠVK thesis. P. J. Šafárik University, Košice (2015)

[3] A. Fellah, H. Jürgensen, and S. Yu, Constructions for alternating finite automata. *Int. J. Comput. Math.* **35** (1990) 117–132.

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

[5] J. Jirásek, G. Jirásková, and A. Szabari, State complexity of concatenation and complementation. *Int. J. Found. Comput. Sci.* **16** (2005) 511–529.

[6] G. Jirásková, State complexity of some operations on binary regular languages. *Theoret. Comput. Sci.* **330** (2005) 287–298.

[7] G. Jirásková, Descriptional complexity of operations on alternating and boolean automata. In *CSR 2012*, edited by E. Hirsch *et al.* Vol. 7353 of *Lect. Notes Comput Sci.* Springer (2012) 196–204.

[8] E. Leiss, Succinct representation of regular languages by boolean automata. *Theoret. Comput. Sci.* **13** (1981) 323–330.

[9] E. Leiss, On generalized language equations. *Theoret. Comput. Sci.* **14** (1981) 63–77.

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

[11] M. Rabin and D. Scott, Finite automata and their decision problems. *IBM Res. Develop.* **3** (1959) 114–129.

[12] M. Sipser, Introduction to the theory of computation. Cengage Learning, Boston (2012)

[13] S. Yu, Regular languages, Chapter 2. In vol. I of *Handbook of Formal Languages*, edited by G. Rozenberg and A. Salomaa. Springer, Heidelberg (1997) 41–110.

[14] S. Yu, Q. Zhuang, and K. Salomaa, The state complexity of some basic operations on regular languages. *Theoret. Comput. Sci.* **125** (1994) 315–328.

Appendix [B]

Michal Hospodár, Galina Jirásková, Peter Mlynárčik:
Descriptional complexity of the forever operator.
International Journal of Foundations of Computer Science, Vol. 30, No. 1, pp. 115–134 (2019).
ISSN: 0129-0541, DOI: 10.1142/S0129054119400069

International Journal of Foundations of Computer Science
Vol. 30, No. 1 (2019) 115–134

DOI: 10.1142/S0129054119400069

World Scientific
www.worldscientific.com

Descriptional Complexity of the Forever Operator

Michal Hospodár*, Galina Jirásková† and Peter Mlynárčik‡

Mathematical Institute, Slovak Academy of Sciences
Grešákova 6, Košice, 040 01, Slovakia
**hosmich@gmail.com*
†*jiraskov@saske.sk*
‡*mlynarcik1972@gmail.com*

Accepted 23 January 2019
Communicated by Michel Rigo

We examine the descriptional complexity of the forever operator, which assigns the language $\overline{\Sigma^*\overline{L}}$ to a regular language L, and we investigate the trade-offs between various models of finite automata. We consider complete and partial deterministic finite automata, nondeterministic finite automata with single or multiple initial states, alternating, and Boolean finite automata. We assume that the argument and the result of this operation are accepted by automata belonging to one of these six models. We investigate all possible trade-offs and provide a tight upper bound for 32 of 36 of them. The most interesting result is the trade-off from nondeterministic to deterministic automata given by the Dedekind number $\mathrm{M}(n-1)$. We also prove that the nondeterministic state complexity of $\overline{\Sigma^*\overline{L}}$ is 2^{n-1} which solves an open problem stated by Birget [The state complexity of $\overline{\Sigma^*\overline{L}}$ and its connection with temporal logic, *Inform. Process. Lett.* **58** (1996) 185–188].

Keywords: Regular languages; forever operator; deterministic automata; nondeterministic automata; Boolean automata; minimal automata; trade-off.

1. Introduction

Formal languages may be recognized by several kinds of formal systems. Different classes of formal systems can be compared either from the point of view of their computational power, or from the descriptional complexity point of view. As for computational power, for example, deterministic and nondeterministic finite automata recognize the same class of languages, while the class of languages recognized by deterministic pushdown automata is strictly included in the class of languages recognized by nondeterministic ones. However, from the descriptional complexity point of view, there is an exponential gap between the cost of description of regular languages by deterministic and nondeterministic finite automata [14, 16–18, 20].

Descriptional complexity, which measures the cost of description of languages by different formal systems, was deeply investigated in last three decades [1, 7, 15, 22] mostly in the class of regular languages. Several kinds of finite automata were proposed and the trade-offs between the costs of description in different classes of automata were examined. Let us mention at least the exact trade-off $\binom{2n}{n+1}$ for the conversion of two-way nondeterministic automata to one-way nondeterministic automata [12], and the exact trade-off for the conversion of self-verifying automata to deterministic automata given by the function that counts the maximal number of maximal cliques in a graph with n vertices [11].

In 1996, Jean-Camille Birget [2] answered the following question of Jean-Éric Pin. Let L be a regular language over an alphabet Σ recognized by a nondeterministic finite automaton (NFA) or a deterministic finite automaton (DFA) with n states. How many states are sufficient and necessary in the worst case for an NFA (DFA) to recognize the language $\overline{\Sigma^*\overline{L}}$? The notation $\overline{L}$ stands for the complement of L. Birget provided the exact trade-off from DFAs to NFAs, and lower and upper bounds for the nondeterministic state complexity of $\overline{\Sigma^*\overline{L}}$.

The motivation of Pin's question came from the word model of Propositional Temporal Logic [5]. The set of all models of a formula φ over a fixed alphabet Σ is a formal language $L(\varphi)$ over Σ which has the non-trivial property of being regular and aperiodic. Some of the temporal operators used in this logic are $\circ$ ("next") and $\diamond$ ("eventually", or "at some moment in the future"); there are also the usual Boolean operations $-$, $\wedge$, $\vee$. A natural dual to the "eventually" operator is the "forever" (or, "always in the future") operator $\Box$, defined to be $-\diamond-$ ("not eventually not"). Formulas and their models are related as follows: $L(\overline{\varphi}) = \overline{L(\varphi)}$, $L(\varphi\wedge\psi) = L(\varphi)\cap L(\psi)$, $L(\varphi\vee\psi) = L(\varphi)\cup L(\psi)$, $L(\circ\varphi) = \Sigma L(\varphi)$, $L(\diamond\varphi) = \Sigma^* L(\varphi)$. Thus $L(\Box\varphi) = L(\overline{\diamond\overline{\varphi}}) = \overline{\Sigma^*\overline{L}}$. Hence in [2], Birget studied the state complexity of the "forever" operator.

Here we continue this research by investigating the complexity of the forever operator for different models of finite automata. We consider complete and partial deterministic finite automata, nondeterministic automata with a single or multiple initial states, and Boolean automata with a single initial state, called *alternating* finite automata in [2, 6], or with an initial function [4]. Similarly as Jean-Éric Pin, we ask the following question: If a language L is represented by an n-state automaton of some model, how many states are sufficient and necessary in the worst case for an automaton of some other model to accept $\overline{\Sigma^*\overline{L}}$?

We study all the possible 36 trade-offs, and except for four cases, we always get tight upper bounds. In particular, we are able to prove that the upper bound on the nondeterministic state complexity of $\overline{\Sigma^*\overline{L}}$ is 2^{n-1}. This improves Birget's upper bound $2^{n+1}+1$ and meets his lower bound for DFA-to-NFA trade-off. The most interesting result of this paper is the tight upper bound for the NFA-to-DFA trade-off given by the Dedekind number $\mathrm{M}(n-1)$; recall that the Dedekind number $\mathrm{M}(n)$ counts the number of antichains of subsets of an n-element set.

2. Preliminaries

Let Σ be a finite alphabet of symbols. Then Σ^* denotes the set of strings over Σ including the empty string ε. A language is any subset of Σ^*. For a language L, the complement of L is the language $\overline{L} = \Sigma^* \setminus L$. The concatenation of languages K and L is the language $KL = \{uv \mid u \in K \text{ and } v \in L\}$. The cardinality of a finite set A is denoted by $|A|$, and its power-set by 2^A. By $\log n$, we denote the binary logarithm of the number n. For details, we refer to [19, 21].

A *nondeterministic finite automaton with multiple initial states* (NNFA) is a 5-tuple $A = (Q, \Sigma, \circ, I, F)$, 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 $L(A) = \{w \in \Sigma^* \mid I \circ w \cap F \neq \emptyset\}$. *If* $|I| \geq 2$, we say that A is a nondeterministic finite automaton with nondeterministic choice of initial state (so we use the abbreviation NNFA, cf. [21]). Otherwise, if $|I| = 1$, we say that A is a *nondeterministic finite automaton* (NFA). Then, if $I = \{s\}$, we write $A = (Q, \Sigma, \circ, s, F)$ instead of $A = (Q, \Sigma, \circ, \{s\}, F)$. In an ε-NFA, we also allow the transitions on the empty string. It is known that the ε-transitions can be removed without increasing the number of states in the resulting NFA [21, Theorem 2.3].

An NFA A is a (complete) *deterministic finite automaton* (DFA) if $|q \circ a| = 1$ for each q in Q and each a in Σ. Next, A is a *partial deterministic finite automaton* (pDFA) if $|q \circ a| \leq 1$ for each q in Q and each a in Σ. We write $p \circ a = q$ in such cases. For DFAs, we use $\cdot$ to denote the transition function that maps $Q \times \Sigma$ to Q.

We call a state of an NNFA *sink state* if it has a loop on every input symbol. From every final sink state, every string is accepted, but from every non-final sink state in a DFA, no string is accepted. Notice that every minimal pDFA has no non-final sink states, and every minimal DFA has at most one non-final sink state.

For a symbol a and states p and q, we say that (p, a, q) is a transition in the NNFA A if $q \in p \cdot a$, and for a string w, we write $p \xrightarrow{w} q$ if $q \in p \cdot w$. We also say that the state q has an *in-transition* on a, and the state p has an *out-transition* on a.

To *omit* a state q of a DFA means to remove it from the state set and to remove also all its in-transitions and out-transitions. To *replace* the state q with a sink state means to remove each of its out-transitions and add a loop (q, a, q) for each a.

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

Every NNFA $A = (Q, \Sigma, \cdot, I, F)$ can be converted to an equivalent DFA $\mathcal{D}(A) = (2^Q, \Sigma, \cdot, I, F')$ where $F' = \{S \in 2^Q \mid S \cap F \neq \emptyset\}$. We call the DFA $\mathcal{D}(A)$ the *subset automaton* of the NNFA A. We use the following proposition to prove reachability of states in a subset automaton in some cases.

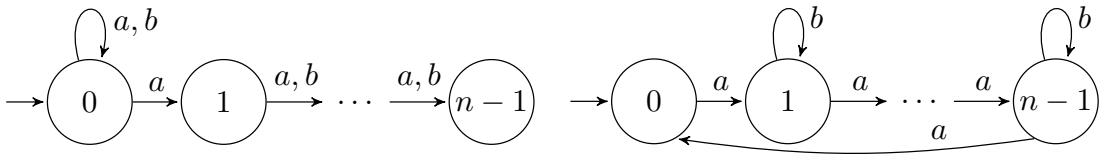


Fig. 1. The NFAs used in Proposition 1.

Proposition 1. *In the subset automaton of the* NFA *shown in Fig.* 1(left), *each subset containing* 0 *is reachable from* $\{0\}$, *and in the subset automaton of the* NFA *shown in Fig.* 1(right), *each subset is reachable from* $\{0, 1, \ldots, n-1\}$.

Proof. The proof of case (1) is by induction on the size of subsets. The subset $\{0\}$ is the initial subset. Each subset $\{0, i_1, i_2, \ldots, i_k\}$ of size $k+1$, where $1 \leq k \leq n-1$ and $1 \leq i_1 < i_2 < \cdots < i_k$, is reached from the subset $\{0, i_2 - i_1, \ldots, i_k - i_1\}$ of size k by the string ab^{i_1-1}. Notice that the proof works for arbitrary transitions on a, b in the state $n-1$. The proof of case (2) is also by induction on the size of subsets. The set $\{0, 1, \ldots, n-1\}$ of size n is the basis. Each subset S with $t \notin S$ of size $k-1$, where $1 \leq k \leq n$, is reachable from the subset $S \cup \{t\}$ of size k by $a^{n-t}ba^t$. □

To prove distinguishability, we use the following notions and observations. A state q of an NFA $A = (Q, \Sigma, \cdot, s, F)$ is called *uniquely distinguishable* (cf. [3]) if there is a string w which is accepted by A from and only from the state q. A transition (p, a, q) in the NFA A is called a *unique in-transition* if there is no state r of A 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 is a sequence of unique in-transitions (p_{i-1}, a_i, p_i) $(1 \leq i \leq k)$ such that $p_0 = p$ and $p_k = q$.

Proposition 2 (cf. [3, Propositions 14 and 15]). *Let* A *be an* NFA *and* $\mathcal{D}(A)$ *be the subset automaton of* A.

(a) *If two subsets of the state set of* A *differ in a uniquely distinguishable state, then the two subsets are distinguishable in* $\mathcal{D}(A)$.
(b) *If a uniquely distinguishable state* q *is uniquely reachable from a state* p, *then the state* p *is uniquely distinguishable as well.*
(c) *If there is a uniquely distinguishable state of an* NFA A *that 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.*

A set of pairs of strings $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ is called a *fooling set* for a language L if for all i, j in $\{1, 2, \ldots, n\}$, we have $x_i y_i \in L$, and if $i \neq j$, then $x_i y_j \notin L$ or $x_j y_i \notin L$. It is well-known [1, 8] that the size of a fooling set for L provides a lower bound on the number of states in any NNFA accepting the language L.

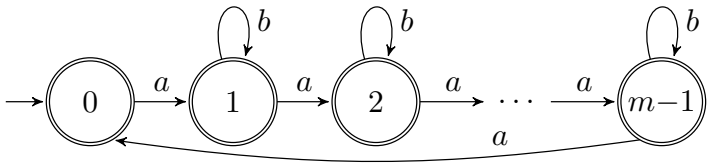


Fig. 2. The NFA for L such that every NFA for L^R has at least $m+1$ states and every DFA for L^R has at least 2^m states.

For NFAs, the following lemma from [10] is useful; notice that $\mathcal{A}$ and $\mathcal{B}$ in [10, Lemma 4] must be disjoint.

Lemma 3 (cf. [10, Lemma 4]). *Let $\mathcal{A}$ and $\mathcal{B}$ be disjoint sets of pairs of strings and let u and v be two strings such that $\mathcal{A} \cup \mathcal{B}$, $\mathcal{A} \cup \{(\varepsilon, u)\}$, and $\mathcal{B} \cup \{(\varepsilon, v)\}$ are fooling sets for a language L. Then every* NFA *for L has at least $|\mathcal{A}|+|\mathcal{B}|+1$ states.*

The next result is used later in our paper.

Proposition 4. *Let L be the language accepted by the* NFA *shown in Fig.* 2. *Then*

(a) *every* NFA *for L^R has at least $m+1$ states, and*
(b) *every* DFA *for L^R has at least 2^m states.*

Proof. We reverse the NFA in Fig. 2 to get the NNFA N for L^R.

(a) Let

$$\begin{aligned}
\mathcal{A} &= \{((ba)^{m-1-i}, (ba)^i) \mid 0 \le i \le m-2\},\\
\mathcal{B} &= \{(b, a(ba)^{m-2})\},\\
u &= a(ba)^{m-2}, \quad \text{and}\\
v &= a^m.
\end{aligned}$$

The reader may verify that $\mathcal{A} \cup \mathcal{B}$, $\mathcal{A} \cup \{(\varepsilon, u)\}$, and $\mathcal{B} \cup \{(\varepsilon, v)\}$ are fooling sets for the language L^R. By Lemma 3, every NFA for L^R needs at least $m+1$ states.

(b) By Proposition 1, the subset automaton $\mathcal{D}(N)$ has 2^m reachable states. To prove distinguishability, notice that the state 0 is uniquely distinguishable in N by ε, and it is uniquely reachable from any other state of N through unique in-transitions on symbol a. By Proposition 2(c-d), the subset automaton $\mathcal{D}(N)$ does not have equivalent states. □

A *Boolean finite automaton* (BFA, cf. [4]) is a quintuple $A = (Q, \Sigma, \delta, g_s, F)$, where Q is a finite non-empty set of states such that $Q = \{q_1, \ldots, q_n\}$, 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 is 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 of strings $L(A) = \{w \in \Sigma^* \mid \delta(g_s, w)(f) = 1\}$. A Boolean finite automaton is called *alternating* (AFA, cf. [6]) if the initial function is a projection $g(q_1, \ldots, q_n) = q_i$.

We use the following observations for trade-offs between various automata throughout this paper. We use the claim in Lemma 5(a) quite often in the paper without referring to Lemma 5(a) again and again.

Lemma 5 (Properties of Finite Automata). *Let L be a regular language.*

(a) *The language L is accepted by an n-state* BFA (AFA) *if and only if L^R is accepted by a* DFA *of 2^n states (of which 2^{n-1} are final, in case of an* AFA*).*
(b) *Let L^R be a regular language accepted by a minimal n-state* DFA*. Then every* BFA *for L has at least $\lceil \log n \rceil$ states.*
(c) *If the minimal DFA for L^R has more than 2^{n-1} final states, then every AFA for L has at least $n + 1$ states.*
(d) *Let L be unary. Then L is accepted by an n-state* BFA (AFA) *if and only if L is accepted by a* DFA *of 2^n states (of which 2^{n-1} are final).*
(e) *If L is accepted by an n-state* BFA (AFA)*, then $\overline{L}$ is accepted by an n-state* BFA (AFA, *respectively*)*.*
(f) *If L is accepted by an n-state* BFA*, then L is accepted by an* AFA *of at most $n + 1$ states, and by an* NNFA *of at most 2^n states.*
(g) *If L is accepted by an n-state* NNFA*, then L is accepted by an* NFA *of at most $n + 1$ states and by a* pDFA *of at most $2^n - 1$ states. If L is accepted by an n-state* pDFA*, then L is accepted by a* DFA *of at most $n + 1$ states.*

Proof. (a) ($\Rightarrow$; cf. [6, Theorem 4.1, Corollary 4.2] and [9, Lemma 1]).

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.

($\Leftarrow$; cf. [9, Lemma 2]) 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 $Q' = \{q_1, \ldots, q_n\}$, $F' = \{q_\ell \,|\, \mathrm{bin}(k)_\ell = 1\}$, 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 $L(A') = L(A^R)$.

(b)–(d) These are corollaries of case (a).

(e) Let L be accepted by an n-state BFA (AFA). Then, by (a), the language L^R is accepted by a DFA of 2^n states (of which 2^{n-1} are final). Then the complement $\overline{L^R}$ is also accepted by a DFA of 2^n states (of which 2^{n-1} are final). Since $\overline{L^R} = \overline{L}^R$, the claim follows again by (a).

(f) Let L be accepted by an n-state BFA. Then L^R is accepted by a DFA of 2^n states by (a). Add some unreachable final and non-final sink states to get an equivalent DFA of 2^{n+1} states of which 2^n are final. Then, by (a), L is accepted by an $(n+1)$-state AFA. By reversing the 2^n-state DFA for L^R, we get a 2^n-state NNFA for L.

(g) These claims are well-known. □

If u, v, and w are strings over Σ such that $w = uv$, then u is a *prefix* of w and v is a *suffix* of w. A language L is *prefix-closed* (*suffix-closed*) if $w \in L$ implies that every prefix (suffix) of w is in L.

In 1996, Birget [2] studied the state complexity of the "forever" operator $\overline{\Sigma^*\overline{L}}$ on DFAs and NFAs. Here we continue this research and to simplify the exposition, we use the following notation:

$$f_L = \overline{\Sigma^*\overline{L}}. \tag{1}$$

3. Descriptional Complexity of the Forever Operator

We start with an investigation of some properties of the "forever" operator.

Lemma 6 (Properties of $\overline{\Sigma^*\overline{L}}$). *Let L be a regular language and $f_L = \overline{\Sigma^*\overline{L}}$. Then*

(a) $f_L = \{w \in L \,|\, \textit{every suffix of } w \textit{ is in } L\}$;
(b) $f_L = \emptyset$ *if and only if* $\varepsilon \notin L$;
(c) $f_L = L$ *if and only if L is suffix-closed.*
(d) *If L^R is accepted by a* DFA *A, then f_L^R is accepted by a* DFA *obtained from A by replacing each non-final state of A with a non-final sink state and by a* pDFA *obtained from A by omitting each non-final state of A.*

Proof. The claim (a) follows directly from the definition of f_L, and (b) and (c) follow directly from (a).

(d) We have $f_L^R = \overline{\overline{L^R}\Sigma^*}$. To get a DFA for $\overline{L^R}$, we interchange final and non-final states in A. Then, to get a DFA for $\overline{L^R}\Sigma^*$, we replace all the out-transitions in every final state with loops on every input symbol. Finally, to get a DFA for f_L^R,

we again interchange final and non-final states. Now, all non-final states are sink states. We can omit all of them to get a pDFA for f_L^R. □

In what follows we consider six models of finite automata: DFAs, pDFAs, NFAs, NNFAs, AFAs, and BFAs. We try to answer the following question. If a language L is represented by an n-state automaton of some model, how many states are sufficient and necessary in the worst case for an automaton of some other model to accept the language $f_L = \overline{\Sigma^* \overline{L}}$? We first consider upper bounds. Although we have 36 possible trade-offs, it is enough to prove only some of them. The remaining trade-offs follow either from inclusions of some models of finite automata or from Lemma 5. For the (N)NFA-to-(p)DFA trade-offs, we need the Dedekind number $\mathrm{M}(n)$ which counts the number of antichains of subsets of an n-element set. The number $\mathrm{M}(n)$ lies in the order of magnitude $2^{2^{\Theta(n)}}$ [13]:

$$2^{n-\log n} \leq \binom{n}{\lfloor n/2 \rfloor} \leq \log \mathrm{M}(n) \leq \binom{n}{\lfloor n/2 \rfloor} \left(1 + O\left(\frac{\log n}{n}\right)\right) \leq 2^{n+1-(\log n)/2}.$$

It follows that $\log \mathrm{M}(n)$ lies in the order of magnitude $2^{n-|\Theta(\log n)|}$. Moreover, we assume that $\varepsilon \in L$ and $L \neq \Sigma^*$ in the statement of the next theorem because otherwise f_L is empty or equals Σ^* by Lemma 6(b) and (c).

Theorem 7 (Upper Bounds). *Let $n \geq 3$ and $f_L = \overline{\Sigma^* \overline{L}}$. Let L be a regular language such that $\varepsilon \in L$ and $L \neq \Sigma^*$. Let L be accepted by a finite automaton A of n states.*

(1) *If A is a* DFA, *then f_L is accepted by a* DFA *of at most 2^{n-1} states.*
(2) *If A is a* pDFA, *then f_L is accepted by a* pDFA *of at most 2^{n-1} states.*
(3) *If A is an* NFA, *then f_L is accepted by*
   (a) *an* NFA *of at most 2^{n-1} states*;
   (b) *a* pDFA *of at most $\mathrm{M}(n-1)-1$ states.*
(4) *If A is an* NNFA, *then f_L is accepted by*
   (a) *an* NNFA *of at most 2^n-2 states.*
   (b) *a* pDFA *of at most $\mathrm{M}(n)-1$ states.*
(5) *If A is an* AFA, *then f_L is accepted by*
   (a) *an* AFA *of at most n states*;
   (b) *an* NNFA *of at most 2^{n-1} states.*
(6) *If A is a* BFA, *then f_L is accepted by*
   (a) *a* BFA *of at most n states*;
   (b) *an* NNFA *of at most 2^n-1 states.*

Proof. (1) We provide a simple alternative proof to [2, Theorem 1(b)]. We first interchange final and non-final states in A to get the DFA $\overline{A}$ for $\overline{L}$. Then we add a loop on every input symbol in the initial state of $\overline{A}$ to get an NFA N for $\Sigma^* \overline{L}$.

In $\mathcal{D}(N)$, only subsets containing the initial state are reachable. Finally, we again interchange the final and non-final states of $\mathcal{D}(N)$.

(2) Let $A = (Q, \Sigma, \cdot, s, F)$ be an n-state pDFA for L. It is enough to show that the language $\Sigma^*\overline{L}$ is accepted by a DFA of at most $2^{n-1}+1$ states, one of which is final sink state. To get an $(n+1)$-state DFA $\overline{A}$ for $\overline{L}$, we first add a new non-final sink state q_d to A. Then, for each transition which is undefined in A, we add the corresponding transition to q_d. Finally, we interchange final and non-final states of the resulting automaton. We construct an $(n+1)$-state NFA N for $\Sigma^*\overline{L}$ from DFA $\overline{A}$, by adding a loop on each input symbol in the initial state s. In the corresponding subset automaton, each reachable subset must contain s. Moreover, the state q_d is a final sink state. It follows that each string is accepted by N from q_d, and therefore each subset containing q_d, is equivalent to $\{q_d\}$. In total, we get at most $2^{n-1}+1$ reachable and pairwise distinguishable states.

(3a) Let $A = (Q, \Sigma, \cdot, s, F)$ be an n-state NFA for L. We reverse A to get an n-state NNFA A^R for L^R with a unique final state s. In the corresponding subset automaton $\mathcal{D}(A^R)$, using Lemma 6(d), we omit all the non-final subsets, that is, all subsets not containing s, to get a 2^{n-1}-state pDFA B for f_L^R. We have two cases. If there is a final subset which is not reachable in B, then we reverse B and add a new initial state to get an NFA for f_L of at most 2^{n-1} states. Otherwise, each final subset, that is, each subset containing s is reachable in B. We show that if a string w is accepted by B from the state $\{s\}$, then w is accepted by B from any other state. The claim holds for $w = \varepsilon$ since all states of B are final. Let $w = a_1 a_2 \cdots a_k$, where $a_i \in \Sigma$, be accepted by B from $\{s\}$. Then in B, we have the following computation:

$$\{s\} \xrightarrow{a_1} S_1 \xrightarrow{a_2} S_2 \xrightarrow{a_3} \cdots \xrightarrow{a_k} S_k,$$

where $s \in S_i$ since each state of B contains s. Notice that any other state S of B is a superset of $\{s\}$. Recall that B is derived from the subset automaton $\mathcal{D}(A^R)$ in which we must have

$$S \xrightarrow{a_1} S'_1 \xrightarrow{a_2} S'_2 \xrightarrow{a_3} \cdots \xrightarrow{a_k} S'_k \tag{2}$$

for some sets $S'_1, S'_2, \ldots, S'_k$ such that $S_i \subseteq S'_i$ $(1 \leq i \leq k)$. Since each S_i contains s, each S'_i must contain s as well. It follows that (2) is a computation in B, so w is accepted by B from S; notice that each set S_i is reachable in B.

We modify pDFA B as follows. We make all states of B non-final, except for $\{s\}$. Next, we add the ε-transition to $\{s\}$ from any other state in B. Denote the resulting NFA by B'. Then $L(B) \subseteq L(B')$. Let us show that $L(B') \subseteq L(B)$. Let w be accepted by B'. If $w = \varepsilon$, then $w \in L(B)$. Otherwise w can be partitioned as $w = w_1 w_2 \cdots w_k$ and in B' we have the following computation on w:

$$F \xrightarrow{w_1} S_1 \xrightarrow{\varepsilon} \{s\} \xrightarrow{w_2} S_2 \xrightarrow{\varepsilon} \{s\} \xrightarrow{w_3} \cdots \xrightarrow{\varepsilon} \{s\} \xrightarrow{w_k} S_k$$

in which while reading each w_i no added ε-transition is used, so $\{s\} \xrightarrow{w_i} S_i$ is a computation in B. As shown above, each w_i is accepted by B from each state of B. It follows that in B we have an accepting computation $F \xrightarrow{w_1} S_1 \xrightarrow{w_2} S'_2 \xrightarrow{w_3} \cdots \xrightarrow{w_k} S'_k$

for some states $S'_2, \ldots, S'_k$ of B. Hence w is accepted by B, so $L(B') \subseteq L(B)$. This means that B' is a 2^{n-1}-state NFA with one final state for f_L^R. By reversing B' and removing ε-transitions, we get a 2^{n-1}-state NFA for f_L.

(3b) It is enough to show that $\Sigma^*\overline{L}$ is accepted by a DFA of at most $\mathrm{M}(n-1)$ states, one of which is a final sink state. Let $A = (Q, \Sigma, \cdot, s, F)$ be an n-state NFA for L, and B be the 2^n-state subset automaton of A. We interchange the final and non-final states in B, to get a 2^n-state DFA $\overline{B}$ for $\overline{L}$. To get a 2^n-state NFA N for $\Sigma^*\overline{L}$, we add a loop on each input symbol in the initial state of the DFA $\overline{B}$. Finally, let C be the subset automaton of N. Then C is a DFA for $\Sigma^*\overline{L}$. Formally, we have

$$
\begin{aligned}
&B = \mathcal{D}(A) = (2^Q, \Sigma, \cdot, \{s\}, F_B) \text{ where } F_B = \{X \subseteq Q \mid X \cap F \neq \emptyset\});\\
&\overline{B} = (2^Q, \Sigma, \cdot, \{s\}, F_{\overline{B}}) \text{ where } F_{\overline{B}} = 2^Q \backslash F_B = \{X \subseteq Q \mid X \subseteq Q \backslash F\});\\
&N = (2^Q, \Sigma, \circ, \{s\}, F_{\overline{B}}) \text{ where for each } X \in 2^Q \text{ and each } a \text{ in } \Sigma,\\
&\qquad\qquad \{s\} \circ a = \{\{s\}, \{s\} \cdot a\} \text{ and } X \circ a = \{X \cdot a\} \text{ if } X \neq \{s\};\\
&C = \mathcal{D}(N) = (2^{2^Q}, \Sigma, \circ, \{\{s\}\}, F_C) \text{ where } F_C = \{X \in 2^{2^Q} \mid X \cap F_{\overline{B}} \neq \emptyset\}.
\end{aligned}
$$

Thus, the states of C are sets of subsets of Q, and a state $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ is final if there is an i such that $S_i \subseteq Q \backslash F$. Our aim is to show that C has at most $\mathrm{M}(n-1)$ reachable and pairwise distinguishable states. We first show that each state of C is equivalent to an antichain in 2^Q.

Let $S \subseteq T \subseteq Q$ and w be accepted by N from the state T. Let us show that w is accepted by N also from the state S. If w is accepted from T, then there is a computation in N on w starting in T and ending in a final state T' of N. If this computation does not use any transition which was added to get N from $\overline{B}$, then we have the same computation in $\overline{B}$. This means that $T \cdot w \subseteq Q \backslash F$, and since B is the subset automaton of A, we have $S \cdot w \subseteq T \cdot w \subseteq Q \backslash F$. Therefore w is accepted by $\overline{B}$ from S. Since in N we have the same computation from S on w, the string w is accepted by N from S. Now assume that w is accepted by N from T by a computation using an added transition. Then w can be partitioned as $w = uv$, where $T \xrightarrow{u} \{s\} \xrightarrow{v} T'$ and $T' \subseteq Q \backslash F$, and moreover, while reading u, no added transition is used. It follows that in $\overline{B}$, we have $T \cdot u = \{s\}$, and therefore also $S \cdot u \subseteq \{s\}$. If $S \cdot u = \emptyset$, then $S \cdot w = \emptyset$, and therefore also $S \circ w = \emptyset$, so w is accepted by N from S. If $S \cdot u = \{s\}$, then in N we have $S \xrightarrow{u} \{s\} \xrightarrow{v} T'$, which means that w is accepted by N from S.

Thus if in a state $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ of C we have $S_i \subseteq S_j$ for some i and j, then $\mathcal{S}$ is equivalent to $\mathcal{S} \backslash \{S_j\}$. It follows that each state of C is equivalent to an antichain in 2^Q. Moreover, since N has a loop on each symbol in its initial state $\{s\}$, and C is the subset automaton of N, each reachable state of C must contain the set $\{s\}$, that is, each reachable antichain has a form $\{\{s\}, S_2, S_3, \ldots, S_k\}$, where $k \geq 1$, and $\{S_2, S_3, \ldots, S_k\}$ is an antichain in $2^{Q \backslash \{s\}}$. This gives the upper bound $\mathrm{M}(n-1)$. Notice that the empty antichain corresponds to the initial state $\{\{s\}\}$. We also have to count the antichain $\{\emptyset\}$ which is an unreachable final sink state, but it is equivalent to the reachable state $\{\{s\}, \emptyset\}$.

(4a) If all states of a given NNFA are initial, then L is suffix-closed and $f_L = L$ by Lemma 6(c). Otherwise, L^R is accepted by an 2^n-state DFA which has at least two non-final states. Omit all the non-final states to get a pDFA for f_L^R (cf. Lemma 6(3b)), and reverse the resulting pDFA to get the desired NNFA for f_L.

(4b) It is enough to show that $\Sigma^*\overline{L}$ is accepted by a DFA of at most $\mathrm{M}(n)$ states, one of which is a final sink state. Let $A = (Q, \Sigma, \cdot, I, F)$ be an n-state NNFA for L, and B be the 2^n-state subset automaton of A. In the same way as in the case (3b),

$$\begin{aligned}
&B = \mathcal{D}(A) = (2^Q, \Sigma, \cdot, I, F_B) \text{ where } F_B = \{X \subseteq Q \mid X \cap F \neq \emptyset\});\\
&\overline{B} = (2^Q, \Sigma, \cdot, I, F_{\overline{B}}) \text{ where } F_{\overline{B}} = 2^Q \backslash F_B = \{X \subseteq Q \mid X \subseteq Q \backslash F\});\\
&N = (2^Q, \Sigma, \circ, I, F_{\overline{B}}) \text{ where for each } X \in 2^Q \text{ and each } a \text{ in } \Sigma,\\
&\qquad I \circ a = \{I, I \cdot a\}, \text{ and } X \circ a = \{X \cdot a\} \text{ if } X \neq I;\\
&C = \mathcal{D}(N) = (2^{2^Q}, \Sigma, \circ, \{I\}, F_C) \text{ where } F_C = \{X \in 2^{2^Q} \mid X \cap F_{\overline{B}} \neq \emptyset\}.
\end{aligned}$$

The states of C are sets of subsets of Q, and a state $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ is final if there is an i such that $S_i \subseteq Q \backslash F$. Our aim is to show that C has at most $\mathrm{M}(n)$ reachable and pairwise distinguishable states. We first show that each state of C is equivalent to an antichain in 2^Q. Let $S \subseteq T \subseteq Q$. We show that the state $\{I, S, T\}$ is equivalent to $\{I, S\}$ in C. To this aim, let w be accepted by N from T. If no added transition is used in this computation, then similarly as in the case (3b), w is accepted by C from S. Otherwise $w = uv$ where $T \xrightarrow{u} I \xrightarrow{v} T'$ with $T' \subseteq Q \backslash F$. Since the state I has a loop on each symbol, in C we have $I \xrightarrow{u} I \xrightarrow{v} T'$, so the string w is accepted by C from I. It follows that the state $\{I, S, T\}$ is equivalent to $\{I, S\}$ in C. Thus if in a state $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ of C we have $S_i \subseteq S_j$ for some i and j, then $\mathcal{S}$ is equivalent to $\mathcal{S} \backslash \{S_j\}$. It follows that each state of C is equivalent to an antichain in 2^Q. Since the number of antichains in 2^Q is $\mathrm{M}(n)$, and the reachable state $\{I, \emptyset\}$ is equivalent to the unreachable antichain $\{\emptyset\}$.

(5a) If L is accepted by an n-state AFA, then L^R is accepted by a DFA of 2^n states of which 2^{n-1} are final. Replace each non-final state with a non-final sink state to get a DFA for f_L^R of 2^n states of which 2^{n-1} are final. Hence f_L is accepted by an n-state AFA.

(5b) In the DFA for f_L^R obtained as in case (5a), we omit the non-final sink states to get an equivalent pDFA of 2^{n-1} states. By reversing this pDFA, we get a 2^{n-1}-state NNFA for f_L.

(6a) If A is an n-state BFA, then L^R is accepted by a DFA of 2^n states. Replace each non-final state with a non-final sink state to get a DFA for f_L^R of 2^n states given by Lemma 5(f). Hence f_L is accepted by an n-state BFA.

(6b) In the DFA for f_L^R obtained as in case (6a), we omit the non-final sink states to get an equivalent pDFA of at most $2^n - 1$ states; recall that $L \neq \Sigma^*$. By reversing this pDFA, we get the desired NNFA for f_L. □

Now we turn our attention to lower bounds. We again need to prove only some of them. All the remaining bounds follow from the inclusions of models or from

Lemma 5. However, in some cases, we use witnesses over a smaller alphabet for the bound that follows from some other trade-off. In 32 of 36 cases, our lower bounds meet the upper bounds given by Theorem 7. The remaining four cases are the trade-offs from NNFA to DFA, pDFA, NFA, and NNFA. Except for four trade-offs, our witnesses are defined over a fixed alphabet of size one, two, three, or four. The binary case is always optimal in the sense that there is no unary witness language.

Theorem 8 (Lower Bounds). *Let $n \geq 3$ and $f_L = \overline{\Sigma^* \overline{L}}$. There exists a regular language L accepted by an n-state finite automaton A such that A is*

(1) *a ternary* DFA *and every* BFA *for f_L has at least n states*;
(2) *a ternary* DFA *and every* NNFA *for f_L has at least 2^{n-1} states*;
(3) *a binary* DFA *and every* pDFA *for f_L has at least 2^{n-1} states*;
(4) *a binary* pDFA *and every* BFA *for f_L has at least n states*;
(5) *a quaternary* pDFA *and every* DFA *for f_L has at least $2^{n-1}+1$ states*;
(6) *an* NFA *and every* DFA *for f_L has at least $\mathrm{M}(n-1)$ states*;
(7) *a binary* NNFA *and every* AFA *for f_L has at least $n+1$ states*;
(8) *a unary* AFA *and*
   (a) *every* BFA *for f_L has at least n states*;
   (b) *every* NNFA *for f_L has at least 2^{n-1} states*;
(9) *a binary* AFA *and*
   (a) *every* NFA *for f_L has at least $2^{n-1}+1$ states*;
   (b) *every* DFA *for f_L has at least $2^{2^{n-1}}$ states*;
(10) *a unary* BFA *and*
   (a) *every* AFA *for f_L has at least $n+1$ states*;
   (b) *every* NNFA *for f_L has at least 2^n-1 states*;
(11) *a binary* BFA *and*
   (a) *every* NFA *for f_L has at least 2^n states*;
   (b) *every* DFA *for f_L has at least 2^{2^n-1} states.*

Proof. (1) Let L be the language accepted by the DFA A shown in Fig. 3. We reverse A to get an NFA A^R for L^R. In the corresponding subset automaton, all (final) subsets containing 0 are reachable by Proposition 1, and the empty set is reached from $\{0\}$ by c. Notice that no other subset is reachable. Moreover, the subset automaton does not have equivalent states since the state 0 is uniquely

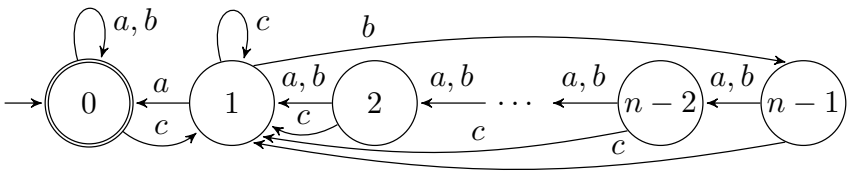


Fig. 3. The DFA for L such that every BFA for $\overline{\Sigma^* \overline{L}}$ has n states.

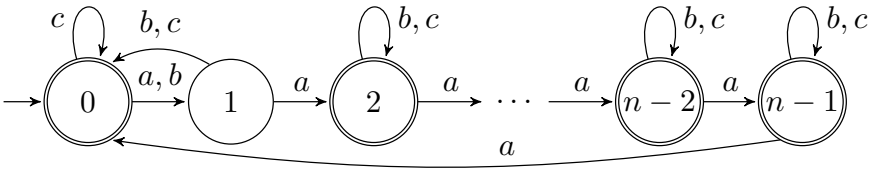


Fig. 4. The DFA from [2] for L such that every NNFA for $\overline{\Sigma^*\overline{L}}$ has 2^{n-1} states.

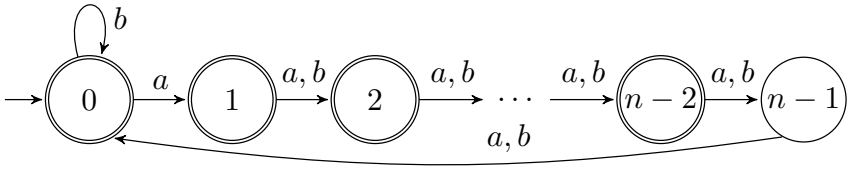


Fig. 5. The binary DFA for L such that every pDFA for $\overline{\Sigma^*\overline{L}}$ has 2^{n-1} states.

distinguishable in A^R by ε, and it is uniquely reachable from any other state of A^R through unique in-transitions $2 \xrightarrow{b} 3 \xrightarrow{b} \cdots \xrightarrow{b} n-1 \xrightarrow{b} 1 \xrightarrow{c} 0$. Since in the minimal DFA for L^R we have all states final but one non-final sink state, the language L^R is prefix-closed. Therefore L is suffix-closed, so $f_L = L$. Since the minimal DFA for L^R has $2^{n-1}+1$ states, every BFA for L, so for f_L, has at least n states.

(2) This case follows from [2, Proof of Theorem 2(a)].

(3) Let L be accepted by the n-state DFA A shown in Fig. 5. We construct an n-state NFA N for $\Sigma^*\overline{L}$ by interchanging final and non-final states in A and by adding the transition $(0, a, 0)$. It is enough to prove that the subset automaton $\mathcal{D}(N)$ has at least 2^{n-1} reachable and pairwise distinguishable states. We prove reachability by using Proposition 1. To prove distinguishability, notice that the state $n-1$ is uniquely distinguishable by ε in N and it is uniquely reachable from any other state through unique in-transitions on a. By Proposition 2, the subset automaton $\mathcal{D}(N)$ does not have equivalent states. Since $\mathcal{D}(N)$ has no non-final sink state, it is also a minimal pDFA. Notice that the lower bound 2^{n-1} for a DFA accepting f_L follows from the proof. In [2, Proof of Theorem 2(b)], it is claimed that this bound is met by the complement of binary language $a\{a,b\}^{n-2}$. However, the minimal DFA for this language has $n+1$ states.

(4) Let L be the language accepted by the pDFA A shown in Fig. 6. We reverse A to get an NNFA for L^R. In the corresponding subset automaton, we replace every non-final state with a single non-final sink state. By Lemma 6(d), we get a DFA B for f_L^R with 2^{n-1} final states. We reach all of them using induction on the size of subsets. The set $\{0, 1, \ldots, n-1\}$ of size n is the basis. Each subset S with $0 \in S$ and $t \notin S$ of size $k-1$, where $1 \leq k \leq n$, is reachable from the subset $S \cup \{t\}$ of size k by the string $a^{t-1}ba^{n-t}$. By doing this, we always keep the state 0 in the set since symbol a performs a loop on state 0 and symbol b moves state 1 to 0. This means that all sets with 0 are reachable in B through final states. The non-final empty set is reached from $\{0\}$ on b. For distinguishability, let S and T be two different subsets

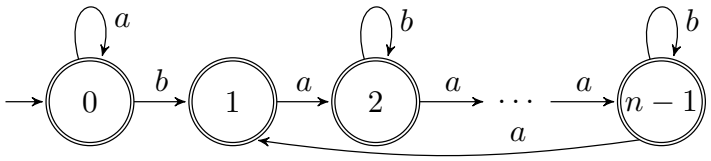


Fig. 6. The pDFA for L such that every BFA for $\overline{\Sigma^*\overline{L}}$ has n states.

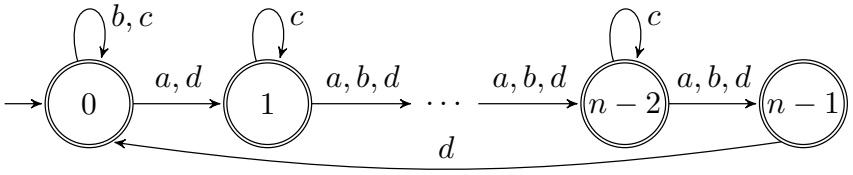


Fig. 7. The pDFA for L such that every DFA for $\overline{\Sigma^*\overline{L}}$ has $2^{n-1}+1$ states.

such that $t \notin S$ and $t \in T$. Then $a^{t-1}b$ is rejected from S and accepted from T. Since a performs a loop on 0 and b moves 1 to 0, we are always in final states. So we distinguish every pair of 2^{n-1} final states. By Lemma 5, every BFA for f_L has at least n states.

(5) Let L be the language accepted by the pDFA A shown in Fig. 7. We construct an $(n+1)$-state NFA N for $\Sigma^*\overline{L}$ as follows. First, we add a new non-final sink state n and the transitions on a, b, c from $n-1$ to n. Then we make state n final, and all the remaining states non-final. Finally, we add the transitions $(0, a, 0)$ and $(0, d, 0)$. By Proposition 1, in the subset automaton $\mathcal{D}(N)$, all the subsets containing 0 are reachable from the initial subset $\{0\}$ via strings over $\{a, b\}$. All the subsets containing n are final and equivalent to $\{n\}$. All the remaining subsets are non-final. Two distinct subsets of $\{0, 1, \ldots, n-1\}$ differ in a state i, and the string $d^{n-1-i}c$ distinguishes the two subsets; notice that as for the states in $\{0, 1, \ldots, n-1\}$, the string c is accepted only from $n-1$, the state $n-1$ is uniquely reachable from any other state through unique in-transitions $0 \xrightarrow{d} 1 \xrightarrow{d} \cdots \xrightarrow{d} n-1$, and therefore $d^{n-1-i}c$ is accepted only from the state i. Thus the subset automaton $\mathcal{D}(N)$ has at least $2^{n-1}+1$ reachable and pairwise distinguishable states. It is claimed in [2, Theorem 1(b)] that the upper bound in this case is 2^{n-1}. The proof of [2, Theorem 1(b)] does not work in the case of a partial automaton for L since in such a case we can reach an accepting state in the automaton $\mathbf{B}$ also by a string which is not in f_L. Our witness fixes this small inaccuracy.

(6) Let L be accepted by the n-state NFA $A = (Q, \Sigma, \cdot, 0, F)$, where $Q = \{0, 1, \ldots, n-1\}$, $\Sigma = \{a_X, b_X \mid X \subseteq Q\}$, $F = Q \setminus \{n-1\}$, and the transition function is defined as follows:

$$0 \cdot a_X = X \text{ and } i \cdot a_X = \{i\} \text{ if } i \neq 0,$$

$$i \cdot b_X = \begin{cases} \{n-1\}, & \text{if } i \in X; \\ \{0\}, & \text{if } i \notin X; \end{cases}$$

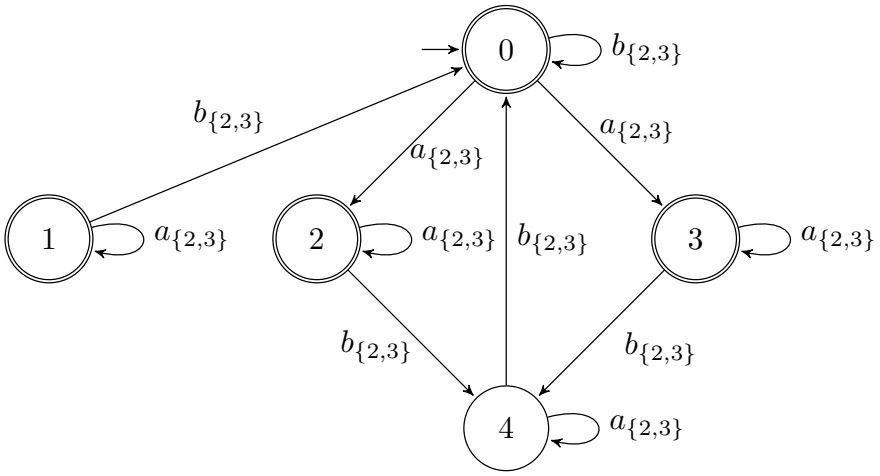


Fig. 8. The example of NFA for L such that every DFA for $\overline{\Sigma^*\overline{L}}$ has $\mathrm{M}(n-1)$ states. The alphabet is $\{a_X, b_X \mid X \subseteq Q\}$, only transitions on $a_{\{2,3\}}$ and $b_{\{2,3\}}$ are shown.

see Fig. 8 for an illustration. Then

$$
\begin{aligned}
&B = \mathcal{D}(A) = (2^Q, \Sigma, \cdot, \{0\}, 2^Q \backslash \{\{n-1\}, \emptyset\});\\
&\overline{B} = (2^Q, \Sigma, \cdot, \{0\}, \{\{n-1\}, \emptyset\});\\
&N = (2^Q, \Sigma, \circ, \{0\}, \{\{n-1\}, \emptyset\}) \text{ where } \{0\} \circ a = \{0\} \cup \{0\} \cdot a \text{ and}\\
&\qquad\qquad X \circ a = X \cdot a \text{ if } X \neq \{0\};\\
&C = \mathcal{D}(N) = (2^{2^Q}, \Sigma, \circ, \{\{0\}\}, \{X \in 2^{2^Q} \mid X \cap \{\{n-1\}, \emptyset\} \neq \emptyset\}).
\end{aligned}
$$

Our aim is to show that C has at least $\mathrm{M}(n-1)$ reachable and distinguishable states. Let $S_1, S_2, \ldots, S_k$ be subsets of Q such that $0 \notin S_i$ for every i. Then in C we have $\{\{0\}\} \xrightarrow{a_{S_1}} \{\{0\}, S_1\} \xrightarrow{a_{S_2}} \{\{0\}, S_1, S_2\} \xrightarrow{a_{S_3}} \cdots \xrightarrow{a_{S_k}} \{\{0\}, S_1, S_2, \ldots, S_k\}$. It follows that every state $\mathcal{S} = \{\{0\}, S_1, S_2, \ldots, S_k\}$ where $\{S_1, S_2, \ldots, S_k\}$ is an antichain of subsets of $\{1, 2, \ldots, n-1\}$ is reachable. To prove distinguishability, let $\mathcal{S} = \{\{0\}, S_1, S_2, \ldots, S_k\}$ and $\mathcal{T} = \{\{0\}, T_1, T_2, \ldots, T_\ell\}$ be two distinct reachable antichains in C. Then there exists a subset X of $\{1, 2, \ldots, n-1\}$ such that, without loss of generality, $X \in \mathcal{S} \backslash \mathcal{T}$. We have two cases. (i) No subset of X is in $\mathcal{T}$. Then b_X is accepted from $\mathcal{S}$ since $X \in \mathcal{S}$, $X \circ b_X = X \cdot b_X = \{n-1\}$, and therefore $\{n-1\} \in \mathcal{S} \circ b_X$. Hence b_X is accepted by C from $\mathcal{S}$. On the other hand, we have $0 \in \{0\} \circ b_X$. Next, since T_j $(1 \leq j \leq \ell)$ is not a subset of X, there is a state i such that $i \in T_j \backslash X$. Since $i \cdot b_X = \{0\}$, we must have $0 \in T_j \circ b_X$. Hence b_X is rejected by C from $\mathcal{T}$. (ii) There is a subset Y of X such that $Y \in \mathcal{T}$. Then no subset of Y is in $\mathcal{S}$ since $\mathcal{S}$ is an antichain and $X \in \mathcal{S}$. By the former case, b_Y is accepted from $\mathcal{T}$ and rejected from $\mathcal{S}$. Thus C has $\mathrm{M}(n-1)$ reachable and distinguishable states.

(7) Let L be accepted by the NNFA A from Fig. 9. Since each state of A is initial, L is suffix-closed, so $f_L = L$. To show that every AFA for f_L, so for L, has $n+1$ states, it is enough to show that the minimal DFA for L^R has more than 2^{n-1} final states. Since the reverse of A is isomorphic to A, we have $L^R = L$. In

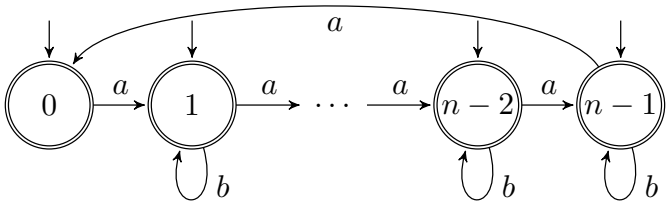


Fig. 9. The NNFA for L such that every AFA for $\overline{\Sigma^*\overline{L}}$ has $n+1$ states.

the subset automaton $\mathcal{D}(A)$, the initial subset is $\{0,1,\ldots,n-1\}$. By Proposition 1, every subset is reachable in $\mathcal{D}(A)$. Next, the state 0 is uniquely distinguishable in A by the string $(ab)^{n-1}$, and it is uniquely reachable from any other state of A. Thus $\mathcal{D}(A)$ is minimal. Since $\mathcal{D}(A)$ has 2^n-1 final states, every AFA for L, so for f_L, has at least $n+1$ states.

(8) Let $L=\{a^i \mid 0 \le i \le 2^{n-1}-1\}$. Then L is a unary language accepted by a 2^n-state DFA with 2^{n-1} final states. So L is accepted by an n-state AFA. Since L is suffix-closed, $f_L = L$. (a) Since the minimal DFA for L has $2^{n-1}+1$ states, every BFA for L has at least n states. (b) The longest string in L is of length $2^{n-1}-1$, and therefore every NNFA for L has at least 2^{n-1} states.

(9) Set $m=2^{n-1}$ in Fig. 2. Let K be accepted by the 2^n-state DFA A in which the transitions on final states $0,1,\ldots,2^{n-1}-1$ are shown in Fig. 2. Moreover, A has 2^{n-1} non-final sink states. Set $L=K^R$. Then L is accepted by an n-state AFA. By Lemma 6(d), if we omit all non-final states of A, we get a pDFA for f_L^R. By Proposition 4, we get that (a) every NFA for f_L has at least $2^{n-1}+1$ states, and (b) every DFA for f_L has $2^{2^{n-1}}$ states.

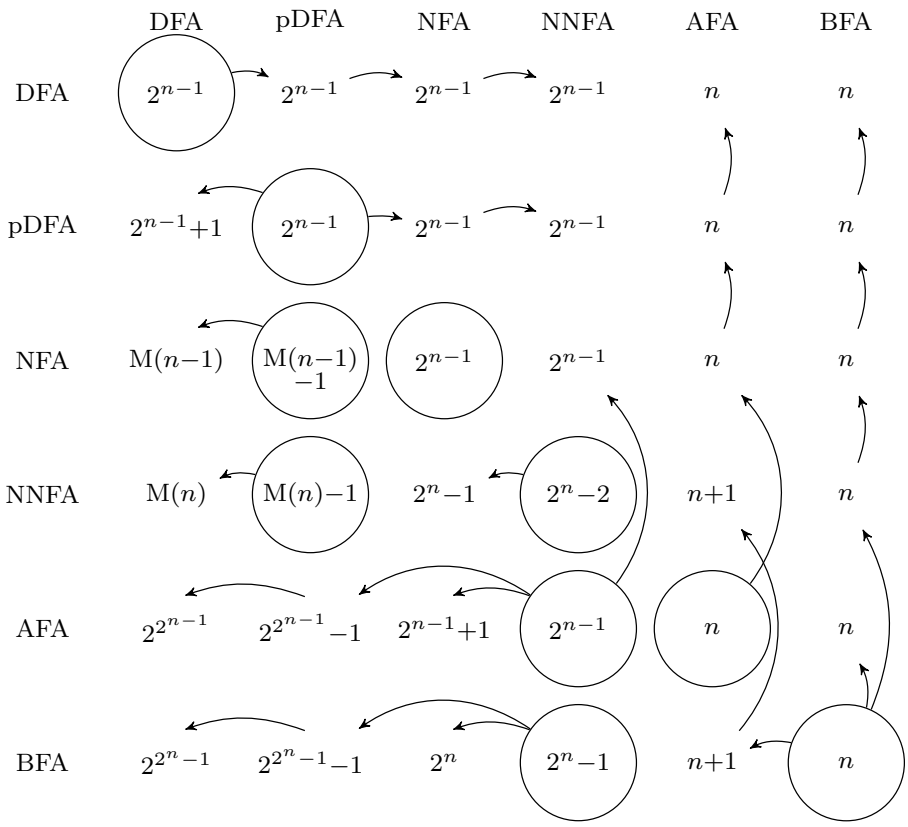
(10) Let $L=\{a^i \mid 0 \le i \le 2^n-2\}$. Then L is a unary language accepted by a minimal 2^n-state DFA A, so L is accepted by an n-state BFA. Since L is suffix-closed, $f_L = L$. (a) Every AFA accepting L has at least $n+1$ states since the number of final states in A is greater than 2^{n-1}. (b) The longest string in L is of length 2^n-2, and therefore every NNFA for L has at least 2^n-1 states.

(11) Set $m=2^n-1$ in Fig. 2. Now the proof goes exactly the same way as in case (9). It results in the lower bound 2^n for NFAs and 2^{2^n-1} for DFAs. □

The upper and lower bounds from Theorems 7 and 8 are shown in Table 1 as circled. If an upper or lower bound in one cell follows from the bound in another cell, this is denoted by an arrow. Another witness for the same lower bound, but using a smaller alphabet, is circled dashed. As a corollary, we get the results that are displayed in Table 2. The table also shows the size of alphabets used for describing witness languages.

Table 3 shows the upper bounds on the complexity of the forever operator on unary regular languages; here $\mathrm{F}(n)$ denotes the Landau function defined as $\mathrm{F}(n) = \max\{\mathrm{lcm}(x_1,\ldots,x_k) \mid n = x_1+\cdots+x_k\}$. In six cases, namely {AFA, BFA}-to-{NNFA, AFA, BFA}, the upper bounds are the same as for general languages and

Table 1. Upper (top) and lower bounds (bottom) on the complexity of $\overline{\Sigma^* \overline{L}}$.



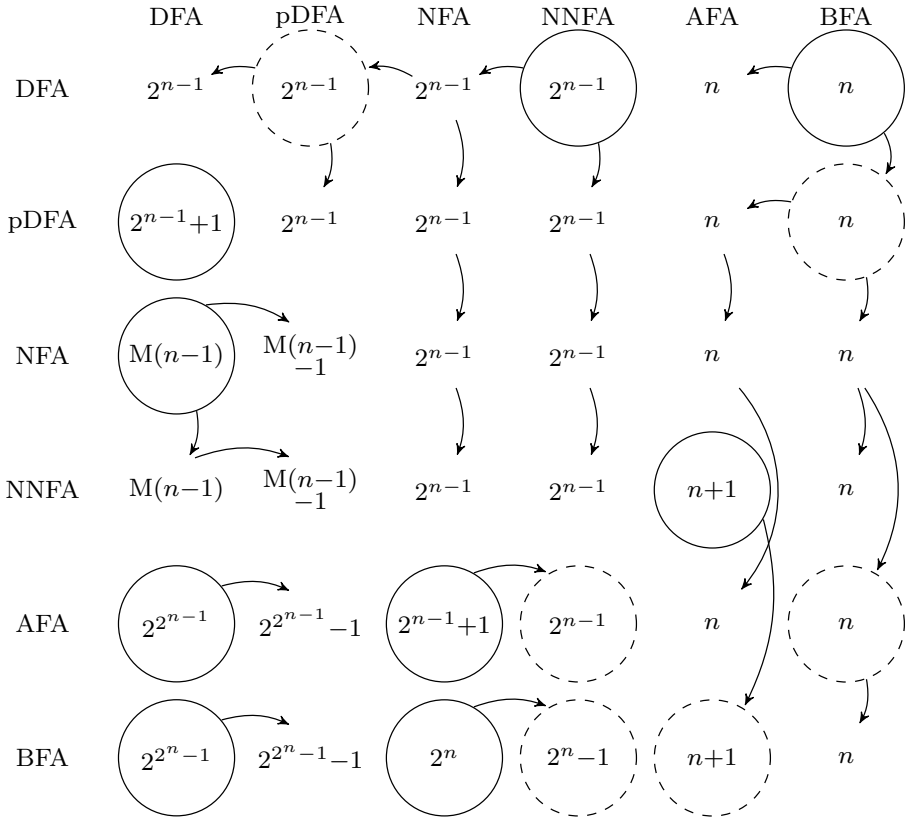


Table 2. The complexity of $\overline{\Sigma^*\overline{L}}$ for various types of finite automata. The DFA-NFA and DFA-NNFA trade-offs are from [2].

$L\backslash\overline{\Sigma^*\overline{L}}$	DFA	$\lvert\Sigma\rvert$	pDFA	$\lvert\Sigma\rvert$	NFA	$\lvert\Sigma\rvert$	NNFA	$\lvert\Sigma\rvert$	AFA	$\lvert\Sigma\rvert$	BFA	
DFA	2^{n-1}	2	2^{n-1}	2	2^{n-1}	3	2^{n-1}	3	n	3	n	3
pDFA	$2^{n-1}+1$	4	2^{n-1}	2	2^{n-1}	3	2^{n-1}	3	n	2	n	2
NFA	$\mathrm{M}(n-1)$	2^{n+1}	$\mathrm{M}(n-1)-1$	2^{n+1}	2^{n-1}	3	2^{n-1}	3	n	2	n	2
NNFA	$\geq \mathrm{M}(n-1)2^{n+1}$ $\leq \mathrm{M}(n)$		$\geq \mathrm{M}(n-1)-12^{n+1}$ $\leq \mathrm{M}(n)-1$		$\geq 2^{n-1}$ $\leq 2^n-1$	3	$\geq 2^{n-1}$ $\leq 2^n-2$	3	$n+1$	2	n	2
AFA	$2^{2^{n-1}}$	2	$2^{2^{n-1}}-1$	2	$2^{n-1}+12$		2^{n-1}	1	n	1	n	1
BFA	2^{2^n-1}	2	$2^{2^n-1}-1$	2	2^n	2	2^n-1	1	$n+1$	1	n	1

Table 3. The upper bounds on the complexity of forever operator in the unary case. We have $\mathrm{D}(n)=\mathrm{F}(n-1)+n^2-2\in 2^{O(\sqrt{n\log n})}$ and $\lceil\log(\mathrm{D}(n))\rceil<n$.

$L\backslash\overline{\Sigma^*\overline{L}}$	DFA	pDFA	NFA	NNFA	AFA	BFA
DFA	n	$n-1$	$n-1$	$n-1$	$\lceil\log n\rceil+1$	$\lceil\log n\rceil$
pDFA	$n+1$	n	n	n	$\lceil\log n\rceil+1$	$\lceil\log(n+1)\rceil$
NFA	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\lceil\log(\mathrm{D}(n))\rceil+1$	$\lceil\log(\mathrm{D}(n))\rceil$
NNFA	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\mathrm{D}(n)$	$\lceil\log(\mathrm{D}(n))\rceil+1$	$\lceil\log(\mathrm{D}(n))\rceil$
AFA	$2^{n-1}+1$	2^{n-1}	2^{n-1}	2^{n-1}	n	n
BFA	2^n	2^n-1	2^n-1	2^n-1	$n+1$	n

they are met by unary witnesses. In the remaining cases, the upper bounds are smaller than those in the general case. It follows that a binary alphabet is optimal whenever it is used to describe witnesses in the general case.

4. Conclusions

We investigated the descriptional complexity of $\overline{\Sigma^*\overline{L}}$ over complete and partial deterministic, nondeterministic, alternating, and Boolean finite automata. For each trade-off, except for those starting with NNFAs, we provided tight upper bounds for complexity of $\overline{\Sigma^*\overline{L}}$ depending on the complexity of L. The most interesting result is the tight upper bound on NFA-to-DFA trade-off given by the Dedekind number $\mathrm{M}(n-1)$. However, we used a growing alphabet of size 2^{n+1} to get the lower bound in this case. Except for (N)NFA-to-(p)DFA trade-offs, all witnesses are described over an alphabet of fixed size. Moreover, binary and unary alphabets are optimal for their respective cases. Whenever we have a larger alphabet, we do not know whether or not it is optimal. The precise complexity for NNFA-to-(p)DFA and NNFA-to-(N)NFA trade-offs remains open as well.

Acknowledgements

A preliminary version of this paper was presented at the conference DLT 2017 and appeared in the proceedings. The research was supported by VEGA Grant 2/0084/15 and grant APVV-15-0091.

References

[1] J. Birget, Intersection and union of regular languages and state complexity, *Inf. Process. Lett.* **43**(4) (1992) 185–190.

[2] J. Birget, The state complexity of $\overline{\Sigma^*\overline{L}}$ and its connection with temporal logic, *Inf. Process. Lett.* **58**(4) (1996) 185–188.

[3] J. A. Brzozowski, G. Jirásková, B. Liu, A. Rajasekaran and M. Szykula, On the state complexity of the shuffle of regular languages, *DCFS 2016*, eds. C. Câmpeanu, F. Manea and J. Shallit, LNCS, Vol. 9777 (Springer, 2016), pp. 73–86.

[4] J. A. Brzozowski and E. L. Leiss, On equations for regular languages, finite automata, and sequential networks, *Theoret. Comput. Sci.* **10** (1980) 19–35.

[5] J. Cohen, D. Perrin and J. Pin, On the expressive power of temporal logic, *J. Comput. Syst. Sci.* **46**(3) (1993) 271–294.

[6] A. Fellah, H. Jürgensen and S. Yu, Constructions for alternating finite automata, *Intern. J. Computer Math.* **35**(1–4) (1990) 117–132.

[7] Y. Gao, N. Moreira, R. Reis and S. Yu, A survey on operational state complexity, *CoRR* abs/1509.03254 (2015).

[8] I. Glaister and J. Shallit, A lower bound technique for the size of nondeterministic finite automata, *Inf. Process. Lett.* **59**(2) (1996) 75–77.

[9] 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, Vol. 7353 (Springer, 2012), pp. 196–204.

[10] G. Jirásková and T. Masopust, Complexity in union-free regular languages, *Internat. J. Found. Comput. Sci.* **22**(7) (2011) 1639–1653.

[11] G. Jirásková and G. Pighizzini, Optimal simulation of self-verifying automata by deterministic automata, *Inform. Comput.* **209**(3) (2011) 528–535.

[12] C. A. Kapoutsis, Removing bidirectionality from nondeterministic finite automata, *MFCS 2005*, eds. J. Jedrzejowicz and A. Szepietowski, LNCS, Vol. 3618 (Springer, 2005), pp. 544–555.

[13] D. Kleitman and G. Markowsky, On Dedekind's problem: The number of isotone Boolean functions. II, *Trans. Amer. Math. Soc.* **213** (1975) 373–390.

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

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

[16] A. R. Meyer and M. J. Fischer, Economy of description by automata, grammars, and formal systems, *Proceedings of the* 12*th Annual Symposium on Switching and Automata Theory* (IEEE Computer Society Press, 1971), pp. 188–191.

[17] F. R. Moore, On the bounds for state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way finite automata, *IEEE Transaction on Computing* **C-20** (1971) 1211–1219.

[18] M. O. Rabin and D. Scott, Finite automata and their decision problems, *IBM* **3** (1959) 114–125.
[19] M. Sipser, *Introduction to the Theory of Computation* (Cengage Learning, 2012).
[20] Yu. L. Yershov, On a conjecture of V. A. Uspenskii, *Algebra i Logika* (*Seminar*) **1** (1962) 45–48 (in Russian).
[21] S. Yu, *Handbook of Formal Languages*: *Volume* 1 *Word*, *Language*, *Grammar* (Springer, Heidelberg, 1997), Heidelberg, ch. Regular Languages, pp. 41–110.
[22] S. Yu, Q. Zhuang and K. Salomaa, The state complexities of some basic operations on regular languages, *Theoret. Comput. Sci.* **125**(2) (1994) 315–328.