Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Supplementary Information and Transducers

Bachelor's Thesis

2019
Matúš Juran

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Supplementary Information and Transducers

Bachelor's Thesis

Study programme:	Computer Science
Study field:	2508 Computer Science
Department:	Department of Computer Science
Supervisor:	prof. RNDr. Branislav Rovan, PhD.

Bratislava, 2019
Matúš Juran

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname:	Matúš Juran
Study programme:	Computer Science (Single degree study, bachelor I. deg., full time form)
Field of Study:	Computer Science, Informatics
Type of Thesis:	Bachelor´s thesis
Language of Thesis:	English
Secondary language:	Slovak

Title: Supplementary Information and Transducers

Annotation: The thesis continues the line of research concerning various aspects of the notion of information pursued at the Department of Computer Science of the Comenius University. The aim of the thesis is to define and develop a new approach to defining the notion of usefulness of information using transducers and exhibit similarities and differences to the existing setting using deterministic and nondeterministic finite automata.

Supervisor:	prof. RNDr. Branislav Rovan, PhD.
Department:	FMFI.KI - Department of Computer Science
Head of department:	prof. RNDr. Martin Škoviera, PhD.
Assigned:	23.10.2018

Approved: 24.10.2018 doc. RNDr. Daniel Olejár, PhD.
Guarantor of Study Programme

.....
Student

.....
Supervisor

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Matúš Juran
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Supplementary Information and Transducers
Dodatočná informácia a prekladače

Anotácia: Práca je pokračovaním výskumu rôznych aspektov pojmu informácia na Katedre informatiky FMFI UK. Cieľom práce je definovať a preskúmať nový pohľad na pojem užitočnosti informácie využitím prekladačov a nájsť podobnosť resp. rozdiel s doterajším prístupom využívajúcim deterministické a nedeterministické konečné automaty.

Vedúci: prof. RNDr. Branislav Rovan, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: prof. RNDr. Martin Škoviera, PhD.

Spôsob sprístupnenia elektronickej verzie práce:
bez obmedzenia

Dátum zadania: 23.10.2018

Dátum schválenia: 24.10.2018

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Acknowledgement: I would like to thank my supervisor prof. RNDr. Branislav Rovan, PhD. for his guidance and useful advice.

Abstract

This thesis is a part of the research on various aspects of the notion of information. Supplementary information (advice) may reduce the complexity of solving a problem in some cases. The notion of usefulness of supplementary information was studied in the finite automata setting and in the deterministic push-down automata setting. In this thesis, we study the possibility of exploiting supplementary information in finite transducers setting. The notion of advice is formalized as a regular language which the transducer transforms to a different regular language. We consider the advice to be useful if the transducer requires fewer states than it would require to transform the language of all strings over the input alphabet. This approach can be used with either deterministic or nondeterministic transducer as a computational model. We define families of languages for which useful advice exists and show particular languages that belong to these families. We compare these families to the families defined by decomposability of automata studied earlier and study their closure properties. We show that some modifications of the advice can affect its usefulness. We also study the families defined by a fixed advice language.

Keywords: supplementary information, transducers, state complexity, regular languages

Abstrakt

Táto práca je súčasťou výskumu rôznych aspektov informácie. Prídavná informácia (rada) môže v niektorých prípadoch zjednodušiť riešenie problému. Koncept užitočnosti prídavnej informácie bol už skúmaný v súvislosti s konečnými automatmi a deterministickými zásobníkovými automatmi. V tejto práci skúmame možnosť využitia dodatočnej informácie v súvislosti s prekladačmi. Prídavná informácia formalizujeme ako regulárny jazyk, ktorý prekladač transformuje na iný jazyk. Informáciu považujeme za užitočnú ak nový prekladač bude potrebovať menej stavov než by bolo potrebných na transformovanie jazyka všetkých slov nad danou abecedou. Tento prístup je použiteľný pre deterministické aj nedeterministické prekladače. V práci definujeme triedy jazykov, pre ktoré existuje užitočná rada a ukazujeme konkrétne jazyky patriace do týchto tried. Porovnávame tieto triedy s už skúmanými triedami určenými rozložiteľnosťou automatov a skúmame ich uzáverové vlastnosti. Ukazujeme, ako niektoré modifikácie môžu ovplyvniť užitočnosť rady. Skúmame tiež triedy jazykov, pre ktoré je radou jeden konkrétny jazyk.

Kľúčové slová: prídavná informácia, prekladače, stavová zložitosť, regulárne jayzky

Contents

Introduction

The notion of supplementary information and its usefulness has been studied in the setting of deterministic finite automata [1], nondeterministic finite automata [7] and deterministic pushdown automata [3]. In these settings, advice was formalized as a regular language L_{adv}. In a computation with advice, it was assumed that every input word shall belong to the advice language. Without any supplementary information, any word can appear as input. Standard computations without advice are therefore equivalent to computations with the advice Σ_L^*. In finite automata setting, the advice L_{adv} was considered useful for the language L if the minimal automaton accepting the language L using the advice required fewer states than the minimal automaton accepting L without advice. Moreover, the state complexity of L_{adv} was required to be lower than the state complexity of L. This approach is equivalent to finding two simpler regular languages such that $L_1 \cap L_2 = L$. However, the intersection operation is rather limiting. Most importantly, it must hold that $L \subseteq L_{adv}$. For example, the language $\{a^4\}$ can not be used as advice for the language $\{a^8\}$, although they can be intuitively seen as similar.

Finite state transducer is a computational model which can be viewed as a finite state automaton augmented with an output function. Besides intersection with a regular language, transducers can also realize homomorphisms and inverse homomorphisms. [6] The possibility of using them to transform the advice was already studied [8]. In this thesis, we shall study the use of transducers in a different way. Since they can realize an intersection with a regular language, it can be easily seen that for every regular language L there is a transducer M that can transform Σ_L^* into L. We shall consider the language L_{adv} to be a useful supplementary information if a transducer that transforms it to L having fewer states than the one with no advice exists.

First, we shall provide necessary preliminaries and definitions. Next, we shall show particular examples of languages for which useful advice exists and for which it does not. We shall also study the relationship between the families $\mathscr{L}_{DTA}, \mathscr{L}_{NTA}$, deterministically decomposable and nondeterministically decomposable languages. Then, we shall study closure properties of the families $\mathscr{L}_{DTA}$ and $\mathscr{L}_{NTA}$. Finally, we shall study the families generated by a fixed advice language and the effects of advice modifications.

Chapter 1

Preliminaries

In this chapter, we shall provide necessary definitions and present the relevant results in the area of automata and language decompositions. We shall also present known techniques for calculating the minimal number of states necessary to recognize a regular language by a deterministic and a nondeterministic finite state automaton. However, it is assumed that the reader already has some elementary knowledge of automata theory.

1.1 Definitions

In this section, we shall present formal definitions of models and concepts used throughout this thesis.

Transducers are models of computation used to transform languages. In general, they can be viewed as finite automata augmented by an output function. The models we shall use are modifications of the *sequential transducer* defined by Ginsburg. [2]

Definition 1. *A deterministic sequential transducer is a 6-tuple* $M = (K, \Sigma_1, \Sigma_2, H, q_0, F)$ *where* K *is a finite set of states,* Σ_1 *is the input alphabet,* Σ_2 *is the output alphabet,* q_0 *is the initial state,* F *is the set of accepting states and* $H \subseteq K \times (\Sigma_1 \cup \{\varepsilon\}) \times (\Sigma_2 \cup \{\varepsilon\}) \times K$ *is a transition relation.* H *must also fulfil the following conditions:*

1. *If* $(q, c_1, c_2, p_1) \in H$ *and* $(q, c_1, c_3, p_2) \in H$, *then* $c_2 = c_3$ *and* $p_1 = p_2$
2. *If* $(q, \varepsilon, c_1, p_1) \in H$ *and* $(q, c_2, c_3, p_2) \in H$, *then* $c_2 = \varepsilon, c_1 = c_3$ *and* $p_1 = p_2$
3. *There does not exist a sequence of states* $q_1, \ldots, q_n$ *and a sequence of symbols* $c_1, \ldots, c_{n-1}$ *such that* $q_1 \in F, q_n \in F$ *and* $\forall i \in \{1, \ldots n-1\} : (q_i, \varepsilon, c_i, q_{i+1}) \in H$

If ε can be read in some state, no alphabet symbol may be read in this state. Moreover, for every state and input symbol, the next state and the output are

uniquely determined. This is comparable to the way determinism is achieved for deterministic pushdown automata, which is described in [6]. Without rules of the form (q, ε, c, p), it would be impossible to produce an output word longer than the input word. The condition 3. is required to guarantee there shall be at most one accepting computation for any input word. Due to this condition, it can be assumed that the computation shall halt once the entire input is read and an accepting state is reached. At most one symbol can be read and written at a time in order to achieve state complexity comparable to that of finite state automata.

Definition 2. *A configuration of a transducer is a 3-tuple (q, u, v), where q is a state, $u \in \Sigma_1^*$ is the remaining input and $v \in \Sigma_2^*$ is the content of the output tape.*

Definition 3. *A computation step of a transducer is a relation $\vdash$ on the set of configurations such that $(q_1, cu, v) \vdash (q_2, u, vd)$ iff $(q_1, c, d, q_2) \in H$.*

We shall also consider the nondeterministic variant of transducers.

Definition 4. *A nondeterministic sequential transducer is a 6-tuple $M = (K, \Sigma_1, \Sigma_2, H, q_0, F)$ where $K, \Sigma_1, \Sigma_2, q_0$ and F are interpreted the same way as for deterministic transducers and H is a subset of $K \times (\Sigma_1 \cup \{\varepsilon\}) \times (\Sigma_2 \cup \{\varepsilon\}) \times K$ without any further restrictions. A configuration and a computation step are defined the same way as for deterministic transducers.*

Definition 5. *The image of a language L is defined as $M(L) = \{w \mid \exists q_F \in F \ \exists u \in L : (q_0, u, \varepsilon) \vdash^* (q_F, \varepsilon, w)\}$.*

Notation 1. *We shall denote the number of states of a transducer M by $\#_S(M)$. The same notation shall be used to denote the number of states of a finite automaton.*

As noted in [6], the transducers defined this way are a normal form of transducers with rules of the form $H \subseteq K \times \Sigma_1^* \times \Sigma_2^* \times K$. This normal form was chosen to keep the number of states comparable to the number of states of finite automata without limiting the model's computational power. Thus, regular languages are closed under sequential transduction and for every pair of regular languages L_1 and L_2 where $L_1 \neq \emptyset$, there is a nondeterministic transducer M such that $M(L_1) = L_2$. However, we shall show that there exist languages L_1 and L_2 such that L_1 can not be transformed into L_2 by any deterministic transducer in Chapter 2.

The concept of supplementary information in the finite automata setting is closely related to automata decompositions. These areas were studied by Gaži [1] and Sádovský [7].

Definition 6. *Deterministic state complexity of a regular language L, denoted by $sc(L)$, is defined as $min\{\#_S(A) \mid L(A) = L\}$ where A is a finite state automaton. Nondeterministic state complexity is defined similarly and denoted by $nsc(L)$.*

Definition 7. *A regular language L is decomposable if there exist languages L_1 and L_2 such that $L = L_1 \cap L_2$, $sc(L_1) < sc(L)$ and $sc(L_2) < sc(L)$.*

Definition 8. *A regular language L is nondeterministically decomposable if there exist languages L_1 and L_2 such that $L = L_1 \cap L_2$, $nsc(L_1) < nsc(L)$ and $nsc(L_2) < nsc(L)$.*

With advice L_{adv}, it can be assumed that every input shall belong to L_{adv}. Using a suitable advice may allow us to construct an automaton accepting L with fewer states than would be necessary without any advice. However, the state complexity of the advice must also be strictly lower than the state complexity of L. It can be easily seen that the acceptance of the language L can be simplified by some supplementary information iff it is decomposable. In this thesis, we shall be using the supplementary information in a different way. Given a regular language L and advice L_{adv}, we shall be interested in whether there exists a simpler (in terms of state complexity) transducer that can transform L_{adv} into L.

Definition 9. *Transducer state complexity of language L, denoted by $tsc(L)$, is defined as $min\{\#_S(M) \mid M(\Sigma_1^*) = L\}$ where M is a deterministic sequential transducer. For nondeterministic sequential transducers, state complexity of language L is defined the same way and denoted by $ntsc(L)$.*

Next, we shall formally define the notion of usefulness of information in the sequential transducer setting.

Definition 10. *Let L_{adv} be a regular language. The family $\mathscr{L}_D(L_{adv})$ is defined as $\mathscr{L}_D(L_{adv}) = \{L \mid sc(L_{adv}) < tsc(L), \exists M : M$ is a deterministic sequential transducer, $M(L_{adv}) = L$, $|\Sigma_{L_{adv}}| \leq |\Sigma_L|$, $\#_S(M) < tsc(L)\}$ and the family $\mathscr{L}_N(L_{adv})$ is defined as $\mathscr{L}_N(L_{adv}) = \{L \mid nsc(L_{adv}) < ntsc(L), \exists M : M$ is a nondeterministic sequential transducer, $M(L_{adv}) = L$, $|\Sigma_{L_{adv}}| \leq |\Sigma_L|$, $\#_S(M) < ntsc(L)\}$.*

Note that we are only interested in cases where the state complexity of both the automaton and the new transducer is strictly lower than the state complexity of the original transducer. Any language could be used as advice for itself, but we shall consider such advice to be trivial. Also note that the advice alphabet must not contain more symbols than the original one does. It would also be possible to omit this requirement. Later. we shall show that these two definitions would not be equivalent.

Definition 11. *The family $\mathscr{L}_{DTA}$ is defined as $\mathscr{L}_{DTA} = \{L \mid \exists L_{adv} : L \in \mathscr{L}_D(L_{adv})\}$. The family $\mathscr{L}_{NTA}$ is defined as $\mathscr{L}_{NTA} = \{L \mid \exists L_{adv} : L \in \mathscr{L}_N(L_{adv})\}$.*

1.2 Previous results and existing techniques

In this section, we shall present previous results that we shall refer to later and existing techniques for computing the state complexity of languages.

We shall use the following lemmas to study the relationship between the family $\mathscr{L}_{NTA}$ and nondeterministically decomposable languages. They are taken from the article [7].

Lemma 1. *Let L be a language containing a single word w. The language L is nondeterministically decomposable if and only if w contains at least two distinct symbols.*

Lemma 2. *Let $L = \{a^{kn} \mid n \in \mathbb{N}\}$. The language L is nondeterministically decomposable if and only if n is not a power of a prime.*

The following lemma is a corollary of the Myhill–Nerode theorem. The Myhill–Nerode theorem itself and its proof can be found in [6].

Lemma 3. *Let L be a regular language, let R be a relation on Σ_L^* such that $\forall u, v \in \Sigma_L^* : uRv \Leftrightarrow (\forall w \in \Sigma_L^* : uw \in L \Leftrightarrow vw \in L)$. Let n be the number of equivalence classes of R. Then $sc(L) = n$.*

The Myhill–Nerode theorem can not be used to compute the state complexity of nondeterministic automata. The *extended fooling set technique* can be used instead. The definition is taken from Palioudakis [5]. However, the lower bound of the state complexity computed in this way is not necessarily tight.

Definition 12. *An extended fooling set for language L is a set $P = \{(x_i, y_i) \mid 1 \leq i \leq n\}$ such that $x_i y_i \in L$ and if $i \neq j$, then $x_i y_j \notin L$ or $x_j y_i \notin L$.*

Theorem 1. *Let P be an extended fooling set for language L. Then $nsc(L) \geq |P|$.*

Proof. By contradiction. Let P be an extended fooling set for L, let $|P| > nsc(L)$. Let A be the minimal automaton accepting L. Using the pigeonhole principle, it can be shown that there exist two distinct indices i, j such that $\delta(q_0, x_i) = \delta(q_0, x_j) = q$ for some state q. Since $x_i y_i \in L$ and $x_j y_j \in L$, it must hold that $x_i y_j \in L$ and $x_j y_i \in L$. This contradicts the definition of an extended fooling set. □

The following simple observation allows us to compute a lower bound of $sc(L)$ based on the value of $nsc(L)$.

Lemma 4. *For every regular language L, $sc(L) \geq nsc(L)$.*

Proof. Let $A = (K, \Sigma, \delta, q_0, F)$ be a minimal DFA accepting L and let $A' = (K, \Sigma, \delta', q_0, F)$ be an NFA. Let $\forall q \in K\ \forall c \in \Sigma : \delta'(q, c) = \{\delta(q, c)\}$. Obviously, $\#_S(A) = \#_S(A')$ and $L(A') = L$. □

Chapter 2

Supplementary information for transducers

In this chapter, we shall present examples of languages both in and outside of the families $\mathscr{L}_{DTA}$ and $\mathscr{L}_{NTA}$. Afterwards, we shall combine our results with existing ones to study the relationship of families $\mathscr{L}_{DTA}$ and $\mathscr{L}_{NTA}$ to the families of deterministically and nondeterministically decomposable languages. We shall also study the effects of modifying the advice language.

2.1 Languages in $\mathscr{L}_{DTA}$

The following simple lemma allows us to use the extended fooling set technique to reason about the state complexity of deterministic transducers.

Lemma 5. *Transducer state complexity of a language is greater than or equal to its nondeterministic state complexity.*

Proof. For the sake of contradiction, let L be a regular language such that $tsc(L) < nsc(L)$ and let M be a transducer such that $M(\Sigma_1^*) = L$. Let A be a nondeterministic automaton such that $K_A = K_M, q_{0_A} = q_{0_M}, F_A = F_M, \Sigma_A = \Sigma_2$ and $\delta_A(q_1, c) = q_2$ if and only if there is $d \in \Sigma_1$ such that $\delta_M(q_1, d) = (q_2, c)$. The automaton A accepts the language L and its state complexity is equal to that of the transducer M. □

Languages consisting of a single word were studied in the finite automata setting. In the following theorem, we shall show that with transducer as a computational model, the existence of advice depends only on the length of the word, not the number of different symbols.

Theorem 2. *If $|w| \geq 3$, the singleton language $L = \{w\}$ is in $\mathscr{L}_{DTA}$.*

Proof. Let $w = c_1c_2 \dots c_n$, let $P = \{(c_1 \dots c_i, c_{i+1} \dots c_n) \mid 0 \leq i \leq n\}$. Then P is a fooling set for $\{w\}$ and therefore $nsc(\{w\}) \geq n + 1$. By Lemma 5, it also holds that $tsc(\{a^n\}) \geq n + 1$. Let $L_{adv} = \{c_n\}$. The following diagrams show an automaton accepting L_{adv} and the transducer M_{new}.

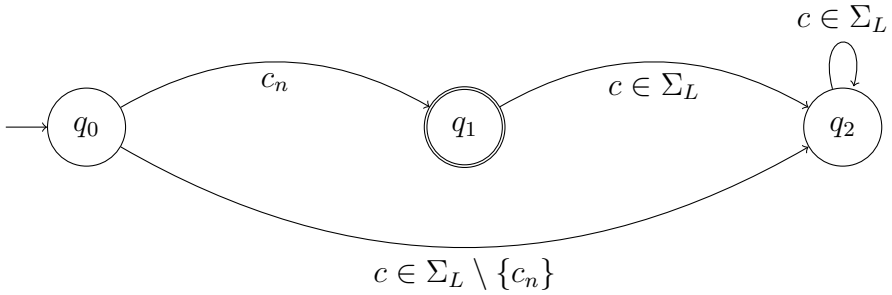

Figure 2.1: The automaton accepting L_{adv}

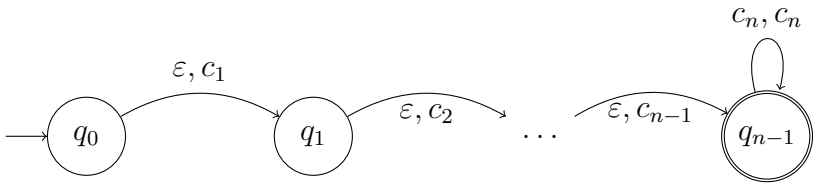

Figure 2.2: The transducer M_{new}

First, M_{new} writes the prefix $c_1 \dots c_{n-1}$ to the output without reading the input at all. Then, an accepting state is reached and the input is copied to the output. Since the input is not empty, the computation must continue. After reading the symbol c_n, the computation shall halt in an accepting state with the word $c_1 \dots c_n = w$ written on the output tape. Thus, $M_{new}(L_{adv}) = \{w\}$. Since $\#_S(M_{new}) = n < n + 1$ and $sc(L_{adv}) = 3 \leq n < n + 1$, the advice is useful. □

Theorem 3. *Let $L_n = \{a^{kn} \mid k \in \mathbb{N}\}$. If n is not a prime, L_n is in $\mathscr{L}_{DTA}$.*

Proof. Since n is not a prime, there exist integers p_1 and p_2 such that $n = p_1p_2$. Let $L_{adv} = \{a^{kp_1} \mid k \in \mathbb{N}\}$. The transducer M_{new} is shown in the diagram.

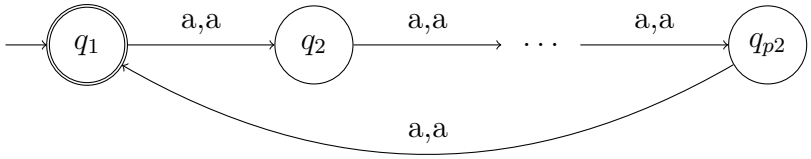

Figure 2.3: The transducer M_{new} that realizes the intersection of L_{adv} and $\{a^{kp_2} \mid k \in \mathbb{N}\}$.

It can be easily seen that the state complexity of both the advice language and the new transducer is lower than the state complexity of the original transducer. □

2.2 Languages not in $\mathscr{L}_{DTA}$

We shall use the following lemma to reason about the cardinality and finiteness of the advice language.

Lemma 6. *If $L \in \mathscr{L}_{DTA}$ and $M_{new}(L_{adv}) = L$, the cardinality of L_{adv} must be greater than or equal to the cardinality of L.*

Proof. For every word w in L, there must exist a word v in L_{adv} such that $M_{new}(v) = w$. Since M_{new} is deterministic, there is only one possible accepting run for every word. Thus, M_{new} represents a surjective mapping. □

For every deterministic transducer, there exists an equivalent nondeterministic transducer. It can be easily seen that there exists a nondeterministic transducer M_1 such that $M(\{\varepsilon\}) = \{\varepsilon, a\}$. By Lemma 6, no deterministic transducer M_2 such that $M_2(\{\varepsilon\}) = \{\varepsilon, a\}$ exists. Thus, deterministic transducers are a less powerful computational model.
In the following theorem, we shall prove that if the state complexity of a language is already low enough, no useful advice can exist.

Theorem 4. *If $tsc(L) \leq 2$, L is not in $\mathscr{L}_{DTA}$.*

Proof. The theorem trivially holds when $tsc(L) = 1$. If $tsc(L) = 2$, then there must exist a transducer M_{new} such that $\#_S(M_{new}) = 1$ and a regular language L_{adv} such that $sc(L_{adv}) = 1$ and $M_{new}(L_{adv}) = L$. The only languages with deterministic state complexity equal to 1 are Σ_1^* and $\emptyset$. It holds that $M_{new}(\emptyset) = \emptyset$. Thus, it must also hold that $L = \emptyset$. However, $tsc(\emptyset) = 1$. If $L_{adv} = \Sigma_1^*$, a contradiction with the assumption that $tsc(L) = 2$ is reached as well. □

Corollary 1. *The languages $\Sigma_2^*, \Sigma_2^+, \emptyset$, $\{\varepsilon\}$ and $\{a\}$ are not in $\mathscr{L}_{DTA}$.*

We have shown that the languages of unary-coded multiples of composite numbers are in the family $\mathscr{L}_{DTA}$. Now, we shall show that this does not hold for the multiples of primes.

Theorem 5. *Let $L_p = \{a^{kp} \mid k \in \mathbb{N}\}$ where p is a prime. Then L is not in $\mathscr{L}_{DTA}$.*

Proof. The language L_{adv} must contain some word a^x such that $x \geq p$ by Lemma 6 and the fact that L_p is infinite. Since $tsc(L_p) = p$, the automaton that accepts L_{adv} must have fewer than p states. Thus, a part of a^x of length r_1 can be pumped and

$\forall k \in \mathbb{N} : a^{x+kr_1} \in L_{adv}$. Since M_{new} must also have fewer than p states, a cycle where r_2 symbols are read and r_3 symbols are written must be reached during the run on a^x; $0 < r_2 < p$, $0 < r_3 < p$. Let $w = a^{x+r_1r_2}$. During the run on w, the cycle must be reached additional r_1 times and the output shall be $a^{qp+r_1r_3}$ for some q. Both r_1 and r_3 are less than p and p is a prime, therefore $p \nmid r_1r_3$ and L_p can not contain $a^{qp+r_1r_3}$. If $n \geq 2$, let $L_{adv} = \{a^{kp^{n-1}} \mid k \in \mathbb{N}\}$. Then, $sc(L_{adv}) = p^{n-1}$. The transducer M_{new} is shown in the diagram.

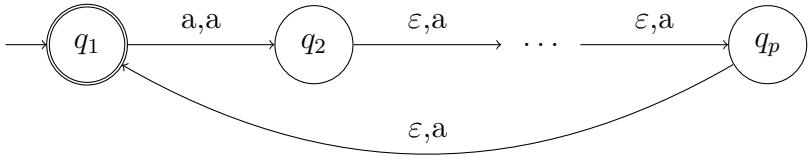

Figure 2.4: The transducer M_{new} that applies the homomorphism $h(a) = a^p$.

□

Note that $L_{p^n} = \{a^{kp^n} \mid k \in \mathbb{N}\}$ was shown to be nondeterminstically undecomposable for arbitrary prime p and positive integer n by Rovan and Sádovský [7]. Also note that we assumed that the alphabet of L_{adv} contained only one symbol. However, using a larger alphabet would allow us to find useful advice (if $p \geq 3$).
Let $L_{adv} = \{ab\}^*$. The transducer M_{new} is shown in the following diagram.

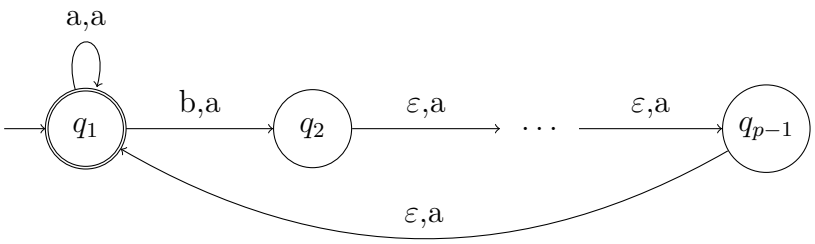

Figure 2.5: The transducer M_{new} that applies the homomorphism $h(a) = a$, $h(b) = a^{p-1}$.

Clearly $M_{new}(L_{adv}) = L_p$. Since $tsc(L) = p \geq 3$, $sc(L_{adv}) = 2 < 3 \leq tsc(L)$. It also holds that $\#_S(M_{new}) < tsc(L)$. Thus, L_{adv} would be useful advice if not for the fact that $|\Sigma_{L_{adv}}| > |\Sigma_{L_p}|$,

2.3 Languages in $\mathscr{L}_{NTA}$

In this section, we shall study the nondeterministic transducer state complexity and particular languages in the family $\mathscr{L}_{NTA}$.

In the following lemma, we shall show that nondeterministic state complexity of a regular language is equal to its nondeterministic transducer state complexity.

Lemma 7. *If L is a regular language, then $nsc(L) = ntsc(L)$.*

Proof. If $A = (K, \Sigma, \delta, q_0, F)$ is a minimal NFA such that $L(A) = L$ and $M = (K, \Sigma, \Sigma, H, q_0, F)$ where $H = \{(q, c, c, p) \mid q, p \in K, c \in \Sigma \cup \{\varepsilon\}, p \in \delta(q, c)$, then $M(\Sigma^*) = L$. Therefore $nsc(L) \geq ntsc(L)$ for every regular language L.
For the sake of contradiction, let $M = (K, \Sigma_1, \Sigma, H, q_0, F)$ be such a transducer that $M(\Sigma_1^*) = L$ and $\#_S(M) < nsc(L)$. Then a NFA $A = (K, \Sigma, \delta, q_0, F)$ where $\forall p, q \in K \forall c \in \Sigma \cup \{\varepsilon\} : p \in \delta(q, c) \Leftrightarrow \exists c' \in \Sigma_1 \cup \{\varepsilon\} : (q, c', c, p) \in H$ accepts the language L and at the same time, $\#_S(A) < nsc(L)$ holds, which is a contradiction. □

In the following theorems, we shall show that the languages consisting of one word of length at least 2 and the languages of unary-coded multiples of composite numbers belong to the family $\mathscr{L}_{NTA}$.

Theorem 6. *Let $L = \{w\}$, let $|w| \geq 2$. Then $L \in \mathscr{L}_{NTA}$.*

Proof. Let $w = a_1 \dots a_n$ for symbols $a_1, \dots, a_n$. Let $P = \{(a_1 \dots a_k, a_{k+1} \dots a_n) \mid 0 \leq k \leq n\}$. It can be easily seen that P is an extended fooling set for L and therefore $ntsc(L) \geq n + 1$. Let $L_{adv} = \{a_n\}$. Clearly $nsc(L_{adv}) = 2$. The transducer M_{new} is shown in the following diagram.

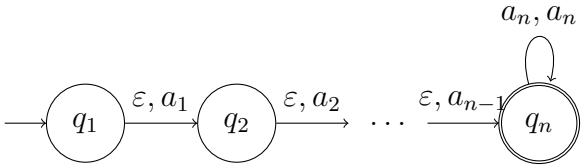

Figure 2.6: The transducer M_{new} that adds a prefix to the input word

Since $\#_S(M_{new}) = n$ and $M_{new}(L_{adv}) = L$, $L \in \mathscr{L}_{NTA}$. □

Theorem 7. *Let n be a composite number, let $L = \{a^{kn} \mid k \in \mathbb{N}\}$. Then $L \in \mathscr{L}_{NTA}$.*

Proof. The proof is analogous to the proof of Theorem 3. □

2.4 Languages not in $\mathscr{L}_{NTA}$

In this section, we shall show examples of languages not in the family $\mathscr{L}_{NTA}$.

Theorem 8. *If $nsc(L) \leq 2$, then L is not in $\mathscr{L}_{NTA}$.*

Proof. The proof is analogous to the proof of Theorem 4. □

Theorem 9. *Let $L_p = \{a^{kp} \mid k \in \mathbb{N}\}$ where p is a prime. Then L_p is not in $\mathscr{L}_{NTA}$.*

Proof. The proof is similar to the proof of Theorem 5. For the sake of contradiction, let L_{adv} be a useful advice language. If the transducer contains a reachable cycle where ε is read and a^{r_1} $(1 \leq r_1 < p)$ is written to the output tape, a^{r_1} can be written to the output tape one more time and the resulting output shall no longer belong to L_p. Now, let us suppose that M_{new} contains no ε-cycle. If L_{adv} is infinite, the automaton for L_{adv} must contain a cycle. The transducer M_{new} has to contain some cycle as well, otherwise the output language would be finite. Thus, there exist constants s and r_2 such that $\forall k \in \mathbb{N} : a^{s+kr_2} \in L_{adv}$ and constants r_3 and r_4 such that M_{new} contains a cycle where r_3 symbols are read and r_4 symbols are written. It must also hold that $1 \leq r_2, r_3, r_4 < p$ and that there is some n such that $a^{np} \in M_{new}(a^s)$. During the run on $a^{s+r_2r_3}$, M_{new} can enter the cycle additional r_2 times and thus add $a^{r_2r_4}$ to the output. Neither r_2 nor r_4 is divisible by p, therefore r_2r_4 is not divisible by p either. Thus, $a^{np+r_2r_4} \in M_{new}(a^{s+r2_r3)})$. Since $a^{s+r_2r_3} \in L_{adv}$ but $a^{np+r_2r_4} \notin L_p$, L_{adv} is not a useful advice language. If L_{adv} is finite there exists a constant c such that for every c symbols written, the transducer has to read at least one. Thus, $M_{new}(L_{adv})$ is finite and L_{adv} is not a useful advice language. □

Analogously to the note after Theorem 5, adding an additional symbol to the input alphabet would allow us to find a useful advice language.

2.5 Relationship to deterministically decomposable languages

In this section, we shall study the relationship of the family $\mathscr{L}_{DTA}$ to deterministically decomposable languages.

Theorem 10. *There exists a language in $\mathscr{L}_{DTA}$ that is not deterministically decomposable.*

Proof. Let $L = \{a^{94}\}$. By Theorem 2, $L \in \mathscr{L}_{DTA}$. However, it was shown by Gaži and Rovan that L is not deterministically decomposable. [1] □

Theorem 11. *If the language L is deterministically decomposable and $sc(L) = tsc(L)$, then $L \in \mathscr{L}_{DTA}$.*

Proof. Since L is deterministically decomposable, there must exist languages L_1, L_2 such that $sc(L_1) < sc(L), sc(L_2) < sc(L)$ and $L_1 \cap L_2 = L$. Then $sc(L_1) < tsc(L)$. It is trivial to construct a transducer M_{new} with $sc(L_2)$ states that realizes the intersection of the input language with L_2. Thus, $M_{new}(L_1) = L$ and $L \in \mathscr{L}_{DTA}$. □

However, it does not always hold that $sc(L) = tsc(L)$. Since a transducer may halt without processing the entire input, a trash state is not neeeded. Moreover, there exist languages such that $sc(L) - tsc(L) > 1$. An example of such language is $L = c(a^2\{b\}^*\{a, b\})^*$. Using the Myhill–Nerode theorem, it can be shown that $sc(L) \geq 7$ - the automaton has to finish reading the words $\varepsilon, c, ca, caa, caab, caaba$ and cc in pairwise distinct states. The following table contains a suffix w for every distinct pair of words u, v such that $uw \in L$ and $vw \notin L$ or $uw \notin L$ and $vw \in L$.

	ε	c	ca	caa	caab	caaba	cc
ε	-	ε	aa	a	ε	ε	c
c	-	-	ε	a	b	aa	ε
ca	-	-	-	a	ε	ε	aa
caa	-	-	-	-	ε	ε	a
caab	-	-	-	-	-	aa	ε
caaba	-	-	-	-	-	-	ε
cc	-	-	-	-	-	-	-

However, $tsc(L) \leq 4$. The transducer is shown in the following diagram.

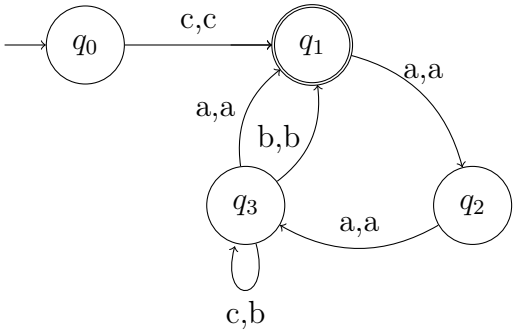

Figure 2.7: The transducer M which transforms $\{a, b, c\}^*$ into L

In general, the reason for this inequality is that there may exist words $w_1, w_2, u \in \Sigma^*_{L_{adv}}$, a word $v \in \Sigma^*_L$ and distinct states p and q such that $(q_0, w_1u, \varepsilon) \vdash^* (p, u, v)$ and $(q_0, w_2u, \varepsilon) \vdash^* (q, u, v)$ - in other words, a deterministic transducer may reach two different states with the same remaining input and with the same content of the output tape based on the processed part of the input. The language L is deterministically decomposable. Informally, the input word can be split into two parts separated by the first occurence of the symbol b. The new automata shall ensure the correct structure of the first and the second part, respectively. The formal proof of correctness of this decomposition would be analogous to the proof of decomposability of $L_i = (a^{i-1}b^*\{a, b\})^*$ in [7]. The decomposition is shown in Figure 2.8. For clarity, the trash states are omitted.

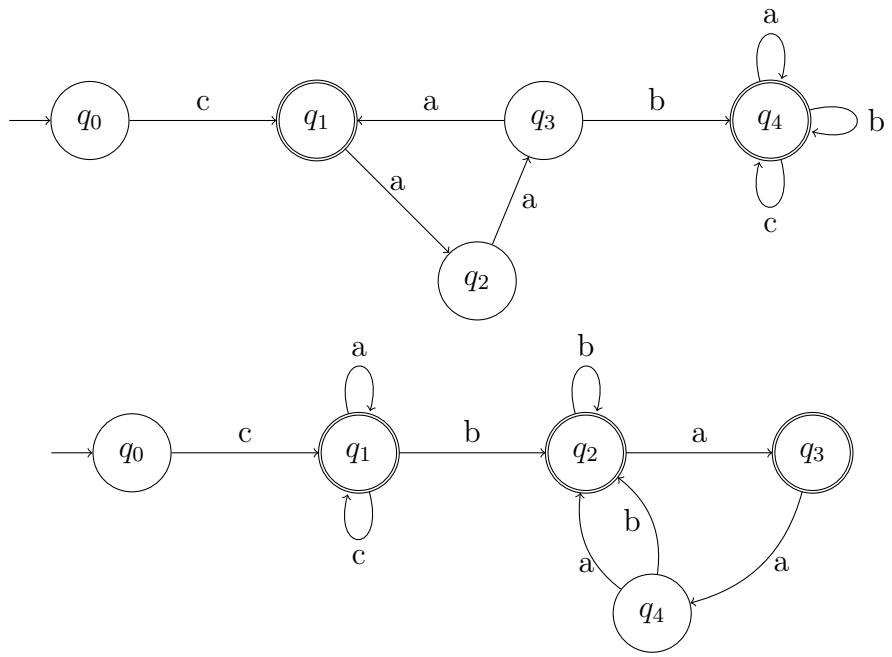

Figure 2.8: The automata A_1 and A_2

Our conjecture is that $L \notin \mathscr{L}_{DTA}$, which would mean that deterministically decomposable languages and the family $\mathscr{L}_{DTA}$ are uncomparable. However, we were not able to formally prove this conjecture. In the following theorems, we shall show that many problems related to regular supplementary information are decidable.

Theorem 12. *Let L and L_{adv} be regular languages. It is decidable whether $L \in \mathscr{L}_D(L_{adv})$ and whether $L \in \mathscr{L}_N(L_{adv})$.*

Proof. Since we consider obly transducers where $H \subseteq K \times (\Sigma_1 \cup \{\varepsilon\}) \times (\Sigma_2 \cup \{\varepsilon\}) \times K$, there are only finitely many (up to isomorphism) transducers $M_{new} = (K, \Sigma_{L_{adv}}, \Sigma_L, H, q_0, F)$ such that $\#_S(M_{new}) < (n)tsc(L)$. Transduction can also be viewed as the application of an inverse homomorphism, intersection with a regular language and a homomorphism. The family of regular languages is closed under these operations and it is decidable whether two finite state automata accept the same language [6]. Thus, it can be verified in finite time whether a suitable transducer such that $M_{new}(L_{adv}) = L$ exists. □

Theorem 13. *Let L be a regular language. It is decidable whether $L \in \mathscr{L}_{DTA}$ and whether $L \in \mathscr{L}_{NTA}$.*

Proof. Without loss of generality, let $\Sigma_{L_{adv}} = \Sigma_L$. Then, there are only finitely many automata A (up to isomorphism) such that $\#_S(A) < (n)tsc(L)$ and thus only finitely many possible advice languages L_{adv}. By Theorem 12, it is decidable whether $L \in \mathscr{L}_D(L_{adv})$ and whether $L \in \mathscr{L}_N(L_{adv})$ for all such advice languages. □

Theorem 14. *Let L be a regular language. It is decidable whether L is deterministically decomposable.*

Proof. There are only finitely many (up to isomorphism) automata A where $\Sigma_A = \Sigma_L$ such that $\#_S(A) < sc(L)$. The family of regular languages is closed under intersection and thus for every pair of such automata A_1, A_2 there is an automaton A' such that $L(A') = L(A_1) \cap L(A_2)$. It is also decidable whether $L(A') = L$. □

Thus, it is possible to computationally verify whether $L \notin \mathscr{L}_{DTA}$ and either confirm our conjecture or show that the proposed counterexample is not valid. However, we were merely concerned with the decidability of the considered problems and did not take the time complexity into account. The actual computation could take a significant amount of time.

2.6 Relationship to nondeterministically decomposable languages

In this section, we shall prove that nondeterministically decomposable languages are a strict subset of the family $\mathscr{L}_{NTA}$.

Theorem 15. *The family of nondeterministically decomposable languages is a subset of $\mathscr{L}_{NTA}$.*

Proof. Let L_1, L_2 be regular languages such that $nsc(L_1) < nsc(L), nsc(L_2) < nsc(L)$ and $L_1 \cap L_2 = L$. Let $A_1 = (K, \Sigma, \delta, q_0, F)$ be an NFA such that $L(A) = L_1$ and $\#_S(A_1) < nsc(L)$. Let $M = (K, \Sigma, \Sigma, H, q_0, F)$ where
$H = \{(q, c, c, p) \mid q, p \in K, c \in \Sigma \cup \{\varepsilon\}, p \in \delta(q, c)\}$, Then $M(L_2) = L$,
$nsc(L_2) < ntsc(L)$ and $\#_S(M) < ntsc(L)$. □

Theorem 16. *There exists a language in $\mathscr{L}_{NTA}$ that is not nondeterministically decomposable.*

Proof. As shown by Rovan and Sádovský [7], the language $\{a^{94}\}$ is nondeterministically undecomposable. However, it can be easily seen that if $L_{adv} = \{a^{47}\}$, there exists a nondeterministic transducer M_{new} such that $M_{new}(L_{adv}) = \{a^{94}\}$. □

Corollary 2. *The family of nondeterministically decomposable languages is a strict subset of $\mathscr{L}_{NTA}$.*

2.7 Relationship between $\mathscr{L}_{DTA}$ and $\mathscr{L}_{NTA}$

The relationship between deterministically and nondeterministically decomposable languages was studied by Rovan and Sádovský [7]. It was shown that some languages are deterministically decomposable but nondeterministically undecomposable. Such languages include $L = (a\{a,b\}a\{a,b\})^*$. Since $sc(L) = 5$ but $nsc(L) = 4$, more complex advice can be used in the deterministic setting. However, the acceptance of the language L can not be simplified in the nondeterministic setting. In the following theorem, we shall show that a different situation may arise in the transducers setting: advice that is useful for a nondeterministic transducer may be too complex for a deterministic transducer.

Theorem 17. *There exists a language L such that $L \in \mathscr{L}_{NTA}$ but $L \notin \mathscr{L}_{DTA}$.*

Proof. One such language is $L = \{a^2\}$. By Theorem 6, $L \in \mathscr{L}_{NTA}$. For the sake of contradiction, let L_{adv} be useful advice. Since $tsc(L) = 3, sc(L_{adv}) \leq 2$. Neither $\{a\}^*$ nor $\emptyset$ can be used as advice, therefore $sc(L_{adv}) = 2$. Without loss of generality, let $A_{adv} = \{\{q_0, q_1\}, \{a\}, \delta, q_0, F\}$ and let $L_{adv} = L(A_{adv})$. If $F = \{q_0, q_1\}$, then $L_{adv} = \{a\}^*$ and if $F = \emptyset$, then $L_{adv} = \emptyset$, neither of which is useful advice. Therefore, exactly one state is accepting. If $\delta(q_0, a) = q_0$, the state q_1 would never be reached. Thus, it must hold that $\delta(q_0, a) = q_1$.
There are only 4 possible languages L_{adv}. If $F = \{q_0\}$ and $\delta(q_1, a) = q_1$, then $L_{adv} = \{\varepsilon\}$. If $F = \{q_0\}$ and $\delta(q_1, a) = q_0$, then $L_{adv} = \{a^{2n} \mid n \in \mathbb{N}\}$. If $F = \{q_1\}$ and $\delta(q_1, a) = q_1$, then $L_{adv} = \{a\}^+$. If $F = \{q_1\}$ and $\delta(q_1, a) = q_0$, then $L_{adv} = \{a^{2n+1} \mid n \in \mathbb{N}\}$.
First, we shall analyze the case when $L_{adv} = \{\varepsilon\}$. The transducer M_{new} must have at most 2 states, at least one of them has to be accepting and $M_{new}(\varepsilon) = a^2$. If the initial state q_0 is accepting, the computation must halt immediately without any output being written. If the initial state q_0 is not accepting, then the state q_1 clearly has to be. If the state q_1 is ever to be reached, it must be reached after the first step of the computation. Otherwise, it would never be reached due to the transducer's determinism. Then, the computation must halt since an accepting state is reached and there is no remaining input. However, the output can contain at most one symbol at that point. Thus, $L_{adv} \neq \{\varepsilon\}$.
In the other cases, L_{adv} is infinite and it holds that if $a^k \in L_{adv}$, then $a^{k+2} \in L_{adv}$. Since M_{new} must write at least two symbols to the output, it must contain some cycle where ε, a or a^2 is read and at least one symbol is written to the output. It must also hold that $M_{new}(a^k) = a^2$ for some k. During the run on a^{k+2}, the cycle shall be reached again and some additional symbol shall be written to the output. □

2.8 Advice modifications

To acquire a better understanding of what makes the advice language useful, we shall study whether some common modifications preserve the usefulness of the advice. In Theorem 18, only deterministic transducers are considered. In the remaining theorems, either deterministic or nondeterministic transducer may be used as a model.

Theorem 18. *There exists a pair of languages L, L_{adv} such that L_{adv} is useful advice for L but L_{adv}^R is not.*

Proof. Let $L = \{ba^{3n} \mid n \in \mathbb{N}\}$, let $L_{adv} = \{b\}\{a\}^*$. It can be easily seen that $\{(\varepsilon, b), (b, aaa), (ba, aa), (baa, a)\}$ is a fooling set for L and therefore $tsc(L) \geq 4$. The transducer M_{new} is shown in the diagram.

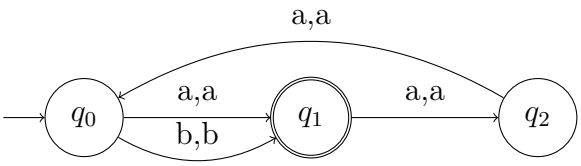

Figure 2.9: The transducer M_{new}.

Since $M_{new}(L_{adv}) = L$, $\#_S(M_{new}) = 3$ and $sc(L_{adv}) < 4$, L_{adv} is useful advice for L. If $\{a\}^*\{b\}$ is used as advice, there must exist a state q where a or ε is read and b is written to the output. If b could be written only while reading b, b could be written only after the entire input was read. However, since the run where ba^6 is written must end in an accepting state and it must stop once the input is read and an accepting state is reached, it would be impossible to write a^6 after reading the entire input. At least 6 steps are needed and by the pigeonhole principle, some state has to be reached twice. If it is an accepting state, a contradiction is reached since the run would have to stop after reaching such a state for the first time. If only a non-accepting state is reached twice, the part of the run in between shall be repeated indefinitely due to the transducer's determinism and the fact that no input is read during any of the steps. Thus, an accepting state shall never be reached.
Once b is written, q can not be reached again unless q is accepting and the entire input is read, thus only the two remaining states can be used while writing the a^6 part. Then, some state has to be reached twice with a or aa read and a or aa written in between. The necessary number of a's can be added to the input to produce a word that does not belong to L. □

Theorem 19. *There exist languages L, L_{adv_1}, L_{adv_2} such that L_{adv_1} and L_{adv_2} are useful advice for L but $L_{adv_1} \cup L_{adv_2}$ is not.*

Proof. Let $L = \{a^{10n} \mid n \in \mathbb{N}\}$, $L_{adv_1} = \{a^{2n} \mid n \in \mathbb{N}\}$, $L_{adv_2} = \{a^{2n+1} \mid n \in \mathbb{N}\}$. The advice L_{adv_1} can be transformed into L by applying the homomorphism $h(a) = a^5$. The advice L_{adv_2} can be transformed into L by ignoring the first symbol and applying the same homomorphism to the rest of the input. However, $L_{adv_1} \cup L_{adv_2} = a^*$. By the definition of transducer state complexity, a^* can not be useful advice. □

Theorem 20. *There exist languages L, L_{adv_1}, L_{adv_2} such that L_{adv_1} and L_{adv_2} are useful advice for L but $L_{adv_1} \cap L_{adv_2}$ is not.*

Proof. Let $L = \{a^{24}\}$, $L_{adv_1} = \{a^3\}, L_{adv_2} = \{a^4\}$. Then $L_{adv_1} \cap L_{adv_2} = \emptyset$. By the definition of transducers, $M(\emptyset) = \emptyset$. □

Theorem 21. *There exist languages L and L_{adv} and a homomorphism h such that L_{adv} is useful advice for L but $h(L_{adv})$ is not.*

Proof. Let $L = \{a^{4n} \mid n \in \mathbb{N}\}, L_{adv} = \{a^{2n} \mid n \in \mathbb{N}\}, h(a) = \varepsilon$. Then $h(L_{adv}) = \{\varepsilon\}$. In case a transducer M with fewer than 4 states such that $M(\varepsilon) = L$ existed, an NFA accepting L with fewer than 4 states could be constructed and a contradiction with the nondeterministic complexity of L (which is obviously 4) would be reached. □

None of the considered operations preserved the usefulness of advice. In the following theorem, we shall show that supplementary information shall remain useful even if the symbols are renamed.

Theorem 22. *Let $h : \Sigma_1 \to \Sigma_3$ be a bijective homomorphism. If L_{adv} is useful advice for L, then $h(L_{adv})$ is useful advice as well.*

Proof. Let $M_{new} = (K, \Sigma_1, \Sigma_2, H, q_0, F)$ be the transducer used to transform L_{adv} into L. Let $H' = \{(p, h(c), d, q) \mid (p, c, d, q) \in H\}$. It can be easily seen that the transducer $M'_{new} = (K, \Sigma_3, \Sigma_2, H', q_0, F)$ transforms $h(L_{adv})$ into L. □

Chapter 3

Properties of the families $\mathscr{L}_{DTA}$ and $\mathscr{L}_{NTA}$

In this chapter, we shall study the closure properties of the families $\mathscr{L}_{DTA}$ and $\mathscr{L}_{NTA}$.

3.1 Properties of the family $\mathscr{L}_{DTA}$

In this section, we shall show that for every typical operation, the family $\mathscr{L}_{DTA}$ is either not closed under it or the closure property remains open.

Theorem 23. *The family $\mathscr{L}_{DTA}$ is not closed under homomorphism.*

Proof. Let h be a homomorphism. For every c in Σ_2, let $h(c) = \varepsilon$. Therefore, $h(L) = \{\varepsilon\}$ for any language L. The language $\{\varepsilon\}$ is not in $\mathscr{L}_{DTA}$ by Theorem 4 □

Theorem 24. *The family $\mathscr{L}_{DTA}$ is not closed under ε-free homomorphism.*

Proof. Let $L = \{abb\}^*$, let $L_{adv} = \{ab\}^*$. It is easy to see that L_{adv} is useful advice for L and thus $L \in \mathscr{L}_{DTA}$. Let $h(a) = a$ and $h(b) = a$. Then $h(L) = \{a^{3n} \mid n \in \mathbb{N}\}$. However, the language $\{a^{3n} \mid n \in \mathbb{N}\}$ is not in $\mathscr{L}_{DTA}$ by Theorem 5. □

Theorem 25. *The family $\mathscr{L}_{DTA}$ is not closed under inverse homomorphism.*

Proof. Let h be the homomorphism from the proof of Theorem 23. Let $L = \{a^{94}\}$. By Theorem 2, $L \in \mathscr{L}_{DTA}$. However, $h^{-1}(L) = \{w \mid h(w) = a^{94}\} = \emptyset$, which is not in $\mathscr{L}_{DTA}$ by Theorem 4. □

Theorem 26. *The family $\mathscr{L}_{DTA}$ is not closed under intersection.*

Proof. Let $L_1 = \{a^{94}\}$ and $L_2 = \{a^{96}\}$. It can be easily seen that both L_1 and L_2 belong to the family $\mathscr{L}_{DTA}$. However, $L_1 \cap L_2 = \emptyset$, which is not in $\mathscr{L}_{DTA}$ by Theorem 4. □

Theorem 27. *The family $\mathscr{L}_{DTA}$ is not closed under the Kleene star.*

Proof. Let $L = \{\varepsilon, a^2, a^4\}$. Then $tsc(L) = 5$. Let $L_{adv} = \{\varepsilon, a, a^2\}$. Since the homomorphism $h(a) = a^2$ can be applied by a transducer, $L \in \mathscr{L}_D(L_{adv})$ and therefore $L \in \mathscr{L}_{DTA}$. However, $L^* = \{a^{2n} \mid n \in \mathbb{N}\}$. Thus, L^* is not in $\mathscr{L}_{DTA}$ by Theorem 4. □

Theorem 28. *The family $\mathscr{L}_{DTA}$ is not closed under the Kleene plus.*

Proof. Let $L = \{\varepsilon, a^2, a^4\}$. As shown in the proof of the previous theorem, $L \in \mathscr{L}_{DTA}$. Since $L^+ = \{a^{2n} \mid n \in \mathbb{N}\}$, L^+ is not in $\mathscr{L}_{DTA}$ by Theorem 4. □

Theorem 29. *The family $\mathscr{L}_{DTA}$ is not closed under union.*

Proof. Let $L_1 = \{a^{4n} \mid n \in \mathbb{N}\}$, let $L_2 = \{a^{4n+2} \mid n \in \mathbb{N}\}$. Let $L_{adv,2} = \{a^{2n+1} \mid n \in \mathbb{N}\}$. The transducer M_{new_2} applies the homomorphism $h(a) = a^2$, therefore $L_{adv,2}$ is useful advice for L_2 and $L_2 \in \mathscr{L}_{DTA}$. However, $L_1 \cup L_2 = \{a^{2n} \mid n \in \mathbb{N}\}$, which is not in $\mathscr{L}_{DTA}$ by Theorem 4. □

Theorem 30. *The family $\mathscr{L}_{DTA}$ is not closed under concatenation.*

Proof. Let $L_1 = \{a^{4n} \mid n \in \mathbb{N}\}$, $L_2 = \{a, a^3, a^5\}$ and $L_{adv_2} = \{a, a^2, a^3\}$. The transducer M_{new_2} is shown in the diagram:

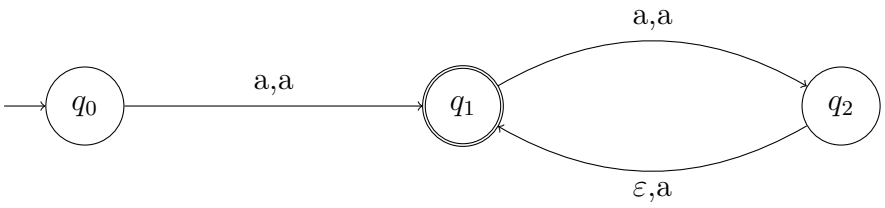

Figure 3.1: The transducer M_{new_2}

Since $tsc(L_2) = 6$ and $sc(L_{adv_2}) \leq 5$, $L_2 \in \mathscr{L}_{DTA}$. However, $L_1L_2 = \{a^{2n+1} \mid n \in \mathbb{N}\}$, which is not in $\mathscr{L}_{DTA}$ by Theorem 4. □

Theorem 31. *The family $\mathscr{L}_{DTA}$ is closed under bijective homomorphism.*

Proof. Let $L \in \mathscr{L}_{DTA}$, let $M_{new} = (K, \Sigma_1, \Sigma_2, H, q_0, F)$ be the transducer that transforms the advice L_{adv} into L. Let $h : \Sigma_2 \to \Sigma_3$ be a bijective homomorphism. If $H' = \{(p, c, h(d), q) \mid (p, c, d, q) \in H\}$, then the transducer $M'_{new} = (K, \Sigma_1, \Sigma_3, H', q_0, F)$ transforms L_{adv} into $h(L)$. □

3.2 Properties of the family $\mathscr{L}_{NTA}$

Counterexamples similar to those used for the family $\mathscr{L}_{DTA}$ can be used to show that the family $\mathscr{L}_{NTA}$ is not closed under the considered operations either.

Theorem 32. *The family $\mathscr{L}_{NTA}$ is not closed under ε-free homomorphism.*

Proof. Let $L = \{ab^2\}^*$. It can be easily seen that the language $L_{adv} = \{ab\}^*$ can be used as advice and therefore $L \in \mathscr{L}_{NTA}$. Let $h(a) = h(b) = a$. Then $h(L) = \{a^{3n} \mid n \in \mathbb{N}\}$. By Theorem 9, $h(L) \notin \mathscr{L}_{NTA}$. □

Theorem 33. *The family $\mathscr{L}_{NTA}$ is not closed under inverse homomorphism.*

Proof. Let $L = \{a^{94}\}$, let $h(a) = \varepsilon$. By Theorem 6, $L \in \mathscr{L}_{NTA}$. However, $h(L) = \emptyset$. By Theorem 8, $\emptyset \notin \mathscr{L}_{NTA}$. □

Theorem 34. *The family $\mathscr{L}_{NTA}$ is not closed under union.*

Proof. Let $L_1 = \{a^{4n} \mid n \in \mathbb{N}\}$, let $L_2 = \{a^{4n+2} \mid n \in \mathbb{N}\}$. It can be easily seen that the languages $L_{adv,1} = \{a^{2n} \mid n \in \mathbb{N}\}$ and $L_{adv,2} = \{a^{2n+1} \mid n \in \mathbb{N}\}$ can be used as advice. However, $L_1 \cup L_2 = \{a^{2n} \mid n \in \mathbb{N}\}$. By Theorem 8, $L_1 \cup L_2 \notin \mathscr{L}_{NTA}$. □

Theorem 35. *The family $\mathscr{L}_{NTA}$ is not closed under intersection.*

Proof. Let $L_1 = \{a^{94}\}$, let $L_2 = \{a^{96}\}$. By Theorem 6, $L_1 \in \mathscr{L}_{NTA}$ and $L_2 \in \mathscr{L}_{NTA}$. However, $L_1 \cap L_2 = \emptyset$ and by Theorem 8, $\emptyset \notin \mathscr{L}_{NTA}$. □

Theorem 36. *The family $\mathscr{L}_{NTA}$ is not closed under concatenation.*

Proof. Let $L_1 = \{a^{4n} \mid n \in \mathbb{N}\}$, let $L_2 = \{a, a^3, a^5\}$. By Theorem 7, $L_1 \in \mathscr{L}_{NTA}$. The language $L_{adv} = \{\varepsilon, a, a^2\}$ can be used as advice for L_2 by writing the symbol a to the output tape and subsequently applying the homomorphism $h(a) = a^2$. Thus, $L_2 \in \mathscr{L}_{NTA}$ as well. However, $L_1L_2 = \{a^{2n+1} \mid n \in \mathbb{N}\}$. By Theorem 8, $L_1L_2 \notin \mathscr{L}_{NTA}$. □

Theorem 37. *The family $\mathscr{L}_{NTA}$ is not closed under the Kleene star.*

Proof. Let $L = \{\varepsilon, a^2, a^4\}$. Since $L_{adv} = \{\varepsilon, a, a^2\}$ can be used as advice, $L \in \mathscr{L}_{NTA}$. However, $L^* = \{a^{2n} \mid n \in \mathbb{N}\}$. By Theorem 8, $L^* \notin \mathscr{L}_{NTA}$. □

Theorem 38. *The family $\mathscr{L}_{NTA}$ is not closed under the Kleene plus.*

Proof. Let $L = \{\varepsilon, a^2, a^4\}$. Then $L^+ = \{a^{2n} \mid n \in \mathbb{N}\}$. By Theorem 8, $L^+ \notin \mathscr{L}_{NTA}$. □

Similarly to the family $\mathscr{L}_{DTA}$, the family $\mathscr{L}_{NTA}$ is closed under bijective homomorphism.

Theorem 39. *The family $\mathscr{L}_{NTA}$ is closed under bijective homomorphism.*

Proof. The proof is analogous to the proof of Theorem 31. □

Chapter 4

Fixed advice

In the thesis by Martiš [4], the families generated by a fixed advice language in the context of deterministic automata were studied. Such family was defined as $\mathscr{L}(L_{adv}) = \{L \mid L_{adv} \text{ is useful advice for } L\}$. In this chapter, we shall study such families in the context of deterministic and nondeterministic transducers.

4.1 Fixed deterministic advice

In this section, we shall show that the family $\mathscr{L}_D(L_{adv})$ is not necessarily closed under common operations. We shall also show that it is always closed under bijective homomorphism.

Theorem 40. *There exists a language L_{adv} such that $\mathscr{L}_D(L_{adv})$ is not closed under complement, union, intersection, Kleene star and Kleene plus.*

Proof. By Lemma 6, if $L \in \mathscr{L}_D(L_{adv})$, then $|L| \leq |L_{adv}|$. Let $L_{adv} = \{a^8\}$, let $L_1 = \{a^{16}\}$ and let $L_2 = \{a^{24}\}$. Clearly, L_{adv} is useful for L_1 and L_2 but not for L_1^C, L_1^+, L_1^* or $L_1 \cup L_2$ since their cardinality is greater than that of L_{adv}. It can also be easily seen that $\mathscr{L}_D(L_{adv})$ is not closed under intersection, since $L_1 \cap L_2 = \emptyset$. By Theorem 4, no useful advice exists for $\emptyset$. □

Theorem 41. *There exists a language L_{adv} such that $\mathscr{L}_D(L_{adv})$ is not closed under concatenation.*

Proof. Let $L_{adv} = \{a^4, a^8\}$, let $L_1 = \{a^8, a^{16}\}$ and let $L_2 = \{a^{16}, a^{32}\}$. Clearly, L_{adv} is useful for both L_1 and L_2. However, $L_1L_2 = \{a^{24}, a^{32}, a^{40}, a^{48}\}$ and by Lemma 6, L_{adv} is not useful for L_1L_2. □

Theorem 42. *There exists a language L_{adv} such that $\mathscr{L}_D(L_{adv})$ is not closed under homomorphism and inverse homomorphism.*

Proof. Let $L_{adv} = \{a^4\}$, let $L = \{a^8\}$. Clearly $L \in \mathscr{L}_D(L_{adv})$. Let $h(a) = \varepsilon$. Then $h(L) = \{\varepsilon\}$ and $h^{-1}(L) = \emptyset$. By Theorem 4, neither $h(L)$ nor $h^{-1}(L)$ is in $\mathscr{L}_D(L_{adv})$. □

Theorem 43. *For every language L_{adv}, the family $\mathscr{L}_D(Ladv)$ is closed under bijective homomorphism.*

Proof. Follows from the proof of Theorem 31. □

4.2 Fixed nondeterministic advice

In this section, we shall show that the family $\mathscr{L}_N(L_{adv})$ does not have to be closed under common operations either and that it is also always closed under bijective homomorphism.

Lemma 8. *If $\mathscr{L}_{NTA}$ is not closed under an unary operation f, then there exists a language L_{adv} such that $\mathscr{L}_N(L_{adv})$ is not closed under that operation.*

Proof. Since $\mathscr{L}_{NTA}$ is not closed under the operation, there must exist a language L such that advice L_{adv} exists for L but no advice exists for $f(L)$. Thus, $L \in \mathscr{L}_N(L_{adv})$ but $f(L) \notin \mathscr{L}_N(L_{adv})$. □

Corollary 3. *By Lemma 8 and the results in Chapter 3, there exist languages L_{adv} such that $\mathscr{L}_N(L_{adv})$ is not closed under homomorphism, inverse homomorphism, ε-free homomorphism, Kleene star and Kleene plus.*

Theorem 44. *There exists a language L_{adv} such that $\mathscr{L}_N(L_{adv})$ is not closed under intersection.*

Proof. Let $L_{adv} = \{a^4\}$, let $L_1 = \{a^8\}$, let $L_2 = \{a^{12}\}$. Then $L_1 \in \mathscr{L}_N(L_{adv})$ and $L_2 \in \mathscr{L}_N(L_{adv})$, but $L_1 \cap L_2 = \emptyset \notin \mathscr{L}_N(L_{adv})$. □

Theorem 45. *There exists a language L_{adv} such that $\mathscr{L}_N(L_{adv})$ is not closed under union.*

Proof. Let $L_{adv} = \{a^{2n+1} \mid n \in \mathbb{N}\}$, let $L_1 = \{a^{4n+2} \mid n \in \mathbb{N}\}$, let $L_2 = \{a^{4n} \mid n \in \mathbb{N}\}$. Then $L_1 \in \mathscr{L}_N(L_{adv})$ and $L_2 \in \mathscr{L}_N(L_{adv})$, but
$L_1 \cup L_2 = \{a^{2n} \mid n \in \mathbb{N}\} \notin \mathscr{L}_N(L_{adv})$. □

Theorem 46. *There exists a language L_{adv} such that $\mathscr{L}_N(L_{adv})$ is not closed under concatenation.*

Proof. Let $L_{adv} = \{b(ab)^n \mid n \in \mathbb{N}\} \cup \{\varepsilon\}$, let $L_1 = \{(ab)^{2n} \mid n \in \mathbb{N}\}$, and let $L_2 = \{\varepsilon, ab\}$. It can be easily seen that $nsc(L_{adv}) = 2$. Since $P_1 = \{(abab, \varepsilon), (aba, b), (ab, ab), (a, bab)\}$ is an extended fooling set for L_1, $ntsc(L_1) \geq 4$ and since $P_2 = \{(\varepsilon, ab), (a.b), (ab, \varepsilon)\}$ is an extended fooling set for L_2, $ntsc(L_2) \geq 3$. The following figures show how L_{adv} can be used as advice.

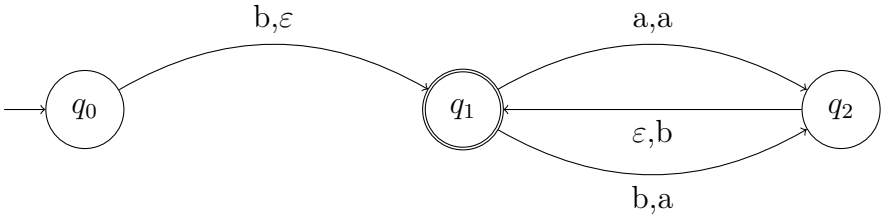

Figure 4.1: The transducer M_{new_1}

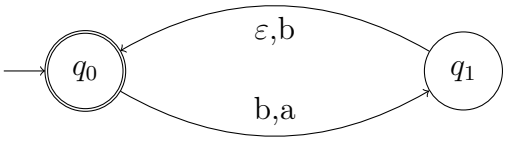

Figure 4.2: The transducer M_{new_2}

The transducer M_{new_1} shall ignore the initial symbol b and then apply the homomorphism $h(a) = h(b) = ab$. Since the length of the remaining input is guaranteed to be even, $M_{new_1}(L_{adv}) = L_1$. It can be easily seen that $M_{new_2}(\{\varepsilon\}) = \{\varepsilon\}, M_{new_2}(\{b\}) = \{ab\}$ and $M_{new_2}(\{b(ab)^{n+1} \mid n \in \mathbb{N}\}) = \emptyset$, since the computation shall halt once the symbol a appears on the input. Thus, $M_{new_2}(L_{adv}) = L_2$.
We have shown that $L_1 \in \mathscr{L}_N(L_{adv})$ and $L_2 \in \mathscr{L}_N(L_{adv})$. However, $L_1L_2 = \{(ab)^n \mid n \in \mathbb{N}\}$ and therefore $ntsc(L_1L_2) = 2$. By Theorem 8, no advice exists for L_1L_2 and thus $L_1L_2 \notin \mathscr{L}_N(L_{adv})$. □

Theorem 47. *For every language L_{adv}, the family $\mathscr{L}_N(Ladv)$ is closed under bijective homomorphism.*

Proof. Follows from the proof of Theorem 39. □

4.3 Relationships

In this section, we shall investigate whether some relations between advice languages $L_{adv,1}$ and $L_{adv,2}$ imply such relations between the families $\mathscr{L}_D(L_{adv,1})$ and $\mathscr{L}_D(L_{adv,2})$.

Lemma 9. *There exist languages L_1 and L_2 such that $L_1 \subseteq L_2$ and $\mathscr{L}_D(L_1) \not\subseteq \mathscr{L}_D(L_2)$*

Proof. Let $L_1 = \{a^{3n} \mid n \in \mathbb{N}\}$, let $L_2 = \{a\}^*$. By the definition of deterministic transducer state complexity $\mathscr{L}_D(L_2) = \emptyset$. However, it can be easily seen that $\mathscr{L}_D(L_1) \neq \emptyset$. □

Lemma 10. *There exist languages L_1 and L_2 such that $L_1 \not\subseteq L_2$ and $\mathscr{L}_D(L_1) \not\subseteq \mathscr{L}_D(L_2)$*

Proof. Let $L_1 = \{a^{3n} \mid n \in \mathbb{N}\}$, let $L_2 = \{a^3\}$. Then L_1 can be used as advice for the language $\{a^{6n} \mid n \in \mathbb{N}\}$. By Lemma 6, L_2 can only be used as advice for finite languages. □

By Theorem 20, there exist languages L_1 and L_2 such that $\mathscr{L}_D(L_1) \cap \mathscr{L}_D(L_2) \not\subseteq \mathscr{L}_D(L_1 \cap L_2)$. We shall show that it may also hold that $\mathscr{L}_D(L_1) \cap \mathscr{L}_D(L_2) \subseteq \mathscr{L}_D(L_1 \cap L_2)$.

Lemma 11. *There exist languages L_1 and L_2 such that $\mathscr{L}_D(L_1) \cap \mathscr{L}_D(L_2) \subseteq \mathscr{L}_D(L_1 \cap L_2)$.*

Proof. Let $L_1 = \{a^4\}$, let $L_2 = \{a\}^*$. Then $\mathscr{L}_D(L_2) = \emptyset$ and thus $\mathscr{L}_D(L_1) \cap \mathscr{L}_D(L_2) = \emptyset$. However, $L_1 \cap L_2 = L_1$ and it can be easily seen that $\mathscr{L}_D(L_1) \neq \emptyset$. □

Lemma 12. *There exist languages L_1 and L_2 such that $\mathscr{L}_D(L_1) \cup \mathscr{L}_D(L_2) \not\subseteq \mathscr{L}_D(L_1 \cup L_2)$.*

Proof. Let $L_1 = \{a^{2n} \mid n \in \mathbb{N}\}$, let $L_2 = \{a^{2n+1} \mid n \in \mathbb{N}\}$. Then $L_1 \cup L_2 = a^*$ and $\mathscr{L}_D(a^*) = \emptyset$. It can be easily seen that $\mathscr{L}_D(L_1) \cup \mathscr{L}_D(L_2) \neq \emptyset$. □

Lemma 13. *There exist languages L_1 and L_2 such that $\mathscr{L}_D(L_1 \cap L_2) \not\subseteq \mathscr{L}_D(L_1) \cap \mathscr{L}_D(L_2)$.*

Proof. Let $L_1 = \{a^4\}$, let $L_2 = \{a\}^*$. Then $\{a^8\} \in \mathscr{L}_D(L_1) = \mathscr{L}_D(L_1 \cap L_2) \neq \emptyset$. By the definition of transducer state complexity, $\mathscr{L}_D(L_2) = \emptyset$ and therefore $\mathscr{L}_D(L_1) \cap \mathscr{L}_D(L_2) = \emptyset$. As a result, $\mathscr{L}_D(L_1 \cap L_2) \not\subseteq \mathscr{L}_D(L_1) \cap \mathscr{L}_D(L_2)$. □

Lemma 14. *There exist languages L_1 and L_2 such that $\mathscr{L}_D(L_1 \cup L_2) \not\subseteq \mathscr{L}_D(L_1) \cup \mathscr{L}_D(L_2)$.*

Proof. Let $L_1 = a^*$, let $L_2 = b^*$ and let $L = \{a^{4n} \mid n \in \mathbb{N}\} \cup \{b^{6n} \mid n \in \mathbb{N}\}$. Neither L_1 nor L_2 can be used as advice for any language, therefore $\mathscr{L}_D(L_1) \cup \mathscr{L}_D(L_2) = \emptyset$. Since $P = \{(\varepsilon, b^6), (b, b^5), (b^2, b^4), (b^3, b^3), (a, a^3), (a^2, a^2)\}$ is an extended fooling set for L, $tsc(L) \geq 7$. The transducer M_{new} such that $M_{new}(L_1 \cup L_2) = L$ is shown in Figure 4.3. Thus, $\mathscr{L}_D(L_1 \cup L_2) \neq \emptyset$. □

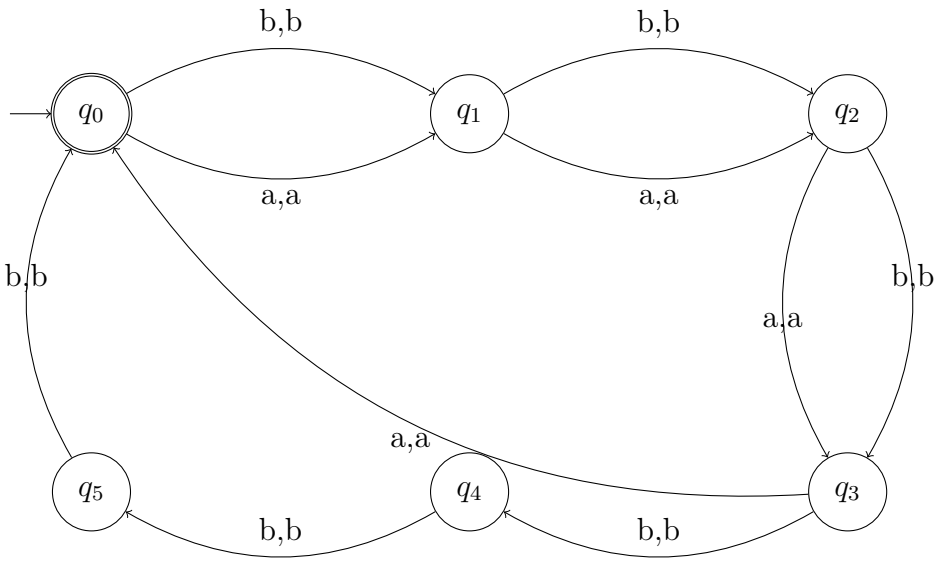

Figure 4.3: The transducer M_{new}

We have shown that some relationships may or may not hold depending on the choice of languages. This is in part caused by the fact that $\mathscr{L}_D(\emptyset) = \mathscr{L}_D(\Sigma^*) = \emptyset$. Note that with the exception of Lemma 10, the same examples could be used to prove the same propositions in the nondeterministic transducers setting.

Conclusion

In this thesis, we have formalized and studied the use of supplementary information in finite transducers setting. We have found some particular examples of languages for which useful advice exists. Using these examples, we have shown that some types of advice that can be intuitively seen as useful, such as homomorphic images, can be useful in our setting but not necessarily useful in the finite automata setting. Furthermore, we have shown that the family $\mathscr{L}_{NTA}$ is a proper superset of nondeterministically decomposable languages and that the family $\mathscr{L}_{DTA}$ is not a subset of deterministically decomposable languages. Our conjecture is that deterministically decomposable languages are not a subset of $\mathscr{L}_{DTA}$ either. We were unable to formally prove this conjecture, but we have presented a possible counterexample and shown that it is decidable whether the counterexample is valid. We have also shown that using advice over an alphabet with more symbols than there are in Σ_L may allow us to find advice for more languages.. Using counterexamples, we have shown that the families defined by our approach are not closed under any of the considered operations with the exception of bijective homomorphism. This is in part due to the fact that they do not contain languages such as Σ^*, $\emptyset$ and $\{\varepsilon\}$. However, we did not find an efficient way to decide whether a language is useful advice for another. The closure under reversal and concatenation remains open. Note that these problems are still open in the nondeterministic finite automata setting as well. We have also studied the families generated by a fixed advice language and we have shown that they do not have to be closed under standard operations. Finally, we have studied the effects of advice modifications and shown that, with the exception of bijective homomorphism, they do not necessarily preserve the usefulness of the information.

In our work, we were focused on whether *any* advice exists. A possible continuation of our research could focus on how many states can be saved by using particular advice. The remaining open problems regarding closure properties can be studied as well. Other possibilities for further study include a full characterization of the families $\mathscr{L}_{DTA}$ and $\mathscr{L}_{NTA}$ and a generalization of the use of supplementary information using concepts from promise problems.

Bibliography

[1] Peter Gaži and Branislav Rovan. Assisted problem solving and decompositions of finite automata. *SOFSEM 2008*, LNCS 4910:292–303, 2008.

[2] Seymour Ginsburg. *The Mathematical Theory of Context-free Languages*. McGraw-Hill Book Company, 1966.

[3] Pavel Labath. Zjednodušenie výpočtov prídavnou informáciou. Master's Thesis, Comenius University in Bratislava, 2010.

[4] Jozef Martiš. Dodatočná informácia a triedy jazykov. Bachelor's Thesis, Comenius University in Bratislava, 2018.

[5] Alexandros Palioudakis. Nondeterministic state complexity and quantifying non-determinism in finite automata. Technical Report 2012-596, School of Computing, Queen's University, Kingston, ON, Canada, 2012.

[6] Branislav Rovan and Michal Forišek. Formálne jazyky a automaty, 2013. [Accessed 2018-12-01] Available at: `http://foja.dcs.fmph.uniba.sk/materialy/skripta.pdf` (in Slovak).

[7] Branislav Rovan and Šimon Sádovský. On usefulness of information: Framework and NFA case. *Adventures Between Lower Bounds and Higher Altitudes*, LNCS 11011:85–99, 2018.

[8] Boris Vida. Using transformation in solving problems with supplementary information. Master's Thesis, Comenius University in Bratislava, 2015.