Acknowledgements

This work has been made possible by the cooperation, support and encouragement given to me by many people during these past four years in which I have elaborated this work. I would like to express my gratitude to all of them.

First of all, I would like to thank the person who encouraged me to pursue this research, who introduced me into the field of Mathematical Linguistics, and who gave me assistance and invaluable advices. This person is my supervisor, Carlos Martín-Vide.

Thanks also to the "Ministerio de Educación y Ciencia" (MEC) which provided me the financial support to do this work, granting me a FPU ("Formación de Profesorado Universitario", AP2001-1880) pre-doctoral fellowship. Due to this support, I have had the opportunity to do brief stays at several other universities, which have been very important to this research.

I am grateful to the "Grupo de Investigación en Reconocimiento de Formas e Inteligencia Artificial" (RFIA) of the University of Alicante (Spain). During my stay at this university (first year of my fellowship), I was able to compile information and background literature on Grammatical Inference, which defined the research path that I would follow during the completion of this dissertation.

I am indebted to the support given to me by Professor Takashi Yokomori and his research group at Waseda University in Japan. During this stay

(second year of my fellowship), I acquired a solid foundation in the field of Grammatical Inference, which allowed me to specify a concrete topic for my dissertation.

My gratitude also to the support of Professor Tim Oates and his group, the "Cognition Robotics and Learning" (CORAL) laboratory at the University of Maryland Baltimore County (USA). During this stay (third year of my fellowship), I was able to examine the technical details of my dissertation topic, I was able to improve my English, and with the help of all of them – especially Tim Oates and Tom Armstrong– I made large progress towards the completion of my dissertation.

I also would like to thank everybody at the "Équipe Universitaire de Recherche en Informatique de Saint-Étienne" (EURISE) of the University Jean Monnet (France). While working with this research group during the fourth year of my fellowship, I was able to further extend my knowledge on Grammatical Inference, analyze preliminary results, extend other important aspects of my work and complete the writing of this dissertation.

During all these stays I have been very fortunate to meet fantastic people from different countries, who are responsible for making my experiences in these places unforgettable. Space restrictions do not allow me to mention all them, but I would like just thank Kaoru Onodera for her company and kindness during my stay in Tokyo, and Eric Eaton for his unconditional help during my stay in Baltimore, and for his assistance in editing this dissertation.

I cannot forget the people in my research group. My gratitude to all members of the "Research Group on Mathematical Linguistics" (GRLMC) at Rovira i Virgili University (Victor Mitrana, Artiom Alhazov, Remco Loos...), and special thanks to Adrian Horia Dediu and Cristina Bibire, for all their support. I also express my gratitude for all professors of our International PhD School for discussions and interactions.

Finally, I would like to thank my family and friends for their personal support throughout these four years. To them and to all the people that have believed in me, once again, GRACIAS.

Contents

List of Figures

List of Abbreviations

CF: Context-free languages

CQ: Correction Query

CS: Context-Sensitive languages

DFA: Deterministic Finite Automata

EC: External Contextual languages

EC_p: p-dimensional External Contextual languages

EFS: Elementary Formal Systems

EQ: Equivalence Query

FIN: Finite languages

L^*: Angluin's algorithm for learning DFA from queries

LCA: Learning from Corrections Algorithm

LIN: Linear languages

$LSMG$: Linear Simple Matrix Grammar

MAT: Minimally Adequate Teacher

MCS: Mildly Context-Sensitive languages

MQ: Membership Query

PAC: Probably Approximately Correct

RE: Recursively Enumerable languages

REG: Regular languages

SEC_p: Simple p-dimensional External Contextual languages

Part I

Introduction

Chapter 1

Motivation and Structure

1.1. Context and motivation

Natural language learning constitutes one of the most typical human abilities, and despite research efforts in this domain, human learning mechanisms are poorly understood.

Several questions arise from the beginning, among others: how complex are natural languages? The properties of natural language could give us an answer to that question.

Natural languages, for example Spanish or English, have a great expressive power. The number of sentences that we can construct with a natural language is infinite, but the set of words that we use to construct those sentences is finite. However, not all the combinations of words are allowed; word combinations must be correct (with respect to the syntax) and have sense (with respect to the semantics).

The set of syntactically and semantically correct sentences is indeterminate, *a priori*. We cannot define beforehand all the set of possible constructions of a natural language. One of the main reasons for this is the ambiguity of natural languages.

There are different types of ambiguities. One such types is *semantic ambiguity*. Any given word may have several different meanings, e.g., *banco* in Spanish means "asiento" (seat) or "entidad financiera" (bank). We have to select the meaning which makes the most sense in context. Also, the same syntactic structure can have different meanings, e.g, *Todos los estudiantes de la escuela hablan dos lenguas* (all the students of the school can speak two languages) could means that each student can speak two languages, or that only two certain languages are spoken.

The ambiguity can also be syntactic. The same sentence can have multiple possible parse trees (more than one associated syntactic structure). For example, in the sentence *Juan vio a un hombre con el telescopio* (Juan saw a man with a telescope), who is with the telescope, the man or Juan?. Choosing the most appropriate meaning usually requires semantic and contextual information.

This is only a small demonstration of the complexity of natural languages.

Despite the complexity of natural languages, how are children able to learn language so fluently and effortlessly, without explicit instruction?

A child growing up in a linguistic community acquires the language spoken by the community from samples of speech presented to her. There are several remarkable facts in the process of children's language acquisition:

- Children learn language easily. The ease with which children learn language belies the underlying complexity of the task.
- Children are capable of learning any natural language given adequate input. A child with an English environment will learn to speak English; the same child with a Japanese environment would learn to speak Japanese.
- Children learn one or more of the languages that they are exposed to, without actively deciding whether they want to learn the language.

- Children acquire their native language on the basis of exposure to limited data, without any specific training and in a short amount of time.

Therefore, children acquire their native language efficiently and successfully. Nevertheless, other cognitive tasks that are less complex than language acquisition are harder for them.

> *About two years after conception, or a year after birth, children will say their first words. The skill and the swiftness with which children learn to speak have always fascinated adults, who sometimes forget to marvel at the mystery of it all. Even so, what a prodigy the child is. Producing words, combining them into original sentences, understanding other people's words: these are much more remarkable feats than those that children accomplish much later and with greater difficulty. The fact that the sum of two and two is four seems a simple notion. Nonetheless, it becomes consciously accessible to children only well after they have uttered hundreds of distinct sentences. Before knowing how to coordinate their hands to catch a ball, children will have understood almost all the sentences that adults address to them, and they will have virtually mastered their language before knowing how to tie their shoelaces.*[de Boysson-Bardies, 1999, p. 5].

Linguists, in spite of all research efforts, do not understand all the rules, strategies, and other processes that underlie children's language acquisition. Several linguistic theories of language acquisition have been proposed in the last century, but there is not a single accepted theory. In Chapter 2 we explain the main ideas of the most representative theories.

Why is there this contradiction between the facility with which children acquire language (known as Plato's problem) and the difficulty to explain it (known as Orwell's problem)?

The publication of *Syntactic Structures* by N. Chomsky in 1957 inaugurated the use of a mathematical model in the study of natural language. This new methodology radically changed the way linguists study natural languages.

Formal languages are behind the first model of Chomsky. Formal languages are symbolic systems used primarily in mathematics and computer science.

The process of generation and development of formal languages is inverse to natural languages. Whereas the origin and development of natural languages is *natural*, namely, without the control of any theory (theories of natural languages were established *a posteriori*, after the language had already matured), formal languages were developed through the establishment of a theory that gives the basis for these languages.

Words in formal languages are precisely defined. The meaning of symbols is determined exclusively by the syntax, without any reference to the semantics. Only the operators and relations (such as equality, pertinence, etc.) have special meanings.

A fundamental property of formal languages is the finiteness of their alphabet and of their generative rules.

Moreover, in opposition to natural languages, formal languages specifications are easier to translate to computer languages. In Chapter 2 we present the basic notions of formal language theory.

Access to an abstract conception of a language can provide a better comprehension of its structure. In this manner, formal languages became an important tool in the study of natural languages.

Language acquisition is now studied in a variety of fields, including linguistics, psychology, and, computer science.

One of the main topics in cognitive science, epistemology, linguistic and psycholinguistic theory as well as of machine learning

> *and algorithmic learning theory is language acquisition. The human ability to acquire their mother tongue as well as other languages has attracted a huge amount of interest in all these scientific disciplines. In particular, the main goal of the research undertaken is to gain a better understanding of what learning really is.*[Lange and Zeugmann, 1996, p. 89].

Linguists distinguish between *language acquisition* and *language learning*, but there is not such distinction in computer science, which focuses only on language learning. For linguists, language acquisition refers to *first* language(s) learning (by children); it is as a subconscious process in which language acquirers are not consciously aware of the grammatical rules of the language. Language learning refers to second language(s) learning (by adults); conscious process, knowing the rules, being aware of them, and being able to talk about them. Since second language learning is a process very similar to other human learning processes, we consider that acquisition of first languages is of much more interest, because the underlying mechanisms are still not well understood. For this reason, this dissertation focuses on the problem of language acquisition.

The desire to better understand the process of natural language acquisition motivated research in formal models of language learning. By studying formal models of language acquisition, several key questions on natural language learning can be answered. Moreover, these formal models could provide an operational framework for the numerous practical applications of language learning (e.g., language learning by machines).

> *The issues and practical difficulties associated with formal language learning models can provide useful insights for the development of language understanding systems. Several key questions in natural language learning such as the role of prior knowledge,*

> *the types of input available to the learner, and the impact of semantic information on learning the syntax of a language can possibly be answered by studying formal models of language acquisition.*[Parekh and Honavar, 2000, p. 728].

The field of Machine Learning has a specialized subfield that deals with the learning of formal languages. This field is known as *Grammatical Inference* or grammar induction. It refers to the process of learning grammars and languages from data.

> *The problem of grammatical inference is roughly to infer (discover) a grammar that generates a given set of sample sentences in some manner that is supposed to be realized by some algorithmic device, usually called inference algorithm.* [Yokomori, 2004] p. 507

The initial theoretical foundations of Grammatical Inference were given by M.E. Gold [Gold, 1967], who was primarily motivated by the problem of first language acquisition. A remarkable amount of research has been done since his seminal work to establish a theory of Grammatical Inference, to find effective and efficient methods for inferring grammars, and to apply those methods to practical problems (e.g., Natural Language Processing, Computational Biology).

As T. Yokomori stated:

> *Therefore, grammatical inference can be taken as one of the typical formulations for a broader word "learning", and provides a good theoretical framework for investigating a learning process* [Yokomori, 2004, p. 507].

Grammatical Inference has been investigated within many research fields, including machine learning, computational learning theory, pattern recognition, computational linguistics, neural networks, formal language theory, and

many others. Excellent surveys on the field of Grammatical Inference can be found in [Miclet, 1986], [Sakakibara, 1997], and [Yokomori, 2004].

Based on all these ideas, with this dissertation, we will try to bring together the Theory of the Grammatical Inference and Studies of language acquisition, in pursuit of our final goal: to go deeper in the understanding of the process of language acquisition by using the theory of inference of formal grammars.

This work is highly interdisciplinary, drawing from computer science, linguistics and cognitive science. Such interdisciplinary research might help close the

> *undesirable gap between the communities of linguists and computer scientists, more specifically the communities of computational linguists and formal language theoreticians* [Martín-Vide, 1996, p. 462].

By its nature, the study of language learning is interdisciplinary. Efforts of researchers from different areas could help to decipher the mystery of the language.

> *Language learning is considered by many to be one of the central problems of linguistics and, more generally, cognitive science. Yet, the very same interdisciplinary nature that makes this field of study so interesting, makes it somehow difficult for researchers to reach a thorough understanding of the issues at play. This follows from the fact that research in the field by necessity has to draw on techniques and results that come from traditionally disparate fields such as linguistics, psychology and computer science.* [Bertolo, 2001, Preface].

As S. Bertolo states, applications of formal learning theory to the problem of human language learning can be described as an exercise in which linguists, psychologists and learnability researchers cooperatively construct a theory of

human language learning. He compares the interaction among these three parties with the interaction between a patron, an architect and a structural engineer that want to design a museum together:

> *(...) the architect would start by designing very bold and innovative plans for the museum; the engineer would remind him or her, calculator in hand, that some of those designs would be physically impossible to build and the patron would visit every so often to make sure that the plans the engineer and the architect have agreed upon would result in a museum that could be built within budget and according to a specified construction schedule. In our case, linguists would correspond to the architect: based on their study of human languages or on more speculative reasons, they specify what they take the possible range of variation among human languages to be. Psychologists would correspond to the patron: they collect experimental data to show that it is not just that humans learn the language(s) of the linguistic community in which they are brought up, but that they do so according to a typical time schedule and relying on linguistic data of a certain, restricted, kind. Finally, learnability researchers correspond to the engineer: some theories of language variation they would be able to rule out directly, by showing that no conceivable mechanism could single out a correct hypothesis from such a large and dense range of choice; some other theories they would pronounce tenable, but only under certain assumptions on the resources available for learning, assumptions that need to be empirically validated by work in developmental psycholinguistics.*[Bertolo, 2001, p. 1].

Since our background is primarily in linguistics, we intend to enrich Grammatical Inference studies with our ideas from this field.

1.2. Structure of the dissertation

This dissertation is organized into four parts and one Appendix.

- **Part I** includes this chapter and Chapter 2, in which we provide linguistic and formal language prerequisites needed to understand some concepts and formalizations presented throughout the dissertation.
- **Part II** and **Part III** are directly connected. This parts are explained in the sequel.
- **Part IV** presents conclusions that follows from precedents parts and some directions for future work.
- **Appendix** offers comparative tables for the results presented in Chapter 8 and also the automata that we used for tests.

A Grammatical Inference problem can be specified by providing the following items:

- **The class of languages or grammars**: what class of languages or grammars is to be learned.
- **Learning Setting**: what kind of data is used in the learning process, and how these data are provided to the learner.
- **The criteria for a successful inference**: under what conditions we say that a learner has been successful in the language learning task.

Part II presents the state-of-the-art of each item.

Regarding the first item, the main focus of research in Grammatical Inference deals with regular and context-free grammars. However, these are mechanisms with a limited representational power to describe some of the aspects of natural language constructions. Context-sensitive grammars are

able to model many aspects of natural language constructions, yet the computational complexity is too high. Therefore, the Chomsky Hierarchy has some limitations when we deal with natural language.

Motivated by linguistic ideas, in the 1980s, researchers introduced a class of formal grammars called *Mildly Context-Sensitive* (MCS), situated halfway between context-free and context-sensitive grammars. This non-standard class has been considered to be appropriate to describe natural languages due to the class' properties (it includes non-context-free constructions that are found in the syntax of natural language, and is computationally feasible). There are well known mechanisms to fabricate MCS families (e.g., *tree adjoining grammars* ([Joshi and Schabes, 1997]), *head grammars* [Roach, 1987], *combinatory categorial grammars* [Steedman, 1985], etc).

All these studies are based on the idea that the class of natural languages is located in the Chomsky Hierarchy. However, as some authors pointed out (for instance, see [Manaster-Ramer, 1999]), this assumption is not necessarily true, as natural languages could occupy an orthogonal position in the Chomsky Hierarchy. In this case, a new hierarchy would be needed.

Many-dimensional External Contextual grammars are a non-standard mechanism that generate a class of languages occupying an orthogonal position with respect the Chomsky Hierarchy. They constitute a MCS language family.

A more general overview of all these ideas will be presented in Chapter 3.

Taking these ideas into account, we consider that the study of natural language syntax from a formal point of view should be focused on a class of languages that occupy an orthogonal position in the Chomsky Hierarchy, and that this class is MCS. Unfortunately, most research on Grammatical Inference is not based on a class of languages with such features.

Three important formal models have been developed in the last four decades within Computational Learning Theory: Gold's model of *identification in the limit* [Gold, 1967], the *query learning model* of Angluin [Angluin, 1987, Angluin, 1988], and the *PAC learning model* of Valiant [Valiant, 1984]. All these models have been thoroughly investigated in the field of Grammatical Inference. We review them in Chapter 4, and present the state-of-the-art aspects of the last two items that define a grammatical inference problem (learning setting and criteria for a successful inference).

Each of these models is based on different learning settings and different criteria for a successful inference. The following question arises: what model is the most adequate one to study children's language acquisition? We discuss in the same chapter some linguistic aspects of these models. In that way, we will try to find an answer to that question.

In Chapter 5 we present current results within the field of Grammatical Inference. Based on some linguistic assumptions such that the availability of positive data (sentences that are grammatically correct) in the process of language learning and the usefulness of using queries in order to get additional information in the learning process, we will focus on results concerning the learnability of languages from only positive data and from queries.

After the presentation and discussion of classes of languages or grammars that could be subject of study, models that are used in Grammatical Inference and some results in that field, we present our contributions to each one of these items. This overview will justify and motivate the novel ideas we introduce in this dissertation.

Part III presents this dissertation's contributions.

We propose the study of a new class of languages, called *Simple External Contextual* (see Chapter 6). This class might contribute to improve our understanding of some aspects of natural language acquisition. From a linguistic

point of view, studying this class is more interesting than to focus on classes such as regular or context-free ones.

Another contribution is the application of the idea of correcting a child during the learning process to the studies of Grammatical Inference, for instance, to the query learning model. Since the type of queries that are used in this model are very simple for real learning environments, we introduce a new type of query called *correction query*, which involves a new way of answering. We believe that correction queries might be more adequate than standard queries in a real learning process (see Chapter 7).

Finally, we present our results regarding learnability of Simple External Contextual from only positive data and learnability of Deterministic Finite Automata (DFA) from correction queries (see Chapter 8).

The structure of this work is summarized in Figure 1.1.

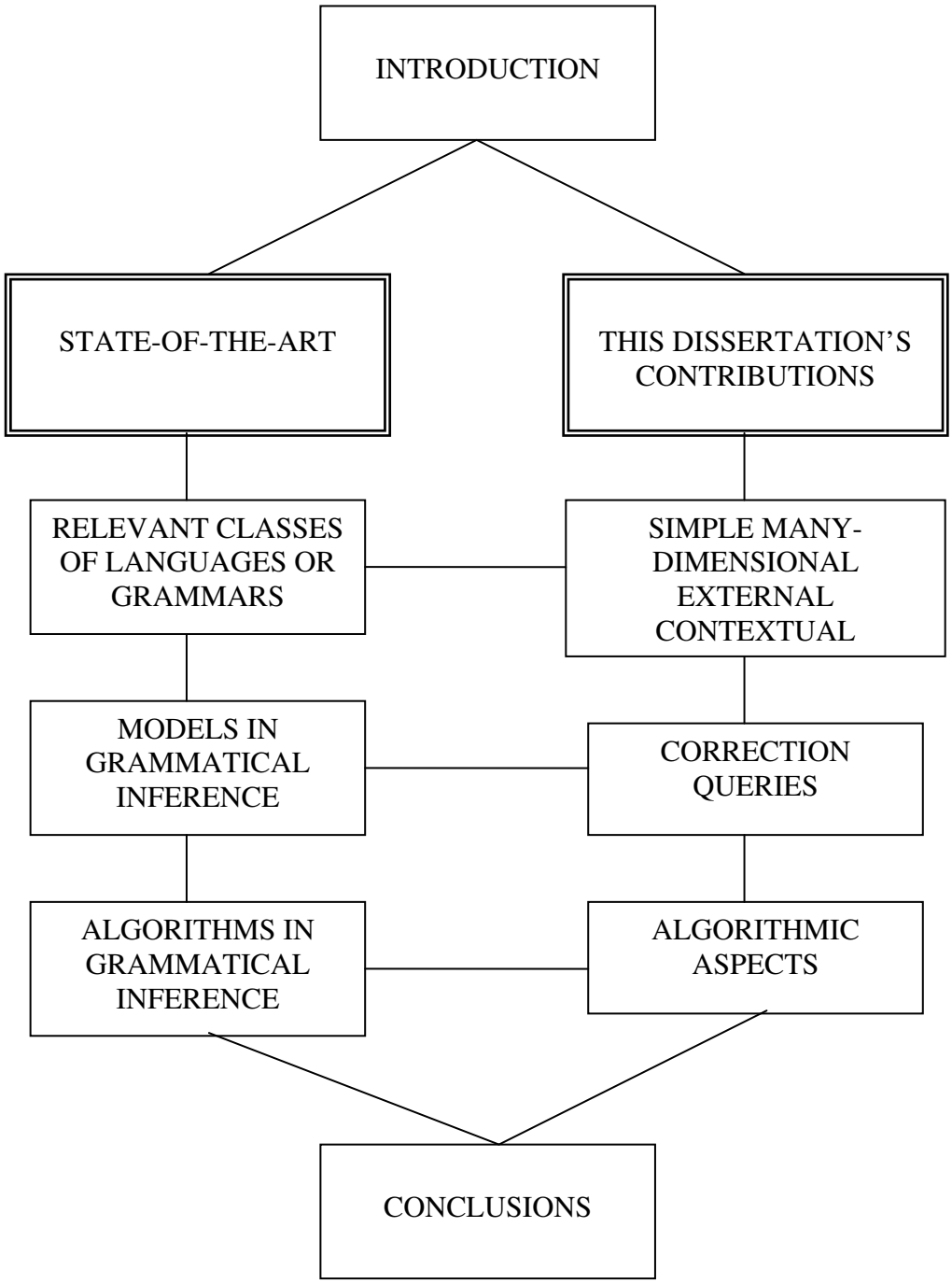


Figure 1.1: Structure of this dissertation

Chapter 2

Prerequisites

2.1. Linguistic Prerequisites

> *How is it possible for children to acquire their native language on the basis of casual exposure to limited data in a short amount of time?*

In the space of a few years, children learn the language they are exposed to, without any explicit instruction. They only hear, not grammars rules, but sentences of English (Spanish, French, Japanese, etc.). Therefore, the problem that children have to face is to figure out (unconsciously) the grammar on the basis of some finite set of sentences. The problem of getting from these limited data to the grammar is known as the *projection problem*.

> *A multitude of subsidiary debates have sprung up around this central issue covering questions about critical periods - the ages at which this can take place, the exact nature of the evidence available to the child, and the various phases of linguistic use through which the infant child passes. In the opinion of many researchers, explaining this ability is one of the most important challenges facing linguists and cognitive scientists today.*[Clark, 2004, p. 26].

Despite all research efforts in this domain, there is not a clear answer to that question. In the last century, two opposite philosophic tendencies arise: nativism, which holds that language is a biological capability with which the human being is born; and empiricism, which defends that the social environment is the unique factor in the development of language.

From both tendencies the contributions of the main theories of the acquisition of language come off. We will deal with three of them:

- Behaviorism
- Innatism
- Evolutionary Psychology

2.1.1. Behaviorism

The American psychologist B.F. Skinner was mainly responsible for the development of the behaviorist theory.

The behaviorism was based on a model of operant conditioning. Operant conditioning is a behavior modification technique based on reinforcement and punishment.

- *Reinforcement.* It is a consequence that causes a behavior to occur with greater frequency. Two kinds of reinforcement: positive reinforcement, which occurs when a behavior (response) is followed by a pleasant stimulus that rewards it (e.g., rat press a lever and receive a food reward); negative reinforcement, which occurs when a behavior (response) is followed by an unpleasant stimulus being removed (e.g., a loud noise continuously sounding until the rat press the lever, then the noise ceases).
- *Punishment.* It is a consequence that causes a behavior to occur with less frequency. Two possible kind of punishment: positive punishment, which adds an aversive stimulus, such as introducing a shock or loud

noise; negative punishment, which removes a pleasant stimulus, such as taking away a child's toy.

Skinner did not advocate the use of punishment, considering that punishment was an ineffective way of controlling behavior. However, reinforcement, both positive and negative (the latter of which is often confused with punishment), proves to be more effective in bringing about lasting changes in behaviour.

Skinner used the model of operant conditioning to train animals and he concludes that similar results could be obtained by applying it to children by means of the process of stimulus-answer.

> *(...) the basic processes and relations which give verbal behavior its special characteristics are now fairly well understood ... the results [of this experimental work] have been surprisingly free of species restrictions. Recent work has shown that the methods can be extended to human behavior without serious modification.* [Skinner, 1957, p. 3].

In [Skinner, 1957], he presents his ideas about language. For Skinner, speech, along with other forms of communication, was simply a behavior. Skinner argue that children acquire language by means of a process of adaptation to extern stimulus of correction and repetition of the adult, in different situations of communication. That is, there is a process of imitation of the children where later they associate certain words to situations, objects or actions. In that way, children learn some habits and answers, internalizing what adult provide them in order to satisfy one necessity to a particular stimulus (for instance, hunger, pain, etc.).

Children learn the vocabulary and the grammar by means of operant conditioning. The adult reward, for example, constructions grammatically correct, but disapprove the incorrect ones.

Therefore, the main ideas of the Skinner's model about the process of language acquisition are:

- The acquisition of human language is not so different to another behaviors learned by other species.
- Children imitate the language of adults.
- Adults correct the errors of children, and children learn by means of these errors.

For Skinner, the proper object of study is behavior itself, analyzed without reference to mental structure. The influence of the environment plays an important role in the behaviorism approach, as well as the idea that children use the language to satisfy specific necessities that they have.

2.1.2. Innatism

N. Chomsky is considered the father of most nativist theories of language acquisition. As we have seen, before Chomsky, learning language had widely been considered a purely cultural phenomenon based on imitation. Chomsky brought greater attention to the innate capacity of children for learning language.

Chomsky's argument to explain natural language acquisition is based on the idea that a newborn's brain is already programmed to learn language. In the same way that children develop the ability to walk (which is a genetic ability) whether or not anybody tries to teach them to do so, children develop the ability to talk. For this reason, many linguists believe that language ability is genetic.

Chomsky compares the task of a linguist with a child that is acquiring a language.

> *The construction of a grammar of a language by a linguist is in some respects analogous to the acquisition of a language by the child. The linguist has a corpus of data; the child is presented with unanalyzed data of language use. The linguist tries to formulate the rules of the language; the child constructs a mental representation of the grammar of the language. The linguist applies certain principles and assumptions to select a grammar among the many possible candidates compatible with his data; the child must also select among the grammars compatible with the data.* [Chomsky, 1975, pag. 11].

Chomsky considers that language is a faculty – a knowledge that is in the mind even when it is not used.

> *The study of human language is particularly interesting in this regard. In the first place, it is a true species property an one central to human thought and understanding. Furthermore, in the case of language we can proceed rather far toward characterizing the system of knowledge attained -knowledge of English, of Japanese, etc.- and determining the evidence that was available to the child who gained this knowledge; we also have a wide range of evidence available about the variety of attainable systems. We are thus in a good position to ascertain the nature of the biological endowment that constitutes the human "language faculty", the innate component of the mind/brain that yields knowledge of language when presented with linguistic experience, that converts experience to a system of knowledge.* [Chomsky, 1986, p. xxvi].

According to Chomsky, language is innate in the biological make up of the brain. Children learn through their natural ability to organize the laws of language, but cannot fully utilize this talent without the presence of other humans.

> *Language learning is not really something that the child does; it is something that happens to the child body grows and matures in a predetermined way when provided with appropriate nutrition and environment stimulation. This is not to say that the nature of the environment is irrelevant. The environment determines the way the parameters of universal grammar are set, yielding different languages.* [Chomsky, 1988, p. 134].

Chomsky claims that children are born with a hard-wired language acquisition device (LAD) in their brains. They are born with the major principles of language in place, but with many parameters to set. According to nativist theory, when the young child is exposed to a language, their LAD makes it possible for them to set the parameters and deduce the grammatical principles, because the principles are innate.

> *(...) language acquisition is interpreted as the process of fixing the parameters of the initial state in one of the permissible ways. A specific choice of parameter settings determines a language in the technical sense that concerns us here: an I-language [...] where I is understood to suggest "internal", "individual", and "intensional".* [Chomsky, 1995, p. 6].

This innate knowledge is often referred to as Universal Grammar. Chomsky defines the Universal Grammar as:

> *(...) principles and elements common to attainable human languages.(...) UG [Universal Grammar] may be regarded as a characterization of the genetically determined language faculty.* [Chomsky, 1986, p. 3]

Namely, human are born with a set of rules already built into them. These rules allow human beings the ability to learn any language.

The Principles and Parameters approach [Chomsky, 1981] make strong claims regarding universal grammar. The central idea is that the syntactic knowledge of a person can be modelled with two formal mechanisms:

- A finite set of fundamental *principles* that are common to all languages. For example, a sentence must always have a subject, even if it is not pronounced.
- A finite set of *parameters* that determine syntactic variability between languages. For instance, the head parameter states that in universal grammar the set of parameters describes the placement of the head in phrase structure. In that way, English is a head-initial language, meaning that the head of the phrase precedes the complement (e.g., the head of the prepositional phrase *in the house* would be the preposition *in*). Whereas Japanese is a head-final language whereby the head of the phrase follows the complement (e.g., in the prepositional phrase *nihon ni* -Japan in-, the preposition *ni* follows the complement *nihon*).

Following the last example, the innate knowledge allows us to understand that there are phrases in all languages, regardless of whether they are head-initial or head-final. It is the parameters settings that allows us to decipher the head position in phrases, even though we may have only heard any particular phrase one or twice.

An important idea of the innatism is the fact that children, through a short period of time, have the ability to produce, perceive and comprehend an infinite number of sentences. If humans were born with a clean slate or *tabula rasa*, as it was once believed, they would not be able to produce or comprehend an infinite number of sentences.

Therefore, we can state that the innatism differs totally of the behaviorism. The behaviorist approach explain the process of natural language acquisition based on superficial features; they consider that children learn the answers of

the adults and in that way they acquired the language (this approach does not take into account the generative capacity of human beings). On the contrary, the innatism considers, first, the mental structure of the human beings and their predisposition to acquire the language, and second, it emphasizes the active role of the learners and their generative capacity to construct an infinite number of sentences.

> *The shift in focus was from the study of E-language to the study of I-language, from the study of language regarded as an externalized object to the study of the system of knowledge of language attained and internally represented in the mind/brain.* [Chomsky, 1986, p. 24].

Chomsky's ideas have had a strong influence on researchers investigating the acquisition of language in children, though some researchers who work in this area today do not support Chomsky's theories.

A popular argument in favor of linguistic nativism is the *Argument from Poverty of the Stimulus* (or APS). The name of the "APS" was coined by Chomsky in [Chomsky, 1980]. The APS emerged out of several of Chomsky's writings on the issue of language acquisition.

The argument from the poverty of stimulus is that there are principles of grammar that cannot be learned on the basis of positive input alone. Therefore, children have insufficient evidence in the primary linguistic data to induce the grammar of their native language.

Though Chomsky reiterated the argument in a variety of different manners, a common structure to the argument is always present and it can be summed up as follows:

1. The grammars of human languages produce hierarchical tree structures and are capable of infinite recursion.

2. For any given set of sentences generated by a hierarchical grammar capable of infinite recursion there are an indefinite number of grammars which could have produced the same data. As such, positive evidence (evidence of those sentences accepted by the grammar) cannot provide enough data to learn the correct grammar, negative evidence (evidence of those sentences not accepted by the grammar) is required.

3. Children are only ever presented with positive evidence, e.g. they only hear others speaking using sentences that are "right" not those that are "wrong".

4. Children do learn the correct grammars for their native languages.

Therefore, human beings must have some form of innate linguistic capacity which provides additional knowledge to language learners.

Researchers believe that there may be a *critical period* during which language acquisition is effortless. The linguist E. Lennenberg states that the crucial period of language acquisition ends around the age of 12 years. After this period it is much harder to learn a new language, due to changes occur in the structure of the brain during puberty.

An interesting example of this is the case of Genie. She was discover in her house when he was thirteen-year old. She appeared to be entirely without language. Her father had judged her retarded at birth and has chosen to isolate her. After her discovery, sadly, she was unable to acquire language completely.

If it is true that children are born with a lot of language knowledge built in, that will help to explain how it is possible to acquire quickly and easily a system of language so complex that no other animal or machine has ever mastered it.

2.1.3. Evolutionary Psychology

During the 1950s and 1960s, a different view of learning began developing. Many theorists disagreed with several aspects of the behaviorist approach due to its failure to incorporate mental events into its learning theories.

The behaviorist approach consider that the study of learning should be objective and that learning theories should be developed from the findings of empirical research. This means that they do not support that mental processes are suitable for scientific or objective study.

The behaviorist perspective could not easily explain why people attempt to organize and make sense of the information they learn. Therefore, mental events or cognition became important.

Cognitive psychologists share with behaviorists the belief that the study of learning should be objective and that learning theories should be founded in the results of empirical research. However, cognitivists argue that by observing the individual's responses to a variety of stimulus conditions they can draw inferences about the nature of the internal cognitive processes that produce those responses.

In cognitive theories, knowledge is viewed as symbolic mental constructs in the learner's mind, and the learning process is the means by which these symbolic representations are committed to memory. Changes in behavior are observed, but only as an indicator to what is going on in the learner's head. The cognitivist approach of the human mind is an input/output model of information or symbol processing.

As opposed to behaviorism, knowledge acquisition is measured by what learners know, not necessarily what they do.

The learner is viewed as an active participant in the knowledge acquisition process. In addition, instructional material that utilizes demonstrations, il-

lustrative examples and corrective feedback are helpful in providing mental models that the learner can follow.

The use of feedback to guide and support the learner to create accurate mental connections is a key component in the cognitive theory.

Jean Piaget was one of the most influential cognitive psychologist [Piaget, 1953], [Piaget, 1962]. Piaget emphasizes on two main functions:

- Organization: it refers to the fact that all cognitive structures are interrelated and that any new knowledge must be fitted into the existing system.
- Adaptation: it refers to the tendency of the organism to fit with its environment in ways that promote survival. It is composed of two terms; assimilation (to understand something new by fitting it into what we already know) and accommodation (if new information cannot be made to fit into existing schemes, a new, more appropriate structure must be developed).

Piaget did many experiments on children's ways of thinking and concluded that human beings go through several distinct stages of cognitive development. These stages are known as:

- Sensorimotor stage (0-2 years): children experience through their senses.
- Preoperational stage (2-7 years): motor skills are acquired.
- Concrete operational stage (7-11 years): children think logically about concrete events.
- Formal Operational stage (after age 11): abstract reasoning is developed here.

Anywise, this psycholinguistic perspective complements the innatist approach. Indeed, besides the linguistic competence, it argues for a general

cognitive competence which is needed to learn and develop the knowledge of language.

Piaget emphasized the importance of the interaction between biological and social (nature and nurture) aspects of language acquisition, a view that is held today.

2.2. Formal Language Prerequisites

In this chapter we present some of the basic notions of Formal Language Theory and we also describe the notation and terminology used throughout this work. As necessary we will introduce specific concepts and definitions in future chapters. Supplementary information can be found in [Hopcroft et al., 2001], [Martín-Vide et al., 2004], and [Rozenberg and Salomaa, 1997].

Formal languages are defined with respect to a given *alphabet*. The alphabet is a finite nonempty set of symbols, denoted Σ. A finite sequence of symbols chosen from some alphabet is called a *string* (or sometimes *word*). The *empty string*, denoted λ, is the string with zero occurrences of symbols. The *length* of a string is the number of positions for symbols in the string, and is denoted $|w|$. For example, $|\lambda| = 0$.

Given an alphabet Σ, the set of all strings over the alphabet Σ is denoted by Σ^*. The set of nonempty strings from alphabet Σ is denoted Σ^+. Thus, $\Sigma^+ = \Sigma^* - \{\lambda\}$. Each subset of Σ^* is called a *language* over the alphabet Σ.

For $x, y \in \Sigma^*$, $x = a_1a_2...a_i$, $y = b_1b_2...b_j$, $i, j \geq 0$, the string $a_1a_2...a_ib_1b_2...b_j$ is denoted by xy and is called the *concatenation* of x and y. If $x = x_1x_2$, for some $x_1, x_2 \in \Sigma^*$, then x_1 is called a *prefix* of x and x_2 is called a suffix of x; if $x = x_1x_2x_3$ for some $x_1, x_2, x_3 \in \Sigma^*$, then x_2 is called a *substring* of x. A *context* is a pair of words, i.e., (u, v), where $u, v \in \Sigma^*$.

$\mathbb{N}$ denotes the set of natural numbers. Assume that $\Sigma = \{a_1, a_2, ..., a_k\}$. The *Parikh mapping*, denoted by Ψ, is:

$$\Psi : \Sigma^* \to \mathbb{N}^k, \Psi(w) = (|w|_{a_1}, |w|_{a_2}, ..., |w|_{a_k})$$

If L is a language, then the *Parikh set* of L is defined by:

$$\Psi(L) = \{\Psi(w) \mid w \in L\}$$

A *linear set* is a set $M \subseteq \mathbb{N}^k$ such that $M = \{v_0 + \sum_{i=1}^{m} v_i x_i \mid x_i \in \mathbb{N}\}$, for some $v_0, v_1, ..., v_m$ in $\mathbb{N}^k$. A *semilinear set* is a finite union of linear sets, and a *semilinear language* is a language L such that $\Psi(L)$ is a semilinear set.

In general, a *grammar* is a finite mechanism by means of which we can generate the elements of the language. The Chomsky grammars are particular cases of *rewriting systems*, where the operation used in processing the strings is the rewriting (the replacement of a "short" substring of the processed string by another short substring).

A *Chomsky grammar* is a quadruple $G = (N, T, S, P)$, where N and T are disjoint alphabets of *nonterminals* symbols and *terminals* symbols, respectively. $S \in N$ is the *axiom* of the grammar, and P is the set of *production rules*. The rules (or productions) of P are written in the form $u \to v$, where u is a string in $(N \cup T)^*$ with at least one nonterminal symbol, and v is a string in $(N \cup T)^*$ that can be the empty string.

The *direct derivation relation* with respect to a grammar G is denoted by $\Rightarrow_G$. For $x, y \in (N \cup T)^*$ we write $x \Rightarrow_G y$ iff $x = x_1 u x_2, y = x_1 v x_2$, for some $x_1, x_2 \in (N \cup T)^*$ and $u \to v$ a rule of G. If G is understood, then we write $\Rightarrow$ instead of $\Rightarrow_G$. The reflexive and transitive closure of the relation $\Rightarrow$ is denoted by $\Rightarrow^*$. The language generated by G, denoted $L(G)$, is defined by $L(G) = \{x \in T^* | S \Rightarrow^* x\}$.

Two grammars G_1, G_2 are called *equivalent* if $L(G_1) - \{\lambda\} = L(G_2) - \{\lambda\}$ (the two languages coincide modulo the empty string).

According to the form of their rules, the Chomsky grammars are classified as follows. A grammar $G = (N, T, S, P)$ is called:

- *length-increasing*, if for all $u \to v \in P$ we have $|u| \leq |v|$.
- *context-sensitive*, if each $u \to v \in P$ has $u = u_1 A u_2, v = u_1 x u_2$, for $u_1, u_2 \in (N \cup T)^*, A \in N$, and $x \in (N \cup T)^+$. (In length-increasing and context-sensitive grammars the production $S \to \lambda$ is allowed, providing that S does not appear in the right-hand members of rules in P.)
- *context-free*, if each production $u \to v \in P$ has $u \in N$).
- *linear*, if each rule $u \to v \in P$ has $u \in N$ and $v \in T^* \cup T^* N T^*$.
- *right-linear*, if each rule $u \to v \in P$ has $u \in N$ and $v \in T^* \cup T^* N$.
- *left-linear*, if each rule $u \to v \in P$ has $u \in N$ and $v \in T^* \cup N T^*$.
- *regular*, if each rule $u \to v \in P$ has $u \in N$ and $v \in T \cup TN \cup \{\lambda\}$.

The arbitrary, length-increasing, context-free, and regular grammars are also said to be of *type 0, type 1, type 2, and type 3*, respectively.

The family of languages generated by length-increasing grammars is equal to the family of languages generated by context-sensitive grammars; the families of languages generated by right- or by left-linear grammars coincide and they are equal to the family of languages generated by regular grammars, as well as with the family of regular languages.

We denote by RE, CS, CF, LIN, and REG the families of languages generated by arbitrary, context-sensitive, context-free, linear, and regular grammars, respectively (RE stands for *recursively enumerable*). By FIN we denote the family of finite languages.

The following strict inclusions hold:

$$FIN \subset REG \subset LIN \subset CF \subset CS \subset RE$$

We call this the *Chomsky hierarchy*. It is schematically depicted in Figure 2.1.

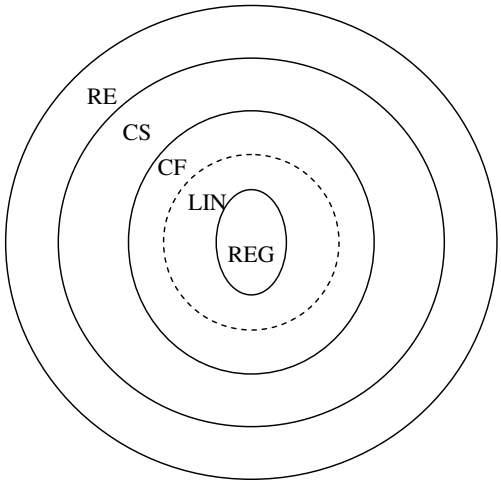


Figure 2.1: The Chomsky Hierarchy

Chomsky Hierarchy is the usual framework to define other families of languages that are not in this classification. Even many people try to locate natural languages in the Chomsky Hierarchy (this topic will be discuss in the next chapter). We will propose in this work some classes of languages that do not fit in this classification.

Automata are computing devices which start from the strings over a given alphabet and *analyze* them (we also say *recognize*), telling us whether or not the input string belongs to a specified language.

The five basic families of languages in the Chomsky Hierarchy, REG, LIN, CF, CS, RE, are also characterized by recognizing automata. These automata are: the finite automaton, the one-turn pushdown automaton, the

pushdown automaton, the linearly bounded automaton, and the Turing machine, respectively. We present here only two of these devices, those which, in some sense, define the two poles of computability: finite automata and Turing machines.

A (deterministic or nondeterministic) finite automaton consists of a finite set of states, a finite alphabet of input symbols, and a set of transition rules. If the next state is always uniquely determined by the current state and the current input symbol, we say that the automaton is deterministic.

Formally, we define a deterministic finite automaton as follows:

A *deterministic finite automaton* (DFA) A is a quintuple $(Q, \Sigma, \delta, q_0, F)$, where:

- Q is the finite set of states.
- Σ is the input alphabet.
- $\delta : Q \times \Sigma \longrightarrow Q$ is the state transition function.
- $q_0 \in Q$ is the starting state.
- $F \subseteq Q$ is the set of final states.

A relation $\vdash$ is defined in the following way: for $px, qy \in Q\Sigma^*$, $px \vdash qy$ if $x = ay$ for some $a \in \Sigma$ and $\delta(p, a) = q$. The reflexive and transitive closure of the $\vdash$ is denoted $\vdash^*$.

The language accepted by a DFA $A = (Q, \Sigma, \delta, q_0, F)$, denoted $L(A)$, is defined as follows:

$$L(A) = \{w \mid q_0 w \vdash^* f, \text{ for some } f \in F\}$$

For convenience, we define the extension of δ, $\delta^* : Q \times \Sigma^* \to Q$, inductively as follows. We set $\delta^*(q, \lambda) = q$ and $\delta^*(q, xa) = \delta(\delta^*(q, x), a)$, for $q \in Q$, $a \in \Sigma$, and $x \in \Sigma^*$. Then, we can also write

$$L(A) = \{w \mid \delta^*(q_0, w) = f \text{ for some } f \in F\}$$

The collection of all languages accepted by DFA is denoted $\mathcal{L}_{DFA}$. We call it the *family of DFA languages*.

We say that a DFA $A = (Q, \Sigma, \delta, q_0, F)$ is *complete* if for all q in Q and a in Σ, $\delta(q, a)$ is defined (that is δ is a total function). For any DFA A, there exists a minimum state DFA A' (also called the canonical DFA), such that $L(A) = L(A')$.

A state q is called a *live state* if there exist strings x and y such that $\delta(q_0, x) = q$ and $\delta(q, y) \in F$. The set of all the live states is called the *liveSet(A)*. A state that is not in the *liveSet* is called a *dead state*. The set of all dead states is called the *deadSet(A)*. Note that for a canonical DFA A, $deadSet(A)$ has at most one element.

The nondeterministic finite automata (NFA) model is a generalization of the DFA model, for a given state and an input symbol, the number of possible transitions can be greater than one.

Formally, a *nondeterministic finite automaton* A is a quintuple $(Q, \Sigma, \delta, q_0, F)$ where Q, Σ, q_0, and F are defined exactly the same way as for a DFA, and $\delta : Q \times \Sigma \to 2^Q$ is the transition function, where 2^Q denotes the power set of Q.

A DFA can be considered an NFA, where each value of the transition function is either a singleton or the empty set.

The computation relation $\vdash$: $Q\Sigma^* \times Q\Sigma^*$ of a NFA A is defined as follows: $px \vdash qy$ if $x = ay$ and $q \in \delta(p, a)$ for $p, q \in Q$, $x, y \in \Sigma^*$, $a \in \Sigma$. Then, the language accepted by A is

$$L(A) = \{w \mid q_0 w \vdash^* f, \text{ for some } f \in F\}$$

The family of languages accepted by NFA are denoted by $\mathcal{L}_{NFA}$.

Two automata are said to be equivalent if they accept the same language.

It is known that both deterministic and nondeterministic finite automata characterize the same family of languages, namely REG.

An important related notion is that of a *sequential transducer* which is nothing else than a finite automaton with outputs associated with its moves; we do not enter here into details and refer the reader to the general formal language theory literature.

Turing machines were devised by Alan Turing in 1936 in a paper [Turing, 1936] which lays the foundations of computer science.

A *Turing machine* is a construct $A = (Q, \Sigma, T, B, q_0, F, \delta)$, where:

- Q, Σ are disjoint alphabets (the set of states and the tape alphabet).
-$T \subseteq \Sigma$ is the input alphabet.
-$B \in \Sigma - T$ is the blank symbol.
-$q_0 \in Q$ is the initial state.
-$F \subseteq Q$ is the set of final states.
-δ is a partial mapping from $Q \times \Sigma$ to the power set of $Q \times \Sigma \times \{L, R\}$ (the move mapping; if $(q, b, d) \in \delta(p, a)$, for $p, q \in Q$, $a, b \in \Sigma$, and $d \in \{L, R\}$, then the machine reads the symbol a in state p and passes to state q, replaces a with b and moves the read-write head to the left when $d = L$ and to the right when $d = R$). If $|(\delta(p, a))| \leq 1$ for all $p \in Q$, $a \in \Sigma$, then A is said to be *deterministic*, where $|(\delta(p, a))|$ represents the number of elements of the transition function from the state p with the character a.

An *instantaneous description* of a Turing machine as above is a string xpy, where $x \in \Sigma^*$, $y \in \Sigma^*(\Sigma - \{B\}) \cup \{\lambda\}$, and $p \in Q$. In this way we identify the contents of the tape, the state, and the position of the read-write head: it scans the first symbol of y. Observe that the blank symbol may appear in x, y, but not in the last position of y; both x and y may be empty. We denote by ID_M the set of all instantaneous descriptions of M.

On the ID_M one defines the *direct transition* relation $\vdash_M$ as follows:

$$xpay \vdash_M xbqy \text{ iff } (q, b, R) \in \delta(p, a),$$
$$xp \vdash_M xbq \text{ iff } (q, b, R) \in \delta(p, B),$$
$$xcpay \vdash_M xqcby \text{ iff } (q, b, L) \in \delta(p, a),$$
$$xcp \vdash_M xqcb \text{ iff } (q, b, L) \in \delta(p, B),$$

where $x, y \in \Sigma^*$, $a, b, c \in \Sigma$, $p, q \in Q$.

The language recognized by a Turing machine A is defined by

$$L(A) = \{w \in T^* \mid q_0 w \vdash_M xpy \text{ for some } p \in F,\ x, y \in \Sigma^*\}.$$

(This is the set of all strings such that the machine reaches a final state when starting to work in the initial state, scanning the first symbol of the input string).

It is also customary to define the language accepted by a Turing machine as consisting of the input strings $w \in T^*$ such that the machine, starting from the configuration $q_0 w$, reaches a configuration where no further move is possible (we say that the machine *halts*); in this case, the set F of final states is no longer necessary. The two models of defining the language $L(M)$ are equivalent, the identified families of languages are the same, namely RE, and this is true both for deterministic and nondeterministic machines.

The difference between a finite automaton and a Turing machine is visible only in their functioning: the Turing machine can move its head in both directions and it can rewrite the scanned symbol, possibly erasing it (replacing it with the blank symbol).

Turing machines play a key role in computer science. As an abstract model which was later implemented in reality, it is a yardstick for various purposes. In particular, this is the case in the theory of complexity where the complexity of algorithms is measured with respect to an implementation of the algorithm in a Turing machine program.

Part II

State-of-the-art

Chapter 3

Relevant classes of languages or grammars

3.1. Main focus on Grammatical Inference

The field of Grammatical Inference has focused its research on learning regular grammars or deterministic finite automata (DFA).

> *Reasons justifying that most attention has been spent on this class of grammars are that this problem may seem simple enough but theoretical results make it already too hard for usual Machine Learning settings (...); on the other the class of DFA seems to be in some way maximal for certain forms of polynomial learning* [de la Higuera, 2005, p. 1335]

The problem of identifying DFA from examples has been studied quite extensively (see, e.g., [Angluin and Smith, 1983, Pitt, 1989]). A general review of the main results can be found in [Sakakibara, 1997].

The problem of identifying CFG has been considered as well, and several positive results have been obtained (see [Sakakibara, 1997]). Nonetheless,

there is not too many studies about identifying classes of grammars more powerful than CF by using grammatical inference techniques.

3.2. The Chomsky Hierarchy and its main limitations from a linguistic viewpoint

Despite the fact that most state-of-the-art grammatical inference algorithms apply to regular and context-free language (which belongs to the Chomsky Hierarchy), in this section we are going to consider the limitations of the Chomsky Hierarchy, especially when we want to study natural language syntax.

3.2.1. Where are Natural Languages located in the Chomsky Hierarchy?

One of the main limitations of the Chomsky Hierarchy emerges when we try to locate natural languages in this hierarchy.

The question of determining the location of natural languages in the Chomsky Hierarchy has been a subject of discussion since it was posed by Chomsky in his 1956 paper "Three Models for the Description of Language" [Chomsky, 1956].

> *(...) Noam Chomsky posed an interesting open question: When we consider the human languages purely as sets of strings of words (henceforth string-sets), do they always fall within the class called context-free languages (CFL's)?* [Pullum and Gazdar, 1982, p. 471].

In his 1957 "Syntactic Structures", Chomsky declared that he did not know the answer to this question.

> *English is not a regular language... I do not know whether or not English is itself literally outside the range of such analysis.* [Chomsky, 1957, p. 21].

And, in his 1959 "On certain formal properties of grammars" he stated that:

> *The main problem of immediate relevance to the theory of language is that of determining where in the hierarchy of devices the grammars of natural languages lie.* [Chomsky, 1959, p. 138].

This debate, which lasted for more than twenty years, was focused on the context-freeness of natural languages: "Are natural language context-free?".

In the 60's and 70's there was many attempts to prove that natural languages are not context-free. G.K. Pullum and G. Gazdar showed that all these attempts had failed:

> *What it has shown is that every published argument purporting to demonstrate the non-context-freeness of some natural languages is invalid, either formally or empirically or both. Whether non-context-free characteristics can be found in the stringset of some natural languages remains an open question, just as it was a quarter century ago.* [Pullum and Gazdar, 1982, p. 497].

And they concluded that for all they knew up to the time of that paper's publication, natural languages (conceived as string sets) might be context-free.

> *In the meantime, it seems reasonable to assume that the natural languages are a proper subset of the infinite-cardinality context-free languages, until such time they are validly shown not to be.* [Pullum and Gazdar, 1982, p. 499].

However, it was soon realized that natural languages, English included, are not context-free.

3.2.2. Examples of non-context-free constructions in natural languages

In the late 80's, some clear examples of natural language structures that cannot be described using a context-free grammar were discovered. Some examples of such constructions are list next:

- **Dutch**: Bresnan et al. studied cross-serial dependencies in Dutch, giving in this way an argument against the context-freeness of natural language.

  > *While Dutch may or may not be CF in the weak sense, it is not strongly CF: there is no CFG that can assign the correct structural descriptions to Dutch cross-serial dependency constructions.* [Bresnan et al., 1987, p. 314]

  The following example shows a duplication-like structure $\{w\bar{w} \mid w \in \{a, b\}^*\}$, where $\bar{w}$ is the word obtained from w by replacing each letter with its barred copy.

  > *...dat Jan Piet Marie de Kinderen zag helpen laten zwemmen*
  > (That Jan saw Piet help Marie make the children swim)

  This is only *weakly* non-context-free, i.e., only in the deep structure.

- **Bambara**: Bambara, an African language of the Mande family, was studied by Culy in [Culy, 1987]. He provided another argument against the context-freeness based on the morphology of words in that language.

  > *In this paper I look at the possibility of considering the vocabulary of a natural language as a sort of language itself. In particular, I study the weak generative capacity of the vocabulary of Bambara, and show that the vocabulary is not context-free. This result has important ramifications for the theory of syntax of natural language.* [Culy, 1987, p. 349].

A duplication structure is found in the vocabulary of Bambara, demonstrating a strong non-context-freeness, i.e., on the surface and in the deep structure:

> *malonyininafilèla o malonyininafilèla o*
> (one who searches for rice watchers + one who searches for rice watchers = whoever searches for rice watchers)

This has the structure $\{wcw \mid w \in \{a,b\}^*\}$. But also the *crossed agreement* structure $\{a^n b^m c^n d^m \mid \text{m, n}>0\ \}$ can be inferred.

- **Swiss German**: The paper by Shieber [Shieber, 1987], offers evidence for the non-context-freeness of natural language. He collected data from native Swiss German speakers, and he provided a formal proof of the non-context-freeness of Swiss German.

  > *Using a particular construction of Swiss German, the cross-serial subordinate clause, we have presented an argument providing evidence that natural languages can indeed cross the context-free barrier. The linguistic assumptions on which our proof rests are small in number and quite weak; most of the proof is purely formal. In fact, the argument would still hold even if Swiss German were significantly different from the way it actually is, i.e., allowing many more constituent orders, cases and constructions, and even if the meanings of the sentences were completely different.* [Shieber, 1987, p. 330].

  The following example is a strong non-context-free structure, again showing crossed agreement:

  > *Jan säit das mer (d'chind)^m (em Hans)^n es huus haend wele (laa)^m (hälfe)^n aastriiche*

(Jan said that we wanted to let the children help Hans paint the house)

This has the structure $xwa^m b^n y c^m d^n z$, where *a, b* stand for accusative, dative noun phrases, respectively, and *c, d* for the corresponding accusative, dative verb phrases, respectively.

In this way, all these works provide a negative answer to the question "Are natural languages context-free?". Besides, they suggest that more generative capacity than context-free grammar is required to describe natural languages.

Some authors go further on and conclude that "the world is not context-free". They discuss seven circumstances where context-free grammars are not enough (natural languages, programming languages, logic, formalizing the mapping graphs in symbolic terms, development biology, modelling economic processes, formal semiotic approaches to fairy-tales, music and visual arts).

> *The theory of context-free grammars is the most developed part of formal language theory due to the wide applicability and to the mathematical appeal of these grammars. However, "the world is not context-free": there is a lot of circumstances where naturally non-context-free languages appear.* [Dassow and Păun, 1989, p. 18].

Hence, the question of "How much power beyond context-free is necessary to describe these non-context-free constructions that appear in natural language?" became important.

Therefore, the Chomsky hierarchy does not provide a specific demarcation of language families having some desired properties from a linguistic point of view (for instance, to be able to describe these examples of constructions that have led to the non-context-freeness of natural language). The family of context-free languages has good computational properties, but it does not

contain some important formal languages that appear in human languages. On the other hand, the family of context-sensitive languages contains all important constructions that occur in natural languages, but it is believed that the membership problem for languages in this family cannot be solved in deterministic polynomial time.

3.3. Mildly Context-Sensitive Languages: a grammatical environment for natural language constructions

3.3.1. Introduction

Mildly Context-Sensitive Grammars and Languages ($MCSG$, $MCSL$) arose out the study of formal grammars adequate to model natural language structures.

As we have seen in the previous section, some clear examples of natural languages were discovered that required more formal power than CFG. Therefore, there was considerable interest in the development and study of grammatical formalisms with more generative power than CFG.

It would be desirable to have a family of languages that contains the most significant languages that appear in the study of natural languages and, also, languages in such a family to have good computational properties, i.e., the membership problem for languages in this family not be solvable in deterministic polynomial time complexity.

The idea of generating context-free and non-context-free structures, keeping under control the generative power, has led to the notion of Mildly Context-Sensitive devices.

Joshi introduced the notion of mild context-sensitivity [Joshi, 1985]. Based on the formal properties of a grammatical formalism called tree adjoining grammars (TAGs), he proposed that the class of grammars that is neces-

sary for describing natural languages might be characterized as the class of $MCSG$:

> *I would like to propose that the three properties*
> *1. limited crossed-serial dependencies*
> *2. constant growth, and*
> *3. polynomial parsing*
> *roughly characterize a class of grammars (and associated languages) that are only slightly more powerful than context-free grammars (context-free languages). I will call these "mildly context-sensitive grammars (languages)", MCSGs (MCSLs). This is only a rough characterization because conditions 1 and 3 depend on the grammars, while condition 2 depends on the languages; further, condition 1 needs to be specified much more precisely than I have done so far. I now would like to claim that grammars that are both weakly and strongly adequate for natural language structures will be found in the class of MCSGs.* [Joshi, 1985, p. 225].

3.3.2. Formal definition

Definition 3.3.1 *By a Mildly Context-Sensitive family of languages we mean a family $\mathcal{L}$ of languages that satisfies the following conditions:*

(i) *each language in $\mathcal{L}$ is semilinear*

(ii) *for each language in $\mathcal{L}$ the membership problem is solvable in deterministic polynomial time*

(iii) *$\mathcal{L}$ contains the following three non-context-free languages:*
- *multiple agreements:* $L_1 = \{a^n b^n c^n \mid n \geq 0\}$
- *crossed agreements:* $L_2 = \{a^n b^m c^n d^m \mid n, m \geq 0\}$
- *duplication:* $L_3 = \{ww \mid w \in \{a, b\}^*\}$

Figure 3.1 shows the location of the MCS family in the Chomsky hierarchy.

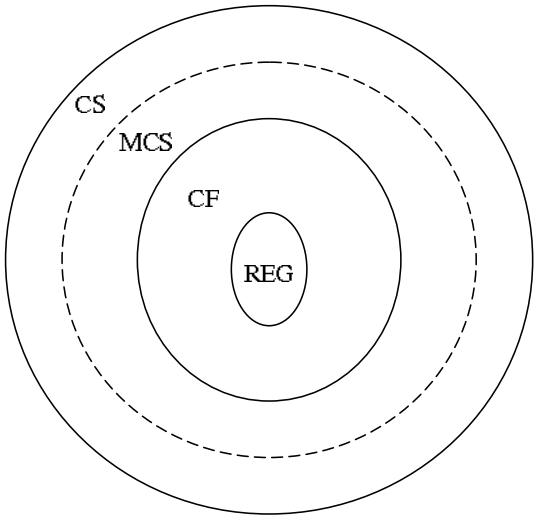


Figure 3.1: Location of $MCSL$ in the Chomsky Hierarchy

3.3.3. Generative devices

Several formalisms have been introduced and used to fabricate MCS families: *tree adjoining grammars* ([Joshi and Schabes, 1997]), *head grammars* [Roach, 1987], *combinatory categorial grammars* [Steedman, 1985], *linear indexed grammars* [Gazdar and Pullum, 1985], *simple matrix grammars* [Ibarra, 1970], etc. The first four ones were proved to be equivalent in terms of computational power [Joshi et al., 1991].

However, some authors consider that all these investigations are based on an implicit assumption which is not necessarily true ([Manaster-Ramer, 1999]).

> *As Manaster Ramer states, "the question as posed by Chomsky seems to suggest that the class of natural languages will be found somewhere in the Chomsky hierarchy. Yet this need not be the case,*

> *and probably is not. It is entirely possible, for example, that a realistic theory of natural languages would define a class of languages which is incommensurate with the Chomsky types, e.g., a few regular languages, a few non-regular context-free languages, a few non-context-free languages, and so on"* [Păun, 1997, xi].

Hence, natural languages could occupy an orthogonal position in the Chomsky Hierarchy. Therefore, we need a new hierarchy, which should certainly hold strong relationships with the Chomsky Hierarchy, but which should not coincide with it. In a certain sense, the new hierarchy should be incomparable with the Chomsky hierarchy and pass across it.

Since *external contextual grammars* have the property of transversality (they generate a class of languages occupying an orthogonal position with respect to the Chomsky Hierarchy), they appear to be appropriate candidates to model natural language syntax.

3.4. Contextual Grammars

3.4.1. Introduction

In the 50's there were great steps in the investigation of natural languages using mathematical rules. In 1957, Chomsky published his pioneering book [Chomsky, 1957], presenting a new generative approach to syntactic structures. And the research of some Russian mathematicians gave the start in the development of analytical mathematical models of languages.

Attempts have been made to bridge the gap between these two trends, and contextual grammars are part of this process.

> *Generative grammars are a rupture from the linguistic tradition of the first half of 20th century, while analytical models are just the development, the continuation of this tradition. It was natural to*

> *expect an effort to bridge this gap. This effort came from both parts and, as we shall see, contextual grammars are a component of this process* [Marcus, 1997, p. 215].

Contextual grammars were introduced by S. Marcus in [Marcus, 1969]. The circumstances and the motivation of introducing contextual grammars have been explained in detail in [Marcus, 1997].

> *Contextual grammars (shortly CGs) have their origin in the attempt to transform in generative devices some procedures developed within the framework of analytical models. The idea to connect in this way the analytical study with the generative approach to natural languages was one of the main problems investigated in mathematical linguistics from 1957 to 1970. (...)*
> *CGs try to exploit two ideas that were at the very beginning of the tradition of descriptive distributional linguistics in USA., in both the 1940s and the 1950: the idea of a string on a given finite nonempty alphabet A and the idea of a context on A, conceived as an ordered pair (u, v) of strings over A.* [Marcus, 1997, p. 216].

Therefore, contextual grammars are based on the idea of modelling some natural aspects from descriptive linguistics, for instance, the acceptance of a word (construction) only in certain contexts.

> *In descriptive linguistics the association of certain strings with certain contexts (pair of strings) with respect to a more or less explicit idea of well-formedness is a basic ingredient of most (if not all) linguistic analysis. Contextual grammars try to capture this strings-contexts interaction, under the form of a generative device (complementing in this way its analytic status in structural linguistics).* [Păun, 1997, pp.xi-xii].

Roughly speaking, a contextual grammar produces a language starting from a finite set of words (*axioms*) and iteratively adding *contexts* (pair of words) to the currently generated words. Despite the fact that these mechanisms generate a proper subclass of simple matrix languages, they are still MCS. These models are (technically) much simpler than any other models found in the literature on MCS families of languages. Unlike the Chomsky grammars, contextual grammars do not involve nonterminals and they do not have rules of derivation except one general rule: to adjoin contexts.

There are many variants of contextual grammars, but all of them are based on context adjoining. The differences are in the way of adjoining contexts, the sites where contexts are adjoined, the use of selectors, etc. For a detailed introduction to the topic, see the monograph [Păun, 1997].

As Gh. Păun points out in [Păun, 1997], it is paradoxical that although contextual grammars were motivated by natural language investigations, they were studied for about 25 years as a mathematical object, without exploiting their linguistic relevance. He indicates two possible types of linguistic relevance of a contextual grammar in order to explain this situation.

> *There is first the relevance that can follow from the linguistic significance of various contextual relations and operations involved, directly or indirectly, in the basic components of such a grammar. The most important of them is perhaps the quasiorder relation of contextual domination: a string x contextually dominates string y with respect to a given language L if any context accepting x in L also accepts y in L. This quasiorder relation makes it possible to compare strings from the standpoint of their contextual ambiguity (...). We can easily understand why this important source of linguistic relevance of contextual grammars has so far been ignored: the associated mathematical models, mainly developed in the fifties and sixties, as well as the corresponding linguistic facts, have been*

> *largely ignored by the new generations of researchers in the field of formal grammars, guided mainly by motivations coming from computer science.*
> *A second type of linguistic relevance of contextual grammars refers to their capacity to capture various types of recursive behaviours occurring in natural languages. These recursive aspects lead to formal languages as* $\{ww \mid w \in \{a,b\}^*\}, \{a^n b^n c^n \mid n \geq 1\}, \{a^n b^m c^n d^m \mid n, m \geq 1\}$*, etc. (...)* [Păun, 1997, p. xii].

The mathematical richness of the field of contextual grammars has been demonstrated, but there is a need of more research on the relevance for natural language modelling, both of a mathematical type (for example, efficient parsing algorithms) and of a linguistic type (trying to build contextual grammars for certain fragments of given natural languages).

> *Indeed, contextual grammars, in the many variants considered in the literature, were investigated mainly from a mathematical point of view; see Paun (1982, 1985, 1994), Paun, Rozenberg and Salomaa (1994), and their references. A complete source of information is the monograph Paun (1997). A few applications of contextual grammars were developed in connection with action theory (Paun 1979), with the study of theatrical works (Paun 1976), and with computer program evolution (Balanescu and Gheorghe 1987), but up to now no attempt has been made to check the relevance of contextual grammars in the very field where they were motivated: linguistics, the study of natural languages. A sort of a posteriori explanation is given: the variants of contextual grammars investigated so far are not powerful enough, hence they are not interesting enough; what they can do, a regular or a context-free grammar can do as well.* [Marcus et al., 1998, p. 246].

3.4.2. Formal Definitions

In the derivation process of the contextual grammars, the contexts can be added in two different ways: as introduced in [Marcus, 1969], at the end of the current string - we call these grammars *external*; or as introduced in [Păun and Nguyen, 1980], inside the current string - we call these grammars *internal*. Details about these basic variants of contextual grammars can be found in [Păun, 1997].

Many variants have been investigated in the last decade: determinism, parallelism, normal forms, modularity, use of selectors, etc. (see, [Păun, 1997] for details). All these variants have the main goal of finding a class of contextual languages appropriate from natural language point of view.

In this dissertation we investigate the external type of contextual grammars and, especially, the many dimensional case. We will point out the generative capacity of these grammars and their relevance for natural language modelling. As we will see later, this type of grammar has an important property from a linguistic point of view: the three basic features of natural (and artificial) languages that lead to their non-context-freeness (multiple agreements, crossed agreements and duplication) can be covered by such grammars. And moreover, all of them can be generated by many-dimensional external contextual grammars in a simple way.

We now start by reviewing the notion of a contextual grammar. Later, the extension of these grammars to the many-dimensional case is explained.

3.4.2.1. External contextual grammars (EC)

Definition 3.4.1 *A External Contextual grammar is* $G = (\Sigma, B, C)$*, where* Σ *is the alphabet of* G*,* B *is a finite subset of* Σ^* *called the base of* G*, and* C *is a finite set of contexts, i.e. a finite set of pairs of words over* Σ*.* C *is called the set of contexts of* G*.*

The direct derivation relation with respect to G is a binary relation between words over Σ*, denoted* $\Rightarrow_G$*, or* $\Rightarrow$ *if G is understood from the context. By definition,* $x \Rightarrow_G y$*, where* $x, y \in \Sigma^*$*, iff* $y = uxv$ *for some* $(u, v) \in C$*. The derivation relation with respect to G, denoted* $\Rightarrow_G^*$*, or* $\Rightarrow^*$ *if G is understood from the context, is the reflexive and transitive closure of* $\Rightarrow_G$*.*

Definition 3.4.2 *Let* $G = (\Sigma, B, C)$ *be a External Contextual grammar. The language generated by G, denoted by* $L(G)$*, is defined as:*

$L(G) = \{\ y \in \Sigma^* \mid$ *there exists* $x \in B$ *such that* $x \Rightarrow_G^* y\ \}$.

One can verify that the language generated by $G = (\Sigma, B, C)$ is the smallest language L over Σ such that:

(i) $B \subseteq L$

(ii) if $x \in L$ and $(u, v) \in C$, then $uxv \in L$

The family of all External Contextual languages is denoted by EC.

Remark 3.4.3 $EC = MinLIN$*, which is a strict subfamily of* LIN*, incomparable with REG (see [Păun, 1997]).*

3.4.2.2. Many-dimensional external contextual grammars (EC_p)

Many-dimensional External Contextual grammars are an extension of External Contextual grammars, but they work with vectors of words and vectors of contexts [Kudlek et al., 2002] .

Let $p \geq 1$ be a fixed integer, and let Σ be an alphabet. A *p-word* x over Σ is a p-dimensional vector whose components are words over Σ, i.e., $x = (x_1, x_2, ..., x_p)$, where $x_i \in \Sigma^*, 1 \leq i \leq p$. A *p-context* c over Σ is a p-dimensional vector whose components are contexts over Σ, i.e., $c = [c_1, c_2, ..., c_p]$ where $c_i = (u_i, v_i), u_i, v_i \in \Sigma^*$, $1 \leq i \leq p$. We denote vectors of words with round brackets, and vectors of contexts with square brackets.

Definition 3.4.4 *Let $p \geq 1$ be an integer. A p-dimensional External Contextual grammar is $G = (\Sigma, B, C)$, where Σ is the alphabet of G, B is a finite set of p-words over Σ called the base of G, and C is a finite set of p-contexts over Σ. C is called the set of contexts of G.*

The direct derivation relation with respect to G is a binary relation between p-words over Σ, denoted by $\Rightarrow_G$, or $\Rightarrow$ if G is understood from the context. Let $x = (x_1, x_2, ..., x_p)$ and $y = (y_1, y_2, ..., y_p)$ be two p-words over Σ. By definition, $x \Rightarrow_G y$ iff $y = (u_1x_1v_1, u_2x_2v_2, ..., u_px_pv_p)$ for some p-context $c = [(u_1, v_1), (u_2, v_2), ..., (u_p, v_p)] \in C$. The derivation relation with respect to G, denoted by $\Rightarrow_G^*$, or $\Rightarrow^*$ if no confusion is possible, is the reflexive and transitive closure of $\Rightarrow_G$.

Definition 3.4.5 *Let $G = (\Sigma, B, C)$ be a p-dimensional External Contextual grammar. The language generated by G, denoted $L(G)$, is defined as:*

$L(G) = \{y \in \Sigma^* |$ *there exists* $(x_1, x_2, ..., x_p) \in B$ *such that* $(x_1, x_2, ..., x_p) \Rightarrow_G^* (y_1, y_2, ..., y_p)$ *and* $y = y_1y_2...y_p\}$.

The family of all p-dimensional External Contextual languages is denoted by EC_p.

Remark 3.4.6 *Any family EC_p for $p \geq 2$ is a subfamily of linear simple matrix languages (see [Kudlek et al., 2002]).*

Chapter 4

Models in Grammatical Inference

In this chapter we present the three important formal models that have been widely investigated in the Grammatical Inference framework. We also discuss some linguistic aspects of these models.

4.1. Identification in the limit

4.1.1. Learning in the Limit Model

In 1967 Gold introduced the model of *identification in the limit* [Gold, 1967], with the ultimate goal of explaining the learning process of the natural language.

> *The study of language identification described here derives its motivation from artificial intelligence. The results and the methods used also have implications in computational linguistics, in particular the construction of discovery procedures, and in psycholinguistics, in particular the study of child learning (...).*
> *I wish to construct a precise model for the intuitive notion "able to*

> *speak a language" in order to be able to investigate theoretically how it can be achieved artificially. Since we cannot explicitly write down the rules of English which we require one to know before we say he can "speak English", an artificial intelligence which is designed to speak English will have to learn its rules from implicit information. That is, its information will consist of examples of the use of English and/or of an informant who can state whether a given usage satisfies certain rules of English, but cannot state these rules explicitly.* [Gold, 1967, pp. 447–448].

Identification in the limit views learning as an infinite process. In this model, an infinite sequence of examples of the unknown language is presented to the learner (inference algorithm), and the eventual or limiting behavior of the algorithm is used as the criterion of its success.

> *A class of possible languages is specified, together with a method of presenting information to the learner about an unknown language, which is to be chosen from the class. The question is now asked, "Is the information sufficient to determine which of the possible languages is the unknown language?" Many definitions of learnability are possible, but only the following is considered here: Time is quantized and has a finite starting time. At each time the learner receives a unit of information and is to make a guess to the identity of the unknown language on the basis of the information received so far. This process continues forever. The class of languages will be considered learnable with respect to the specified method of information presentation if there is an algorithm that the learner can use to make his guesses, the algorithm having the following property: Given any language of the class, there is some finite time after which the guesses will all be the same and they will be correct* [Gold, 1967, p. 447].

Therefore, identification in the limit can be defined as follows:

Definition 4.1.1 *Method M identifies language L in the limit if, after a finite number of examples, M makes a correct guess and does not alter its guess thereafter. A class of languages is identifiable in the limit if there is a method M such that given any language of the class and given any admissible example sequence for this language, M identifies the language in the limit.*

Note that under this criterion the learner can never be certain of having correctly guessed the language, since new examples could appear at any time step.

> *In the case of identifiability in the limit the learner does not necessary know when his guess is correct. He must go on processing information forever because there is always the possibility that information will appear which will force him to change his guess. (...) My justification for studying identifiability in the limit is this: A person does not know when he is speaking a language correctly; there is always the possibility that he will find that his grammar contains an error. But we can guarantee that a child will eventually learn a natural language, even if it will not know when it is correct.* [Gold, 1967, p. 450].

In this model, only positive examples (sentences that are in the language to be learned) are given to the learner. This is known as learning from *text* or *positive presentation*. This assumption corresponds to the empirical fact that in first language acquisition no systematic negative evidence (examples of sentences that are not in the language) is available to the children.

> *Recently, psycholinguists have begun to study the acquisition of grammar by children (e.g., McNeill, 1966). Those working in the field generally agree that most children are rarely informed when*

> *they make grammatical errors, and those that are informed take little heed. In other words, it is believed that it is possible to learn the syntax of a natural language solely from positive instances, i.e., a "text".* [Gold, 1967, p. 453].

From the point of view of language acquisition, *text* might correspond to the following situation. We imagine that a child is presented with grammatically correct sentences of a language in an arbitrary order (repetitions are allowed). Negative information (i.e., ungrammatical sentences) is not presented. Each sentence of the language appears eventually, without any restriction on the order of their arrival.

Gold also considered learning from positive and negative examples, which is know as learning from *informant* or *complete presentation*. If complete data is available (we know whether or not a sentence belongs to the target language), learning turns out to be much easier. However, as Kanazawa states, *"complete data has little relevance to first language acquisition and linguistic theory"* [Kanazawa, 1998, p. 3]. Empirical evidence suggests that natural language is not acquired in this way (see [Brown and Hanlon, 1970], [Goodluck, 1991]).

Although it is desirable that learning can be achieved using only positive data, Gold proves in [Gold, 1967] that soon as a class of languages contains all finite languages and at least one infinite language (called a *superfinite* class), it is not identifiable in the limit from positive data.

Theorem 4.1.2 *A superfinite family of languages is not learnable in the limit from positive data.*

Hence, it implies that even the smallest class in the Chomsky Hierarchy, i.e. the class of regular languages, is not identifiable in the limit from positive data. The problem is that no positive example can refute a too general

hypothesis. As Angluin point out in [Angluin, 1980], when only positive examples are available, there is danger of overgeneralization.

> *Intuitively, an added difficulty in trying to do inference from positive rather than positive and negative data is the problem of "overgeneralization". If in the course of making guesses the inferring process makes a guess that is overly general, i.e, specifies a language that is a proper subset of the true answer, then with positive and negative data there will eventually be a counterexample to the guess, i.e., a string that is contained in the guessed language but is not a member of the true language. No such specific conflict with the examples will occur in the case of inference from positive data.* [Angluin, 1980, p. 118].

Pursuant to his result, none of the language classes defined by Chomsky to model natural language is identifiable in the limit from only positive data.

How do children overcome Gold's theoretical hurdle? Gold suggested several hypothesis to overcome this difficulty:

> *If one accepts identification in the limit as a model of learnability, then this conflict must lead to at least one of the following conclusions:*
>
> 1. *The class of possible natural languages is much smaller than one would expect from our present models of syntax. That is, even if English is context-sensitive, it is not true that any context-sensitive language can occur naturally. Equivalently, we may say that the child starts out with more information than that the language it will be presented is context-sensitive. In particular, the results on learnability from text imply the following: The class of possible natural languages, if it contains languages*

of infinite cardinality, cannot contain all languages of finite cardinality.

2. *The child receives negative instances by being corrected in a way we do not recognize. If we can assume that the child receives both positive and negative instances, then it is being presented information by an "informant". The class of primitive recursive languages, which includes the class of context-sensitive languages, is identifiable in the limit from an informant. The child may receive the equivalent of negative instances for the purpose of grammar acquisition when it does not get the desired response to an utterance. It is difficult to interpret the actual training program of a child in terms of the naive model of a language assumed here.*
3. *There is an a priori restriction on the class of texts which can occur, such as a restriction on the order of text presentation. The child may learn that a certain string is not acceptable by the fact that it never occurs in a certain context. This would constitute a negative instance.*

[Gold, 1967, p. 453–454].

Works in this direction have showed that the first path (the class of potential natural language is more restrictive than those defined by Chomsky) can be successful (see, [Angluin, 1982, Sakakibara, 1992, Kanazawa, 1998]). In linguistics, it is generally also assumed that the first conclusion holds (i.e., the class of humanly learnable languages is severely restricted).

Now it seems evident to many linguists (notably, Chomsky [40,43]) that children are not genetically prepared to acquire any arbitrary language on the basis of the kind of casual linguistic exposure typically afforded the young. Instead, a relatively small class

> *$\mathcal{H}$ of languages may be singled out as "humanly possible" on the basis of their amenability to acquisition by children, and it falls to the science of linguistics to propose a nontrivial description of $\mathcal{H}$* [Jain et al., 1999, p.29].

Due to Gold's result, learning from only positive data had been considered too difficult to be of much theoretical interest.

> *Since all non-trivial classes in the Chomsky-hierarchy are superfinite this result was interpreted as showing that identification in the limit is just 'too hard' and thus a trivial model of learning.* [Costa-Florêncio, 2003] p. 3

However, in the early eighties, there was renewed interest in the paradigm, mainly because of work by D. Angluin. In [Angluin, 1979, Angluin, 1980, Angluin, 1982], she provide examples of nontrivial classes of languages for which correct inference from positive data is possible.

Therefore, Angluin's work show that the initial pessimism about the Gold's model was premature.

> *In fact, it seems that searching for specific classes of languages which are inferrable from positive data may help to generate new approaches to concrete problems of inductive inference that avoid the difficulties of the apparent computational intractability of some previous approaches.* [Angluin, 1980, p. 118].

Later, Shinohara was able to come up with an impressive result; languages generated by context-sensitive grammars with a bounded number of rules can be identified from positive data (see [Shinohara, 1994]).

One of the most basic strategy in learning in the limit models is learning by enumeration. Both *identification in the limit* and *identification by enu-*

meration have been very important in theoretical work developed later in the field of inductive inference.

> *Identification by enumeration is an abstraction of our first method for guessing polynomial sequences, namely, systematically searching the space of possible rules until one is found that agrees with all the data so far. Suppose that a particular domain of rules is specified, and there is an enumeration of descriptions, say,* $d_1, d_2, d_3, ...$, *such that each rule in the domain has one or more descriptions in this enumeration (...). Given any collection of examples of a rule, the method of identification by enumeration goes down the list to find the first description, say* d_i*, that is compatible with the given examples and then conjectures* d_i*.* [Angluin and Smith, 1983, p. 241].

The enumeration method is guaranteed to achieve correct identification in the limit if the following two conditions are satisfied: a correct hypothesis is always compatible with the examples given; any incorrect hypothesis is incompatible with some sufficiently large collection of examples and with all larger collections. Although this method is very powerful, it is inefficient in time because of its exhaustive nature of enumeration.

> *The method of identification by enumeration is very general and powerful but also rather impractical because the size of the space that must be searched is typically exponential in the length of the description of the correct guess.* [Angluin and Smith, 1983, p. 241].

There are at least two directions to improve this time-efficiency. On the one hand, to find an appropriate subfamily of languages for which an efficient learning is possible. On the other hand, to enhance a learning protocol so that more information is available to a learning algorithm. Some works in these directions have been developed later.

As we have seen, the identification in the limit model is not a trivial model of learning. Moreover, we can find some similarity between learning in Gold's model and first language acquisition. In both cases there is a process of *improvement*: in identification in the limit model, the new conjecture is better that the previous guess; in the case of first language acquisition there is a progressive improvement of the language acquired by the child.

In the case of second language learning the way to master the language is quite different, there is not this process of making guesses of grammars; adults learn second languages from grammar books, attending some courses, establishing some analogies with the first languages that they already know, etc. Moreover, usually adults learn second languages with some difficulty and not with the same success that in first language acquisition.

However, there are some aspects of Gold's model that are controversial from a linguistic point of view. We point out some of them below.

In Gold's model *language* is defined as a set of strings (e.g., sentences) on some finite alphabet (e.g., words). This concept of language is quite different of what is understood by language from a linguistic point of view:

> *The fact that different languages have different vocabulary items is not what makes them different within this framework [Gold's sense]. Rather, the vocabulary is held constant within the language class, and each language is defined in terms of which combinations of vocabulary items they allow in sentences. A class of languages is primarily defined in terms of the kinds of rules that are allowed in the grammar.* [Gordon, 1990, p. 217].

The characteristics of the *learner* given by Gold's paradigm are also controversial from a linguistic point of view.

> *A learner is a general computing device (e.g. a Turing machine) that accepts input sentences from the environment and guesses*

> *which language they are from. If a new sentence is consistent with the previous guess, the learner will stick with the same guess; otherwise it will try a new language. Notice that the form of the learning function here really says nothing about the kinds of rule-inductive processes with which language acquisition theorists are concerned. For all intents and purposes, the learner already knows the functions (i.e., grammars) that generate the languages within the class. All it has to do is figure out which one it is being presented with.*[Gordon, 1990, p. 218].

Therefore, *learning* within Gold's paradigm is not really what we normally associate with this term. As we have seen, *in the limit* denotes the criterion of success, which suppose that there is no limit on how long it can take the learner to guess the correct language.

> *That is, a language has been correctly identified when the learner no longer changes its guess through the presentation of all of the (possible infinite) strings in the language. If the learner is lucky, the first guess could be correct. Alternatively, it might take several billions of years to come up with the correct guess.*[Gordon, 1990, p. 218].

Hence, considerations of efficiency form a somewhat separate line of analysis from Gold's work, which was concerned with limiting behavior rather than speed of learning. However, from natural language acquisition point of view efficiency is also important.

Although learning natural language is an infinite process, we are able to learn the language in an efficient way. Therefore, learning from polynomial time will play a fundamental role. Why polynomial time? On one hand, exponential concerns the intractable, and normal children learn the language they are exposed to. On the other hand, linear is too simple, and we know

that mastering a natural language is a complex task. Hence, polynomial is what remains in the "middle", it is the standard.

Natural language learning is mainly based on positive examples. However, positive data only is less than what a child actually gets in the learning process and, on the other hand, informant is much more than what a learner can expect. Therefore, learning from only positive data could be enough in a first stage. Though, as we have pointed out before, when only positive data is available there is danger of overgeneralization; learner could overgeneralize and positive data would not be helpful to refute a too general hypothesis. How could we avoid that problem? In Chapter 7 we propose an answer to that question.

Learning from text or informant is also known as passive learning, as the learner passively received strings of the language. We know that natural language learning is more than that. Children also interact with their environment. They produce sentences that could be grammatically correct or not, and they can also ask questions to the adults, etc. Therefore, there is an interaction between child and adult, that is not gained by *identification in the limit from only positive data*.

Moreover, in identification in the limit the current hypothesis has to be consistent with all the examples seen so far. From a linguistic point of view this assumption is unrealistic, due to children are unlikely to remember the entire record of sentences ever addressed to them.

Therefore, the definition of identification in the limit postulates greatly idealized conditions, as compared to the conditions under which children learn language.

If we compare Gold's model with Chomsky's argument to explain natural language acquisition, we can see that there are some differences which will be interesting to point out here. In Gold's model there is not any consideration

about innate knowledge; the learner learns only from the data received. In Chomsky's theory the data is only an excuse: Chomsky considers that the linguistic data available to the child is poor and it is only a stimulus to activate the development of the grammar (Chomsky's *argument from the poverty of the stimulus*). Therefore, thanks to the data there is an activation of the language acquisition device (LAD). This leads to the Universal Grammar that is underlaid in the grammar of each natural language (see Chapter 2).

4.1.2. Conditions for Positive Data Learnability in the Limit

As we have seen, superfinite languages cannot be learned in the limit from positive data. Therefore, what makes the target language learnable in the limit from positive data only? Angluin was the first who considered this problem and presented necessary and sufficient conditions for learnability in the limit from positive data.

In the sequel, we present several conditions for the language family to be learnable in the limit from positive data. We are going to concentrate on [Yokomori, 2004]

For a language family $\mathcal{C}$, consider the following conditions:

- **Condition C1** (Finite tell-tale property): $\mathcal{C}$ has the finite tell-tale property if, for any L in $\mathcal{C}$, there exist a finite set T of L such that for any L' in $\mathcal{C}$, $T \subseteq L'$ implies that $L' \not\subset L$.
- **Condition C2** (Characteristic sample property): $\mathcal{C}$ has the characteristic sample property if, for any L in $\mathcal{C}$, there exist a finite set T of L such that for any L' in $\mathcal{C}$, $T \subseteq L'$ implies that $L \subseteq L'$.
- **Condition C3** (Finite elasticity property): $\mathcal{C}$ has the finite elasticity property if $\mathcal{C}$ does not have the infinite elasticity, where $\mathcal{C}$ has the *infinite elasticity* iff there exist infinite sequences $w_0, w_1, w_2, ...$ and $L_1, L_2, L_3, ...$

in $\mathcal{C}$ such that for any $n \geq 1$, $\{w_0, w_1, w_2, ..., w_{n-1}\} \subseteq L_n$ and $w_n \notin L_n$ (see Figure 4.1).

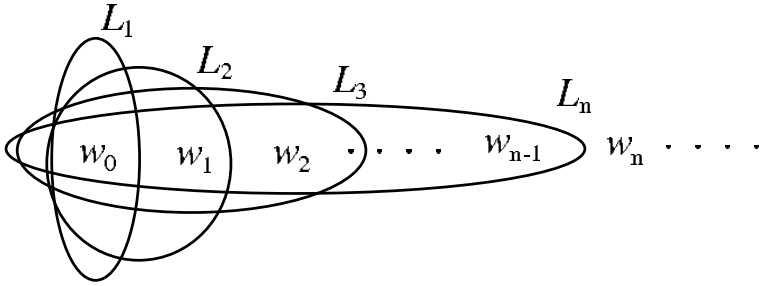


Figure 4.1: Infinite elasticity property

- **Condition C4** (Finite thickness property): $\mathcal{C}$ has the finite thickness property if, for any w, the family $\{L|L \in \mathcal{C} \text{ and } w \in L\}$ is finite.

Finally, we consider a weak version of Condition C1:

Condition EC1 (Existential C1): $\mathcal{C}$ has the *existential* finite tell-tale property if, for any L in $\mathcal{C}$, there exists a finite set T of L such that for any L' in $\mathcal{C}$, $T \subseteq L'$ implies that $L' \not\subseteq L$.

Based on all these conditions, the following theorem is obtained:

Theorem 4.1.3 *(i) Condition C4 implies Condition C3 (the finite thickness property implies the finite elasticity property).*

(ii) Condition C3 implies Condition C2 (the finite elasticity property implies the characteristic sample property).

(iii) Condition C2 implies Condition C1 (the characteristic sample property implies the finite tell-tale property).

(iv) Condition C1 implies Condition EC1.

The proof of this theorem can be found in [Yokomori, 2004].

4.1.3. Efficient Learning in the Limit

When we want to learn something *efficiently* in any sense, but still *in the limit* framework, a significant question arise: What sort of *definition* for efficient learning in the limit can be possible? The answer to this question could be: polynomial time learning in the limit. But, in what sense we should analyze the time complexity of an "in the limit" algorithm?

One may define the notion of *polynomial time identification in the limit* in various ways. It was not until late 80's that the polynomial time identifiability in the limit was reasonably defined by Pitt [Pitt, 1989]. By making a slight modification of his definition, T. Yokomori proposed the following definition for polynomial time identification in the limit from positive data [Yokomori, 1991]:

Definition 4.1.4 *A class $\mathcal{R}$ is polynomial-time identifiable in the limit if and only if there is an algorithm $\mathcal{A}$ such that:*

- *$\mathcal{A}$ identifies $\mathcal{R}$ in the limit.*
- *$\mathcal{A}$ has polynomial update time (time used by A between receiving the ith example w_i and outputting the ith conjecture r_i).*
- *$\mathcal{A}$ makes a polynomial amount of implicit prediction errors (A makes an implicit prediction error at step i if the conjecture r_i is not consistent with the $(i+1)$st example, i.e., if $L(r_i)$ fails to contain the $(i+1)$st example).*

From the point of view of language acquisition, an important item of this approach is to consider the implicit prediction errors that the learner makes. When we want to study the efficiency of a learning process it has sense to take into account this kind of error made by the learner; if the learner classifies correctly the examples that receives but these examples are not enough to find the correct grammar, this is not a fault of the learner.

Based on [Goldman and Kearns, 1995], [Goldman and Mathias, 1996], Colin de la Higuera proposes the model of *identification in the limit model from polynomial time and data* [de la Higuera, 1997]. For a class of grammars to be learnable in this setting it is required that each grammar in the class admits some robust characteristic set of polynomial size. The set is characteristic in the sense that from it the learning algorithm will return some equivalent grammar, and robust in the sense that this remains true whenever this characteristic set is included in any correctly labelled learning set.

Taking into account the fact that the length of the examples must depend polynomially on the size of the concept to be learned, Colin de la Higuera proposed the following definition, which is just a generalization of Gold's results [Gold, 1978] and a natural restriction of the definition of polynomial update time [Pitt, 1989].

Definition 4.1.5 *A representation class* $\boldsymbol{R}$ *is identifiable in the limit from polynomial time and data iff there exist two polynomials* $p()$ *and* $q()$ *and an algorithm* A *such that:*

1. *Given any sample (S+, S-), of size* m*,* A *returns a representation* R *in* $\boldsymbol{R}$ *compatible with (S+, S-) in* $O(p(m))$ *time.*
2. *For each representation* R *of size* n*, there exists a characteristic sample (CS+, CS-) of size less than* $q(n)$ *for which, if* $S+ \supseteq CS+, S- \supseteq CS-$*,* A *returns a representation* R' *equivalent with* R*.*

By this definition algorithm A is a polynomial learner.

The idea of characteristic sample in natural language is not intuitive at all. It might be related to a set of same sentences schemes. However, it is not an idea that corresponds to the current research concerns in linguistics.

4.2. Query Learning

The results on learning in the limit model discussed in the previous section suggest that, instead of passively receiving strings of the language, the learner could be allowed to make queries to the teacher. In that way, additional information is available in the learning process.

> *Imagine a human expert in some domain, for example, cancer diagnosis, attempting to communicate the method he or she uses in that domain to an expert system. Specific positive and negative examples will form an important component of the communication, in addition to advice about general rules, explanations of significant and irrelevant features, justifications of lines of reasoning, clarifications of exceptions, and so on. Moreover, the examples given are likely to be chosen so that they are "central" or "crucial" rather than random or arbitrary, in an attempt to speed convergence of the system to a correct hypothesis.*
> *Studies of learning general rules from examples have generally assumed a source of examples that is arbitrary or random [2]. The scenario above suggests that it is reasonable to investigate learning methods that assume that the source of examples is "helpful". To emphasize this aspect, the source of examples will be called the Teacher and the learning algorithm the Learner* [Angluin, 1987, p. 87].

Learning from queries was introduced by Angluin [Angluin, 1987]. In query learning, there is a teacher (oracle) that knows the language and has to answer correctly specific kind of queries asked by the learner.

The possible queries available for the learner are: membership query, equivalence query, subset query, superset query, disjointness query, exhaustiveness query (see, [Angluin, 1988]).

A teacher that can answer membership and equivalence queries is often called *Minimally Adequate Teacher* (MAT). These two typical types of queries include the following:

(i) *Membership query* (MQ). The input is a string $w \in \Sigma^*$ and the output is "yes" if w is in L (target language) and "no" otherwise. See Figure 4.2.

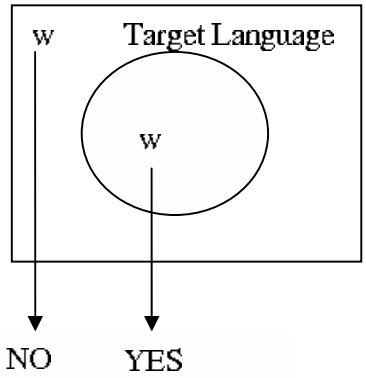


Figure 4.2: Membership Query

(ii) *Equivalence query* (EQ). The input is a grammar *G'* and the output is "yes" if *G'* generates the same language as G (*G'* is equivalent to G) and "no" otherwise. If the answer is "no" a string w in the symmetric difference of the language *L(G)* generated by G and the language *L(G')* generated by *G'* is returned. This returned string w is called a *counterexample*. See Figure 4.3.

.

In [Angluin, 1987], Angluin gave an algorithm for learning DFA from MQs and EQs which, given any regular language, learns from MAT a minimum DFA accepting the target language in polynomial time in the number of states of the minimum DFA and the maximum length of any counterexample provided by the teacher. She was the first who proved learnability of DFA

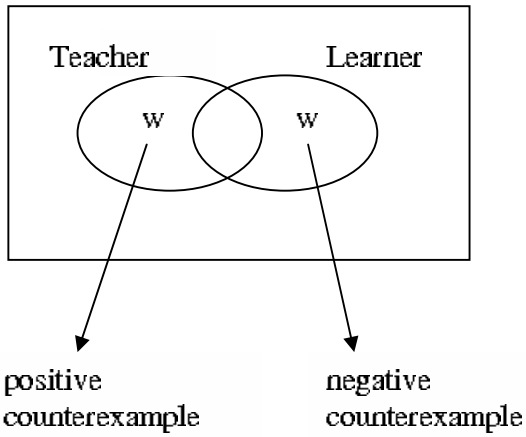


Figure 4.3: Counterexample

via queries. In the next chapter we will explain the main important ideas of her algorithm, which is referred as L^*.

Theorem 4.2.1 *The family of regular languages is MAT-learnable in polynomial time using the class of DFA.*

MQs and EQs have established themselves as the standard combination to be used for exact learning. Angluin proves that both of these queries are necessary in order to infer DFA in polynomial time (either MQs o EQs alone do not allow for polynomial inference).

Angluin's model addresses an important tool available to a child, i.e., queries to a teacher (usually, a family adult member). Therefore, we consider that query learning model might be useful when representing several aspects of the process of children's language acquisition.

However, the type of queries introduced with this model are quite unnatural for real learning environments. For instance, when the learner asks about a word in the language, the teacher's answer is very simple, *yes* or

no. We consider that this hypothesis is oversimplified for a normal learning process. And, on the other hand, a query such as a equivalency query, will never be produce in a real situation; a child will never ask to the adult if her grammar is the correct one. Therefore, what about learning from another kind of queries? Why do not model a more natural way of answering? (according to the linguistic reality). In Chapter 7 we propose several answers to that questions.

Although this model is known as active learning since the learner can ask queries to the teacher, we can see that the learner does not really interact with the teacher; he can ask MQs or EQs, but he does not really communicate with the teacher producing some sentences, etc. In the communication between children and adults we can see that the role of the children is more active, and not limited to ask this kind of queries.

Moreover, we consider that this model does not adequately reflect the fact that a learner, during the process of language acquisition, potentially gets access to all correct statements. In Chapter 7 we propose a solution to that problem.

Angluin's model is known as exact learning. From a linguistic point of view, every person has (small) imperfections in their linguistic competence, despite the fact that normally they have a good command of their language. Therefore, this is an argument showing that learning the exactly correct language is a hard task.

The teacher in this model is assumed to know everything and always gives the correct answers. Therefore, he is an ideal teacher that does not correspond with a real situation.

4.3. PAC learning

Exact learning has always been considered a hard problem. Valiant introduced *probably approximately correct learning* (PAC learning, in short) [Valiant, 1984], which is a distribution-independent probabilistic model of learning from random examples. It is is widely used for learning several different concept classes approximately.

In this model, the inference algorithm takes a sample as input and produces a grammar as output. A successful inference algorithm is one that with high probability finds a grammar whose error is small.

> *An unknown distribution over the examples exists, and examples are sampled under this distribution. Learning is done from this sample, and the result is tested under the same distribution. It is required to be able to learn under any distribution, but, since one may be unlucky during the sampling processes, exactitude is not required: a small (ϵ) error is permitted, and one should not do worse than this error rate in more than very few (δ) cases. Number and size of examples and run-time should be polynomial in* $1/\epsilon$, $1/\delta$, *and the size of the target* [de la Higuera, 2005, p. 7].

In this PAC learning model, more negative results have been proved than positive results (for grammatical inference). Even for the case of DFA, most results are negative. The requirement that the learning algorithm must learn under any arbitrary (but fixed) probability distribution seems too strong. Some positive results in this model can be found in [Ron et al., 1994].

We consider that, taking into account that exact identification is "too hard" in natural language learning, approximately learning could be also a good approach to deal children's language acquisition. However, we guess that the PAC model is perhaps too restrictive to study natural language learning.

To consider that the examples have the same distribution throughout the process, is too restrictive for practical situations.

> *It is conceivable that most practical learning scenarios do not place such stringent restrictions on the learnability of concept classes. On the contrary, practical learning scenarios feature helpful learning environments (for example, a knowledgeable teacher might guide the learner by answering queries or by carefully selecting training examples that would enable the learner to learn quickly and efficiently).* [Parekh and Honavar, 2000, p. 208].

As some authors state:

> *Even minor adaptations of this model potentially trivialize the learning task (see e.g. Parekh and Honavar (2000)), thus the paradigm is not flexible enough to cover a wider range of learning situations.* [Costa-Florêncio, 2003, p. 4].

Although we do not reject the model of PAC learning, this does not constitute a subject of our research.

Chapter 5

Algorithms in Grammatical Inference

In the sequel we present an overview of the learnability results that have been obtained in the framework of Grammatical Inference using only positive data and queries.

We are going to concentrate on this kind of results because:

a) – Most grammatical inference systems are based on only positive data.

– The availability of positive data in children's language acquisition is trivially accepted.

b) – Learning from queries is another important learning paradigm that has been exhaustively studied in the field of Grammatical Inference.

– Queries provide additional information to the learning process and could be considered an important tool available to the child.

5.1. Learning from only positive data

Learning from only positive data is an attractive paradigm of learning. As C. de la Higuera pointed out,

> *(...) the problem of learning from positive data alone is probably the most practical of grammatical inference settings* [de la Higuera, 2005, p. 12].

The importance of learning from positive data in natural learning is well recognized. As we have seen, Gold pointed out in [Gold, 1967] that the acquisition of a first language is based on sentences which are syntactically correct. Nevertheless, Gold [Gold, 1967] proves that many classes of formal languages, such as regular and context-free, cannot be learned solely from positive examples. Therefore, as natural languages are certainly more complex than regular languages, the non learnability of regular raises a problem.

Despite this result, the most part of grammatical inference systems are based on only positive data, mainly due to in the most part of applications the available data is positive. In the 70's and 80's we can find many papers dealing with this subject.

The main focus of research in the field of Grammatical Inference has been set on learning REG grammars or DFA. The problem of identifying DFA from examples have been studied quite extensively (see, e.g., [Angluin and Smith, 1983, Pitt, 1989]).

Since the class of regular languages are not learnable from positive data in Gold's model, we need to restrict this class somehow to subclasses to establish identifiability results from positive data.

Angluin [Angluin, 1982] has introduced subclasses of regular languages, called *k-reversible languages* (where $k \geq 0$) and shown these classes are identifiable in the limit from positive data, requiring a polynomial time for up-

dating conjectures. She also gave a characterization of identifiability from positive data [Angluin, 1980].

Another interesting class of regular languages which can be identified in the limit from positive data is the class of *strictly deterministic regular languages* investigated by T. Yokomori [Yokomori, 1995].

Further, another subclass of regular languages called *strictly k-testable languages* is studied and shown to be identifiable in the limit from positive data, together with its application to pattern recognition [García and Vidal, 1990] and DNA sequence analysis [Yokomori and Kobayashi, 1998].

Denis et al. define subclasses of languages that can be identified through non-deterministic finite automata by only positive data [Denis et al., 2002]. Fernau proposed a generalization of these results in [Fernau, 2000].

Other interesting results on identification from positive data are Angluin's *pattern languages*[Angluin, 1979] and Oncina et al.'s *subsequential transducers* [Oncina et al., 1993].

Taking into account that context-free grammars are more expressive than regular grammars, the question of whether there are analogous results for context-free grammars become interesting and important.

The class of context-free languages are not learnable from positive data in Gold's model and, therefore, if we want to study this class in the limit from only positive data we need to restrict it to subclasses of context-free grammars.

Yokomori [Yokomori, 1991] has considered a subclass of simple deterministic grammars, called *very simple grammars*, and studied the problem of identifying the subclass in the limit from positive data. The class of very simple languages forms a proper subclass of simple deterministic languages and is incomparable to the class of regular languages. This class of languages is also known as the class of left Szilard languages of context-free grammars.

Yokomori has shown that the class of very simple grammars is polynomial time identifiable in the limit from positive data (in the sense described in section 4.1.3 – definition 4.1.4). Therefore, this result has provided the first instance of language class containing non-regular languages that can be identified in the limit from positive data in polynomial time.

Sakakibara has shown that there exist a class of CF grammars called *reversible context-free grammars*, which can be identified in the limit from positive presentations of structured strings [Sakakibara, 1992]. Makinen investigated a subclass of reversible context-free grammars called *type invertible grammars*, that can be identified from positive presentation of structured strings in time linear in the size of the inputs [Makinen, 1992].

Therefore, research on learning from only positive data has been focus on regular or context-free languages. Surveys on the subject can be found in [Sakakibara, 1997], [de la Higuera, 2005].

5.1.1. Learning context-sensitive languages

Since more generative capacity than context-free grammar is required to describe natural languages, a question arises: what about learning context-sensitive languages using positive data? If we take a look on the literature of the field, there is not too many references about learning context-sensitive languages. Due to the complexity of this class, the results are scarce. However, there is an important result that we would like to present here.

Arikawa et al. [Arikawa et al., 1989] developed a framework for language learning called *elementary formal system* (or EFS), which was originally introduced by Smullyan [Smullyan, 1961]. This framework was refined, among others, in [Shinohara, 1990]. In a word, EFS is a logic program over Σ^+. Arikawa showed that EFS can be used as a natural device to define formal languages. We will present here some of the more impressive results without

going into too much detail (we will based on ([Shinohara, 1994]).

An EFS is a finite set of definite clauses. For example,

$$\Gamma = \{p(a,b,c) \leftarrow; p(ax,by,cz) \leftarrow p(x,y,z); q(x,y,z) \leftarrow p(x,y,z)\}$$

is an EFS, where:

- a, b, c are constant symbols taken from an alphabet Σ.
- x, y, z are variables.
- p, q are predicate symbols.

Finite strings consisting of constant symbols and variables are also called patterns. Two inference rules are used in EFS: one is an application of a substitution for variables by nonempty words, and the other is modus ponens. The language defined by Γ and q is:

$$L(\Gamma, q) = \{w \in \Sigma^+ | q(w) \text{ is provable from } \Gamma\} = \{a^n b^n c^n | n \geq 1\}.$$

Let the length of an atom be the sum of lengths of patterns in it. A definite clause $A \leftarrow B_1, ..., B_n$ is called *length-bounded* if the total length of $B_1\theta, ..., B_n\theta$ does not exceed the length of $A\theta$ for any substitution θ.

The EFS Γ in the above example is length-bounded. The class of languages definable by length-bounded EFS coincides with the class of context-sensitive languages.

Shinohara showed in [Shinohara, 1990] that the class of languages definable by length-bounded EFS consisting of at most n clauses is inferible from positive data for any n.

In [Shinohara, 1994] Shinohara tried to extend these results. First, he showed a sufficient condition for inferability from positive data in a more general setting.

A *concept defining framework* is a triple (U, E, M) of a universe U of objects, a universe E of expressions, and a semantic mapping M from finite sets of expressions to concepts. A finite set of expressions is called a *formal system.*

M is *monotonic* if for any formal systems Γ' and Γ, $\Gamma' \subseteq \Gamma$ implies $M(\Gamma') \subseteq M(\Gamma)$.

A formal system $\Gamma \subseteq E$ is said to be *reduced with respect to* a finite set $X \subseteq U$, if $X \subseteq M(\Gamma)$ but $X \not\subseteq M(\Gamma')$ for any $\Gamma' \subseteq \Gamma$.

A concept defining framework (U, E, M) is said to have *bounded finite thickness*, if M is monotonic, and for any finite set $X \subseteq U$ and any $n \geq 0$

$$\{R \subseteq U | R = M(\Gamma), \Gamma \subseteq E, card(\Gamma) \leq n, \Gamma \text{ is reduced with respect to } X\}$$

consists of finitely many concepts.

In any concept defining framework (U, E, M) that has bounded finite thickness, the class of concepts defined by formal systems consisting of at most n expressions is shown to be inferable from positive data for any n.

This general result is applied to several concept defining frameworks. As corollaries:

Corollary 5.1.1 *For any $n \geq 0$, the class of languages definable by length-bounded EFS consisting in at most n clauses is inferable from positive data.*

Corollary 5.1.2 *For any $n \geq 0$, the class of languages defined by context-sensitive grammar consisting of at most n productions is inferable from positive data.*

Corollary 5.1.3 *For any $n \geq 1$, the class of minimal models of linear Prolog programs consisting of at most n clauses is inferable from positive data.*

Since context-sensitive grammars are linguistically interesting, Corollary 5.1.2. will be relevant to our work. One of our results presented in Chapter 8 is based on Shinohara's result.

5.2. Learning via queries

As Balcázar et al. stated:

> *One of the main positive results in the computational learning framework is that deterministic finite automata (dfa) can be learned from membership and equivalence queries.* [Balcázar et al., 1997, p. 1].

The learnability of DFA has been successfully studied in the context of query learning. As we have pointed out before, Angluin proved learnability of DFA via queries. Later, [Rivest and E.Schapire, 1993], [Hellerstein et al., 1995] or [Balcázar et al., 1997] developed more efficient versions of the same algorithm trying to increase the parallelism level, to reduce the number of EQs, etc.

A question arises from Angluin's result: can we extend the polynomial time MAT learnability beyond the class of DFA? Following Angluin's seminal work on learning DFA, research on MAT learnability has been extended beyond the class of DFA.

Yokomori [Yokomori, 1994] has studied efficient identification of non-deterministic finite automata from MQs and EQs. Also in [Yokomori, 1996], Yokomori learns 2-tape automata from both queries and counterexamples.

Angluin [Angluin, 1990] has shown that the class of CF grammars cannot identified in polynomial time using only EQs. Moreover, Angluin and Kharitonov [Angluin and Kharitonov, 1991] have shown that the problem of identifying the class of CF grammars from MQs and EQs is computationally

as hard as the cryptographic problems (for which there is currently no known polynomial-time algorithm). On the other hand, if structural information is available, Sakakibara [Sakakibara, 1990] proves that the class of context-free grammars is identifiable in the Angluin's model.

In [Angluin, 1990], Angluin showed that the presence of the approximate fingerprint property is a sufficient condition for languages to be learnable by EQs. The condition is proved necessary by Gavaldà in [Gavaldà, 1993].

Ishizaka has investigated a subclass of CF grammars called *simple deterministic grammars*, and showed that it can be identified in polynomial time using MQs and EQs in terms of general CF grammars [Ishizaka, 1990].

Another technique often used to obtain positive results is to reduce an inference problem to some other inference problem whose result is known. Takada [Takada, 1988] and Sempere and García [Sempere and García, 1994] showed that the learning problem of *even linear languages* can be solved by reducing it to the learning of REG languages. Note that the class of even linear languages properly contains the class of REG languages and is a proper subclass of context-free languages. In [Fernau, 1999] is showed also that simple even linear matrix languages can be inferred in polynomial time using Angluin's model.

5.2.1. The Learning Algorithm L^*

Since it is conjectured that richer classes than DFA cannot be inferred through a polynomial use of MAT [Angluin, 1987], Angluin's algorithm for learning DFA from MAT, known as L^*, become an important reference and one of the most important results in the framework of learning from queries.

In the sequel we briefly review the learning algorithm L^*. This algorithm, its correctness proof, and complexity analysis are described in detail in [Angluin, 1987].

Let L be the unknown regular set and let Σ be the alphabet of L.

5.2.1.1. Observation table

The information is organized into an *observation table* consisting of three parts: a nonempty finite prefix-closed set S of strings, a nonempty finite suffix-closed set E of strings, and a finite function T mapping $((S \cup S\Sigma) \cdot E)$ to $\{0, 1\}$. The observation table will be denoted (S, E, T).

The interpretation of T is that $T(w)$ is 1 if and only if w is a member of the unknown regular set. The observation table initially has $S = E = \{\lambda\}$, and it is augmented as the algorithm runs.

An observation table can be visualized as a two-dimensional array with rows labeled by elements of $S \cup S\Sigma$ and columns labeled by elements of E with the entry for row s and column e equal to $T(s \cdot e)$. If s is an element of $(S \cup S\Sigma)$ then $row(s)$ denotes the finite function from E to $\{0, 1\}$ defined by $f(e) = T(s \cdot e)$.

The algorithm L^* uses the observation table to build a DFA. Rows labeled by the elements of S are the candidates for states of the automaton being constructed, and columns labeled by the elements of E correspond to distinguishing experiments for these states. Rows labeled by elements of $S\Sigma$ are used to construct the transition function.

Closed, consistent observation tables. An observation table is called *closed* if for every t in $(S\Sigma - S)$ there exists an s in S such that $row(t) = row(s)$. An observation table is called *consistent* if for any s_1, s_2 in S such that $row(s_1) = row(s_2)$, we have $row(s_1 \cdot a) = row(s_2 \cdot a), \forall a \in \Sigma$.

If (S, E, C) is a closed, consistent observation table, we define a corresponding automaton $A(S, E, C) = (Q, \Sigma, \delta, q_0, F)$, where Q, q_0, F and δ are defined as follows:

$Q = \{row(s) \mid s \in S\}$

$q_0 = row(\lambda)$
$F = \{row(s) \mid s \in S \text{ and } T(s) = 1\}$
$\delta(row(s), a) = row(s \cdot a)$

See [Angluin, 1987] for details.

5.2.1.2. The Learner L^*

The learner algorithm uses as its main data structure the observation table that we described in the previous subsection. Initially $S = E = \{\lambda\}$. To determine T, L^* asks MQs for λ and each a in Σ. This initial observation table may or may not be closed and consistent.

The main loop of L^* tests the current observation table (S, E, T) in order to see if it is closed and consistent. If (S, E, T) is not closed, then L^* adds a new string to S and updates the table asking MQs for missing elements. If (S, E, T) is not consistent, then L^* adds a new string to E and updates the table using MQs for missing elements.

When the learner's automaton is closed and consistent the learner asks an EQ. The teacher's answers can be "yes" (in which case the algorithm terminates with the output $A(S, E, T)$) or "no"(in which case a counterexample is provided, all its prefixes are added to S and the table is updated using MQs).

For the proof of the *correctness* and *termination* of L^*, see [Angluin, 1987].

The running time of L^* is bounded by a fixed polynomial of two parameters: n (it is the number of states of the minimum DFA accepting L), and m (it is the length of the longest counterexample presented by the teacher). For the details of the proof, see [Angluin, 1987].

The algorithm L^* is discribed in Figure 5.1.

```
Procedure Learning with L*
 1)  Initialize S and E to {λ};
 2)  Ask MQs for λ and each a ∈ Σ;
 3)  Construct the initial observation table (S, E, T);

 4)  Repeat
 5)    While (S, E, T) is not closed or not consistent
 6)      If (S, E, T) is not closed then
 7)        find s in (SΣ − S) such that row(s) ∉ rows(S);
 8)        add s to S; //actually remove first s from SΣ
 9)        extend SΣ accordingly and T to (S ∪ SΣ)E using MQs;
10)      If (S, E, C) is not consistent then
11)        find s1, s2 ∈ S, a ∈ Σ and e ∈ E such that
12)          row(s1) = row(s2), and C(s1 · a · e) ≠ T(s2 · a · e);
13)        add a · e to E;
14)        extend T to (S ∪ SΣ)E using MQs;

15)    Once (S, E, T) is closed and consistent, let A = A(S, E, T).
16)    Make the conjecture A.
17)    If the Teacher replies with a counterexample t, then
18)      add t and all its prefixes to S
19)      and extend T to (S ∪ SΣ)E using MQs;
20)  Until the teacher replies yes to the conjecture A
21)  Halt and output M.
```

Figure 5.1: The Learner L^*

5.2.1.3. Running Example

We explain how the algorithm L^* runs using an example. Let the alphabet $\Sigma = \{0, 1\}$, and a language $L = (0 + 110)^+$. The minimal automaton associated with the mentioned language is showed in Figure 5.2.

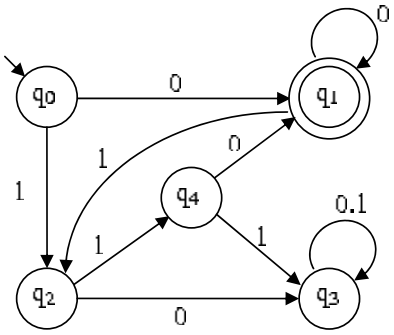


Figure 5.2: Minimal automaton associated to the language $L_1 = (0 + 110)^+$

Initially the learner starts with the following observation table described as Table 5.1.

Table 5.1: $S = \{\lambda\}$, $E = \{\lambda\}$

T_1	λ
λ	0
0	1
1	0

This table is not closed because $row(0)$ does not belong to $rows(S)$. L^* chooses to add the string 0 to S, 00 and 01 to $S\Sigma - S$, and then queries 00 and 01 to construct the observation table T_2 shown in Table 5.2.

Table 5.2: $S = \{\lambda, 0\}$, $E = \{\lambda\}$

T_2	λ
λ	0
0	1
1	0
00	1
01	0

This observation table is closed and consistent, so L^* makes a conjecture of the automaton A_1, shown in Figure 5.3.

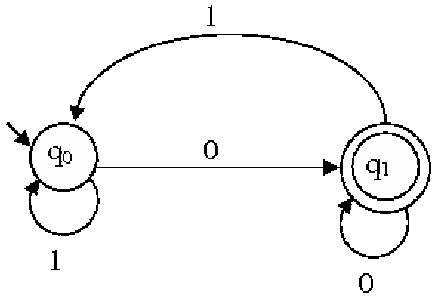


Figure 5.3: Associated automaton: A_1

A_1 is not a correct automaton for L, so the teacher selects a counterexample. In this case we assume that the counterexample 10 is returned (it is not in L but accepted by A_1).

To process the counterexample 10, L^* adds the strings 1 and 10 to S (the string λ is already in S), and queries the strings 11, 100 and 101 to construct the observation table T_3 shown in Table 5.3.

Table 5.3: $S = \{\lambda, 0, 1, 10\}$, $E = \{\lambda\}$

T_3	λ
λ	0
0	1
1	0
10	0
00	1
01	0
11	0
100	0
101	0

This observation table is closed, but not consistent since $row(\lambda) = row(1)$ but $row(0) \neq row(10)$. Thus L^* adds the string 0 to E, and queries the necessary strings to construct the observation table T_4 shown in Table 5.4.

Table 5.4: $S = \{\lambda, 0, 1, 10\}$, $E = \{\lambda, 0\}$

T_4	λ	0
λ	0	1
0	1	1
1	0	0
10	0	0
00	1	1
01	0	0
11	0	1
100	0	0
101	0	0

This observation table is closed, but not consistent since $row(1) = row(10)$ but $row(11) \neq row(101)$. Thus L^* adds the string 10 to E, and queries the necessary strings to construct the observation table T_5 shown in Table 5.5.

Table 5.5: $S = \{\lambda, 0, 1, 10\}$, $E = \{\lambda, 0, 10\}$

T_5	λ	0	10
λ	0	1	0
0	1	1	0
1	0	0	1
10	0	0	0
00	1	1	0
01	0	0	1
11	0	1	0
100	0	0	0
101	0	0	0

This observation table is closed and consistent, so L^* conjectures the automaton A_2 shown in Figure 5.4.

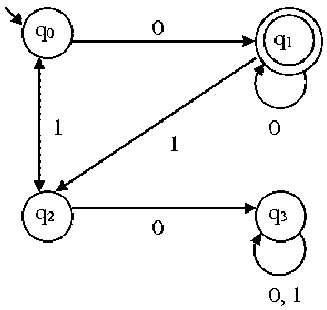


Figure 5.4: Associated automaton: A_2

A_2 is not a correct acceptor for L, so the teacher answers the conjecture with a counterexample. We suppose that the counterexample supplied is 11110, which is not in L but is accepted by A_2.

L^* adds the counterexample and all its prefixes to S and constructs the observation table T_6 shown in Table 5.6.

This table is found to be closed but not consistent, since $row(\lambda) = row(11)$ but $row(1) \neq row(111)$.

Thus L^* adds the string 110 to E and queries the necessary strings to construct the observation table T_7 shown in Table 5.7.

This table is closed and consistent. The automaton conjectured by L^* now corresponds to the correct acceptor for the language L, so the Teacher replies to this conjecture with *yes* and L^* terminates with this automaton as its output.

The total number of queries during this run of L^* is 3 EQs (the last one was successful) and 44 MQs.

Table 5.6:
$S = \{\lambda, 0, 1, 10, 11, 111, 1111, 11110\}$,
$E = \{\lambda, 0, 10\}$

T_6	λ	0	10
λ	0	1	0
0	1	1	0
1	0	0	1
10	0	0	0
11	0	1	0
111	0	0	0
1111	0	0	0
11110	0	0	0
00	1	1	0
01	0	0	1
100	0	0	0
101	0	0	0
110	1	1	0
1110	0	0	0
11111	0	0	0
111100	0	0	0
111101	0	0	0

Table 5.7:
$S = \{\lambda, 0, 1, 10, 11, 111, 1111, 11110\}$,
$E = \{\lambda, 0, 10, 110\}$

T_7	λ	0	10	110
λ	0	1	0	1
0	1	1	0	1
1	0	0	1	0
10	0	0	0	0
11	0	1	0	0
111	0	0	0	0
1111	0	0	0	0
11110	0	0	0	0
00	1	1	0	1
01	0	0	1	0
100	0	0	0	0
101	0	0	0	0
110	1	1	0	1
1110	0	0	0	0
11111	0	0	0	0
111100	0	0	0	0
111101	0	0	0	0

Part III

This dissertation's contributions

Chapter 6

Simple many-dimensional External Contextual grammars (SEC_p)

6.1. Introduction

Gold proves in [Gold, 1978] that superfinite classes are not identifiable in the limit from positive data.

According to the general definition, the EC_p grammar family is superfinite, since the base of G can be any finite set of p-words. We denote by p the dimension and by q the number of contexts.

Theorem 6.1.1 *The class EC_p is superfinite.*

Proof. Let $p = q = 1$. For any finite set S of strings over Σ, consider a EC_p with a base set S and an empty context set. Then, such a EC_p generates a finite language S. A EC_p with a base λ and a context set $\{[(a, \lambda)]\}$ can generate an infinite language a^*. Therefore, the language class is superfinite. □

Corollary 6.1.2 *EC_p is not identifiable in the limit from positive data.*

Hence, we need to put some restrictions to make it possible to learn this class in the limit from only positive data.

6.2. Formal Definition

Definition 6.2.1 *A Simple p-dimensional External Contextual grammar is $G = (\Sigma, B, C)$, where Σ is the alphabet of G, B is a singleton of p-words over Σ called the base of G, and C is a finite set of p-contexts over Σ. C is called the set of contexts of G.*

Therefore, a *Simple many-dimensional External Contextual* grammar is a subfamily of EC_p. The main difference is that the base of a Simple p-dimensional External Contextual grammar is restricted to a single p-word.

The family of all Simple p-dimensional External Contextual languages is denoted by SEC_p.

In what follows, we use the notation SEC_p for referring to families of both languages and grammars, as far as no confusion arises from the context.

6.3. Properties of SEC_p grammars

This section is based on [Kudlek et al., 2002].

Remark 6.3.1 *Note that:*

- $EC = EC_1 = SEC_1$
- $SEC_p \subseteq SEC_r$, *for all* $1 \leq p \leq r$.

Theorem 6.3.2 *For every integer $p \geq 2$, the family SEC_p is a MCS family of languages.*

Proof. 1. $SEC_p \subseteq EC_p$ and EC_p contains semilinear languages only (see [Kudlek et al., 2002]).

2. By *membership problem* is understood the following: given a language $L \subseteq \Sigma^*$ (defined by a certain type of grammar, automaton, etc.) and a word $w \in \Sigma^*$, decide algorithmically whether w is in L or not. Since the membership problem is polynomially decidable for EC_p, it follows that each family SEC_p, $p \geq 1$, is parsable in polynomial time (see [Kudlek et al., 2002], [Victor, 2005]).

3. The following languages are in SEC_p for every $p \geq 2$:
   - *multiple agreements*: $L_1 = \{a^n b^n c^n \mid n \geq 0\}$
   - *crossed agreements*: $L_2 = \{a^n b^m c^n d^m \mid n, m \geq 0\}$
   - *duplication*: $L_3 = \{ww \mid w \in \{a, b\}^*\}$

   It is easy to construct SEC_p grammars for each of these languages:

   (i) $L_1 = \{a^n b^n c^n \mid n \geq 0\}$. It is generated by the SEC_p grammar $G_1 = (\{a, b, c\}, B, C)$, where:

   - B = $\{(\lambda, \lambda)\}$
   - C = { $c_1 = [(a, b), (c, \lambda)]$}

   (ii) $L_2 = \{a^n b^m c^n d^m \mid n, m \geq 0\}$. It is generated by the SEC_p grammar $G_2 = (\{a, b, c, d\}, B, C)$, where:

   - B = $\{(\lambda, \lambda)\}$
   - C = { $c_1 = [(a, \lambda), (c, \lambda)]$, $c_2 = [(\lambda, b), (\lambda, d)]$}

   (iii) $L_3 = \{ww \mid w \in \{a, b\}^*\}$. It is generated by the SEC_p grammar $G_3 = (\{a, b\}, B, C)$, where:

- B = $\{(\lambda, \lambda)\}$
- C = $\{\ c_1 = [(a, \lambda), (a, \lambda)],\ c_2 = [(b, \lambda), (b, \lambda)]\}$

□

Comment 6.3.3 *One can easily prove that the language of k multiple agreements* $L'_1 = \{a_1^n a_2^n ... a_k^n | n \geq 0\}$*, where* $k \geq 3$*, is in* SEC_p *if* $k = 2p$*, and in* SEC_{p+1} *if* $k = 2p + 1$*. In other words,* SEC_p *can count to* $2p$*.*

Proof. Let $k \geq 3$, then we distinguish the following two cases.

Case I) $k = 2p : L'_1 \in SEC_p$

Let $G = (\Sigma, B, C)$, with $B = \{(\lambda_1, \lambda_2, ..., \lambda_p)\}$ and $C = [(a_1, a_2), (a_3, a_4), ..., (a_{2p-1}, a_{2p})]$. Then it is clear that $L(G) = L'_1$ and $L(G) \in SEC_p$.

Case II) $k = 2p + 1 : L'_1 \in SEC_{p+1}$

Let $G = (\Sigma, B, C)$, with $B = \{(\lambda_1, \lambda_2, ..., \lambda_{p+1})\}$ and $C = [(a_1, a_2), (a_3, a_4), ..., (a_{2p-1}, a_{2p}), (a_{2p+1}, \lambda)]$. Then it is clear that $L(G) = L'_1$ and $L(G) \in SEC_{p+1}$.

□

Also, it can be proved that the language of marked duplications $L'_3 = \{wcw | w \in \{a, b\}^*\}$ *is in* SEC_2*.*

Proof. Let $G = (\Sigma, B, C)$, with $B = \{(c, \lambda)\}$ and $C = \{c_1 = [(a, \lambda), (a, \lambda)], c_2 = [(b, \lambda), (b, \lambda)]\}$. Then it is clear that $L(G) = L'_3$ and $L(G) \in SEC_2$. □

Necessary conditions for a language to be a SEC_p language:

Lemma 6.3.4 *Let $L \subseteq \Sigma^*$ be a language in SEC_p, $p \geq 1$. There exist two integers $n \geq 1$ and $k \geq 1$ such that:*

(i) (pumping an arbitrary context) If $w \in L$ such that $|w| > n$, then w has a decomposition $w = x_1u_1y_1v_1x_2u_2y_2v_2 \dots x_pu_py_pv_px_{p+1}$, with $0 < |u_1| + |v_1| + |u_2| + |v_2| + \dots + |u_p| + |v_p| \leq k$, such that, for all $i \geq 0$, the following words are in L:

$$w_i = x_1u_1^iy_1v_1^ix_2u_2^iy_2v_2^i \dots x_pu_p^iy_pv_p^ix_{p+1}$$

(ii) (pumping an innermost context) If $w \in L$ such that $|w| > n$, then w has a decomposition $w = x_1u_1y_1v_1x_2u_2y_2v_2 \dots x_pu_py_pv_px_{p+1}$, with $0 < |u_1| + |v_1| + |u_2| + |v_2| + \dots + |u_p| + |v_p| \leq k$, and $|y_1| + |y_2| + \dots + |y_p| \leq n$, such that, for all $i \geq 0$, the following words are in L:

$$w_i = x_1u_1^iy_1v_1^ix_2u_2^iy_2v_2^i \dots x_pu_p^iy_pv_p^ix_{p+1}$$

(iii) (pumping an outermost context) If $w \in L$ such that $|w| > n$, then w has a decomposition $w = u_1y_1v_1u_2y_2v_2 \dots u_py_pv_p$, with $0 < |u_1| + |v_1| + |u_2| + |v_2| + \dots + |u_p| + |v_p| \leq k$, such that, for all $i \geq 0$, the following words are in L:

$$w_i = u_1^iy_1v_1^iu_2^iy_2v_2^i \dots u_p^iy_pv_p^i$$

(iv) (pumping all occurring contexts) If $w \in L$ such that $|w| > n$, then w has a decomposition $w = u_1y_1v_1u_2y_2v_2 \dots u_py_pv_p$, with $0 < |y_1| + |y_2| + \dots + |y_p| \leq n$, such that, for all $i \geq 0$, the following words are in L:

$$w_i = u_1^iy_1v_1^iu_2^iy_2v_2^i \dots u_p^iy_pv_p^i$$

(v) (interchanging contexts) If $w, w' \in L$ such that $|w| > n$ and $|w'| > n$, then w and w' has decompositions $w = x_1u_1y_1v_1x_2u_2y_2v_2 \dots x_pu_py_pv_px_{p+1}$, and $w' = x_1'u_1'y_1'v_1'x_2'u_2'y_2'v_2' \dots x_p'u_p'y_p'v_p'x_{p+1}'$ with $0 < |u_1| + |v_1| + |u_2| + |v_2| + \dots + |u_p| + |v_p| \leq k$, and with $0 < |u_1'| +$

$|v'_1| + |u'_2| + |v'_2| + \ldots + |u'_p| + |v'_p| \leq k$ *such that also the following two words* z *and* z' *are in* L*:*

$$z = x_1u'_1y_1v'_1x_2u'_2y_2v'_2 \ldots x_pu'_py_pv'_px_{p+1},$$
$$z' = x'_1u_1y'_1v_1x'_2u_2y'_2v_2 \ldots x'_pu_py'_pv_px'_{p+1}.$$

For the proof we refer to [Kudlek et al., 2002].

Theorem 6.3.5 *If* $p, r \geq 1$ *such that* $p < r$*, then* $SEC_p \subset SEC_r$ *and the inclusion is strict.*

Proof. Clearly, $SEC_p \subseteq SEC_r$ (see Remark 6.3.1). It remains to be shown that the inclusion is strict. Consider the language:

$$L = \{a_1^n b_1^n a_2^n b_2^n ... a_r^n b_r^n | n \geq 0\}$$

Let $G = \{\Sigma, B, C\}$ be a SEC_r grammar where:

- $\Sigma = \{a_1, a_2, ..., a_r, b_1, b_2, ..., b_r\}$
- $B = \{(\lambda_1, \lambda_2, ..., \lambda_r)\}$
- $C = \{[(a_1, b_1), (a_2, b_2), ..., (a_r, b_r)]\}$

It is easy to see that $L(G) = L$ and hence $L \in SEC_r$. On the other hand, L is not in SEC_p, since L does not satisfy the condition from Corollary 6.3.4 (i). Therefore, SEC_p is strictly included in SEC_r.

□

By combining Theorem 6.3.2 and Theorem 6.3.5, we obtain the following theorem:

Theorem 6.3.6 *The families* $(SEC_p)_{p \geq 2}$ *define an infinite hierarchy of* $MCSL$*.*

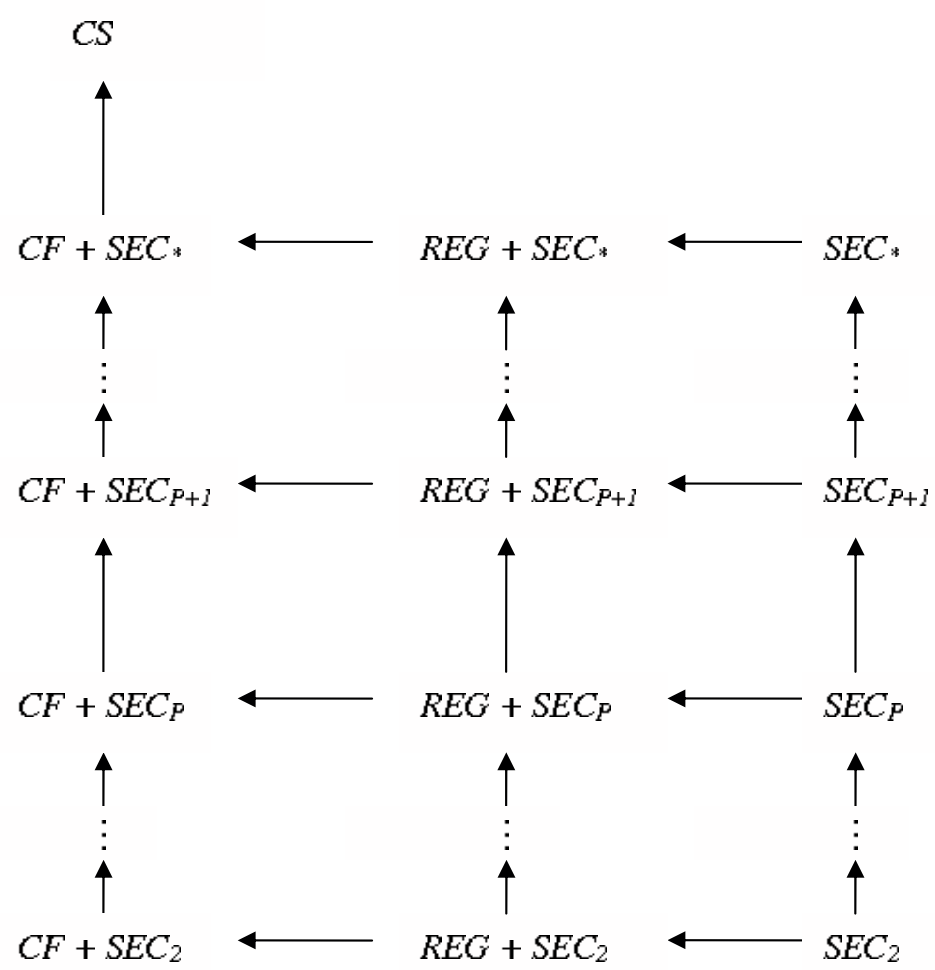


Figure 6.1: Inifinite hierarchy of the families SEC_p

We can see in Figure 6.1 that all families of languages depicted are MCS (except CS). Note that every arrow denotes a strict inclusion.

Now, we investigate the interrelationships between the families SEC_p, $p \geq 1$, and the families of languages in the Chomsky hierarchy.

Theorem 6.3.7 *1. $SEC_p \subset CS$, for every $p \geq 1$.*

2. *Each family SEC_p, $p \geq 2$, is incomparable with the family CF. The family SEC_1 is strictly contained in CF.*
3. *Each family SEC_p, $p \geq 1$, is incomparable with the family REG.*

Proof. 1. Since no deletion is observed in the derivation process of a string in a SEC_p grammar, the first statement follows.

2. From Theorem 6.3.2 it follows that every family SEC_p, $p \geq 2$, contains noncontext-free languages. Consider now the context-free language $L = \{a^n b^n | n \geq 0\}^*$. Assume that L can be generated by a SEC_p grammar $G = (\Sigma, B, C)$. Consider the following word from L:

$$w = a^{i1}b^{i1}a^{i2}b^{i2}...a^{ir}b^{ir},$$

where $p < r$. One can easily see that w does not satisfy Corollary 6.3.4 (iv), and hence L is not in SEC_p, for any $p \geq 2$.

The second part of this statement follows from Remark 3.4.3.

3. Note that each family SEC_p, $p \geq 1$, contains nonregular languages. Now, consider the regular language $L = a^* \cup b^*$. One can verify that L does not satisfy Corollary 6.3.4 (v), i.e. the property of interchanging the contexts, and thus L is not in SEC_p, for any $p \geq 1$.

□

Let SEC_* be the family:

$$SEC_* = \bigcup_{p \geq 1} SEC_p$$

Remark 6.3.8 *One can easily see that SEC_* is also a MCS family of languages. Additionally, SEC_* is strictly contained in the family CS and is incomparable with the families CF and REG.*

Theorem 6.3.9 $\bigcup SEC_{p,r} \subset CS$

Proof. The family of CS languages contains non-SEC languages. Consider for example $L = \{a^{2^n} | n \geq 1\}$. We can easily see that L could not be generated by a SEC grammar. One of the reasons is that SEC languages have the property of bounded length increase (i.e., there is a constant t such that for each $x \in L$, $|x| > t$, there is $y \in L$ with $0 < |x| - |y| \leq t$). Clearly, this is not fulfilled by the chosen language. □

Moreover, the SEC_p grammar has another property with regard to EC_p grammars. We can find some languages showing the proper inclusion:

$$SEC_p \subset EC_p$$

For example, $L = \{a, b, c\}$. It is generated by an EC_p grammar, but never could be generated by a SEC_p grammar because of the restricted features of SEC_p grammars. This demonstrates that SEC_p is not superfinite.

The two relevant closure properties for learning are the decidability of membership and equivalence. The membership problem for SEC, as we mention before, is decidable in polynomial time. However, the decidability of equivalence for SEC is still an open problem.

Figure 6.2 shows the location of the SEC_p family in the Chomsky hierarchy.

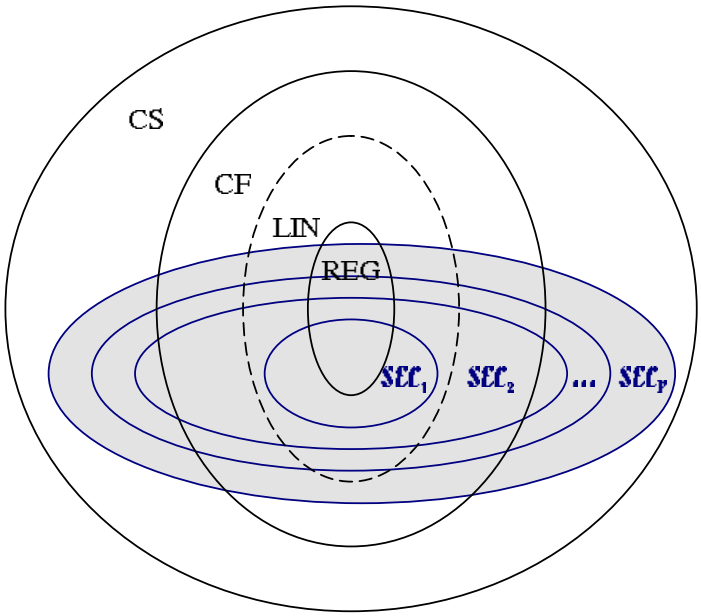


Figure 6.2: The SEC_p family occupies an orthogonal position in the Chomsky hierarchy.

Chapter 7

Correction queries

7.1. What kind of data is available in the process of children's language acquisition?

The availability of positive evidence (sentences in the language to be learned) in children's language acquisition is trivially accepted. As far as we know, no researcher has suggested that children do not receive this kind of data. And it is reasonably, since children are exposed to a large amount of grammatical sentences uttered by adults.

Pinker give us the following definition of positive evidence.

> *The term "positive evidence" refers to the information available to the child about which strings of words are grammatical sentences of the target language.*
> *By "grammatical", incidentally, linguists and psycholinguists mean only those sentences that sound natural in colloquial speech, not necessarily those that would be deemed "proper English" in formal written prose. Thus split infinitives, dangling participles, slang, and so on, are "grammatical" in this sense (and indeed, are as logical, systematic, expressive, and precise as "correct" written English, of-*

> *ten more so; see Pinker, 1994a). Similarly, elliptical utterances, such as when the question Where are you going? is answered with To the store), count as grammatical. Ellipsis is not just random snipping from sentences, but is governed by rules that are part of the grammar of one's language or dialect. For example, the grammar of casual British English allows you to answer the question Will he go? by saying He might do, whereas the grammar of American English doesn't allow it.* [Pinker, 1995, 145]

He considers, as other authors [Wexler and Culicover, 1980], that the only kind of information available to children is grammatical sentences from the language they are learning.

> *That is, when a parent uses a sentence, can the child assume that it is part of the language to be learned, or do parents use so many ungrammatical sentences random fragments, slips of the tongue, hesitations, and false starts that the child would have to take much of it with a grain of salt? Fortunately for the child, the vast majority of the speech they hear during the language-learning years is fluent, complete, and grammatically well-formed:* 99.93%, *according to one estimate (Newport, Gleitman, & Gleitman, 1977). Indeed, this is true of conversation among adults in general (Labov, 1969).*
> *Thus language acquisition is ordinarily driven by a grammatical sample of the target language.* [Pinker, 1995, p. 146].

Positive data does not directly give information to children about which strings are ungrammatical. If children hear a sentence, they can assume that it belongs to the language. But if they do not hear a sentence, it does not imply that the sentence is ungrammatical (perhaps they have not heard the sentence).

Therefore, it is important to know whether children get and need negative data. The absence of negative examples during the process of children's language acquisition has been used to argue in favor of innate knowledge in children. If negative data is not available to the child, any child that overgeneralizes will have no way of knowing that she is wrong ([Gold, 1967], [Pinker, 1979], [Pinker, 1989]). Therefore, they must have some mental mechanism that avoids such situations.

> *The putative absence from children's linguistic environments of negative evidence, about which sentences are ungrammatical in the ambient language, has been used to argue that some form of innate grammatical knowledge is necessary for them to recover from overgeneralization errors. These arguments are facilitated in practice by the assumption that adult speech is to be identified with positive evidence, but that negative evidence, if it existed, would have to be internal to children's minds.* [Marcotte, 2004, p. 1].

For example, G. F. Marcus states that a child would require internal mechanisms to unlearn grammatical errors and that negative evidence is not necessary to the language acquisition task.

> *Several recent studies argue that parents provide noisy feedback, that is, certain discourse patterns that differ in frequency depending on the grammaticality of children's utterances. However, no one has explicitly discussed how children could use noisy feedback, and I show that noisy feedback is unlikely to be necessary for language learning because (a) if noisy feedback exists it is too weak: a child would have to repeat a given sentence verbatim at least 85 times to decide with reasonable certainty that it is ungrammatical; (b) no kind of noisy feedback is provided to all children at all ages for all types of errors; and (c) noisy feedback may be an artifact of defining*

> *parental reply categories relative to the child's utterance. For example, because nearly all parental speech is grammatical, exact repetitions (verbatim repetitions of child utterances) necessarily follow more of children's grammatical utterances than their ungrammatical utterances. There is no evidence that noisy feedback is required for language learning, or even that noisy feedback exists. Thus internal mechanisms are necessary to account for the unlearning of ungrammatical utterances.* [Marcus, 1993, p. 53].

Demonstrating the availability of negative evidence for children would make it unnecessary to postulate many of the innate constraints.

Therefore, whereas the existence of positive information is widely accepted, the availability of negative evidence remains a matter of substantial controversy.

However, first of all we should ask ourselves: what is exactly negative evidence?

As J.P. Marcotte stated in [Marcotte, 2004], there was no consistency in the terms defined in previous works. For example, G.F. Marcus [Marcus, 1993] uses the terms negative evidence and indirect negative evidence to refer to two different kinds of evidence. He gives the following definitions for negative evidence: *information about which strings of words are not grammatical sentences*([Marcus, 1993, p. 53]), *information about which sentences do not belong to [a] language* ([Marcus, 1993, p. 54]), and *a parental behavior that provides information about when sentences are not in the language* ([Marcus, 1993, p. 58]). And G.F. Marcus defines indirect negative evidence as *information about which sentences have not appeared in the input [...] based on mechanisms internal to the child, rather than input external to the child* ([Marcus, 1993, p. 55]).

As one can see, the last definition of negative evidence (which requires

parental behavior as its source) seems to contradict the definition of indirect negative evidence. However, G.F. Marcus intends no contradiction and he draws the distinction between them in the following way: *for the purposes of this paper, I collapse [indirect negative evidence] with positive evidence.* ([Marcus, 1993, p. 55]).

Therefore, all this terminology is, at best, confusing. That is why it is important to clearly define what we understand by negative evidence.

M. Saxton gave, for example, an alternative definition of negative evidence, based on the following idea:

> *(...) the unique discourse structure created in the juxtaposition of child error and adult correct form can reveal to the child the contrast, or conflict between the two forms, and hence provide a basis for rejecting the erroneous form.* [Saxton, 1997, p. 139].

Negative evidence in this sense can be understood as corrective input. He performed experiments with 36 children (0-5 years), in order to compare the effects of this kind of negative evidence with those of positive input, on the acquisition of six novel irregular past tense forms. The results obtained were that children reproduced the correct irregular model more often, and persisted with fewer errors, following negative evidence rather tan positive input.

In [Saxton et al., 1998], M. Saxton concluded that corrective input was more efficient than positive input for learning over time, due to the fact that corrective input more clearly identifies the location of the error for the learner.

> *Performing a time series analysis on observational data, Morgan et al. (1995) conclude that corrective recasts are not related to future improvements in grammaticality. It is argued here, though, that the data sets analysed in this study are inherently ill-suited to the demands of time series analyses. The present study adopts*

> *an experimental approach in order to compare the effects of negative evidence versus positive input on the acquisition of irregular past tense verb forms. Twenty-six children (mean age 3;10) participated in a within-subjects design over a period of five weeks. It was found that improvements in the grammaticality of child speech were considerably greater in cases where negative evidence had been provided. Moreover, children's intuitions concerning the status of irregular and overregularized forms more closely approximated adult intuitions when corrective input was available.* [Saxton et al., 1998, p. 701]

L. Pearl stated that, since negative evidence does not exist naturally (e.g., ADULT:"Don't say *He should have drinked*"), researchers have proposed that children use subtle cues of failed communication (e.g., CHILD:"He should have drinked it", ADULT: "He should have what?"), or that they learn from only positive input (e.g., ADULT:"He should have drunk it", CHILD: "Yeah, he should have") or from only corrective input (CHILD: "He should have drinked it", ADULT: "Right, he should have drunk it"). L. Pearl et al., made an experiment that examined children's ability to use these three input types to learn real English past participles over time. The results obtained demonstrate the viability of learning from both positive and corrective evidence.

> *As in previous research, we found that identifying the location of the error -which corrective evidence provides- was extremely effective for improvements over time for all children and resulted in high overall percentage correct for some children. However, positive evidence (hearing the correct form) also yielded some improvements over time for all children, as well as high overall percentage correct for some children. The subtler cue of failed communication proved to be ineffective for learning.* [Pearl, 2004]

Therefore, beliefs about whether or not children receive negative evidence depends crucially on how one defines that concept.

If we consider that negative evidence is completely incorrect utterances from the adult, or adult replies to a child's ungrammatical utterance like "That's wrong", "No, this is not how you should speak", "I'm about to say something ungrammatical, so pay attention", we can state that this source of evidence is very rare.

> *It is clear that parents do not simply present labeled nonsentences to children in a systematic manner; no parents (or other speakers) say "Here is a sentence, and it is ungrammatical, and here is another one, and this one is ungrammatical, and here is a third, which is grammatical".* [Wexler and Culicover, 1980].

However, there is growing evidence that corrective input for grammatical errors is widely available to children ([Farrar, 1992], [Morgan et al., 1995]).

> *(...) it is often remarked that children get feedback about what they say. For example, parents commonly repeat what they children say with corrections.* [Denis, 2001, p. 38].

7.2. Relevance of corrections in learning processes

We will consider in this work that corrections are available to the child. By *correction* we understand a repetition of children's ungrammatical utterance with correction. For instance, when children overgeneralize (conjugate irregular verbs as regular verbs) the adults correct them by repeating again the same sentence, but instead of the ungrammatical form they say the correct form.

The kind of correction that we are going to consider is exemplified in the sequel:

CHILD: They goed to the cinema.
ADULT: Right, they went to the cinema.

or in Spanish

CHILD: Ya hemos hacido la compra.
ADULT: Sí, ya hemos hecho la compra.

When the adult corrects to the child, it has not to be understood as a negative evidence. The correction give us both, positive and negative information:

- positive: adult returns a correction. This corrected string is information about a string that is grammatically correct.
- negative: if a correction is received, this means that the string uttered by the child was not grammatically correct.

Therefore, corrections should be considered as an especial case of evidence in which positive and negative information will be available to the learner, and not as only negative evidence in the strictest sense.

> *Some researchers (Pinker, Lebeaux, & Frost, 1987; Wexler & Culicover, 1980) have argued that adult repetitions do not explicitly deny the permissibility of children's speech and, therefore, cannot be considered negative evidence. In this strict sense of "negative evidence", it is true that children are rarely told, "No, you said that wrong". However, the information contained in adult repetitious sequences is much greater than that in simple denials.* [Bohannon and Stanowicz, 1988, p. 688].

We are aware of the fact that the concept of correction could be larger than the one considered here. For example, a child might overhear a correction made to another child. But at this point we feel that the kind of

corrections we consider, we capture the essence of this phenomenon (without overly complicating matters).

Although positive examples are an essential part of the language learning process, several questions arise: could corrections play an important role in the process of language learning? Could the child learn some aspects of the language faster by means of corrections? Can corrections help to fix the children's grammar (in order not to repeat again the same errors)?

One can imagine all kinds of situations in which receiving a correction might lead the child to revise a rule ([Nelson et al., 1995]). In fact, the possibility of this kind of evidence is even noted by Chomsky, who pointed out that certain strings of words might be *classed as nonsentences, as a result of correction of the learner's attempts on the part of the linguistic community* [Chomsky, 1965, p. 31].

Several researchers state there is some correlation between the availability of this kind of data and children's language development.

> *Researchers have long sought specific connections between aspects of children's language environment and children's language development (Bates, Beeghley-Smith, Bretherton, & McNew, 1982; Bohannon & Hirsh-Pasek, 1984; Hoff-Ginsburg & Shatz, 1982). Although such relations are difficult to discern because of numerous statistical problems (see Bohannon & Hirsh-Pasek, 1984; Bohannon & Warren-Leubecker, in press), a common finding across many studies is that adult repetition of children correlates with children's language growth (Bates et al., 19982). Children's whose parents repeat after them more often develop language more quickly than those with less repetitious parents. Additionally, laboratory experiments directly manipulating the amount of recasted repetition provided to children have shown enhanced learning of the recasted syntactic forms (Nelson et al., 1984). The mechanisms by*

> *which such repetitions may exert their influence in more natural settings might be through differential provision following children's flawed speech. In this fashion, children are informed about their language errors (i.e., given negative evidence) and simultaneously provided with the correct form to contrast with their immediate error.* [Bohannon and Stanowicz, 1988] p. 688

Some studies [Bohannon and Stanowicz, 1988], [Farrar, 1992] suggest that children may indeed be sensitive to the differential adult behaviors that follow language errors. And moreover, some of them try to prove that natural language are formally learnable without innate knowledge when some form of corrective feedback is available ([Moerk, 1983], [Bohannon and Stanowicz, 1988]).

We will not concern ourselves with this issues here (whether or not innate knowledge is required in natural language learning). Instead, we will focus on the role of corrections in language acquisition.

We do not consider that corrections are the main source of information received during the process of language acquisition; of course, positive data play the main role in that process. However, we have found several linguistic arguments that support the presence of corrections in children's language learning. Therefore, perhaps, we should take into account corrections in studies of language learning, and consider them as a helpful additional information available to children during the learning process. And of course, we do not consider the corrections as a purely negative data, but combination of both positive and negative information.

When we consider how a child learns a language, this linguistic exercise is guided by the adult. There are four possibilities for how the child's language could differ from the adult language. Let's denote by H the child's hypothesis language and by T the target language (language to be acquired).

1. H is disjoint from T . That situation would correspond to the child learning English, for example, who cannot say a single well-formed English sentence. See Figure 7.1

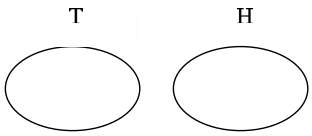


Figure 7.1: H disjoint from T

2. H and T intersect. The child would be able to say some English sentences, e.g., "he went". And she also utter some sentences that are not correct, e.g., "we breaked it". See Figure 7.2

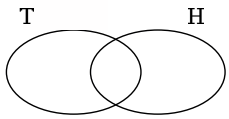


Figure 7.2: H and T intersect

3. H is a subset of T. Here the child would have mastered some of English (which would be part of English), but not all of it. Namely, she would be able to say some grammatical sentences, e.g., "we went", and errors such that "we goed" would not occur. But she might not be able to say "we broke it". See Figure 7.3
4. H is a superset of T. The child could say "we broke it", "we went", "we breaked it" and "we goed". See Figure 7.4

In cases $1, 2, 3$, the child can realize that the hypothesis is not correct by means of the positive evidence received from the adults. This is not possible in case 4. With only positive evidence the child cannot refute a too general

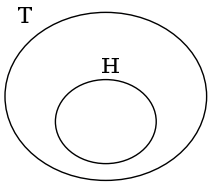


Figure 7.3: H is a subset of T

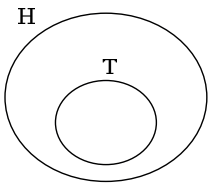


Figure 7.4: H is a superset of T

hypothesis. Therefore, what kind of information would be useful to avoid the situation 4?

We consider that corrections (in the sense that we have described) could play an important role here. The information embedded in a correction could allow children to avoid overgeneralization.

As we have seen, a correction includes positive and negative information. When children overgeneralize, thanks to the negative information from the correction (i.e., if a correction is received then it means that child's utterance was not correct), the child could know that what she said was not correct and she should try not to repeat again the same error.

Therefore, this information could improve learnability. If this kind of information is available to the child, even some aspects of the language could be learned faster (for instance, morphological and phonological aspects).

7.3. Correction queries and Grammatical Inference

Based on all these ideas, we will try to apply the idea of corrections in the studies of Grammatical Inference. We consider that models of Grammatical Inference might benefit from corrections. For instance, the query learning model proposed by Angluin [Angluin, 1987].

As we have seen in Chapter 4, this model provides to the learner an important tool: he is allowed to make queries to the teacher (in a real learning process, the figure of the teacher would correspond to the adult and the figure of the learner would correspond to the child). However, the queries available to the learner are quite unnatural for real learning environments as we have discussed in Chapter 4. Our goal will be to model a more natural way of answering.

Several aspects of the process of children's language acquisition could be represented in the query learning model. For instance, in that stage in which children overgeneralize, we consider that adults correct them and children apply the corrections to their previous knowledge of the language in order not to repeat again the same errors. Our idea is to reflect this process in the query learning model.

It is known since Angluin's results [Angluin, 1987] that DFA can be inferred in polynomial time using a Minimal Adequate Teacher (MAT). Angluin also conjectured [Angluin, 1987] not to be the case for richer classes. This is one of the reasons why we will try to learn DFA from corrections. This will be only a first step of a future direction of research.

Our choice to apply the corrections first to DFA is also motivated by the simplicity of these machines and their adequacy for various applications of natural language processing, although regular languages have limited expressiveness (as we have pointed out in Chapter 3).

> *Regular languages are particularly appealing for natural language processing for two main reason. First, it turns out that most phonological and morphological processes can be straight-forwardly described using the operations that regular languages are closed under, and in particular concatenation. With very few exceptions (such as the interdigitation word-formation processes of Semitic languages or the duplication phenomena of some Asian languages), the morphology of most natural languages is limited to simple concatenation of affixes, with some morpho-phonological alternations, usually on a morpheme boundary. Such phenomena are easy to model with regular languages, and hence are easy to implement with finite-state automata. The other advantage of using regular languages is the inherent efficiency of processing with finite-state automata. Most of the algorithms one would want to apply to finite state automata take time proportional to the length of the word being processed, independently of the size of the automaton. In computational terminology, this is called linear time complexity, and is as good as things can get.* [Wintner, 2002, p. 20].

Our idea is to introduce an extension of MQs called *correction queries* (CQs). In that way, we will try to make a model of the richest information received by a child learning a language: corrections.

7.4. Learning from positive data and correction queries

We have seen evidence that positive data and corrections are available to the child. Why not combine them in a single model? As we have seen, there are models based on text presentation, or informant presentation or queries, etc. We can even find some works ([Jain and Kinber, 2004]) in which learning from positive data and learning from queries is combined into a

computational model, but the kind of queries that they use are the same as the queries introduced by Angluin [Angluin, 1987],[Angluin, 1988].

Since none of the models proposed in Grammatical Inference has considered the combination of both positive data and corrections, we would like to open here a new line of research. We propose the idea of an interactive model between learner-teacher based on the availability of positive data and CQs. We consider that such a model could reflect better the real interaction between child-adult.

As Dale et al. stated, the acquisition is immersed in an interactive context:

> *Language acquisition does not take place in a social vacuum. Instead, children are acquiring their native language while interacting with both people and things in the environment.* [Dale and Christiansen, 2004, p. 262].

Therefore, the novel learning model proposed is inspired by Gold's model and Angluin's model: *learnability in the limit from positive data and CQs*. The development of the algorithm which will implement this model corresponds to a work to be carried out in the future. Here, we would like to introduce the main ideas of the model proposed.

The learning protocol consists in the following elements:

- Hypothesis made by the learner (H_i). Learner will make a hypothesis from the data received. If his hypothesis is not consistent with new data, he will fix it in order to also correctly generate the new information.
- Strings obtained from the teacher (x_i). These strings correspond to positive data (the union of all these strings will be equal to the language that we want to learn).
- Strings that learner submit to the teacher (y_i). The learner has an active role in the learning process, and he will submit to the teacher only strings that belongs to his hypothesis.

- Corrections (c_i). If the string submitted by the learner does not belong to the language, a correction will be returned.

The general idea of the algorithm would be:

- The learner gets information from the teacher (either x_i or c_i) and submits information to him (y_i), but all this is done in an interactive way (the presentation of the teacher and the learner does not occur independently, but in the same time. Recall that our idea is to reflect the interaction between learner-teacher in a real learning process).
- From all the information received, the learner constructs a hypothesis consistent with all this data and after receiving new strings, this hypothesis will be updated. If the new string cannot be generated by his hypothesis, then the hypothesis should be fixed in order to be consistent with the new information received.

In this model, the complexity of the algorithm would take into account the number of CQs and implicit prediction errors made by the learner (we say that the system makes an implicit prediction error if $x_i \notin H_i$ or if y_i is corrected).

An important feature of this algorithm would be that the learning is incremental. In that way, it would also correspond with the incremental learning that takes place in natural language acquisition.

We are aware that many more considerations should be taken into account in the construction of such a model. Of course, some readjustments and a great research effort would be necessary if this model is to be taken into consideration in future work. In Chapter 9, we discuss in more detail some possible lines of research using this combined model.

Chapter 8

Algorithmic aspects

8.1. Learning SEC from only positive data

As we have seen in Chapter 6, SEC grammars could have a chance in the study of syntax of natural language. In the sequel we study their learnability in the limit from positive data.

We have obtained two main results. First, we present a positive learnability result based on Shinohara's results (see chapter 5 for an explanation of Shinohara's results). Second, we present a stronger result based on the property of finite elasticity, which constitutes a sufficient condition for positive data learnability (see chapter 4 for more information about that property).

8.1.1. From Shinohara's results

From a result by Shinohara [Shinohara, 1994], the class of languages generated by CS grammars with a fixed number of rules is learnable from only positive data. Hence, if we can transform a given SEC grammar with dimension p and degree q into an equivalent $LSMG$ (linear simple matrix grammar [Dassow and Păun, 1989]) with dimension p' and degree q' and this into an equivalent CS grammar with a fixed number of rules, we will achieve our

goal.

Definition 8.1.1 *A Linear Simple Matrix Grammar of degree n, $n \geq 1$, is a grammar $G = (N_1, ..., N_p, \Sigma, M, S)$, where:*

- N_i*: nonterminal alphabet.*
- Σ*: terminal alphabet.*
- *S: start symbol.*
- *M: finite set of matrices of the form*
  1. $(S \rightarrow A_1...A_p)$*, for* $A_i \in N_i, 1 \leq i \leq p$*, or*
  2. $(A_1 \rightarrow x_1, A_2 \rightarrow x_2, ..., A_p \rightarrow x_p), for A_i \in N_i, x_i \in \Sigma^*, 1 \leq i \leq p$*, or*
  3. $(A_1 \rightarrow x_1B_1y_1, A_2 \rightarrow x_2B_2y_2, ..., A_p \rightarrow x_pB_py_p)$*, for* $A_i, B_i \in N_i, x_i, y_i \in \Sigma^*, 1 \leq i \leq p$.

We will give the following constructive demonstration to prove that $SEC_{p,q} \subset LSMG_{p',q'} \subset CS$ grammars with a fixed number of rules.

First, we need to define *p*, *q*, *p'* and *q'*.

(i) $SEC_{p,q}$:
- p: dimension (in the same sense as SEC_p),
- q: degree (the number of contexts).

(ii) $LSMG_{p',q'}$:
- p': number of nonterminals in the right hand of the unique rule of the $LSMG$ started by S.
- q': number of matrices.

Let $G = (\Sigma, B, C)$ be a $SEC_{p,q}$ grammar, where

- B = $\{(\gamma_1, ..., \gamma_p)\}$

- C = { $c_1 = [(\alpha_1^1, \beta_1^1), ..., (\alpha_p^1, \beta_p^1)], ..., \ c_q = [(\alpha_1^q, \beta_1^q), ..., (\alpha_p^q, \beta_p^q)]$ }

We can transform this SEC grammar with dimension p and degree q into an equivalent $LSMG$ with dimension p' and degree q'.

G' = $(N_1, ..., N_p, \Sigma, \text{P}, \text{S})$, where

- P = { $\text{S} \longrightarrow A_1...A_p$,
$(A_1 \longrightarrow \gamma_1, ..., A_p \longrightarrow \gamma_p)$,
$(A_1 \longrightarrow \alpha_1^1 A_1 \beta_1^1, ..., A_p \longrightarrow \alpha_p^1 A_p \beta_p^1)$,
(...),
$(A_1 \longrightarrow \alpha_1^q A_1 \beta_1^q, ..., A_p \longrightarrow \alpha_p^q A_p \beta_p^q)$}

for $A_i \in N_i$, $\gamma_i, \alpha_i^j, \beta_i^j \in \Sigma^*$, $1 \leq i \leq p$, $1 \leq j \leq q$

The number of rules of an equivalent CSG will be proportional to p' · q'. Generally, there exists a CSG with the number of rules $\leq k \cdot p' \cdot q'$ (k is a constant).

We now illustrate this method using a grammar as follows. As a simple example, consider a $SEC_{p,q}$ with $p = 2$ and $q = 2$.

Let $G = (\{a, b, c, d\}, B, C)$ be a $SEC_{p,q}$ grammar, where

- B = $\{(ab, cd)\}$
- C = { $c_1 = [(\text{a}, \lambda), (\text{c}, \lambda)], \ c_2 = [(\lambda, \text{b}), (\lambda, \text{d})]$ }

Note that $L(G) = \{a^m b^n c^m d^n | m, n > 0\}$.

We can transform this SECgrammar with dimension p and degree q into an equivalent $LSMG$ with dimension p' and degree q'.

$G' = (\{S, A, A'\}, \{a, b, c, d\}, P, S)$, where

- P = { m_0: S $\longrightarrow$ AA',
m_1: (A $\longrightarrow$ ab, A' $\longrightarrow$ cd),
m_2: (A $\longrightarrow$ aA, A' $\longrightarrow$ cA'),
m_3: (A $\longrightarrow$ Ab, A' $\longrightarrow$ A'd) }.

Now, we can construct a CSG: G" = (V_N, T, P', S), where

$V_N = \{S, A, A', B, R_1, R_2, R_3\}$

P' = {S $\longrightarrow ABA'$

$AB \longrightarrow abR_1$ $R_1b \longrightarrow bR_1$ $R_1c \longrightarrow cR_1$ $bB \longrightarrow Bb$

$AB \longrightarrow aAR_2$ $R_2b \longrightarrow bR_2$ $R_2c \longrightarrow cR_2$ $cB \longrightarrow Bc$

$AB \longrightarrow AbR_3$ $R_3b \longrightarrow bR_3$ $R_3c \longrightarrow cR_3$

$R_1A' \longrightarrow Bcd$ $R_1A' \longrightarrow cd$

$R_2A' \longrightarrow BcA'$ $R_2A' \longrightarrow cA'$

$R_3A' \longrightarrow BA'd$ $R_3A' \longrightarrow A'd\}$

Note that the set of rules presented here may contain some redundancy. However, we gave a priority to the consistency of the manner of constructing corresponding CSGs for general cases.

It is easy to prove that $L(G) = L(G') = L(G'')$. We will do it in two steps:

1. $L(G) \Leftrightarrow L(G')$
2. $L(G') \Leftrightarrow L(G'')$.

Proof 1.

(i) $L(G) \Rightarrow L(G')$.

Let $G = (\Sigma, B, C)$ be a $SEC_{p,q}$ grammar such that $L(G) = L$. Define the $LSMG_{p',q'}$ $G' = (N_1, ..., N_p, \Sigma, M, S)$ such that $A_i \in N_i, 1 \leq i \leq p$. The set M contains the following matrices:

- $(S \rightarrow A_1A_2...A_p)$. The number of nonterminals in the right hand of the unique rule of the $LSMG_{p',q'}$ started by S, is equal to the dimension of the $SEC_{p,q}$. Therefore, p = p'.
- For the p-word $(x_1, x_2, ..., x_p)$, that constitutes the base of $SEC_{p,q}$, M contains the following matrix of rules: $(A_1 \rightarrow x_1, A_2 \rightarrow x_2, ..., A_p \rightarrow$

x_p). There is only one matrix of this kind due to the base of the $SEC_{p,q}$ is a singleton (only has one p-word).

- For each p-context $c = [(u_1, v_1), (u_2, v_2), ..., (u_p, v_p)] \in C$, M contains the matrix of rules: $(A_1 \to u_1 A_1 v_1, A_2 \to u_2 A_2 v_2, ..., A_p \to u_p A_p v_p)$. In this way, when we apply the contexts $c_1, c_2, ..., c_q$, we obtain the same result that when we apply the matrices $m_2, m_3, ..., m_{q+1}$, respectively.

It is easy to see that L(G') = L. By construction, for every $s \in L(G)$ there exist a derivation of s in G'.

(ii) $L(G) \Leftarrow L(G')$.

Let G' be the $LSMG_{p',q'}$, with $L(G') = L$. We define a $SEC_{p,q}$ grammar $G = (\Sigma, B, C)$ such that:

- For the matrix $(A_1 \to x_1, A_2 \to x_2, ..., A_p \to x_p) \in M$, B contains the p-word $(x_1, x_2, ..., x_p)$. Therefore, the elements of B coincide with the elements on the right hand of the matrix $(A_1 \to x_1, A_2 \to x_2, ..., A_p \to x_p)$.
- For each matrix of rules $(A_1 \to u_1 A_1 v_1, A_2 \to u_2 A_2 v_2, ..., A_p \to u_p A_p v_p) \in M$, the set C of p-contexts contains $c = [(u_1, v_1), (u_2, v_2), ..., (u_p, v_p)]$. Therefore, the number of matrices is equal to the number of contexts + 1.

It is easy to verify that L(G) = L. By construction, for every $s \in L(G')$ there exist a derivation of s in G.

Proof 2.

(i) $L(G') \Rightarrow L(G'')$.

Let $G' = (N_1, ..., N_p, \Sigma, M, S)$ be a $LSMG_{p',q'}$ such that L(G') = L. Define the $CSG\ G'' = (N, \Sigma, P, S)$, where: N is a finite set of nonterminal symbols, Σ is a finite set of terminal symbols that is disjoint from N, P is a finite set of production rules and $S \in N$ is the start symbol. The set P contains the following rules:

- $S \rightarrow A_1BA_2A_3...A_p$. The right hand of S coincide with the right hand of the unique rule started by S of the $LSMG_{p',q'}$. We add the nonterminal B when $p \geq 2$, to allow applications of different rules.
- For each matrix of M, P contains the following rules:
  - For the first rule of each matrix, P contains:
    $A_1B \rightarrow x_1R_1$
    $A_1B \rightarrow u_1A_1v_1R_2$
    (...)
    $A_1B \rightarrow u_1A_1v_1R_{q'}$
    q' is the number of matrices. So, there are correspondences between choosing the rule that contains R_1, for example, and applying matrix m_1.
  - For the second rule of each matrix, P contains:
    $R_1A_2 \rightarrow x_2R_1$
    $R_2A_2 \rightarrow u_2A_2v_2R_2$
    (...)
    $R_{q'}A_2 \rightarrow u_2A_2v_2R_{q'}$
    We apply this kind of rules from the second to the $p-1$ rule of each matrix (note that each matrix has p rules).
  - For the p rule of each matrix, P contains:
    $R_1A_p \rightarrow Bx_p \mid x_p$

$R_2A_p \rightarrow Bu_pA_pv_p \mid u_pA_pv_p$
(...)
$R_{q'}A_p \rightarrow Bu_pA_pv_p \mid u_pA_pv_p$.
If we use the rule that contains the nonterminal B, we will go back and we can apply more rules. Otherwise, we will finish the derivation.

- We will need to add some intermediate rules to allow us to make the necessary derivations. These rules don't have any correspondence with the $LSMG_{p',q'}$. With these intermediate rules, we swap R_i to the right until it is adjacent to an A_i, allowing us to apply another rule. Similarly, we move B to the left until it is adjacent to A_1, and then start to apply this process again.

It is easy to see that $L(G'') = L$. By construction, for every $s \in L(G')$ there exist a derivation of s in G''.

(ii) $L(G') \Leftarrow L(G'')$.

Let G" be the CSG, with L(G") = L. We define a $LSMG_{p',q'}$ $G' = (N_1, ..., N_p, \Sigma, M, S)$ such that:

- For the unique rule started by S of the CSG, M contains the same rule without the nonterminal B.
- For all the rules that contain R_i in the CSG (except intermediate rules), where $1 \leq i \leq q'$, M contains a matrix with all this rules, but B, R_i and repeated rules are deleted.

It is easy to verify that L(G') = L. By construction, for every $s \in L(G'')$ there exist a derivation of s in G'.

Hence, there are clear relationships between $SEC_{p,q}$, $LSMG_{p',q'}$ and CSG.

(i) $p' = p$ (in our example, p is equal to 2; therefore, the number of nonterminals in the right hand of the unique rule of the $LSMG$ started by S is 2).

(ii) $q' = q + 1$ (in our example, q is equal to 2; therefore, the number of matrices of $LSMG$ has to be 3).

(iii) *The fixed number of rules of CSG is proportional to p'·q'.* Generally, one can have G'' with $O(p' \cdot q')$ number of rules. Since p' and q' are given, G'' has a bounded number of rules.

From a result by Shinohara [Shinohara, 1994], we can obtain the following theorem:

Theorem 8.1.2 *Given $p' > 0$ and $q' > 0$, the class of languages generated by linear simple matrix grammars with dimension p' and degree q' is learnable from positive data.*

Corollary 8.1.3 *Given $p > 0$ and $q > 0$, the class of languages generated by simple external contextual grammars with dimension p and degree q is learnable from positive data.*

8.1.2. Finite elasticity

Although what we have proved is enough to show that SEC can be learned from only positive data, we have a stronger result. As we will prove next, SEC with any dimension, but with at most q contexts and m bases, has finite elasticity (sufficient condition for learning from positive data).

We will use the notation $\subset$ to mean a *proper* subset relation in the sequel.

By $\mathcal{S}ec(p, q, m)$, we denote the class of languages which can be generated by $SECs$ with dimension less than or equal to p, with at most q contexts,

and with at most m bases. By $\mathcal{S}ec(*,q,m)$, we denote the class of languages defined by

$$\mathcal{S}ec(*,q,m) = \bigcup_{p=1}^{\infty} \mathcal{S}ec(p,q,m).$$

Let w be a string over Σ. A pair (b,C) of a base b and a set C of contexts is said to *minimally generate* w if and only if w is generated by using a base b and contexts in C and there exists no b' and C' such that $b = b'$, $C' \subset C$ and w is generated by using b' and C'. For a string w, by $MinC(w)$, we denote the set of all pairs (b,C) (b:base, C:set of contexts) which minimally generate w. It is clear that the following lemma holds:

Lemma 8.1.4 For any $w \in \Sigma^*$, $MinC(w)$ is finite.

Theorem 8.1.5 *The class $\mathcal{S}ec(*,q,m)$ has finite elasticity, therefore, it is identifiable in the limit from positive data.*

Proof. Assume that the class $\mathcal{S}ec(*,q,m)$ has infinite elasticity.

There exist an infinite sequence $w_0, w_1, w_2, ...$ of strings in Σ^* and an infinite sequence $L_1, L_2, ...$ of languages in $\mathcal{S}ec(*,q,m)$ such that, for any $k \geq 1$, $\{w_0, w_1, ..., w_{k-1}\} \subseteq L_k$ and $w_k \notin L_k$ hold.

For each $i = 1, 2, ...$, let S_i be some SEC generating L_i. Note that each S_i includes some element of $MinC(w_0)$ in its base and context set. Since $MinC(w_0)$ is finite by the above lemma, there exists $C_0 \in MinC(w_0)$ such that infinitely many S_i's include C_0. Let $\sigma = S_{n_1}, S_{n_2}, ...$ be an infinite sequence of such SEC's including C_0. Note that σ is a subsequence of $S_1, S_2, ...$ (That is $n_1, n_2, ...$ is a subsequence of 1,2,3,....)

The string w_{n_1} is not an element of L_{n_1}, therefore, w_{n_1} is not generated by S_{n_1}. But, the infinite subsequence $S_{n_2}, S_{n_3}, S_{n_4}, ...$ should generate w_{n_1}, and therefore, should include some element of $MinC(w_{n_1})$ in its base and context set. Since $MinC(w_{n_1})$ is finite, there exists $C_1 \in MinC(w_{n_1})$ such

that infinitely many S_{n_j}'s include C_1. Note that C_0 does not generate w_{n_1}, therefore, $|C_0| < |C_0 \cup C_1|$ holds.

Repeating the same discussion, we can find an infinite sequence $C_0, C_1, ...$ satisfying the following conditions:

1. $|C_0| < |C_0 \cup C_1| < |C_0 \cup C_1 \cup C_2| < \cdots$ holds,
2. for any q, there exists infinitely many SEC's in $S_1, S_2, ...$ which include $C_0 \cup \cdots \cup C_q$ as its base and context set.

These conditions contradict to the fact that the number of contexts and bases are upper bounded by q and m, respectively. This completes the proof. □

8.2. Learning DFA from corrections

This section is focused on learning DFA within the framework of query learning. We present a new algorithm for learning DFA, based on Angluin's algorithm to identify DFA from MQs and EQs. In order to improve the learning efficiency of DFA we introduce a new type of query called *correction query* (CQ). Therefore, our learning algorithm has access to a teacher able of answering two type of queries: CQs and EQs.

We present theoretical aspects of our learning algorithm, running examples and comparative results. We prove that it is possible to learn DFA from corrections in polynomial time and that the number of queries are reduced.

8.2.1. Introduction

As we have seen in Chapter 4, one of the main results in computational learning theory is that DFA can be learned from MQs and EQs in polynomial time. We consider that this kind of queries are very unnatural in a real learning process and our goal will be to model a more natural way of answering.

Toward this aim, we introduce an innovative type of queries called CQ and we design an algorithm using this kind of queries called *Learning from Corrections Algorithm* (LCA).

Our algorithm is similar to Angluin's algorithm, the main difference consist of the kind of answers the learner receives. Due to the increased complexity of teacher's answers, we obtained a faster learning process; the increased speed is based on the reduced number of queries (CQs and EQs) between the learner and the teacher until the discovering of the language.

8.2.2. Correction queries

CQ is a new type of query which constitutes an extension of the MQ. The difference between them consists in the type of answer that we receive from the teacher. Instead of a yes/no answer, a string called the *correctingString* is returned to the learner.

For a string $\alpha \in \Sigma^*$, we denote the left quotient of L by α by $L_\alpha = \{\beta | \alpha\beta \in L\} = \{\beta | \delta(q_0, \alpha\beta) \in F\}$, where $A = (Q, \Sigma, q_0, \delta, F)$ is a DFA accepting L.

The *correctingString* of α with respect to L is the minimum word (in lex-length order, denoted by $\preceq$) of the set L_α. In the case that $L_\alpha = \emptyset$ we set the *correctingString* of α w.r.t. L to φ, where φ is a symbol which does not belong to the alphabet Σ. With these considerations, for the sake of simplicity in notations, we use C instead of *correctingString*. Hence, C is a function from Σ^* to $\Sigma^* \cup \varphi$. Note that $C(\alpha) = \lambda$ if and only if $\alpha \in L$.

Remark 8.2.1 *If α, β, γ are strings in Σ^* such that $C(\alpha) = \beta \cdot \gamma$ then $C(\alpha \cdot \beta) = \gamma$.*

Proof. Lets take $\alpha, \beta, \gamma \in \Sigma^*$ such that $C(\alpha) = \beta \cdot \gamma$. It follows immediately that $\alpha \cdot \beta \cdot \gamma \in L$ and $\gamma \in L_{\alpha\cdot\beta}$. Assume that $\gamma \neq C(\alpha \cdot \beta)$. Then $\exists\ \gamma' \in \Sigma^*$ such that $\gamma' \prec \gamma$ and $C(\alpha \cdot \beta) = \gamma'$. We deduce that $\alpha \cdot \beta \cdot \gamma' \in L$ which implies $\beta \cdot \gamma' \in L_\alpha$. But $\beta \cdot \gamma = C(\alpha) \Rightarrow \beta \cdot \gamma \preceq \beta \cdot \gamma' \Rightarrow \gamma \preceq \gamma'$, which is a contradiction. Hence, $C(\alpha \cdot \beta) = \gamma$ □

Remark 8.2.2 *For any $\alpha, \beta \in \Sigma^*$, if $L_\alpha = \emptyset$ then $L_{\alpha\cdot\beta} = \emptyset$.*

Proof. Lets take $\alpha, \beta \in \Sigma^*$ such that $L_\alpha = \emptyset$. Suppose by contrary that $L_{\alpha\cdot\beta} \neq \emptyset$ and let γ be an element of $L_{\alpha\cdot\beta}$. Then $(\alpha \cdot \beta) \cdot \gamma \in L$, which implies $\beta \cdot \gamma \in L_\alpha$ and hence $L_\alpha \neq \emptyset$, contradiction with our hypothesis. □

Remark 8.2.3 *For any $\alpha \in \Sigma^*$, the following results hold:*

1. *If $C(\alpha) \neq \varphi$ then $C(\alpha \cdot C(\alpha)) = \lambda$.*
2. *If $C(\alpha) = \varphi$ then $\forall \beta \in \Sigma^*$, $C(\alpha\beta) = \varphi$ but the converse does not hold.*

Proof. Let α be an arbitrary string in Σ^*.

1. If $C(\alpha) = \beta \neq \varphi$ then $\alpha \cdot \beta \in L$ which implies that λ is in $L_{\alpha\cdot\beta}$. Because λ is the smallest possible string we obtain immediately that $C(\alpha\cdot\beta) = \lambda$.
2. If $C(\alpha) = \varphi$ then $L_\alpha = \emptyset$ which implies, using Remark 8.2.1, that $L_{\alpha\cdot\beta} = \emptyset$ for all β in Σ^* and hence $C(\alpha \cdot \beta) = \varphi$. It is easy to see that for $L = (ab)^*$, $C(aba \cdot a) = \varphi$ but $C(aba) \neq \varphi$.

□

8.2.3. Learning from Corrections Algorithm (LCA)

In the sequel, we describe the learning algorithm LCA and we show that it efficiently learns an initially unknown regular set from any adequate teacher. Let L be the unknown regular set and let Σ be the alphabet of L.

Without loss of generality, we assume that the target DFA which is to be learned is a canonical DFA.

8.2.3.1. Observation Tables

The information is organized into an *observation table* consisting of three parts: a nonempty finite prefix-closed set S of strings, a nonempty finite suffix-closed set E of strings, and the restriction of the mapping C to the set $((S \cup S\Sigma) \cdot E)$. The observation table will be denoted (S, E, C).

An observation table can be visualized as a two-dimensional array with rows labeled by elements of $S \cup S\Sigma$ and columns labeled by elements of E with the entry for row s and column e equal to $C(s \cdot e)$. If s is an element of

$(S \cup S\Sigma)$ then $row(s)$ denotes the finite function from E to $\Sigma^* \cup \{\varphi\}$ defined by $row(s)(e) = C(s \cdot e)$. By $rows(S)$ we understand the set $\{row(s) \mid s \in S\}$.

The algorithm LCA uses the observation table to build a DFA. Rows labeled by the elements of S are the candidates for states of the automaton being constructed, and columns labeled by the elements of E correspond to distinguishing experiments for these states. Rows labeled by elements of $S\Sigma$ are used to construct the transition function.

Closed, consistent observation tables. An observation table is called *closed* if for every t in $(S\Sigma - S)$ there exists an s in S such that $row(t) = row(s)$. An observation table is called *consistent* if for any s_1, s_2 in S such that $row(s_1) = row(s_2)$, we have $row(s_1 \cdot a) = row(s_2 \cdot a), \forall a \in \Sigma$.

If (S, E, C) is a closed, consistent observation table, we define a corresponding automaton $A(S, E, C) = (Q, \Sigma, \delta, q_0, F)$, where Q, q_0, F and δ are defined as follows:
$Q = \{row(s) \mid s \in S\}$
$q_0 = row(\lambda)$
$F = \{row(s) \mid s \in S \text{ and } C(s) = \lambda\}$
$\delta(row(s), a) = row(s \cdot a)$

It can be easily shown that *deadSet(A)* $= \{row(s) \mid s \in S \text{ and } C(s) = \varphi\}$ (from Remark 8.2.3 we know that $C(s) = \varphi \Rightarrow C(s \cdot a) = \varphi$, $\forall a \in \Sigma$).

To see that this is a well defined automaton, note that since S is a nonempty prefix-closed set, it must contain λ, so q_0 is defined. Also, since E is a nonempty suffix-closed set, it must contain λ. Thus, if s_1 and s_2 are elements of S such that $row(s_1)$=$row(s_2)$, then $C(s_1) = C(s_1 \cdot \lambda) = row(s_1)(\lambda)$ and $C(s_2) = C(s_2 \cdot \lambda) = row(s_2)(\lambda)$ are defined and equal to each other, hence F is well defined. To see that δ is well defined, suppose s_1 and s_2 are elements of S such that $row(s_1) = row(s_2)$. Then since the observation table $A(S, E, C)$ is consistent, for each a in A, $row(s_1 \cdot a) = row(s_2 \cdot a)$, and since it is closed, this common value is equal to $row(s)$ for some s in S.

Definition 8.2.4 *Assume that (S, E, C) is a closed and consistent observation table. We say that the automaton $A = (Q, \Sigma, \delta, q_0, F)$ is consistent with the function C if for every s in $S \cup S\Sigma$ and e in E, the following statements hold:*

1. $C(s \cdot e) = \varphi \Leftrightarrow \delta(q_0, s \cdot e) \in deadSet(A)$,
2. $C(s{\cdot}e) = t \Leftrightarrow (\delta(q_0, s{\cdot}e{\cdot}t) \in F$ *and* $\forall t' \in \Sigma^* \ (\delta(q_0, s{\cdot}e{\cdot}t') \in F \Rightarrow t \preceq t'))$.

The important fact about the automaton $A(S, E, C)$ is the following.

Theorem 8.2.5 *If (S, E, C) is a closed and consistent observation table, then the automaton $A(S, E, C)$ is consistent with the finite function C. Any other automaton consistent with C but inequivalent to $A(S, E, C)$ must have more states.*

The theorem follows from the following sequence of straightforward lemmas.

Lemma 8.2.6 *Assume that (S, E, C) is a closed and consistent observation table. For the automaton $A(S, E, C)$ and for every s in $S \cup S\Sigma$, $\delta(q_0, s) = row(s)$.*

This lemma can be proved by induction on the length of s. For details see [Angluin, 1987].

Lemma 8.2.7 *If (S, E, C) is a closed and consistent observation table and if $A(S, E, C) = (Q, \Sigma, \delta, q_0, F)$ then for each s in $S \cup S\Sigma$ and all $e \in E$, there exists s' in S such that $\delta(q_0, s \cdot e) = \delta(q_0, s')$ and $C(s \cdot e) = C(s')$.*

Proof. The proof is by induction on the length of e.

- $e = \lambda$. (S, E, C) is a closed table, $s \in S \cup S\Sigma \Rightarrow \exists s' \in S$ such that $row(s') = row(s) \Rightarrow (C(s') = C(s)$ and $\delta(q_0, s') = \delta(q_0, s))$.

- Suppose the result holds for all e in E of length at most k, and let e be an element of E of length $k+1$. Then, since E is suffix-closed, $e = a \cdot e'$ for some a in Σ and e' in E. We take s' in S such that $row(s)$=$row(s')$ $\Rightarrow C(s \cdot e) = C(s' \cdot e)$.

  $\delta(q_0, s \cdot e) = \delta(\delta(q_0, s), a \cdot e')$
  $= \delta(row(s), a \cdot e')$, by the previous lemma,
  $= \delta(row(s'), a \cdot e')$, since $row(s) = row(s')$,
  $= \delta(\delta(row(s'), a), e')$,
  $= \delta(row(s' \cdot a), e')$, by the definition of δ,
  $= \delta(\delta(q_0, s' \cdot a), e')$, by the previous lemma,
  $= \delta(q_0, s' \cdot a \cdot e')$.

  By the induction hypothesis on e', there exist s'' in S such that $\delta(q_0, (s' \cdot a) \cdot e') = \delta(q_0, s'')$ (which implies $\delta(q_0, s \cdot e) = \delta(q_0, s'')$) and $C(s' \cdot a \cdot e') = C(s \cdot e) = C(s'')$.

□

Lemma 8.2.8 *Assume that (S, E, C) is a closed and consistent observation table. Then the automaton $A = A(S, E, C)$ is consistent with the function C.*

Proof. Let s be in $S \cup S\Sigma$ and e in E. In order to prove the lemma, we will use several times Lemma 8.2.7.

1. Let s' in S be such that $\delta(q_0, s \cdot e) = \delta(q_0, s')$ and $C(s \cdot e) = C(s')$. Then, $C(s \cdot e) = \varphi \Leftrightarrow C(s') = \varphi \Leftrightarrow \delta(q_0, s') \in deadSet(A) \Leftrightarrow \delta(q_0, s \cdot e) \in deadSet(A)$.

2. Because s is in $S \cup S\Sigma$ and e in E, there exists s_0 in S such that $\delta(q_0, s \cdot e) = \delta(q_0, s_0)$ and $C(s \cdot e) = C(s_0)$. For any string t in Σ^*, $t = a_1 \cdot a_2 \cdots a_n$ we can inductively find the strings $s_i \in S$, $i = \overline{1, n}$ such that $\delta(q_0, s_{i-1} \cdot a_i) = \delta(q_0, s_i)$ and $C(s_{i-1} \cdot a_i) = C(s_i)$.

We show that $\delta(q_0, s \cdot e \cdot t) = \delta(q_0, s_n)$.
$\delta(q_0, s \cdot e \cdot t) = \delta(q_0, s \cdot e \cdot a_1 \cdot a_2 \cdots a_n) = \delta(\delta(q_0, s \cdot e), a_1 \cdot a_2 \cdots a_n) =$

$\delta(\delta(q_0, s_0), a_1 \cdot a_2 \cdots a_n) = \delta(q_0, s_0 \cdot a_1 \cdot a_2 \cdots a_n) = \delta(\delta(q_0, s_0 \cdot a_1), a_2 \cdots a_n) = \delta(\delta(q_0, s_1), a_2 \cdots a_n)) = \ldots = \delta(\delta(q_0, s_{n-1}), a_n) = \delta(q_0, s_{n-1} \cdot a_n) = \delta(q_0, s_n)$.

We show that $C(s \cdot e) = t$ implies $\delta(q_0, s \cdot e \cdot t) \in F$.
$C(s \cdot e) = t \Rightarrow C(s_0) = a_1 \cdot a_2 \cdots a_n \Rightarrow C(s_0 \cdot a_1) = a_2 \cdots a_n \Rightarrow C(s_1) = a_2 \cdots a_n \Rightarrow C(s_1 \cdot a_2) = a_3 \cdots a_n \Rightarrow C(s_2) = a_3 \cdots a_n \Rightarrow \ldots \Rightarrow C(s_{n-1} \cdot a_n) = \lambda \Rightarrow C(s_n) = \lambda \Rightarrow row(s_n) \in F \Rightarrow \delta(q_0, s_n) \in F \Rightarrow \delta(q_0, s \cdot e \cdot t) \in F$.

Next we will show that if we take $t = a_1 \cdot a_2 \cdots a_n$ to be the smallest string in lex-length order such that $\delta(q_0, s \cdot e \cdot t) \in F$ then $C(s \cdot e) = t$ (notice that the set $\{t \mid \delta(q_0, s \cdot e \cdot t) \in F\}$ is not empty). Because $\delta(q_0, s \cdot e \cdot t) = \delta(q_0, s_n)$ and $\delta(q_0, s \cdot e \cdot t) \in F$ it follows that $C(s_n) = \lambda$ and hence $C(s_{n-1} \cdot a_n) = \lambda$ which implies $a_n \in L_{s_{n-1}}$. Assuming that $C(s_{n-1}) = x \neq a_n$ leads to a contradiction (we found a smaller string $t' = a_1 \cdots a_{n-1} \cdot x$ such that $\delta(q_0, s \cdot e \cdot t') \in F$). Hence $C(s_{n-1}) = a_n$. Reasoning in the same manner we obtain that $C(s_0) = a_1 \cdot a_2 \cdots a_n$ which implies $C(s \cdot e) = t$. This concludes the proof of the lemma. □

Lemma 8.2.9 *Assume that (S, E, C) is a closed, consistent observation table. Suppose the automaton $A(S, E, C)$ has n states. If $A' = (Q', \Sigma, \delta', q'_0, F')$ is any automaton consistent with C that has n or fewer states, then A' is isomorphic with $A(S, E, C)$.*

Proof. We define the relation $\phi \subseteq Q \times Q'$ as follows. For all $s \in S$, $row(s)\ \phi\ q' \Leftrightarrow q' = \delta'(q'_0, s)$.

ϕ is an injection. One can prove this by assuming the contrary, namely that there exist q' in Q' and $s_1, s_2 \in S$ such that $row(s_1)\ \phi\ q'$, $row(s_2)\ \phi\ q'$ and $row(s_1) \neq row(s_2)$. It is clear that $\delta'(q'_0, s_1) = q' = \delta'(q'_0, s_2)$ and that $\exists e \in E$ such that $row(s_1)(e) \neq row(s_2)(e)$, which is equivalent to $C(s_1 \cdot e) \neq C(s_2 \cdot e)$. The two cases that can be distinguished are: one of the values of $C(s_1 \cdot e)$, $C(s_2 \cdot e)$ is φ or both of them are in Σ^*.

Case I) $C(s_1 \cdot e) = \varphi$, $C(s_2 \cdot e) = t \neq \varphi$. Because A' is consistent with C we know from Lemma 8.2.8 that $\delta'(q'_0, s_1 \cdot e) \in deadSet(A')$ and $\delta'(q'_0, s_2 \cdot e \cdot t) \in F'$. $\delta'(q'_0, s_1) = \delta'(q'_0, s_2) \Rightarrow \delta'(q'_0, s_1 \cdot e) = \delta'(q'_0, s_2 \cdot e) \Rightarrow \delta'(q'_0, s_2 \cdot e) \in deadSet(A')$ $\Rightarrow \delta'(q'_0, s_2 \cdot e \cdot t) \in deadSet(A')$, which contradicts $\delta'(q'_0, s_2 \cdot e \cdot t) \in F'$.

Case II) $C(s_1 \cdot e) = t_1$, $C(s_2 \cdot e) = t_2$, $t_1 \neq t_2$ and $t_1, t_2 \neq \varphi$. Because A' is consistent with C we know from Lemma 8.2.8 that $\delta'(q'_0, s_1 \cdot e \cdot t_1) \in F'$, $\delta'(q'_0, s_2 \cdot e \cdot t_2) \in F'$ and t_1, t_2 are the smallest strings with this property. $\delta'(q'_0, s_1) = \delta'(q'_0, s_2) \Rightarrow \delta'(q'_0, s_1 \cdot e \cdot t_1) = \delta'(q'_0, s_2 \cdot e \cdot t_1) \Rightarrow \delta'(q'_0, s_2 \cdot e \cdot t_1) \in F \Rightarrow t_2 \preceq t_1$. In a similar way it can be shown that $t_1 \preceq t_2$, from which we draw the conclusion that $t_1 = t_2$, which leads to a contradiction.

Because ϕ is an injection we deduce that $|Q| \leq |\phi(Q)|$. From the hypothesis we know that $|Q'| \leq |Q|$ and hence $|Q| = |\phi(Q)| = |Q'|$, which makes our relation ϕ a function. It follows immediately that ϕ is bijective since it is injective and has the domain and range finite and of the same cardinality.

ϕ is an automata isomorphism, that is $\phi(q_0) = q'_0$, $\phi(F) = F'$ and for all $s \in S$, $a \in \Sigma$, $\phi(\delta(row(s), a)) = \delta'(\phi(row(s)), a)$. The proof for the first two is quite straight forward. For the last one, we take $s' \in S$ such that $row(s') = row(s \cdot a)$. We have that $\phi(\delta(row(s), a)) = \phi(row(s \cdot a))$ $= \phi(row(s')) = \delta'(q'_0, s')$ and $\delta'(\phi(row(s)), a) = \delta'(\delta'(q'_0, s), a) = \delta'(q'_0, s \cdot a)$. Since $\delta'(q'_0, s')$ and $\delta'(q'_0, s \cdot a)$ have identical row values, namely $row(s')$ and $row(s \cdot a)$, they must be the same state of A'.

This concludes the proof of the lemma.

□

Now, the proof of Theorem 8.2.5 follows, since Lemma 8.2.8 shows that $A(S, E, C)$ is consistent with C, and Lemma 8.2.9 shows that any other automaton consistent with C is either isomorphic to $A(S, E, C)$ or contains at least one more state. Thus, $A(S, E, C)$ is the unique smallest automaton consistent with C.

8.2.3.2. The Learner LCA

Briefly we recall that the learning algorithm consists actually in a "learner" algorithm communicating with a "teacher" algorithm. The teacher knows a *target language* and the learner wants to discover it. In this description, the teacher is represented as a passive entity, while the learner calls the teacher's methods and properties in order to discover the target language.

The learner algorithm uses as its main data structure the observation table that we described in the previous subsection. Initially $S = E = \lambda$. To determine C, LCA asks CQs for λ and each a in Σ. This initial observation table may or may not be closed and consistent.

The main loop of LCA tests the current observation table (S, E, C) in order to see if it is closed and consistent. If (S, E, C) is not closed, then LCA adds a new string to S and updates the table asking CQs for missing elements. If (S, E, C) is not consistent, then LCA adds a new string to E and updates the table using CQs for missing elements.

When the learner's automaton is closed and consistent the learner asks an EQ. The teacher's answers can be "yes" (in which case the algorithm terminates with the output $A(S, E, C)$) or "no"(in which case a counterexample is provided, all its prefixes are added to S and the table is updated using CQs).

Correctness of LCA. If the teacher answers always correctly then if LCA ever terminates its output is clearly the target one. Recall that the teacher's last answer to an EQ is *yes*, the learner's automaton is isomorphic with the target automaton.

Termination of LCA. To see that LCA terminates, notice that the injectivity of function ϕ defined on Lemma 8.2.9 implies that for any closed and consistent observation table (S, E, C), if n denotes the number of different values of $row(s)$ for s in S then any automaton consistent with C must have at least n states. The proof follows the lines of Angluin's paper [Angluin, 1987].

Now suppose that n is the number of states in the minimum DFA $\mathcal{A}_L$ for the unknown regular language L. We show that the number of distinct values of $row(s)$ for s in S increases monotonically up to a limit of n as LCA runs. (Note that at every iteration of the main loop, (S, E, C) is indeed an observation table).

Suppose a string s is added to S because the table is not closed. Then by definition, $row(s)$ is different from $row(s')$ for all s' in S before S is augmented, so that the number of distinct values $row(s)$ is increased by at least one when s is added to S.

Suppose a string is added to E because the table is not consistent. Then the number of distinct values $row(s)$ for s in S must increase because two previously equal values, $row(s_1)$ and $row(s_2)$, are no longer equal after E is augmented. (Note that two values unequal before E is augmented remain unequal after E is augmented).

Thus the total number of operations of either type over the whole run of the algorithm LCA must be at most $n - 1$, since there is initially at least one value of $row(s)$ and there cannot be more than n. Hence LCA always eventually finds a closed, consistent observation table (S, E, C) and makes a conjecture $A(S, E, C)$.

How many distinct conjectures can LCA make? If a conjecture $A(S, E, C)$ is found to be incorrect by the counterexample $strCounterEx$, then since the correct minimum automaton accepting L, $\mathcal{A}_L$ is consistent with C and inequivalent to $A(S, E, C)$ (since they disagree on $strCounterEx$), by Theorem 8.2.5, $\mathcal{A}_L$ must have at least one more state. That is, $A(S, E, C)$ has at most $n - 1$ states. Furthermore, LCA must make a next conjecture, $A(S', E', C')$ which is consistent with C (since C' extends C) and also classifies $strCounterEx$ the same as $\mathcal{A}_L$ (since $strCounterEx$ is in S' and λ is in E, and so is inequivalent to $A(S, E, C)$). Thus, $A(S', E', C')$ must have at least one more state than $A(S, E, C)$.

That shows that LCA can make a sequence of at most $n-1$ incorrect conjectures, since the number of their states must be monotonically increasing, is initially at least one, and may not exceed $n-1$. Since LCA must, as long as it is running, possibly make another conjecture, it must terminate by making a correct conjecture.

Thus, LCA terminates after making at most n conjectures and executing its main loop a total of at most $n-1$ times.

Time analysis of LCA. The total running time of LCA can be bounded by a polynomial in n and m (n is the number of states in the minimum automaton accepting L; m is the maximum length of any counterexample string presented by the teacher). For the details of the proof, see [Angluin, 1987].

The algorithm LCA is described in Figure 8.1.

```
Procedure Learning from Corrections Algorithm
 1)  Initialize S and E with λ;
 2)  Ask CQs for λ and each a ∈ Σ;
 3)  Construct the initial observation table (S, E, C);
 4)  Repeat
 5)    Repeat
 6)      if Not tableClosed(S, E, C) then
 7)        find s in (SΣ − S) such that row(s) ∉ rows(S);
 8)        add s to S;
 9)        extend C to (S ∪ SΣ)E using CQs;
10)      if Not tableConsistent(S, E, C) then
11)        find s1,s2 ∈ S, a ∈ Σ and e ∈ E such that
12)          row(s1) = row(s2), and C(s1 · a · e) ≠ C(s2 · a · e);
13)        add a · e to E;
14)        extend C to (S ∪ SΣ)E using CQs;
15)    until tableClosed and tableConsistent;
16)    construct Learner's conjecture A(S, E, C);
17)    foundAutomaton:= teacher.askEquiv(A(S, E, C));
18)    If Not foundAutomaton then
19)      strCounterEx:=teacher.counterExample;
20)      for each s ∈ Pref(strCounterEx)
21)       if  s ∉ S then
22)          add s to S;
23)          extend C to (S ∪ SΣ)E using CQs;
24)  until foundAutomaton;
```

Figure 8.1: Procedure Learning from Corrections

8.2.4. Running Example

In all our examples we considered that the initial state is always counted as being the state q_0. In order to simplify the automata description we used only the state number as labels for states. We also introduced the *linear transition table* that is a normal transition table with all the lines written on the same row. From a linear transition table we can restore a normal transition table if we know the cardinality of the alphabet. The final states are deduced from the observation table as being the states having λ on the first column of E (the one corresponding to the experiment λ).

We explain how our algorithm runs by tracing the evolution of the observation table for a language over the alphabet $\Sigma = \{0, 1\}$, $L = (0 + 110)^+$. We can see a minimal automaton associated with the mentioned language in Figure 8.2. We observe that the linear transition table for this automaton is $(1, 2, 1, 2, 3, 4, 3, 3, 1, 3)$ and the set of final states is $F = \{1\}$.

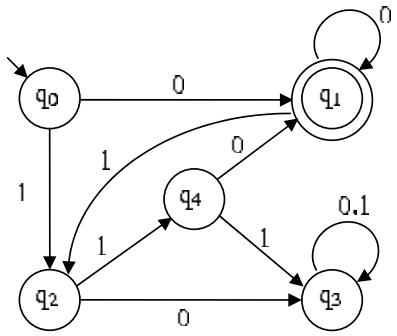


Figure 8.2: Minimal automaton associated to the language $L = (0 + 110)^+$

Initially the learner starts with $S = \{\lambda\}, E = \{\lambda\}$ and the observation table described as Table 8.1.

Table 8.1: $S = \{\lambda\}$, $E = \{\lambda\}$

T_1	λ
λ	$\{0\}$
0	$\{\lambda\}^{(\lambda,\lambda)}$
1	$\{10\}$

We can observe that the information for the string "0" and the experiment "λ" is known from the corresponding query for $row(\lambda)$: because the correction for the string "λ" is the string "0", after an input string "0" we are in a final state, so we need the string "λ" to reach a final state.

The table is not closed because $row(0)$ does not belong to $rows(S)$. We add the string "0" to S and the corresponding rows, $row(0 \cdot 0)$ and $row(0 \cdot 1)$ to $S\Sigma - S$. The algorithm proceeds in a similar manner with $row(1)$ and after two "not closed" steps we get the Table 8.2.

Table 8.2: $S = \{\lambda, 0, 1\}, E = \{\lambda\}$

T_2	λ
λ	$\{0\}$
0	$\{\lambda\}^{(\lambda,\lambda)}$
1	$\{10\}$
00	$\{\lambda\}$
01	$\{10\}$
10	φ
11	$\{0\}^{(1,\lambda)}$

Now we see that the current observation table is not closed since $row(10)$ does not belong to $rows(S)$. Again the algorithm adds it to S and it also adds the corresponding lines, $row(100)$ and $row(101)$, to $S\Sigma - S$. We can see all these operations in Table 8.3.

As an optimization, the learner infers the teacher answers for $row(100)$ and $row(101)$ as being transitions from a dead state.

In this moment, we can see that the observation table is closed and consistent and it follows an EQ. We added the state information for each row and the automaton that the learner discovered until this moment has: the linear transition table $(1, 2, 1, 2, 3, 0, 3, 3)$, the final states set $F = \{1\}$ and the representation given in Figure 8.3.

Table 8.3: $S = \{\lambda, 0, 1, 10\}, E = \{\lambda\}$

T_3	λ	State
λ	$\{0\}$	q_0
0	$\{\lambda\}^{(\lambda,\lambda)}$	$q_1 \in F$
1	$\{10\}$	q_2
10	φ	q_3
00	$\{\lambda\}$	$q_1 \in F$
01	$\{10\}$	q_2
11	$\{0\}^{(1,\lambda)}$	q_0
100	$\varphi^{(10,\lambda)}$	q_3
101	$\varphi^{(10,\lambda)}$	q_3

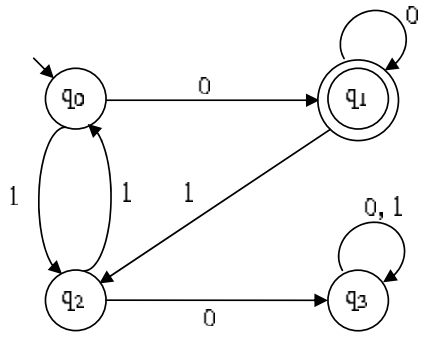


Figure 8.3: Observation Table 8.3 and the associated automaton

The teacher's answer to the EQ is negative and the learner gets as a counterexample the string 11110. Adding the counterexample and all its prefixes to S we get the Table 8.4.

This table is not consistent, since $row(\lambda)$ equals $row(11)$ but $C(\lambda \cdot 1 \cdot \lambda) \neq C(11 \cdot 1 \cdot \lambda)$. In this moment we have to add $1 \cdot \lambda$ to E (Table 8.5).

The automaton of the learner now corresponds to the teacher's automaton, so the answer to the EQ is positive. We notice that during the whole algorithm's execution, the learner asked only two EQs (the last one was successful) and eight CQs. We can see that the number of queries has been reduced considerably with respect to L^* (see running example presented in Chapter 5).

Table 8.4:
$S = \{\lambda, 0, 1, 10, 11, 111, 1111, 11110\}$,
$E = \{\lambda\}$

T_4	λ
λ	$\{0\}$
0	$\{\lambda\}^{(\lambda,\lambda)}$
1	$\{10\}$
10	φ
11	$\{0\}^{(1,\lambda)}$
111	φ
1111	$\varphi^{(111,\lambda)}$
11110	$\varphi^{(111,\lambda)}$
00	$\{\lambda\}$
01	$\{10\}$
100	$\varphi^{(10,\lambda)}$
101	$\varphi^{(10,\lambda)}$
110	$\{\lambda\}^{(1,\lambda)}$
1110	$\varphi^{(111,\lambda)}$
11111	$\varphi^{(111,\lambda)}$
111100	$\varphi^{(111,\lambda)}$
111101	$\varphi^{(111,\lambda)}$

Table 8.5:
$S = \{\lambda, 0, 1, 10, 11, 111, 1111, 11110\}$,
$E = \{\lambda, 1\}$

T_5	λ	1	State
λ	$\{0\}$	$\{10\}^{(1,\lambda)}$	q_0
0	$\{\lambda\}^{(\lambda,\lambda)}$	$\{10\}^{(01,\lambda)}$	$q_1 \in F$
1	$\{10\}$	$\{0\}^{(1,\lambda)}$	q_2
10	φ	$\varphi^{(10,\lambda)}$	q_3
11	$\{0\}^{(1,\lambda)}$	$\varphi^{(111,\lambda)}$	q_4
111	φ	$\varphi^{(111,\lambda)}$	q_3
1111	$\varphi^{(111,\lambda)}$	$\varphi^{(111,\lambda)}$	q_3
11110	$\varphi^{(111,\lambda)}$	$\varphi^{(111,\lambda)}$	q_3
00	$\{\lambda\}$	$\{10\}$	$q_1 \in F$
01	$\{10\}$	$\{0\}^{(01,\lambda)}$	q_2
100	$\varphi^{(10,\lambda)}$	$\varphi^{(10,\lambda)}$	q_3
101	$\varphi^{(10,\lambda)}$	$\varphi^{(10,\lambda)}$	q_3
110	$\{\lambda\}^{(1,\lambda)}$	$\{10\}$	$q_1 \in F$
1110	$\varphi^{(111,\lambda)}$	$\varphi^{(111,\lambda)}$	q_3
11111	$\varphi^{(111,\lambda)}$	$\varphi^{(111,\lambda)}$	q_3
111100	$\varphi^{(111,\lambda)}$	$\varphi^{(111,\lambda)}$	q_3
111101	$\varphi^{(111,\lambda)}$	$\varphi^{(111,\lambda)}$	q_3

8.2.5. Comparative Results

In this section we present some theoretical and practical results. We prove that there exist classes of languages for which the number of queries needed to learn is lower for our algorithm. Using running tests we also show that in practice our algorithm uses less queries.

8.2.5.1. Theoretical Results

We believe that in most of the cases LCA performs not worst that L^* and that there are several subclasses of regular languages for which our algorithm needs smaller number of queries. In the sequel we present one of such subclasses.

Without loss of generality, the teacher is supposed to return the shortest counterexample.

For our technical proofs we introduce first several new definitions.

Definition 8.2.10 *By $MQ_L(m)$ we denote the number of different strings submitted by the learner L^* to the teacher in order to identify the target language L, where m is the size of the minimal complete DFA accepting L.*

Note that each string is counted only once: even if the algorithm reaches a point where the learner should submit to the teacher a string which was previously submitted we will not count this as another MQ.

Definition 8.2.11 *By $CQ_L(m)$ we denote the number of different strings submitted by the learner LCA to the teacher in order to identify the target language L, where m is the size of the minimal complete DFA accepting L.*

In this case, not only we do not count twice the same string, but the learner LCA can also obtain some implicit answers due to Remarks 8.2.1 and 8.2.3. Therefore, the algorithm does not ask these questions and we do not count them as new CQs.

Theorem 8.2.12 *There exists an infinite class of languages which require a polynomial number of MQs but a linear number of CQs in order to be identified.*

Let us consider $\mathcal{S}_\Sigma$ the class of languages over Σ which contain only one string. The theorem follows from the following two lemmas.

Lemma 8.2.13 *For any fixed alphabet Σ of length $k > 1$ and any language L in $\mathcal{S}_\Sigma$ the number of MQs needed by L^* in order to identify L is polynomial in the size m of the minimal automaton. Moreover, we have the following formula:*

$$MQ_L(m) = 2(k-1)m^2 - (4k-7)m + 2k - 5 \tag{8.2.1}$$

Proof. We give the proof only for the case $|\Sigma| = 2$ (similar reasoning for larger alphabets). When $k = 2$ and n is the length of the unique word of the language L (it is clear that $n = m - 2$), what we have to show is:

$$MQ_L(m) = 2n^2 + 7n + 5 \tag{8.2.2}$$

Suppose the string w to be learned is $w = a_1a_2...a_n$ with a_i in $\Sigma = \{a, b\}$ for all $i \in \{1, 2, ..., n\}$. We consider only the case $n \geq 2$, because the proof for $n = 0$ and $n = 1$ is simple.

The algorithm starts by constructing the observation table represented in Table 8.6. The generated automaton $A(S, E, T)$ is represented in Figure 8.4.

Table 8.6: $S = \{\lambda\}$, $E = \{\lambda\}$

T_6	λ	State
λ	$\underline{0}$	q_0
a	$\underline{0}$	q_0
b	$\underline{0}$	q_0

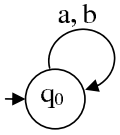


Figure 8.4: Observation Table 8.6 and the associated automaton

This automaton does not accept any string. The counterexample returned (the only possible one in this case) is the string w, so the algorithm L^* adds w and all its prefixes to S and generates the observation Table 8.7.

Table 8.7: $S = \{\lambda, a_1, a_1a_2, ..., a_1a_2...a_n\}$, $E = \{\lambda\}$

T_7	λ
λ	$\underline{0}$
a_1	$\underline{0}$
a_1a_2	$\underline{0}$
$\vdots$	
$a_1a_2...a_n$	$\underline{1}$
$\overline{a}_1$	$\underline{0}$
$a_1\overline{a}_2$	$\underline{0}$
$\vdots$	
$a_1...a_{n-1}\overline{a}_n$	$\underline{0}$
$a_1...a_na$	$\underline{0}$
$a_1...a_nb$	$\underline{0}$

By $\overline{a}_i$ we understand the complement of the letter a_i; the complement of a is b and the complement of b is a (note that the alphabet has only two letters). As a rule, we underline the strings which are not yet counted in the process of counting the MQs.

S has $n+1$ elements and $S\Sigma - S$, $n+2$. Because all the elements of this table are obviously different, it means that up to now the algorithm has asked a total number of $2n+3$ MQs.

This table is not consistent because $row(a_1a_2...a_{n-2}) = row(a_1a_2...a_{n-1})$ but $row(a_1a_2...a_{n-2} \cdot a_n) \neq row(a_1a_2...a_{n-1} \cdot a_n)$ (due to the fact that $T(a_1a_2...a_{n-2} \cdot a_n \cdot \lambda) = 0$ and $T(a_1a_2...a_{n-1} \cdot a_n \cdot \lambda) = 1$).

So we add a_n to E and update the table. The new table is still not consistent, because $row(a_1a_2...a_{n-3}) = row(a_1a_2...a_{n-2})$ but $row(a_1a_2...a_{n-3} \cdot a_{n-1}) \neq row(a_1a_2...a_{n-2} \cdot a_{n-1})$ (the two rows do not coincide in the column a_n). Therefore, we add the experiment $a_{n-1} \cdot a_n$ to E. We continue this procedure until we have $E = \{\lambda, a_n, a_{n-1}a_n, ..., a_2a_3...a_n\}$ and the observation table represented in Table 8.8. This table is closed and consistent and the learner L^* constructs the automaton represented in Figure 8.5.

Table 8.8: $S = \{\lambda, a_1, a_1a_2, ..., a_1a_2...a_n\}$, $E = \{\lambda, a_n, a_{n-1}a_n, ..., a_2a_3...a_n\}$

T_8	λ	a_n	$a_{n-1}a_n$	.....	$a_2a_3...a_n$	State
λ	0	$\underline{0}$	$\underline{0}$		$\underline{0}$	q_0
a_1	0	$\underline{0}$	$\underline{0}$		$\underline{1}$	q_1
a_1a_2	0	$\underline{0}$	$\underline{0}$		$\underline{0}$	q_2
$\vdots$						
$a_1a_2...a_{n-1}$	0	$\underline{1}$	$\underline{0}$		$\underline{0}$	q_{n-1}
$a_1a_2...a_n$	1	$\underline{0}$	$\underline{0}$		$\underline{0}$	q_n
$\overline{a}_1$	0	$\underline{0}$	$\underline{0}$		$\underline{0}$	q_0
$a_1\overline{a}_2$	0	$\underline{0}$	$\underline{0}$		$\underline{0}$	q_0
$\vdots$						
$a_1...a_{n-1}\overline{a}_n$	0	$\underline{0}$	$\underline{0}$		$\underline{0}$	q_0
$a_1...a_na$	0	$\underline{0}$	$\underline{0}$		$\underline{0}$	q_0
$a_1...a_nb$	0	$\underline{0}$	$\underline{0}$		$\underline{0}$	q_0

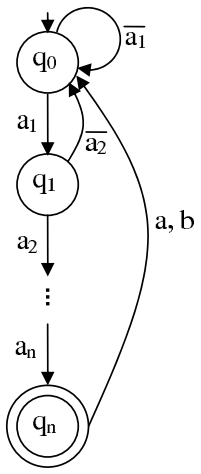


Figure 8.5: Observation Table 8.8 and the associated automaton

How many more distinct MQs is L^* asking in order to obtain this table? (the first column does not count because we have already counted them).

- L^* does not ask a MQ for $\lambda \cdot a_n$, because a_n is either a_1 or $\overline{a}_1$, and for both the answer is known.
- L^* does not ask a MQ for $a_1 \cdot a_n$ because a_n is either a_2 or $\overline{a}_2$, and for both a_1a_2 and $a_1\overline{a}_2$ the answer is known.
- The same for $a_1a_2 \cdot a_n$, $a_1a_2...a_{n-1} \cdot a_n$, $a_1a_2...a_n \cdot a_n$.

Then L^* asks $(n+2)$ MQs for the strings $\overline{a}_1 \cdot a_n$, $a_1\overline{a}_2 \cdot a_n$,..., $a_1a_2...a_nb \cdot a_n$, because they are all distinct and different from any other strings the algorithm has checked so far.

The other columns follow the pattern of the first one: the upper part contains only known strings, and the lower part only unknown ones, which means that there are a total of $(n+2)(n-1)$ MQs. If we add them to what we have counted so far we obtain n^2+3n+4 MQs.

The conjectured automaton is not the target one so the teacher's answer to the EQ is a counterexample. The shortest one is $\overline{a}_1a_1a_2...a_n$, and the algorithm proceeds by adding this counterexample and all its prefixes to S (Table 8.9).

Table 8.9: $S = \{\lambda, a_1, ..., w, \overline{a_1}, \overline{a_1}a_1..., \overline{a_1}w\}, E = \{\lambda, a_n, ..., a_2a_3...a_n\}$

T_9	λ	a_n	$a_{n-1}a_n$	.....	$a_2a_3...a_n$
λ	0	0	0		0
a_1	0	0	0		0
$\vdots$					
$a_1a_2...a_n$	1	0	0		0
$\overline{a}_1$	0	0	0		0
$\overline{a}_1a_1$	$\underline{0}$	$\underline{0}$	$\underline{0}$		$\underline{0}$
$\overline{a}_1a_1a_2$	$\underline{0}$	$\underline{0}$	$\underline{0}$		$\underline{0}$
$\vdots$					
$\overline{a}_1a_1a_2...a_n$	$\underline{0}$	$\underline{0}$	$\underline{0}$		$\underline{0}$
$a_1\overline{a}_2$	0	0	0		0
$a_1a_2\overline{a}_3$	0	0	0		0
$\vdots$					
$a_1...a_{n-1}\overline{a}_n$	0	0	0		0
$a_1...a_{n-1}a_na$	0	0	0		0
$a_1...a_{n-1}a_nb$	0	0	0		0
$\overline{a}_1\overline{a}_1$	$\underline{0}$	$\underline{0}$	$\underline{0}$		$\underline{0}$
$\overline{a}_1a_1\overline{a}_2$	$\underline{0}$	$\underline{0}$	$\underline{0}$		$\underline{0}$
$\vdots$					
$\overline{a}_1a_1...a_{n-1}\overline{a}_n$	$\underline{0}$	$\underline{0}$	$\underline{0}$		$\underline{0}$
$\overline{a}_1a_1...a_{n-1}a_na$	$\underline{0}$	$\underline{0}$	$\underline{0}$		$\underline{0}$
$\overline{a}_1a_1...a_{n-1}a_nb$	$\underline{0}$	$\underline{0}$	$\underline{0}$		$\underline{0}$

This table is not consistent because $row(\lambda) = row(\overline{a_1})$ but $row(\lambda \cdot a_1) \neq row(\overline{a_1} \cdot a_1)$. (they differ in the column corresponding to the experiment $a_2a_3...a_n$). L^* adds the experiment $a_1a_2...a_n$ to E and updates the table (Table 8.10).

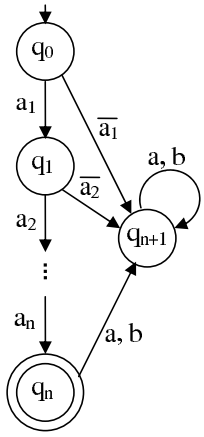


Figure 8.6: Associated automaton to Observation Table 8.10

The conjectured automaton is clearly the target one, so all we have to do now is to count the MQs, more specifically only the underlined ones. We will apply the same strategy, namely counting by columns.

In the column of the experiment λ all the underlined strings are new. The upper part contains n and the lower part $n + 2$ underlined strings.

For the second column one can notice that:

- L^* does not ask a MQ for $\overline{a}_1a_1 \cdot a_n$ because a_n is either a_2 or $\overline{a}_2$, and for both $\overline{a}_1a_1a_2$ and $\overline{a}_1a_1\overline{a}_2$ the answer is known.
- The same for $\overline{a}_1a_1a_2 \cdot a_n$, $\overline{a}_1a_1a_2...a_{n-1} \cdot a_n$, $\overline{a}_1a_1a_2...a_n \cdot a_n$.

Then L^* asks $(n+2)$ MQs for the strings $\overline{a}_1\overline{a}_1 \cdot a_n$, $\overline{a}_1a_1\overline{a}_2 \cdot a_n$,..., $\overline{a}_1a_1a_2...a_nb \cdot a_n$, because they are all distinct and different from any other strings the algorithm has seen so far.

Table 8.10: $S = \{\lambda, a_1, ..., w, \overline{a_1}, \overline{a_1}a_1..., \overline{a_1}w\}, E = \{\lambda, a_n, ..., a_1a_2...a_n\}$

T_{10}	λ	a_n	.....	$a_2a_3...a_n$	$a_1a_2...a_n$	State
λ	0	0		0	$\underline{1}$	q_0
a_1	0	0		1	$\underline{0}$	q_1
$\vdots$						
$a_1a_2...a_n$	1	0		0	$\underline{0}$	q_n
$\overline{a}_1$	0	0		0	$\underline{0}$	q_{n+1}
$\overline{a}_1a_1$	$\underline{0}$	$\underline{0}$		$\underline{0}$	$\underline{0}$	q_{n+1}
$\overline{a}_1a_1a_2$	$\underline{0}$	$\underline{0}$		$\underline{0}$	$\underline{0}$	q_{n+1}
$\vdots$						
$\overline{a}_1a_1a_2...a_n$	$\underline{0}$	$\underline{0}$		$\underline{0}$	$\underline{0}$	q_{n+1}
$a_1\overline{a}_2$	0	0		0	$\underline{0}$	q_{n+1}
$a_1a_2\overline{a}_3$	0	0		0	$\underline{0}$	q_{n+1}
$\vdots$						
$a_1...a_{n-1}\overline{a}_n$	0	0		0	$\underline{0}$	q_{n+1}
$a_1...a_{n-1}a_na$	0	0		0	$\underline{0}$	q_{n+1}
$a_1...a_{n-1}a_nb$	0	0		0	$\underline{0}$	q_{n+1}
$\overline{a}_1\overline{a}_1$	$\underline{0}$	$\underline{0}$		$\underline{0}$	$\underline{0}$	q_{n+1}
$\overline{a}_1a_1\overline{a}_2$	$\underline{0}$	$\underline{0}$		$\underline{0}$	$\underline{0}$	q_{n+1}
$\vdots$						
$\overline{a}_1a_1...a_{n-1}\overline{a}_n$	$\underline{0}$	$\underline{0}$		$\underline{0}$	$\underline{0}$	q_{n+1}
$\overline{a}_1a_1...a_{n-1}a_na$	$\underline{0}$	$\underline{0}$		$\underline{0}$	$\underline{0}$	q_{n+1}
$\overline{a}_1a_1...a_{n-1}a_nb$	$\underline{0}$	$\underline{0}$		$\underline{0}$	$\underline{0}$	q_{n+1}

For the columns corresponding to the experiments $a_{n-1}a_n$, ..., $a_2a_3...a_n$ the situation is the same: the strings from the upper bound of the table are known and the one from the lower part are new.

For the column corresponding to the experiment $a_1a_2...a_n$, we have:

- L^* does not ask a MQ for $a_1\lambda \cdot a_1a_2...a_n$ because the answer is known from the intersection between $row(a_1a_2...a_n)$ and the experiment λ.
- L^* does not ask a MQ for $a_1 \cdot a_1a_2...a_n$ because a_1 is either a_2 or $\overline{a_2}$ and for both $a_1a_2 \cdot a_2...a_n$ and $a_1\overline{a}_2 \cdot a_2...a_n$ the answer is known.
- The same holds for $a_1a_2 \cdot w$, ..., $a_1a_2...a_n \cdot w$.
- L^* does not ask a MQ for $\overline{a_1} \cdot a_1a_2...a_n$ because the answer is known from the intersection between $row(\overline{a}_1a_1)$ and the experiment $a_2...a_n$.
- L^* does not ask a MQ for $\overline{a}_1a_1 \cdot a_1a_2...a_n$ because a_1 is either a_2 or $\overline{a}_2$ and for both $\overline{a}_1a_1a_2 \cdot a_2...a_n$ and $\overline{a}_1a_1\overline{a}_2 \cdot a_2...a_n$ the answer is known.
- The same holds for $\overline{a}_1a_1a_2 \cdot w$, ..., $\overline{a}_1a_1a_2...a_n \cdot w$.

In the lower part of the table, all the strings from the column corresponding to the experiment $a_1a_2...a_n$ are new.

Counting the total number of underlined elements which are new we obtain: $n+n+2$ from the first column, and $n+2$ from the second, third, ... , $n-1$-th column, and $n+1+n+2$ for the last column which makes a total of n^2+4n+1. Adding what we have counted so far (n^2+3n+4) we obtain a total number of $2n^2+7n+5$ MQs.

□

Lemma 8.2.14 *For any fixed alphabet Σ of length $k > 1$ and any language L in $\mathcal{L}_\Sigma$ the number of CQs needed by LCA in order to identify L is linear in the size m of the minimal automaton. Moreover, we have the following formula:*

$$CQ_L(m) = (k-1)(m-1)+2 \tag{8.2.3}$$

Proof. The proof is also given only for the case $|\Sigma| = 2$. Similar arguments can be used for larger alphabets. Suppose the string w to be learned is $w = a_1a_2...a_n$, with $n = m - 2$. Then formula (8.2.3) is equivalent to

$$CQ_L(m) = n + 3 \tag{8.2.4}$$

The algorithm starts by constructing the observation table 8.11.

Table 8.11: $S = \{\lambda\}, E = \{\lambda\}$

T_{11}	λ
λ	$a_1a_2...a_n$
a_1	$a_2...a_n$
$\overline{a}_1$	φ

This observation table is not closed. The algorithm proceeds by adding a_1 and $\overline{a}_1$ to S, and then, a_1a_2, $a_1a_2a_3$, in turn, ... and finally $a_1a_2...a_n$. The obtained observation table (Table 8.12) is closed and consistent. The answer to the EQ is *yes*, which means that $A(S, E, C)$ is the target automaton.

All we have to do now is to count the CQs. Some of the answers are implicit, like those for $C(a_1a_2...a_i)$, for all $i \in \{1, 2, ..., n\}$ (from $C(\lambda) = a_1a_2...a_n$ we know that $C(a_1a_2...a_i) = a_{i+1}...a_n$). Some basic counting shows that L^* is asking a total of $n + 3$ questions in order to identify the target language.

□

Table 8.12: $S = \{\lambda, a_1, a_1a_2, ..., a_1...a_n, \overline{a_1}\}, E = \{\lambda\}$

T_{12}	λ	State
λ	$a_1a_2...a_n$	q_0
a_1	$a_2...a_n$	q_1
$\vdots$		
$a_1...a_{n-1}$	a_n	q_{n-1}
$a_1a_2...a_n$	λ	q_n
$\overline{a}_1$	φ	q_{n+1}
$a_1\overline{a}_2$	φ	q_{n+1}
$a_1a_2\overline{a}_3$	φ	q_{n+1}
$\vdots$		
$a_1...a_{n-1}\overline{a}_n$	φ	q_{n+1}
$a_1\overline{a}_2$	φ	q_{n+1}
$a_1\overline{a}_2$	φ	q_{n+1}

Theorem 8.2.15 *On one letter alphabets, there is an infinite class of languages for which we need a linear number of MQs but a constant number of CQs.*

Suppose $\Sigma = \{a\}$, L is a language from $\mathcal{L}_\Sigma$ and m is the size of the minimal automaton accepting L. The theorem is proved by the following two lemmas.

Lemma 8.2.16 *Given the language L, L^* asks a total number of $3m-3$ MQs in order to learn the language.*

Proof. Let $L = \{a^n\}$, with $n = m-2$. The cases $n = 0$ and $n = 1$ are trivial, so we concentrate only on the case $n \geq 2$.

The algorithm starts by constructing the Table 8.13. As the observation

table is closed and consistent, L^* proceeds by constructing the automaton $A(S, E, C)$ represented in Figure 8.7.

Table 8.13: $S = \{\lambda\}, E = \{\lambda\}$

T_{13}	λ	State
λ	0	q_0
a	0	q_0

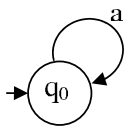


Figure 8.7: Observation Table 8.13 and the associated automaton

We can see that the language accepted by $A(S, E, C)$ is the empty set, so the teacher's answer is a counterexample, namely the string a^n. The algorithm adds this string and all its prefixes to S and obtains the observation Table 8.14.

Table 8.14: $S = \{\lambda, a, ..., a^n\}, E = \{\lambda\}$

T_{14}	λ
λ	0
a	0
$\vdots$	
a^{n-1}	0
a^n	1
a^{n+1}	0

The table is not consistent because $row(a^{n-2}) = row(a^{n-1})$ but $row(a^{n-2}{\cdot}a) \neq row(a^{n-1}{\cdot}a)$ ($T(a^{n-2}{\cdot}a{\cdot}\lambda) = 0$ and $T(a^{n-1}{\cdot}a{\cdot}\lambda) = 1$). L^* proceeds by adding a to E. The table continues to be not consistent and the algorithm adds, one at a time, $a^2, a^3, ..., a^{n-1}$ to E. The observation Table 8.15 is represented bellow.

Table 8.15: $S = \{\lambda, a, ..., a^n\}, E = \{\lambda, a, ..., a^{n-1}\}$

T_{15}	λ	a	a^2	.....	a^{n-1}	State
λ	0	0	0		0	q_0
a	0	0	0		1	q_1
$\vdots$						
a^{n-1}	0	1	0		0	q_{n-1}
a^n	1	0	0		0	q_n
a^{n+1}	0	0	0		0	q_0

This table is closed and consistent so L^* asks an EQ. The counterexample returned by the teacher is a^{2n+1}. L^* adds the string a^{2n+1} and all its prefixes to S and then checks the table for consistency. One can notice that the observation table is not consistent because $row(\lambda) = row(a^{n+1})$ but $row(\lambda \cdot a) \neq row(a^{n+1} \cdot a)$ ($T(\lambda \cdot a \cdot a^{n-1}) = 1$ and $T(a^{n+1} \cdot a \cdot a^{n-1}) = 0$). The algorithm proceeds by adding a^n to E and updating the table. The observation table (Table 8.16) is closed and consistent, and the conjectured automaton, $A(S, E, C)$, is represented in Figure 8.8. The answer to the EQ is *yes* so L^* outputs the automaton $A(S, E, C)$ and halts.

Table 8.16: $S = \{\lambda, a, ..., a^{2n+1}\}, E = \{\lambda, a, ..., a^n\}$

T_{16}	λ	a	a^2	.....	a^{n-1}	a^n	State
λ	0	0	0		0	1	q_0
a	0	0	0		1	0	q_1
$\vdots$							
a^n	1	0	0		0	0	q_n
a^{n+1}	0	0	0		0	0	q_{n+1}
$\vdots$							
a^{2n+1}	0	0	0		0	0	q_{n+1}
a^{2n+2}	0	0	0		0	0	q_{n+1}

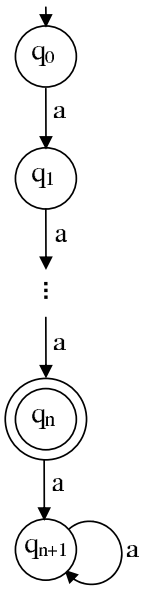


Figure 8.8: Observation Table 8.16 and the associated automaton

In order to count the number of MQs that have been asked it is enough to count how many strings are in the set $\{\lambda, a, a^2, ..., a^{3n+2}\}$. Hence, $MQ(n) = 3n + 3$, which concludes the proof.

□

Lemma 8.2.17 *Given the language L, LCA asks only 2 CQs in order to learn the language.*

Proof. The initial observation table is Table 8.17.

Table 8.17: $S = \{\lambda\}, E = \{\lambda\}$

T_{17}	λ
λ	a^n
a	a^{n-1}

The table is not closed so LCA adds the string a to S. After n non closed tables we obtain Table 8.18.

Table 8.18: $S = \{\lambda, a, ..., a^{n+1}\}, E = \{\lambda\}$

T_{18}	λ	State
λ	a^n	q_0
a	a^{n-1}	q_1
$\vdots$		
a^{n-1}	a	q_{n-1}
a^n	λ	q_n
a^{n+1}	φ	q_{n+1}
a^{n+2}	φ	q_{n+1}

LCA builds the associated automaton and asks an EQ. The answer is of course *yes* so the algorithm outputs the automaton $A(S, E, C)$ and halts.

To identify this language LCA asked the teacher only two CQs, namely $C(\lambda)$ and $C(a^{n+1})$. From $C(\lambda) = a^n$ LCA deduces that $C(a^i) = a^{n-i}$, for all $i \in \{1, 2, ...n\}$ and from $C(a^{n+1}) = \varphi$, LCA deduces that $C(a^{n+2})$ is also φ. □

8.2.5.2. Practical Results

Due to the coding of the states and to the embedded information within the teacher's answers, our practical results reflect the improvement brought by our algorithm.

In order to see the difference between L^* and LCA algorithms, we first tested them on an randomly generated set of 200 DFA, from which 25 are on a one letter alphabet, 100 on two letters alphabet, 50 on three letters alphabet and 25 on four letters alphabet (see Appendix "Test 1 DFA test set). As we will see, generally the results are better with LCA.

In order to generate the DFA test set, we used the public package from http://www.research.att.com/sw/tools/fsm/ having the documentation available in [Mohri et al., 1998].

To generate random automata, we considered a maximum number of states of 20 for our examples. Without loss of generality, the initial state is always the state 0. Then we generate complete transition tables having the destination states generated randomly between 0 and the maximum number of states. For the number of final states, we generated a random number between 0 and the maximum number of states -1. Then, we generated again randomly the final states (without repeating twice the same state).

We generated each automaton in a text file, then we compiled the automaton into an internal format (.fsm) and minimize the compiled automaton

with one of the programs available in the FSM library. After minimizing, we checked the newly found DFA for non-equivalence with the already generated automata. In case of equivalence, we generated new automata. Finally, we checked for completeness and then we loaded the automata in our programs.

Based on this test set, we constructed a comparative table in which one can see the number of queries used by L^* and LCA. We present here a sample of the results in Table 8.19 (see Appendix "Test 1 Comparative results" for all the results).

Table 8.19: A sample from a test set for learning DFA with L^* and LCA algorithms

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
001.Tst	$\{a\}$	3,4,1,1,5,2	2	2	11	1	2
002.Tst	$\{a\}$	0		1	2	1	1
003.Tst	$\{a\}$	0	0	1	2	1	2
004.Tst	$\{a\}$	1,1	0	1	3	1	2
005.Tst	$\{a\}$	2,4,3,1,3	0,2,3	2	8	2	5
006.Tst	$\{a\}$	1,2,1	2	2	5	1	2
007.Tst	$\{a\}$	1,0	0	1	3	1	2
008.Tst	$\{a\}$	1,2,2	1	2	6	1	2
009.Tst	$\{a\}$	1,3,2,4,2	2	2	9	1	2
010.Tst	$\{a\}$	1,6,4,5,3,3,2	0,1,3,4,6	3	14	3	9
...							
050.Tst	$\{a, b\}$	6,1,8,5,4,5,8,3,1,3,3,7,7,0,5,5,8,2	2,3,4,6,7,8	3	59	3	51
051.Tst	$\{a, b\}$	7,1,5,6,2,3,7,2,7,4,7,3,4,5,1,7	3,4	4	71	3	30
052.Tst	$\{a, b\}$	6,6,6,1,5,4,3,6,1,0,3,1,6,2	2	4	71	2	14
053.Tst	$\{a, b\}$	2,0,1,8,1,2,5,6,3,5,8,8,7,5,4,5,7,2	1,2	5	159	3	36
054.Tst	$\{a, b\}$	5,2,4,2,1,1,1,1,3,4,6,3,0,0	2,4	3	49	2	16
055.Tst	$\{a, b\}$	6,2,7,6,7,3,0,0,7,1,4,4,8,7,3,0,2,5	0,1,4,6,7	4	76	4	46
056.Tst	$\{a, b\}$	7,5,2,1,0,1,7,2,0,3,6,4,7,6,1,6	4,7	5	90	3	32
057.Tst	$\{a, b\}$	1,1,0,1	1	1	5	1	4
058.Tst	$\{a, b\}$	8,4,5,3,0,2,6,6,3,5,2,7,0,1,8,0,2,7	0,5	6	161	4	66
059.Tst	$\{a, b\}$	6,0,0,7,0,1,5,4,5,3,3,2,1,3,6,2	4,5	4	89	3	29
060.Tst	$\{a, b\}$	5,2,4,1,0,4,6,7,2,3,7,5,1,2,2,7	3,5,7	4	71	2	16
...							
130.Tst	$\{a, b, c\}$	5,5,4,0,1,1,5,1,1,2,0,1,3,1,3,5,2,3	0,2,4,5	4	67	4	57
131.Tst	$\{a, b, c\}$	0,1,5,2,0,5,5,5,5,1,5,2,0,3,5,5,4,0	0,2,5	3	85	2	42
132.Tst	$\{a, b, c\}$	1,2,0,5,0,1,0,0,1,1,5,1,3,4,2,4,5,2	0,3	5	115	2	28
133.Tst	$\{a, b, c\}$	0,3,4,3,1,2,3,0,5,4,0,1,5,0,2,2,0,4	3,4,5	3	67	2	46
134.Tst	$\{a, b, c\}$	4,3,4,0,0,3,5,1,0,3,5,5,0,4,2,2,5,4	0,2,3	4	76	4	64
135.Tst	$\{a, b, c\}$	5,0,3,4,0,2,4,3,4,4,2,1,5,1,5,0,1,0	4	4	76	3	55
136.Tst	$\{a, b, c\}$	3,2,1,0,5,3,5,4,5,4,1,4,4,1,3,3,2,2	2,3	3	76	2	33
137.Tst	$\{a, b, c\}$	3,4,1,2,5,5,1,2,4,3,2,5,3,4,2,4,3,5	0,2,4	4	104	3	60
138.Tst	$\{a, b, c\}$	5,2,3,3,4,3,1,4,5,3,3,0,0,3,0,3,5,4	0,1,2	3	67	2	25
139.Tst	$\{a, b, c\}$	5,5,0,3,0,5,4,2,5,0,3,4,5,4,5,2,1,3	3,5	3	76	3	60
140.Tst	$\{a, b, c\}$	4,1,2,3,4,4,0,5,4,4,4,2,4,3,4,2,0,2	2,5	3	76	3	55
...							
190.Tst	$\{a, b, c, d\}$	0,4,3,3,4,2,4,3,1,1,4,3,2,0,4,4,2,3,0,4	0,1	3	63	2	41
191.Tst	$\{a, b, c, d\}$	3,0,3,3,0,2,3,0,0,1,3,2,0,2,2,4,2,4,2,4	3	4	121	2	46
192.Tst	$\{a, b, c, d\}$	0,4,0,2,1,2,0,2,1,0,4,4,0,3,1,3,3,0,3,1	1,2,3	3	63	3	65
193.Tst	$\{a, b, c, d\}$	3,2,4,3,2,4,4,1,4,4,4,3,3,3,1,0,4,3,0,3	1,2	4	108	2	41
194.Tst	$\{a, b, c, d\}$	4,2,4,1,2,1,4,4,4,0,3,3,4,3,1,4,1,0,3,4	3	4	108	3	58
195.Tst	$\{a, b, c, d\}$	3,0,0,4,2,1,0,1,4,4,2,0,4,1,3,3,1,4,4,4	0,2,4	4	121	4	109
196.Tst	$\{a, b, c, d\}$	2,2,3,4,3,2,4,1,0,0,4,3,0,3,3,3,4,3,1,4	4	4	121	2	39
197.Tst	$\{a, b, c, d\}$	3,0,0,2,3,4,4,2,3,0,0,4,2,4,2,1,2,2,0,1	1,3,4	2	53	2	51
198.Tst	$\{a, b, c, d\}$	2,2,0,1,0,1,1,0,2,1,2,0	0	2	30	1	11
199.Tst	$\{a, b, c, d\}$	0,1,0,2,2,0,3,0,2,1,1,1,2,3,3,3	3	3	63	2	38
200.Tst	$\{a, b, c, d\}$	2,1,3,0,0,1,1,2,1,2,2,2,1,2,0,2	2	3	63	1	14

In order to visualize the difference between the number of queries used, we generated a graphic for the results obtained with two letters alphabet in a second test. This second test is based on a total of 209 automata: 11 automata for each number of states starting from 2 up to 20 states (see Appendix "Test 2. DFA test set").

We considered average values for automata with the same number of states (see Table 8.20), and we represent the compared results between L^* and LCA for EQs (Figure 8.9) and for MQs and CQs respectively (Figure 8.10). See Appendix "Test 2. Comparative results" for comparative table with all the results.

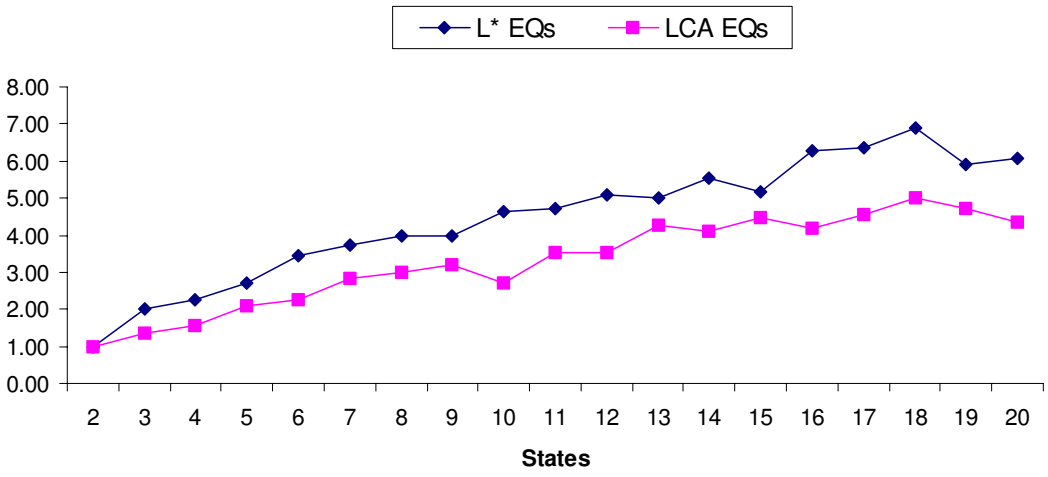


Figure 8.9: EQs average values for automata with the same number of states; comparison between L^* and LCA

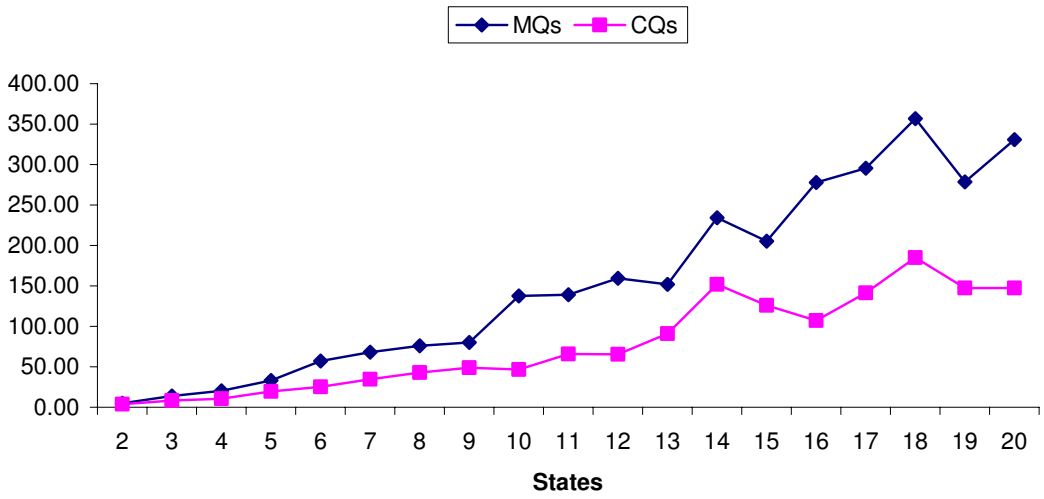


Figure 8.10: MQs and CQs, average values for automata with the same number of states; comparison between L^* and LCA

Table 8.20: Average values. Test 2

States	L^* EQs	L^* MQs	LCA EQs	LCA CQs
2	1.00	5.00	1.00	3.82
3	2.00	14.00	1.36	8.09
4	2.27	20.45	1.55	10.36
5	2.73	33.18	2.09	19.36
6	3.45	57.09	2.27	25.18
7	3.73	68.18	2.82	34.64
8	4.00	76.00	3.00	43.00
9	4.00	80.00	3.18	49.00
10	4.64	137.73	2.73	46.55
11	4.73	139.09	3.55	65.91
12	5.09	159.45	3.55	65.36
13	5.00	151.73	4.27	90.82
14	5.55	234.36	4.09	151.73
15	5.18	205.27	4.45	125.82
16	6.27	277.91	4.18	107.09
17	6.36	295.45	4.55	141.18
18	6.91	356.91	5.00	185.00
19	5.91	278.64	4.73	147.55
20	6.09	330.73	4.36	147.27

Part IV

Concluding remarks

Chapter 9

Conclusions and Further Work

9.1. Conclusions

The main contributions of this dissertation are:

- A new class of languages called *Simple p-dimensional external contextual*.
- A new learning paradigm based on *corrections*.
- Some *algorithmic aspects* based on the two previous contributions.

We summarize next the key aspects of this work.

9.1.1. A new class of languages or grammars

Chapter 3 and Chapter 6 are directly related. In Chapter 3, we have presented the state-of-the-art and discussion on classes of languages or grammars considered in studies of Grammatical Inference. In Chapter 6, we have introduced a new class of languages or grammars for Grammatical Inference.

The main points considered in Chapter 3 were:

- *Limitations of the Chomsky Hierarchy from a linguistic point of view.* The Chomsky Hierarchy has some important limitations that should be

taken into account when we want to study natural language syntax. One of the main limitation emerges when we try to locate natural language in this hierarchy.

- *More generative capacity than CF grammars is required to describe natural languages.* Chomsky developed formal grammars and the grammar hierarchy as tools for formalizing the syntax of natural languages. However, several examples of natural language structures that cannot be described using CF grammar were discovered, which suggests that more generative capacity than CF grammars is required to describe natural languages.
- *Main focus on Grammatical Inference: REG and CF.* Despite the fact that REG and CF grammars are mechanisms with a limited representational power to describe some of the aspects of natural language constructions, research in the field of Grammatical Inference has focused on learning REG or CF grammars (there are not many studies on identifying classes of grammars more powerful than CF by using grammatical inference techniques).

These considerations lead us to the following ideas:

- *Mildly Context-Sensitive (MCS).* We have proposed to focus grammatical inference studies on classes of languages or grammars more relevant from a linguistic point of view, for example, MCS. We consider that this class provides a grammatical environment for natural language syntactical constructions.
- *Incomparability of natural languages to the Chomsky Hierarchy.* In accordance with the idea supported by some other authors, we have also suggested that the Chomsky hierarchy is not the appropriate place for locating natural languages. We believe that natural languages are in fact *incomparable* to the Chomsky types (we can find some examples

of natural language constructions that are neither REG nor CF, and also some REG or CF constructions that do not occur naturally in sentences). Therefore, a new hierarchy is needed, which should hold strong relationships with Chomsky's, but should not coincide with it. This work can be regarded as a step in that long-term research direction.

- *EC_p grammars.* Since the families of languages generated by EC_p grammars have the property of transversality (i.e., they generate a class of languages occupying an orthogonal position in the Chomsky Hierarchy), they could have a chance in the study of natural language syntax. We have also pointed out that this mechanism fabricates MCS families. Therefore, due to its properties, we consider that EC_p could play an important role in grammatical inference studies too.

In Chapter 6, we have presented one of the main contributions of our dissertation. In order to study the learnability of EC_p from positive data, we have needed to introduce a new class of languages. Since Gold proved that superfinite languages are not identifiable in the limit from positive data and EC_p is superfinite, a restriction on the class has been necessary to make it possible to learn this class in the limit from only positive data. For this reason, we have restricted the class of languages to a new class called Simple many-dimensional External Contextual grammars (SEC_p).

We have presented in this chapter the properties of SEC_p and, the most remarkable facts of this family are the following:

- *SEC_p is a MCS family of languages.* As we have seen, MCS was introduced with linguistic motivations and its relevance for studies of natural language has been recognized.
- *SEC_p occupies an orthogonal position in the Chomsky hierarchy.* We have investigated the interrelationship between this family and the families of languages in the Chomsky hierarchy. We have showed that SEC_p

are incomparable with CF and REG, but included in CS. This fact is very important since natural languages could also have this property.

This makes SEC_p an interesting class to study. Moreover, this class might contribute to a better understanding of some aspects of natural language acquisition.

9.1.2. A new learning paradigm

Chapter 4 and Chapter 7 are directly related. In Chapter 4, we have presented the main formal models investigated in the field of Grammatical Inference and have discussed some linguistic aspects of them. In Chapter 7, we have wished to explore the possibility of considering that not only positive data is available in a real learning process, and we have proposed the application of a linguistic motivated idea to the studies of Grammatical Inference.

The main favorable and controversial aspects of each formal model (presented in Chapter 4) are summarized below.

1. **Gold's model**

   - Favorable aspects:

   - *Justification of Gold for studying identifiability in the limit.* Natural language learning is an infinite process. If a inference process is considered infinite, its success can be determined studying its behavior in the limit (however, since children learn their language efficiently, some consideration of computational complexity should be taken into account).
   - *Process of improvement.* The learner hypothesizes a grammar for the language after each presented string, either discarding the previously grammar hypothesis (e.g., if the latest presented string is incompatible with that grammar), or retaining it. Therefore, there

is a process of improvement, similar to the process of children's language acquisition, in which there is a progressive improvement of the language acquired by a child.

- *Identification in the limit model is not a trivial model of learning.* Contrary to what has been suggested in linguistic literature, results from Gold did not show that this model is not feasible for non-trivial classes. We have presented some results (developed later by other authors) that show that non-trivial learnable classes exist.

- Controversial aspects: in Gold's model, the definition of identification in the limit postulates greatly idealized conditions, as compared to the real-word conditions under which children learn language. Moreover, it makes some assumptions that could be somewhat problematic.

- The learner has to identify the target language exactly (even if arbitrary misleading sequences of examples are provided to him).
- The learner receives only positive examples.
- The learner is not limited by any consideration of computational complexity (he has infinite time available).
- The learner hypothesizes complete grammars instantaneously.

2. **Angluin's model**

- Favorable aspects: Angluin's model provides an important tool to the learner; he is allowed to make queries to the teacher. In that way, the main positive aspect of this model is that additional information is available in the learning process thanks to these queries.

- Controversial aspects:

- The learner has to identify the target language exactly.

- The queries introduced in this model are very unnatural for real learning environments.

Therefore, this model provides an important tool to the child, but the kind of queries used in the learning process are inadequate for real learning processes. Moreover, the teacher's answers are oversimplified for a normal learning process.

3. **Valiant's model**

   - Favorable aspects: exact identification is not required in this model; the learner is required to approximate the target grammar with high confidence using an efficient algorithm.

   - Controversial aspects: the requirement that examples are selected randomly from some fixed distribution is too strong for practical situations.

Our conclusion is that none of these models perfectly accounts for natural language acquisition. Each one has aspects that makes it useful to study first language acquisition to a certain extent, but other aspects of the model make it unsuitable for this task (e.g., some aspects of real learning process are not taken into account).

In Chapter 7, we propose a new kind of data to take into account in Grammatical Inference studies. This idea is introduced from linguistic motivations.

First, we have explored the possibility of considering that not only positive data is available in a real learning process. The main items presented here have been:

- *Linguistic discussions about the availability of negative data in child's linguistic environment.* There is no doubt that children learn a language at least in part by hearing sentences of the language to which they are

exposed to. However, there is an aspect of the child's linguistic environment which has been subject of controversy and which is still of importance in discussions of learnability. This is the availability of negative evidence. While it is accepted that positive data is available to the child, the availability of another kind of data has been widely argued. Most of the debates has been focus on whether negative data exists and whether it is necessary or even useful to the child.

- *Definition of negative data.* We have realized that authors reduce the kind of data available to the child to only two types: positive or negative. There is not intermediate possibility. Since it is clear that positive data is linguistic constructions that are grammatically correct, all the remainder is consider negative.

  We have pointed out that first, it should be clarified what we understand by negative evidence. The general definition of negative data is "examples of sentences that are not in the language". What about another kind of data that does not consist on an ungrammatical sentence, as for example, a correction? Should corrections be consider a negative data?

- *Availability of corrections.* There is growing evidence that corrections are available to children. Some linguistic studies in this direction has been shown in Chapter 7.

We have considered in this dissertation that corrections are available to the child. By correction we have understood a repetition of children's ungrammatical utterance with correction (e.g., "CHILD: They breaked the glass; ADULT: Right! They broke the glass").

Important aspects concerning corrections:

- *Should we consider this kind of information given by the adult a negative information?* Corrections should be considered as positive and negative information at the same time. Adults return a correction to the child,

which is information about a string grammatically correct. Indirectly, if a correction is received, this shows that the string uttered by the child was ungrammatical.

- *What role do corrections play in natural language acquisition?* We do not consider that corrections play a more important role than positive data in natural language acquisition. We believe that the main information received during the process of language acquisition is positive data. Corrections will play a complementary role in that process. We should consider them as a additional information available to the learner during the learning process.
- *Can we avoid overgeneralization by means of corrections?* When only positive data is available, the problem of overgeneralization can appear. We consider that it could be solved using the information embedded in a correction. Therefore, corrections could improve learnability. In that way, corrections can play an important role in terms of efficiency (some aspects of the language could be learned faster).

One of the major contributions in this dissertation has been to take into account this type of corrections in learning processes and try to model them.

Based on all these ideas, we have applied the idea of corrections to the studies of Grammatical Inference. Concretely, to the query learning model of Angluin. We have introduced the idea of correction queries (CQ) and we have applied them to learn DFA. In that way, we have tried to make a model of this instructive information received by a child during the learning process (the corrections). It has been developed in Chapter 8. Due to the results obtained, we consider that models in Grammatical Inference framework might benefit from corrections (see conclusions of Chapter 8).

We also consider that could be interesting to develop a model in Grammatical Inference that reflects better the real interaction between child-adult

(interaction play an important role in real learning processes). The combination of positive data and CQs could have a chance in such model.

If positive data and corrections are available in the learning process, the learning could be done in a more efficient way. For this reason, it could be interesting to explore a model of learning based on the availability of these kind of data.

It should be pointed out that learning from positive data and corrections should not be considered as learning from informant; as we have seen, the negative information used in an informant presentation cannot be considered as the same information as the information received from a correction.

Such a model of Grammatical Inference tries to accomplish the following items:

- Use information that is relevant for natural language acquisition.
- Take into account more aspects of real learning processes.
- Use more natural tools (for example, more natural –empirical motivated– questions).

Ideas coming from linguistics can be useful in Grammatical Inference studies in order to obtain new perspectives of the problem and possible new solutions in that way. And also, thanks to these ideas, models of Grammatical Inference might also be more realistic.

9.1.3. Algorithms associated to the new concepts

Chapter 5 and Chapter 8 are directly related. Chapter 5 has been devoted to the review of the most important results concerning learning from positive data and learning from queries. In Chapter 8, we have presented our results concerning learning SEC from only positive data and learning DFA from CQs.

The main two results presented in Chapter 5 are:

- Shinohara's results regarding learning context-sensitive languages (Shinohara's results are relevant to linguistics, since more generative power than CF is necessary to describe natural languages).
- Angluin's results on learning DFA from MQs and EQs (this constitutes one of the main positive results in computational learning theory, since it was conjectured that richer classes than DFA cannot be inferred in polynomial time using these kind of queries).

These two results are the most important and most relevant to our concerns. They are essential to understand our contributions in Chapter 8.

The main conclusions of the results presented in Chapter 8 are:

1. **Learning SEC from positive data**

   We have presented two main results concerning our study on the learnability of SEC in the limit from positive data.

   Our first result shows that the class of languages generated by simple external contextual grammars with fixed dimension and degree is learnable from positive data, from Shinohara's results [Shinohara, 1994]. The learning algorithm straightforwardly derived from our main result is enumerative in nature and therefore not time-efficient, but we have obtained positive learnability result.

   Our second result is stronger, and shows also that SEC with any dimension, but with at most q contexts and m bases, has finite elasticity (sufficient condition for positive data learnability).

   Therefore, thanks to these results we can state that SEC_p can be learned from only positive data. And also that it is important to make that restriction in EC_p in order to learn that class from only positive data.

2. **Learning DFA from corrections**

   We have proposed a new paradigm for the computational learning theory, namely learning from corrections. Our algorithm based on Angluin's L^* learning algorithm for regular languages uses an observation table with *correcting string* instead of 0s and 1s. In our running examples, generally, the number of EQs are less or equal than in Angluin's ones and the number of CQs is significantly smaller than the number of MQs.

   One of the reasons of this reduction is that an answer to a CQ contains embedded much more information. Another advantage of our approach is that we can differentiate better between states.

   The empirical results show that in most of the cases the number of queries used by LCA is smaller. We believe that this is related to the *injectivity property*, meaning that for every distinct two states the correcting string should be different. Of course, not many automata accomplish this condition, but one can see that the rare cases in which our algorithm performs worse, this injectivity property is far from being satisfied.

   Among the improvements previously discussed, we would like to mention here the adequacy of CQs in a real learning process. They reflect in a more accurate manner the process of children's language acquisition. We are aware that this kind of formalism is for an ideal teacher who knows everything and always gives the correct answers and for practical applications our working hypothesis should be adjusted.

9.2. Future Work

We consider that formal results in Grammatical Inference can be relevant to understand first language acquisition. Positive results point to possible ways in which children might learn languages, whereas negative results allow to show certain models of language acquisition to be unlikely or incomplete.

Therefore, Grammatical Inference could be a useful tool for any researcher interested in human language. Results and techniques coming from Grammatical Inference and Studies of language acquisition can help to understand the mechanisms that underlie natural language acquisition.

Several interesting research directions are open in this dissertation. We will present in the sequel some of the main future research directions.

9.2.1. Learning SEC in polynomial time

Although we have proved that SEC is learnable from only positive data, we could not prove that SEC is efficiently learnable. We present in the sequel three possible research directions in order to learn SEC in polynomial time:

a) *Proving the existence of a characteristic set for SEC languages*

In relation with our current research on learning SEC from only positive data, recently we have started to work on a polynomial time algorithm for inferring SEC grammars from positive data.

Our algorithm works given a set of strings, S, that was generated by an $SEC_{p,q}$ grammar (where p and q are known), and a depth d (the number of contexts applied in a derivation of a string is called the *depth* of a string). The list of strings must be exhaustive up to depth d. The smallest string in S is the concatenation of the elements in the base. However, there are multiple possibilities for what the actual base is (depending on the value of p). From the smallest string, a set of possible

bases, B, is generated. For example, if p=2 and the smallest string in S is aa, then the set of possible bases is $\{(\lambda, aa), (a, a), (aa, \lambda)\}$.

The rest of the algorithm is repeated for every $b_j \in B$. The strings in S are processed for b_j. Each s_i is assumed to have been generated by a single application of a context to the base. The possible contexts that could have generated s_i after one application to b_j are stored in a set P. When all the strings in S have been processed, the correct grammar must be found. The elements of P are grouped into grammars with q contexts in all possible ways and stored in a set G. Any element of G that cannot generate every string in S is discarded, leaving only those grammars in G that generate S. Next, every element $g \in G$ is used to generate all possible strings up to depth d. If any of the generated strings are not in the input set S then g is discarded.

For example, consider an input set $S = \{a, aaa, bab, ababa, baaab\}$ for a language generated by an $SEC_{1,2}$ grammar. The input is exhaustive up to depth 2. The shortest string in S is a, and therefore it is the base. Since $p = 1$, a is the only possible base. Assuming that aaa was generated by a single application of a context to a, the possible contexts that can generate the input set are $\{[(\lambda, aa)], [(a, a)], [(aa, \lambda)]\}$. This is done again for bab, resulting in $\{[(b, b)]\}$. All of the possible contexts are then put in a set G. The only $g_i \in G$ that can generate the language is $\{[(a, a)], [(b, b)]\}$; the rest are discarded. All possible strings up to depth 2 have derivations in this grammar. Since it is equal to the input set, this grammar is not discarded and is output by the algorithm. In this case, there was only one possible base. If this grammar had been $SEC_{2,2}$, then the possible bases would have been $\{[(\lambda, a)], [(a, \lambda)]\}$. For each of the possible bases, the set G would have been constructed and every element of that set tested. Unlike the example chosen here, many input sets can be generated by several grammars, some of which have

distinct bases.

The algorithm implemented is showed in Figure 9.1.

INFER-GRAMMAR $(d, p, q, inputSet)$

smallest ← Find-smallest(*inputSet*)
bases ← Generate-possible-bases(p, *smallest*)
Initialize *finalGrammars*
Initialize *result*

for *base* ∈ *bases*
do Initialize *possibleContexts*
  for $i \in inputSet$
  do *possibleContexts.push* [Generate-possible-contexts(i, *base*)]

  possibleGrammars ←
    ← [Generate-all-context-combinations(*possibleContexts*)]
  for $j \in possibleGrammars$
  do if Can-generate-all-strings $(j, base)$
    then *finalGrammars.push*$[(j, base)]$

for $k \in finalGrammars$
do if Generates-exact-list $(d, k, inputSet)$
  then *result.push*[k]
Return(*result*)

Figure 9.1: Pseudocode of the inference algorithm

Supplementary work is necessary to prove the correctness of the algorithm. However, our conjecture is that a finite sample of the language, a characteristic sample, is sufficient to infer a grammar for the language.

Therefore, a future research direction could be to try to prove that there exists a characteristic set for SEC languages.

b) *Using CQs*

We consider that another interesting research direction could be to learn SEC in polynomial time by means of CQs. In the sequel we present an example of the usefulness that can have the information received from the CQ for efficient learning.

Lets suppose that we only provide to previous algorithm positive examples (the parameter d of the above algorithm disappears and we cannot check if the candidate grammar *generates-exact-list*). In that case, the learning algorithm may make an overgeneralization.

The following example shows that situation. We suppose that the correct grammar is the following SEC with 2 dimensions and 2 contexts:

- B = $\{(\lambda, \lambda)\}$
- C = $\{\ c_1 = [(a, \lambda), (c, \lambda)],\ c_2 = [(\lambda, b), (\lambda, d)]\}$

As we can see, the language generated by this grammar is: $L = \{a^n b^m c^n d^m \mid n, m \geq 0\}$

Our first hypothesis could be the following:

- B = $\{(\lambda, \lambda)\}$
- C = $\{\ c_1 = [(a, \lambda), (c, \lambda)],\ c_2 = [(b, \lambda), (\lambda, d)]\}$

Note that with this grammar we can generate all the strings in L, but also we can generate strings that are not in the language, for instance, *bacd*. Therefore, this is an overgeneral hypothesis.

How can we avoid that problem? In order to solve that problem, another future research direction could be to try to combine positive data and

CQs to infer SEC grammars. With only positive data we will not be able to refute a too general hypothesis. What about using also CQs in the learning process? Could correction queries be useful to see that our hypothesis is not correct?

In the above example, the correction of *bacd* would be φ (this means that the string does not belong to L). Therefore, thanks to the correction, we would be able to refute that hypothesis and construct a new hypothesis consistent with all these data.

This example shows that, despite the fact that SEC is learnable from positive data, the information received from the CQs can be useful for efficient learning. Since we are interested on an efficient algorithm, perhaps it could be interesting to explore more this idea in the future.

It should be pointed out also that, if we want to apply CQs to learn SEC, perhaps it will be better to study another kind of corrections. Namely, we have seen that the kind of CQs introduced in Chapter 8 are useful for learning DFA, but perhaps, to study the learnability of SEC, another kind of correction could be more appropriate.

For example, we have seen that, roughly speaking, SEC produces a language starting from an axiom and iteratively adding contexts to the currently generated words. This suggest that perhaps, an appropriate way to correct words generated by SEC grammars could be inserting the necessary symbols to the word (i.e., only insertion is allowed). The kind of correction that should be used for SEC can be also subject of future research.

It could be also interesting to prove that we can learn SEC only from positive data and corrections, without knowing the dimension and the number of contexts from the beginning. This perhaps will be more realistic and it would reflect better the process of natural language acquisition.

c) *Solving two open problems*

It could be also interesting to solve the following open problems:

- What is the status of the equivalence problem for SEC? This is still an open problem. In future work, we would try to prove that equivalence is decidable.
- Do there exist normal forms for SEC? If there do exist, it would facilitate the study of this type of grammars.

These results would be relevant within the field of Grammatical Inference, since we would obtain positive results of a class more expressive than context-free, which is able to describe certain aspects of the syntax of natural language. Perhaps, these results could be extended to other classes of languages.

9.2.2. Exploring the relevance of correction queries within Grammatical Inference

Learning from corrections is our major contribution in this dissertation. The results obtain so far suggest that CQ may play an important role in learning processes. Therefore, *learning from CQ* is a promising model to be further investigated in the field of algorithmic learning theory.

Regarding the results present in Chapter 8, for the future it would be interesting:

- to find subsets of regular languages for which LCA performs always better than L^*.
- to identify which condition could be sufficient in order to use a smaller number of queries.
- to try to extend this result to CF and $MCSL$ (probably another type of correction would be needed).

Other future research directions are:

- To study CQs *per se*. This includes, among other things:
  - Different kind of corrections.
  - Adequacy of each one of them for our study.
  - Complexity of this kind of queries.
  - Comparison with other kind of queries that already exist.
- To demonstrate that CQs can be a very useful tool to learn successfully different classes of languages or grammars.
  - Classes for which learnability positive results have not been obtained so far. To see if they can be learned using corrections.
  - New classes of languages that do not necessarily belong to the Chomsky Hierarchy (i.e, classes of languages with an orthogonal position in that hierarchy). Such classes of languages could be appropriate candidates to model natural language syntax. These results would help us to understand better the process of natural language acquisition.
- To formalize the new proposed model (learning from positive data and CQs) and assess its relevance in the Grammatical Inference framework. We propose:
  - To formally fix that model an do an exhaustive analysis of it.
  - To develop the algorithm that will implement it.
  - To study classes of languages or grammars using this model (classes that belong or not to the Chomsky Hierarchy).

  All this work could demonstrate the effectiveness of this new proposed model of learning, which we consider that reflects in a more accurate manner a real learning process.

- To explore possible applications of our algorithms using real world data.
- To study practical applications of CQs (e.g., to see if they could be a useful tool for Internet search engines, as for example, *Google*).
- To extend all our results to studies related directly to robotics, machine translation, natural language processing, neural networks, bioinformatics, computer-assisted language learning, etc.

$\therefore$

The results obtained in this dissertation encourage us to continue working on all the promising ideas presented and on its applications. We believe that all the future research directions pointed out in this chapter could be helpful to go further in the understanding of language acquisition.

Appendix

Test 1. Comparative results

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
001M.txt	a	3,4,1,1,5,2	2	2	11	1	2
002M.txt	a	0		1	2	1	1
003M.txt	a	0	0	1	2	1	2
004M.txt	a	1,1	0	1	3	1	2
005M.txt	a	2,4,3,1,3	0,2,3	2	8	2	5
006M.txt	a	1,2,1	2	2	5	1	2
007M.txt	a	1,0	0	1	3	1	2
008M.txt	a	1,2,2	1	2	6	1	2
009M.txt	a	1,3,2,4,2	2	2	9	1	2
010M.txt	a	1,6,4,5,3,3,2	0,1,3,4,6	3	14	3	9
011M.txt	a	2,1,1	0,1	2	6	2	5
012M.txt	a	1,2,3,0	1,2	2	6	2	3
013M.txt	a	3,2,3,4,1	2	2	9	1	2
014M.txt	a	1,2,3,0	2	3	9	1	2
015M.txt	a	3,0,1,4,2	4	3	10	1	2
016M.txt	a	1,1	1	1	3	1	2
017M.txt	a	5,1,4,2,1,3	1,3,5	3	12	3	9
018M.txt	a	4,3,1,0,2	0,2	2	9	2	4
019M.txt	a	3,4,1,2,5,0	1,2,3,4,5	2	12	2	10
020M.txt	a	2,3,1,4,4	3	3	12	1	2
021M.txt	a	2,1,1	1	2	5	1	2
022M.txt	a	3,2,0,1	0,1,3	2	7	2	6
023M.txt	a	4,1,5,2,3,1	0,1,4	3	9	3	8
024M.txt	a	3,0,1,2	1,2,3	2	8	2	6
025M.txt	a	2,3,1,0	0,1,3	2	7	2	5
026M.txt	a,b	3,5,3,2,6,6,0,4,4,2,4,6,7,1,7,3	0,3,4,5,6	3	54	3	49
027M.txt	a,b	7,5,8,6,3,7,4,7,0,4,1,6,7,2,2,5,1,2	0,1,2,4,5,7,8	4	76	4	67
028M.txt	a,b	2,3,5,1,0,5,1,3,4,1,3,4	0,4,5	2	27	3	18
029M.txt	a,b	3,0,2,2,1,4,0,1,3,1	1,2	3	31	2	13
030M.txt	a,b	0,7,3,2,5,8,3,6,2,2,4,1,4,0,7,2,3,7	0,1,5	6	111	2	20
031M.txt	a,b	5,2,3,0,3,2,6,1,0,8,0,4,2,5,2,6,4,7	4,5	5	97	2	20

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
032M.txt	a,b	6,4,7,0,5,5,7,2,1,4,1,5,2,3,1,6	0,2,3,4,7	4	76	4	47
033M.txt	a,b	4,3,2,4,3,3,7,4,6,1,5,6,1,3,5,0	0	5	159	1	10
034M.txt	a,b	6,3,4,3,6,6,0,0,3,0,1,2,4,5	5	6	125	2	21
035M.txt	a,b	3,3,5,1,7,6,2,2,6,5,4,0,1,2,3,1	2,3,4,5,6	5	104	5	95
036M.txt	a,b	5,6,5,4,4,3,6,1,6,1,2,0,2,5	0,2,3,5	3	35	3	27
037M.txt	a,b	2,2,2,4,1,4,0,2,3,2	1	3	34	1	7
038M.txt	a,b	8,5,4,2,4,9,0,6,7,2,2,9,1,1,9,0,9,8,3,6	0,3,8	5	127	3	32
039M.txt	a,b	2,5,1,1,5,1,1,6,4,0,2,3,4,6	1,2,4,5	3	39	3	36
040M.txt	a,b	6,5,7,3,6,7,6,4,6,6,7,4,1,2,4,0	0,1,3,7	4	71	4	43
041M.txt	a,b	4,7,5,3,5,5,1,3,0,6,2,1,6,2,7,7	2,6	6	118	2	12
042M.txt	a,b	1,3,2,5,6,2,0,1,6,4,4,7,1,3,5,1	0,2,3	5	89	2	17
043M.txt	a,b	5,1,8,5,6,8,3,1,5,2,4,0,3,2,0,7,1,7	0,5,8	4	83	2	28
044M.txt	a,b	3,5,2,1,7,3,3,9,3,6,6,8,1,1,6,9,2,4,3,8	0,1,2,3,4,5,8	5	118	5	112
045M.txt	a,b	1,3,6,4,5,1,2,1,6,2,5,4,3,1	3	4	69	2	27
046M.txt	a,b	9,5,7,2,2,4,8,0,9,2,1,8,7,7,0,3,2,9,6,2	0	6	229	2	22
047M.txt	a,b	2,7,3,3,4,6,7,1,5,0,7,1,7,5,6,1	5,6,7	4	83	2	34
048M.txt	a,b	4,1,3,4,4,0,0,0,4,2	1,2,3,4	3	39	3	35
049M.txt	a,b	8,5,3,3,0,4,5,2,1,6,0,5,0,7,8,7,1,5	0,1,2,5,7,8	5	97	5	88
050M.txt	a,b	6,1,8,5,4,5,8,3,1,3,3,7,7,0,5,5,8,2	2,3,4,6,7,8	3	59	3	51
051M.txt	a,b	7,1,5,6,2,3,7,2,7,4,7,3,4,5,1,7	3,4	4	71	3	30
052M.txt	a,b	6,6,6,1,5,4,3,6,1,0,3,1,6,2	2	4	71	2	14
053M.txt	a,b	2,0,1,8,1,2,5,6,3,5,8,8,7,5,4,5,7,2	1,2	5	159	3	36
054M.txt	a,b	5,2,4,2,1,1,1,1,3,4,6,3,0,0	2,4	3	49	2	16
055M.txt	a,b	6,2,7,6,7,3,0,0,7,1,4,4,8,7,3,0,2,5	0,1,4,6,7	4	76	4	46
056M.txt	a,b	7,5,2,1,0,1,7,2,0,3,6,4,7,6,1,6	4,7	5	90	3	32
057M.txt	a,b	1,1,0,1	1	1	5	1	4
058M.txt	a,b	8,4,5,3,0,2,6,6,3,5,2,7,0,1,8,0,2,7	0,5	6	161	4	66
059M.txt	a,b	6,0,0,7,0,1,5,4,5,3,3,2,1,3,6,2	4,5	4	89	3	29
060M.txt	a,b	5,2,4,1,0,4,6,7,2,3,7,5,1,2,2,7	3,5,7	4	71	2	16
061M.txt	a,b	8,7,6,5,5,4,3,1,2,4,3,6,4,8,6,3,8,6	1,6,8	5	90	3	29
062M.txt	a,b	2,2,4,5,6,2,2,5,2,2,3,7,2,1,1,4	1,4,7	5	90	4	44
063M.txt	a,b	5,3,2,6,6,0,1,6,3,2,2,4,0,3	0,1,2,4,6	4	65	4	50
064M.txt	a,b	1,2,2,3,2,6,3,0,5,4,2,2,4,0	1,2,5	3	53	2	18
065M.txt	a,b	4,4,3,7,8,0,7,9,2,9,6,0,6,5,5,7,1,8,7,6	1,4,6	4	97	3	47
066M.txt	a,b	3,5,1,3,2,1,5,3,2,4,0,4	1,3,4	3	49	2	19
067M.txt	a,b	0,1,0,1	0	1	5	1	4
068M.txt	a,b	6,1,6,5,4,7,5,7,3,1,2,0,5,0,2,3	1,5,7	3	62	3	28
069M.txt	a,b	0,7,0,3,5,4,2,4,1,6,0,3,6,5,3,6	0,3,5	3	49	3	36
070M.txt	a,b	5,7,2,3,6,8,3,1,5,8,1,4,6,0,3,7,5,3	0,3,7,8	5	111	4	61
071M.txt	a,b	1,2,2,3,0,0,2,0	0,2	3	27	3	23
072M.txt	a,b	1,2,4,1,3,0,3,0,4,4	1,3	3	31	2	15
073M.txt	a,b	0,2,5,5,2,3,7,5,3,5,1,6,4,2,7,1	4,5	3	59	2	16
074M.txt	a,b	2,0,6,5,2,7,7,8,5,1,1,3,9,9,4,4,8,9,3,1	2,3,5	4	90	3	57
075M.txt	a,b	0,4,0,3,4,4,1,4,1,2	3	3	44	1	7
076M.txt	a,b	6,0,7,7,1,4,1,5,7,6,6,4,5,3,2,6	0,2,3,4,5	3	54	3	44
077M.txt	a,b	2,6,2,4,7,5,1,5,4,3,1,7,1,7,3,3	4,5	4	71	3	29

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
078M.txt	a,b	7,5,0,5,6,7,1,2,8,0,4,3,3,7,7,1,9,4,4,9	0,2,3,4,5,6,7,9	6	169	6	157
079M.txt	a,b	4,6,6,6,2,3,6,7,1,6,8,1,7,8,2,4,5,6	8	6	152	2	17
080M.txt	a,b	4,2,6,0,6,1,0,1,4,5,0,4,3,6	1,2,3,4,5	4	71	4	60
081M.txt	a,b	5,4,5,1,1,2,2,3,4,0,2,3	0,1,2,4,5	4	54	4	52
082M.txt	a,b	7,6,3,5,2,4,2,3,2,3,5,6,6,0,4,1	0,2,3,5,6,7	6	111	6	108
083M.txt	a,b	0,2,1,3,1,0,0,1	0	2	19	1	6
084M.txt	a,b	1,0,0,4,2,2,4,2,2,5,1,3	0,1,2	3	35	3	32
085M.txt	a,b	5,9,7,1,1,4,0,4,0,8,3,2,5,7,3,6,0,3,2,1	0,2,5,6,7,8	4	87	4	58
086M.txt	a,b	3,0,6,7,2,5,1,0,3,5,3,2,4,7,4,1	0,1,3,6	4	65	2	31
087M.txt	a,b	5,7,2,5,6,5,2,1,7,0,4,2,5,3,1,5	0,1,2,3,4,5,7	5	135	5	128
088M.txt	a,b	3,4,0,2,1,2,2,0,5,3,6,6,4,0	0,1,2,5	4	71	3	38
089M.txt	a,b	2,4,2,8,7,5,6,4,6,8,5,8,3,9,1,2,5,3,3,2	1,2,3,4,7,8,9	5	118	5	96
090M.txt	a,b	1,2,6,3,1,1,2,4,5,7,0,0,3,7,5,6	2	6	143	2	22
091M.txt	a,b	5,9,3,1,3,8,1,5,0,8,3,2,2,4,0,2,6,2,7,6	1	4	125	2	37
092M.txt	a,b	2,1,1,5,1,6,3,6,0,0,4,1,1,3	2,6	4	65	2	13
093M.txt	a,b	1,4,0,1,2,3,1,3,2,5,4,5	0,2,3,4	4	65	4	48
094M.txt	a,b	3,4,7,3,6,5,3,6,2,1,1,8,6,2,3,1,6,7	0,2,3,4,6,7	3	49	4	61
095M.txt	a,b	3,4,1,2,2,0,1,3,2,3	0,2,4	2	23	2	18
096M.txt	a,b	8,0,7,2,0,3,5,9,2,2,9,7,4,8,9,6,7,9,1,1	7	5	127	2	26
097M.txt	a,b	4,3,4,6,4,1,3,5,6,3,3,1,1,2	0,3	4	71	3	32
098M.txt	a,b	0,4,5,4,1,3,2,3,1,6,3,2,2,2	1,4,6	3	49	2	14
099M.txt	a,b	2,8,8,1,2,2,6,5,3,5,4,4,5,8,7,9,7,6,0,1	2,8,9	4	90	3	30
100M.txt	a,b	5,1,4,5,0,2,3,6,3,1,6,2,2,4	0,2,5	5	71	3	33
101M.txt	a,b	4,4,6,3,2,0,3,1,0,5,3,0,7,4,2,2	2	2	71	1	10
102M.txt	a,b	4,1,2,4,4,3,6,5,1,7,0,3,5,4,3,4	0,1,2,3,4,5,7	3	76	3	75
103M.txt	a,b	1,1,5,4,4,4,2,2,2,3,5,5	0,1,3	3	34	2	12
104M.txt	a,b	2,8,6,5,8,3,3,5,7,4,4,0,7,1,2,6,1,1	3	4	95	2	27
105M.txt	a,b	7,9,1,8,8,0,5,1,0,1,6,7,9,5,1,6,9,2,4,3	1	6	149	3	37
106M.txt	a,b	1,5,8,0,1,7,1,2,2,8,0,6,3,0,4,2,0,5	1,3,8	5	95	4	49
107M.txt	a,b	3,2,2,1,2,5,3,4,3,1,5,5	2	4	65	1	7
108M.txt	a,b	2,6,5,3,4,1,0,0,0,3,3,3,6,6	1,3,5	3	39	3	23
109M.txt	a,b	2,3,6,5,4,2,2,1,1,1,3,1,2,0	0,1,2,3,4,5	3	59	3	55
110M.txt	a,b	8,3,2,6,7,0,2,6,1,7,1,5,8,1,5,0,4,7	1,2,5,8	3	65	3	60
111M.txt	a,b	0,5,2,2,4,3,5,4,2,0,5,1	1,2,4	3	35	2	15
112M.txt	a,b	2,0,0,0,1,1	1	2	11	1	5
113M.txt	a,b	2,3,4,2,5,1,9,8,7,5,1,7,3,0,7,0,4,8,6,4	4	4	95	2	24
114M.txt	a,b	6,1,3,0,3,5,2,3,5,7,5,1,8,4,1,5,6,8	2,5,6,7	5	90	3	42
115M.txt	a,b	5,7,1,4,0,6,5,5,5,6,3,7,4,5,2,1	0,1,3	4	71	3	31
116M.txt	a,b	2,4,0,5,1,0,5,4,1,6,3,1,0,5	0,1,4,5	4	77	4	55
117M.txt	a,b	3,5,5,3,4,6,0,2,3,6,7,3,8,3,9,3,0,1,7,4	0,1,3,7	4	83	3	44
118M.txt	a,b	0,3,3,5,0,3,5,0,3,1,4,2	2,4	3	44	2	15
119M.txt	a,b	3,4,5,4,3,5,1,2,3,4,5,5	1,4	3	39	2	10
120M.txt	a,b	7,6,2,5,6,3,5,7,2,6,1,6,2,0,2,4	2,4,5	5	77	4	46
121M.txt	a,b	1,1,3,2,1,3,3,3	0,2	3	31	2	6
122M.txt	a,b	4,2,7,4,4,1,3,2,2,6,8,3,7,2,1,5,2,4	1,3	6	135	2	33
123M.txt	a,b	4,1,2,0,3,5,0,0,5,6,7,5,2,1,3,3	1,2,3,4,7	4	71	4	51

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
124M.txt	a,b	5,2,2,0,4,5,1,1,2,3,2,1	0,1,3,4,5	4	54	4	47
125M.txt	a,b	0,2,1,8,5,4,1,3,6,1,2,1,3,1,2,8,7,2	6	5	151	2	20
126M.txt	a,b,c	1,1,4,3,5,0,2,0,3,5,0,3,3,1,5,2,0,3	5	5	104	1	14
127M.txt	a,b,c	5,0,2,5,5,1,1,5,3,4,2,4,0,3,5,5,0,0	0,1,4	3	45	2	28
128M.txt	a,b,c	4,3,4,2,2,1,0,1,0,2,1,4,4,3,3	1,2,4	3	45	3	38
129M.txt	a,b,c	0,0,3,0,3,1,1,3,1,2,3,2	1,2,3	2	38	2	36
130M.txt	a,b,c	5,5,4,0,1,1,5,1,1,2,0,1,3,1,3,5,2,3	0,2,4,5	4	67	4	57
131M.txt	a,b,c	0,1,5,2,0,5,5,5,5,1,5,2,0,3,5,5,4,0	0,2,5	3	85	2	42
132M.txt	a,b,c	1,2,0,5,0,1,0,0,1,1,5,1,3,4,2,4,5,2	0,3	5	115	2	28
133M.txt	a,b,c	0,3,4,3,1,2,3,0,5,4,0,1,5,0,2,2,0,4	3,4,5	3	67	2	46
134M.txt	a,b,c	4,3,4,0,0,3,5,1,0,3,5,5,0,4,2,2,5,4	0,2,3	4	76	4	64
135M.txt	a,b,c	5,0,3,4,0,2,4,3,4,4,2,1,5,1,5,0,1,0	4	4	76	3	55
136M.txt	a,b,c	3,2,1,0,5,3,5,4,5,4,1,4,4,1,3,3,2,2	2,3	3	76	2	33
137M.txt	a,b,c	3,4,1,2,5,5,1,2,4,3,2,5,3,4,2,4,3,5	0,2,4	4	104	3	60
138M.txt	a,b,c	5,2,3,3,4,3,1,4,5,3,3,0,0,3,0,3,5,4	0,1,2	3	67	2	25
139M.txt	a,b,c	5,5,0,3,0,5,4,2,5,0,3,4,5,4,5,2,1,3	3,5	3	76	3	60
140M.txt	a,b,c	4,1,2,3,4,4,0,5,4,4,4,2,4,3,4,2,0,2	2,5	3	76	3	55
141M.txt	a,b,c	4,3,3,2,4,2,4,3,2,0,5,1,3,0,4,3,5,2	4	5	137	2	36
142M.txt	a,b,c	3,0,2,4,2,1,0,0,0,0,0,5,5,1,0,4,4,1	2	4	104	1	14
143M.txt	a,b,c	2,2,1,3,4,1,0,3,5,4,1,0,1,1,5,3,0,5	1,2,3,4	3	76	3	69
144M.txt	a,b,c	1,3,0,3,2,2,0,2,2,3,0,0	0,1,3	3	45	3	44
145M.txt	a,b,c	0,3,3,1,2,3,4,1,2,1,4,4,4,3,3	1,2,4	4	67	3	48
146M.txt	a,b,c	0,4,4,2,4,4,2,3,2,1,3,4,2,2,4	2,4	3	45	2	23
147M.txt	a,b,c	2,0,4,0,1,1,3,4,1,1,1,1,3,0,4	0,3	3	59	3	56
148M.txt	a,b,c	3,4,1,4,1,1,2,1,3,0,0,5,0,3,4,5,2,4	0,3,5	3	59	2	34
149M.txt	a,b,c	4,2,3,2,0,4,3,1,4,4,2,3,1,4,0	3	4	85	1	12
150M.txt	a,b,c	3,1,1,1,0,0,3,4,4,1,5,1,4,2,4,4,0,5	3	4	137	3	70
151M.txt	a,b,c	2,1,0,0,1,0,4,1,1,1,5,2,4,5,3,3,1,2	1,5	4	115	3	61
152M.txt	a,b,c	0,2,4,1,2,0,2,3,5,0,5,5,2,2,1,0,0,5	1	4	115	2	32
153M.txt	a,b,c	1,2,3,1,2,2,5,4,0,3,3,4,1,1,1,4,2,4	1,2,3,5	3	67	3	61
154M.txt	a,b,c	0,5,3,1,3,1,3,0,3,1,4,0,3,5,2,5,5,1	1,2,3	4	104	2	49
155M.txt	a,b,c	1,3,1,0,2,3,2,0,0,4,3,0,1,1,3	0,2,3	3	52	2	25
156M.txt	a,b,c	2,1,3,4,0,4,1,0,3,3,4,0,2,3,4	0,4	3	45	2	25
157M.txt	a,b,c	2,2,3,0,0,4,5,0,5,3,1,5,0,5,1,3,2,2	1,3	3	76	2	29
158M.txt	a,b,c	2,5,3,3,4,0,4,4,3,5,4,1,4,4,1,2,0,4	3	3	67	2	33
159M.txt	a,b,c	3,1,3,2,4,4,1,1,1,2,1,0,3,4,2	0	3	76	1	12
160M.txt	a,b,c	2,1,4,3,3,1,0,0,1,2,3,1,5,4,0,2,0,3	0,2,3,5	3	67	3	65
161M.txt	a,b,c	3,3,3,1,4,4,3,3,2,2,2,1,5,4,5,2,3,0	5	3	115	2	33
162M.txt	a,b,c	0,1,2,1,0,0,0,2,2	1	2	22	1	8
163M.txt	a,b,c	4,1,4,2,3,1,3,1,1,1,5,3,3,0,5,3,5,3	3,4,5	3	67	4	69
164M.txt	a,b,c	4,3,5,2,4,4,5,2,3,3,4,1,2,3,1,1,1,4	1,2,3	3	52	3	50
165M.txt	a,b,c	5,0,0,0,1,2,5,2,4,5,5,5,2,1,3,0,2,5	3,5	3	76	3	43
166M.txt	a,b,c	2,4,2,4,2,2,4,2,0,5,1,2,2,3,5,1,4,3	0,2	4	85	2	28
167M.txt	a,b,c	4,5,1,3,0,5,3,0,1,4,1,3,4,3,1,4,2,5	1	4	115	1	14
168M.txt	a,b,c	0,4,5,4,3,5,3,4,4,4,1,4,2,2,3,5,0,0	1,2	4	76	2	31
169M.txt	a,b,c	1,4,1,5,4,4,0,3,3,2,3,2,1,1,0,5,3,0	2	3	115	2	42

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
170M.txt	a,b,c	4,1,3,2,2,0,1,5,1,3,1,3,4,5,4,0,4,4	0,1,2,3	3	76	3	62
171M.txt	a,b,c	1,4,2,1,1,3,5,3,0,0,5,2,0,0,4,2,3,2	2	3	67	2	32
172M.txt	a,b,c	0,1,0,3,1,2,4,1,3,3,0,1,3,0,4	2	4	85	1	12
173M.txt	a,b,c	1,5,0,2,4,3,3,5,5,1,1,5,1,4,0,0,3,3	2,3,4,5	4	76	3	42
174M.txt	a,b,c	2,0,4,2,0,0,0,1,3,2,1,1,1,2,4	0,3,4	3	52	3	42
175M.txt	a,b,c	3,3,3,0,0,0,3,0,5,2,1,1,4,2,2,3,5,4	0,2,4,5	4	94	4	82
176M.txt	a,b,c,d	2,3,3,2,2,3,0,0,2,2,2,1,0,3,3,1	1	3	63	2	37
177M.txt	a,b,c,d	4,2,4,0,1,4,1,4,0,0,3,2,0,0,0,3,0,1,4,2	0,1,3	2	53	2	46
178M.txt	a,b,c,d	1,0,4,3,4,4,2,3,3,1,0,2,2,1,2,1,3,3,2,2	1,3,4	3	82	3	71
179M.txt	a,b,c,d	3,1,2,2,0,3,3,0,3,2,0,0,1,3,2,3	2	3	63	1	14
180M.txt	a,b,c,d	0,4,4,3,3,0,1,4,1,3,3,0,3,1,0,4,0,4,2,4	1,3	3	82	2	40
181M.txt	a,b,c,d	1,1,0,3,1,4,3,4,3,4,1,2,4,3,4,0,3,4,0,2	2	3	95	1	17
182M.txt	a,b,c,d	4,4,3,4,2,4,3,1,0,1,1,3,2,1,2,1,4,4,2,3	0	4	134	2	45
183M.txt	a,b,c,d	2,0,1,1,1,3,0,0,1,0,0,0,1,0,0,1	0,1	3	73	3	68
184M.txt	a,b,c,d	1,2,3,1,0,1,0,2,1,3,2,3,3,1,2,0	0,3	2	37	2	26
185M.txt	a,b,c,d	1,2,2,2,2,2,3,2,2,4,0,4,4,0,3,4,0,0,3,2	1,3	3	63	2	36
186M.txt	a,b,c,d	3,1,1,0,0,2,4,4,3,2,1,1,3,2,3,2,4,3,1,4	1,2,4	4	121	4	116
187M.txt	a,b,c,d	4,2,1,0,0,4,0,2,3,0,4,0,1,1,2,0,3,4,2,1	3,4	3	73	3	53
188M.txt	a,b,c,d	2,1,3,3,4,1,3,2,1,4,3,1,3,1,4,4,4,2,4,2	2,3	2	53	2	41
189M.txt	a,b,c,d	1,3,0,1,1,4,0,4,1,0,2,3,4,1,0,1,1,4,3,2	2	3	95	1	17
190M.txt	a,b,c,d	0,4,3,3,4,2,4,3,1,1,4,3,2,0,4,4,2,3,0,4	0,1	3	63	2	41
191M.txt	a,b,c,d	3,0,3,3,0,2,3,0,0,1,3,2,0,2,2,4,2,4,2,4	3	4	121	2	46
192M.txt	a,b,c,d	0,4,0,2,1,2,0,2,1,0,4,4,0,3,1,3,3,0,3,1	1,2,3	3	63	3	65
193M.txt	a,b,c,d	3,2,4,3,2,4,4,1,4,4,4,3,3,3,1,0,4,3,0,3	1,2	4	108	2	41
194M.txt	a,b,c,d	4,2,4,1,2,1,4,4,4,0,3,3,4,3,1,4,1,0,3,4	3	4	108	3	58
195M.txt	a,b,c,d	3,0,0,4,2,1,0,1,4,4,2,0,4,1,3,3,1,4,4,4	0,2,4	4	121	4	109
196M.txt	a,b,c,d	2,2,3,4,3,2,4,1,0,0,4,3,0,3,3,3,4,3,1,4	4	4	121	2	39
197M.txt	a,b,c,d	3,0,0,2,3,4,4,2,3,0,0,4,2,4,2,1,2,2,0,1	1,3,4	2	53	2	51
198M.txt	a,b,c,d	2,2,0,1,0,1,1,0,2,1,2,0	0	2	30	1	11
199M.txt	a,b,c,d	0,1,0,2,2,0,3,0,2,1,1,1,2,3,3,3	3	3	63	2	38
200M.txt	a,b,c,d	2,1,3,0,0,1,1,2,1,2,2,2,1,2,0,2	2	3	63	1	14

Test 2. Comparative results

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
002.0.txt	a, b	1, 0, 1, 0	1	1	5	1	4
002.1.txt	a, b	1, 0, 0, 1	0	1	5	1	4
002.2.txt	a, b	1, 0, 1, 1	1	1	5	1	4
002.3.txt	a, b	1, 0, 1, 1	0	1	5	1	3
002.4.txt	a, b	1, 0, 0, 0	0	1	5	1	4
002.5.txt	a, b	0, 1, 1, 0	1	1	5	1	4
002.6.txt	a, b	0, 1, 0, 0	1	1	5	1	4
002.7.txt	a, b	1, 1, 1, 1	0	1	5	1	3
002.8.txt	a, b	1, 1, 0, 0	1	1	5	1	4
002.9.txt	a, b	0, 1, 1, 0	0	1	5	1	4
002.10.txt	a, b	1, 0, 0, 0	1	1	5	1	4
003.0.txt	a, b	0, 1, 2, 2, 0, 2	0, 2	2	14	2	13
003.1.txt	a, b	2, 2, 0, 0, 1, 2	0	2	14	1	5
003.2.txt	a, b	0, 1, 2, 1, 0, 2	1	2	14	1	5
003.3.txt	a, b	2, 2, 1, 0, 1, 1	1	2	11	1	5
003.4.txt	a, b	1, 2, 0, 2, 0, 0	0, 2	2	14	2	12
003.5.txt	a, b	0, 1, 0, 2, 1, 0	1	2	17	1	5
003.6.txt	a, b	1, 0, 2, 1, 2, 1	2	2	11	1	5
003.7.txt	a, b	1, 0, 2, 0, 1, 1	1	2	17	2	14
003.8.txt	a, b	0, 1, 2, 2, 0, 0	2	2	11	1	5
003.9.txt	a, b	2, 1, 0, 0, 0, 2	2	2	14	1	5
003.10.txt	a, b	0, 2, 1, 0, 0, 1	1, 2	2	17	2	15
004.0.txt	a, b	3, 0, 0, 2, 3, 2, 0, 1	1, 2	3	27	2	12
004.1.txt	a, b	3, 1, 2, 1, 1, 3, 3, 3	2	3	27	1	5
004.2.txt	a, b	0, 3, 1, 0, 1, 3, 3, 2	0, 2, 3	2	19	2	18
004.3.txt	a, b	0, 1, 0, 2, 3, 0, 2, 3	0, 1	2	14	2	11
004.4.txt	a, b	1, 0, 2, 0, 3, 3, 1, 0	0, 1	2	14	2	11
004.5.txt	a, b	1, 1, 3, 1, 1, 2, 2, 0	2	2	19	1	6
004.6.txt	a, b	0, 3, 0, 2, 2, 2, 1, 3	2	2	19	1	6
004.7.txt	a, b	1, 3, 0, 2, 2, 3, 3, 0	1, 3	2	17	2	15
004.8.txt	a, b	0, 1, 2, 0, 3, 0, 3, 3	2	3	27	1	5
004.9.txt	a, b	3, 1, 0, 3, 1, 2, 2, 2	1, 2, 3	2	23	2	19
004.10.txt	a, b	2, 1, 3, 1, 1, 3, 1, 1	3	2	19	1	6
005.0.txt	a, b	4, 0, 2, 1, 0, 2, 4, 3, 3, 1	0, 1	3	39	2	14
005.1.txt	a, b	2, 0, 1, 4, 0, 3, 1, 0, 4, 2	0, 2, 3	2	23	2	21
005.10.txt	a, b	3, 2, 3, 4, 4, 3, 1, 0, 1, 0	0, 2, 3	3	31	2	13
005.3.txt	a, b	4, 1, 3, 4, 0, 2, 3, 2, 3, 1	1, 2, 3	2	23	2	22
005.4.txt	a, b	2, 3, 4, 0, 1, 0, 3, 2, 2, 3	0, 2	2	23	2	14
005.5.txt	a, b	4, 3, 2, 3, 0, 0, 2, 3, 1, 2	1, 2	3	27	2	16
005.6.txt	a, b	2, 1, 3, 2, 3, 4, 2, 1, 4, 4	2, 3	4	49	2	13
005.7.txt	a, b	1, 3, 4, 0, 2, 3, 2, 3, 0, 1	0, 1, 3, 4	4	54	4	52
005.8.txt	a, b	1, 0, 4, 0, 1, 3, 2, 1, 4, 2	0, 2, 3	2	34	2	31
005.9.txt	a, b	1, 4, 2, 4, 3, 3, 3, 1, 4, 4	0, 2	3	39	2	10
005.2.txt	a, b	3, 3, 2, 4, 0, 1, 2, 1, 1, 2	1	2	23	1	7

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
006.0.txt	a, b	3, 4, 4, 2, 3, 4, 1, 5, 3, 1, 1, 5	0, 1, 5	3	44	3	25
006.1.txt	a, b	1, 4, 0, 2, 5, 0, 2, 2, 0, 3, 1, 3	0, 4, 5	3	31	3	22
006.2.txt	a, b	0, 3, 5, 1, 4, 1, 1, 0, 0, 0, 2, 5	2	3	71	1	8
006.3.txt	a, b	3, 3, 0, 5, 2, 3, 5, 4, 3, 1, 2, 2	0, 1, 3, 5	4	44	3	28
006.4.txt	a, b	4, 3, 1, 0, 2, 1, 5, 4, 1, 2, 1, 0	5	4	77	1	8
006.5.txt	a, b	4, 3, 3, 0, 5, 1, 3, 4, 5, 3, 2, 0	0, 2, 3, 4, 5	3	71	3	69
006.6.txt	a, b	1, 1, 5, 3, 0, 5, 2, 4, 0, 2, 0, 0	1, 3	4	71	3	36
006.7.txt	a, b	5, 5, 3, 4, 0, 2, 1, 2, 0, 3, 4, 4	5	4	83	2	24
006.8.txt	a, b	0, 1, 5, 3, 1, 2, 0, 0, 5, 5, 4, 2	2	4	71	2	16
006.9.txt	a, b	3, 5, 3, 1, 4, 4, 0, 3, 1, 1, 5, 2	1, 3	3	31	2	20
006.10.txt	a, b	2, 1, 1, 3, 4, 3, 2, 5, 4, 0, 4, 3	4, 5	3	34	2	21
007.0.txt	a, b	4, 0, 1, 6, 2, 2, 1, 0, 5, 2, 5, 6, 1, 3	1, 2, 3	4	71	2	20
007.1.txt	a, b	3, 1, 5, 5, 3, 0, 4, 6, 6, 4, 2, 6, 5, 4	0, 1, 4, 5	3	49	3	32
007.2.txt	a, b	6, 5, 0, 6, 6, 6, 6, 1, 2, 2, 0, 4, 3, 5	0, 3, 4, 5, 6	3	39	3	37
007.3.txt	a, b	1, 0, 6, 4, 2, 0, 3, 6, 5, 2, 3, 0, 5, 5	0, 6	5	89	3	38
007.4.txt	a, b	4, 3, 5, 3, 3, 3, 3, 5, 3, 6, 3, 1, 1, 2	3	4	90	3	40
007.5.txt	a, b	3, 6, 5, 4, 6, 5, 2, 1, 6, 3, 5, 5, 5, 4	0, 1, 2, 3, 4, 5	4	83	4	77
007.6.txt	a, b	1, 0, 2, 5, 5, 4, 6, 6, 3, 5, 4, 2, 5, 3	0, 1, 3	4	65	3	23
007.7.txt	a, b	2, 6, 5, 5, 3, 3, 0, 3, 0, 3, 2, 4, 1, 6	0, 4	3	53	2	14
007.8.txt	a, b	6, 4, 5, 2, 6, 2, 4, 2, 1, 1, 4, 3, 0, 5	1, 3, 4, 5	4	83	4	56
007.9.txt	a, b	4, 1, 5, 3, 6, 5, 2, 5, 0, 5, 5, 5, 1, 2	1, 3, 5	3	59	3	35
007.10.txt	a, b	5, 6, 1, 0, 6, 1, 3, 4, 3, 2, 4, 6, 6, 3	4	4	69	1	9
008.0.txt	a, b	2, 1, 1, 6, 5, 4, 3, 1, 3, 3, 7, 4, 5, 1, 5, 3	1, 3, 4, 5, 7	4	90	3	80
008.1.txt	a, b	2, 0, 7, 0, 6, 4, 2, 7, 1, 6, 0, 2, 5, 2, 3, 4	1, 3, 5, 6, 7	5	83	4	45
008.2.txt	a, b	6, 5, 4, 3, 5, 7, 3, 0, 6, 7, 6, 2, 1, 7, 4, 0	1, 2, 3, 4	3	44	3	35
008.3.txt	a, b	1, 7, 3, 6, 4, 2, 0, 3, 5, 7, 7, 2, 7, 1, 0, 4	0, 2, 3, 4	4	59	3	28
008.4.txt	a, b	7, 4, 3, 4, 1, 1, 7, 6, 7, 7, 2, 1, 0, 5, 0, 3	0, 1, 6	4	65	3	30
008.5.txt	a, b	1, 5, 5, 5, 4, 3, 7, 7, 7, 1, 1, 7, 2, 3, 4, 6	2, 3, 4, 5, 6, 7	5	111	4	98
008.6.txt	a, b	1, 6, 0, 4, 5, 6, 1, 2, 0, 7, 0, 6, 6, 2, 5, 3	1, 5, 6, 7	4	77	3	45

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
008.7.txt	a, b	3, 4, 5, 6, 1, 3, 2, 7, 1, 4, 3, 4, 7, 2, 5, 3	0, 1, 4, 6	4	77	2	16
008.8.txt	a, b	5, 3, 1, 7, 6, 4, 3, 3, 1, 5, 4, 2, 4, 5, 4, 5	0, 3, 6	3	69	2	21
008.9.txt	a, b	0, 1, 5, 5, 0, 7, 2, 6, 2, 4, 4, 3, 0, 4, 0, 3	2, 3	4	90	2	18
008.10.txt	a, b	4, 4, 5, 5, 7, 4, 2, 5, 1, 6, 1, 0, 1, 3, 4, 6	0, 1, 2, 4	4	71	4	57
009.0.txt	a, b	1, 3, 2, 7, 0, 4, 8, 7, 2, 2, 7, 3, 8, 1, 5, 6, 0, 5	1, 2, 3, 4, 5, 6, 7	5	125	5	116
009.1.txt	a, b	1, 5, 1, 4, 7, 6, 1, 1, 8, 1, 1, 4, 8, 3, 4, 3, 2, 8	5, 6, 7, 8	4	83	2	26
009.2.txt	a, b	6, 8, 6, 4, 4, 7, 5, 1, 4, 3, 8, 4, 8, 7, 5, 6, 0, 2	5, 8	3	76	3	46
009.3.txt	a, b	1, 4, 5, 4, 3, 3, 6, 7, 4, 4, 6, 7, 8, 5, 2, 4, 8, 4	0, 1, 3, 4, 6, 7	4	59	4	58
009.4.txt	a, b	6, 0, 2, 5, 3, 5, 4, 1, 7, 3, 1, 0, 8, 7, 8, 4, 3, 0	0, 1, 4, 5, 6	4	83	3	40
009.5.txt	a, b	8, 2, 7, 8, 3, 5, 0, 4, 0, 1, 0, 8, 5, 5, 2, 6, 7, 3	0, 1, 2, 3, 5	4	77	3	46
009.6.txt	a, b	5, 6, 7, 7, 1, 0, 1, 8, 3, 0, 4, 6, 8, 2, 0, 0, 6, 6	2, 7	4	95	2	20
009.7.txt	a, b	4, 1, 6, 8, 7, 2, 5, 6, 8, 0, 8, 4, 3, 4, 6, 2, 7, 3	1, 2, 3, 4, 5, 7	5	97	5	83
009.8.txt	a, b	3, 3, 1, 3, 7, 4, 7, 1, 4, 8, 4, 0, 5, 3, 3, 6, 2, 8	3, 4, 5, 6	4	59	2	37
009.9.txt	a, b	0, 6, 8, 5, 7, 6, 4, 2, 3, 3, 1, 8, 3, 0, 1, 5, 4, 4	1, 3, 6, 8	3	49	3	36
009.10.txt	a, b	0, 1, 3, 3, 4, 0, 6, 0, 7, 4, 0, 4, 2, 8, 5, 5, 3, 6	1, 3, 4, 8	4	77	3	31
010.0.txt	a, b	6, 8, 6, 0, 9, 9, 0, 7, 2, 1, 8, 3, 8, 4, 9, 5, 9, 4, 3, 7	0, 1, 2, 3, 4, 6, 7, 8, 9	2	76	2	75
010.1.txt	a, b	0, 7, 5, 9, 8, 4, 6, 1, 6, 6, 8, 7, 8, 9, 8, 3, 5, 0, 2, 6	0, 2, 8	3	65	3	60
010.2.txt	a, b	1, 7, 7, 4, 4, 9, 7, 3, 3, 2, 6, 4, 8, 0, 5, 5, 9, 5, 3, 7	1	6	269	1	12
010.3.txt	a, b	0, 2, 8, 2, 5, 7, 3, 8, 2, 0, 9, 4, 0, 3, 7, 1, 6, 1, 1, 3	1, 3, 5	4	90	3	34
010.4.txt	a, b	0, 1, 0, 5, 3, 7, 8, 9, 9, 3, 2, 0, 4, 2, 3, 9, 8, 0, 1, 6	1, 3, 9	4	95	4	55
010.5.txt	a, b	5, 9, 6, 7, 3, 0, 8, 4, 4, 0, 1, 0, 6, 4, 2, 7, 9, 9, 3, 8	0, 3, 4, 6, 7, 8	6	143	3	59
010.6.txt	a, b	8, 7, 5, 3, 4, 4, 9, 8, 2, 1, 7, 2, 4, 9, 3, 2, 6, 3, 2, 8	3	5	189	2	32
010.7.txt	a, b	2, 7, 3, 8, 6, 9, 7, 2, 0, 5, 5, 7, 2, 6, 6, 1, 9, 7, 4, 5	4, 9	6	161	3	35
010.8.txt	a, b	7, 3, 9, 7, 1, 6, 8, 1, 4, 5, 1, 2, 4, 3, 0, 9, 0, 5, 4, 9	0, 2, 3, 6, 9	5	101	5	76
010.9.txt	a, b	9, 3, 6, 1, 5, 4, 1, 0, 7, 9, 8, 7, 0, 4, 6, 2, 2, 3, 7, 0	0, 4, 7	4	97	2	36
010.10.txt	a, b	5, 0, 1, 2, 9, 7, 3, 1, 0, 5, 0, 3, 9, 8, 2, 8, 6, 6, 3, 4	1	6	229	2	38
011.0.txt	a, b	3, 2, 1, 10, 9, 5, 10, 1, 3, 10, 2, 3, 1, 0, 8, 2, 3, 4, 4, 3, 7, 6	8	5	229	2	30
011.1.txt	a, b	7, 1, 1, 4, 5, 6, 6, 1, 7, 10, 9, 8, 1, 5, 3, 7, 5, 0, 2, 9, 5, 3	2, 3, 5, 6, 7, 8, 9, 10	5	161	6	124

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
011.2.txt	a, b	8, 9, 0, 5, 5, 7, 10, 6, 4, 2, 7, 6, 5, 6, 4, 1, 8, 10, 10, 10, 0, 3	1, 3, 6, 8	5	135	2	25
011.3.txt	a, b	2, 1, 0, 5, 7, 0, 4, 0, 4, 8, 0, 6, 9, 10, 7, 4, 3, 9, 0, 7, 6, 8	4, 7	6	170	3	42
011.4.txt	a, b	8, 4, 6, 2, 10, 3, 4, 1, 5, 0, 5, 9, 1, 7, 4, 5, 3, 4, 0, 5, 2, 4	0, 2, 5, 8, 9, 10	5	104	4	62
011.5.txt	a, b	7, 4, 3, 2, 7, 8, 3, 4, 2, 9, 7, 2, 8, 8, 3, 0, 5, 10, 6, 1, 8, 5	0, 1, 2, 6, 8, 9, 10	4	111	3	61
011.6.txt	a, b	3, 2, 5, 8, 10, 1, 6, 2, 5, 9, 0, 7, 4, 10, 1, 3, 7, 7, 2, 9, 9, 3	1, 2, 10	4	83	3	32
011.7.txt	a, b	9, 4, 5, 7, 7, 5, 0, 3, 3, 10, 3, 8, 2, 8, 6, 4, 9, 1, 6, 1, 0, 6	0, 4, 7, 8, 10	5	143	4	66
011.8.txt	a, b	7, 5, 1, 8, 6, 3, 5, 3, 7, 4, 2, 6, 1, 7, 9, 9, 1, 7, 10, 1, 8, 4	7, 8, 9, 10	5	118	3	39
011.9.txt	a, b	2, 4, 2, 9, 10, 2, 2, 2, 0, 7, 0, 1, 3, 6, 8, 1, 5, 6, 5, 8, 3, 0	0, 1, 2, 4, 5, 6, 7, 8, 9, 10	5	199	5	176
011.10.txt	a, b	6, 0, 7, 3, 7, 0, 9, 0, 10, 1, 10, 0, 4, 2, 3, 8, 0, 2, 4, 10, 5, 6	1, 2, 3, 7, 8, 9, 10	3	77	4	68
012.0.txt	a, b	10, 5, 3, 11, 7, 5, 0, 0, 5, 3, 2, 1, 7, 4, 7, 11, 5, 11, 6, 8, 5, 9, 6, 10	0, 1, 2, 4, 6, 8, 9, 11	5	167	5	94
012.1.txt	a, b	10, 4, 2, 2, 0, 10, 3, 7, 2, 3, 7, 11, 2, 11, 8, 8, 1, 11, 2, 5, 9, 8, 6, 11	9	5	186	3	60
012.2.txt	a, b	2, 8, 1, 9, 11, 7, 7, 7, 0, 1, 6, 0, 10, 2, 7, 10, 4, 3, 10, 8, 9, 8, 11, 5	1, 2, 3, 9	6	215	3	55
012.3.txt	a, b	4, 7, 9, 5, 0, 1, 9, 2, 6, 10, 8, 5, 7, 9, 4, 11, 3, 8, 0, 5, 4, 10, 2, 3	2, 9	5	161	3	49
012.4.txt	a, b	7, 5, 5, 6, 6, 10, 8, 11, 7, 8, 2, 4, 0, 10, 9, 10, 10, 0, 1, 8, 0, 3, 3, 4	1, 9	6	197	4	63
012.5.txt	a, b	4, 8, 2, 6, 4, 11, 0, 4, 11, 9, 1, 7, 5, 11, 3, 10, 3, 7, 4, 10, 8, 4, 4, 1	1, 2, 3, 4, 5, 8, 10, 11	5	188	5	178
012.6.txt	a, b	2, 0, 3, 11, 8, 2, 2, 6, 0, 11, 7, 6, 6, 10, 7, 9, 5, 2, 4, 11, 1, 5, 2, 3	1, 3, 6, 7, 9, 10	5	119	3	51
012.7.txt	a, b	10, 10, 2, 3, 5, 3, 3, 8, 11, 10, 6, 4, 2, 4, 1, 3, 3, 9, 7, 9, 4, 3, 10, 8	0, 6, 7, 9, 10	5	170	3	43
012.8.txt	a, b	9, 5, 7, 11, 6, 7, 2, 1, 8, 1, 6, 8, 5, 5, 10, 11, 11, 2, 4, 0, 10, 8, 3, 11	5, 9, 10	4	118	4	43
012.9.txt	a, b	0, 7, 8, 10, 5, 6, 6, 11, 2, 5, 9, 1, 0, 11, 4, 5, 3, 1, 11, 11, 1, 6, 8, 8	0, 3, 4, 5, 7, 9	4	90	3	44

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
012.10.txt	a, b	3, 0, 11, 10, 5, 10, 10, 10, 6, 11, 7, 3, 1, 1, 2, 8, 6, 1, 2, 2, 9, 3, 3, 4	1, 6, 9	6	143	3	39
013.0.txt	a, b	1, 5, 6, 10, 4, 12, 0, 3, 7, 5, 6, 8, 10, 9, 2, 1, 4, 6, 3, 4, 4, 3, 12, 5, 11, 7	0, 1, 2, 7, 8, 12	6	175	6	147
013.1.txt	a, b	5, 1, 4, 12, 2, 10, 8, 0, 7, 9, 5, 11, 11, 0, 1, 6, 5, 5, 2, 3, 10, 11, 5, 6, 1, 7	0, 5, 9, 11	5	132	4	67
013.2.txt	a, b	8, 7, 12, 0, 4, 0, 10, 4, 1, 1, 3, 6, 0, 11, 9, 4, 2, 1, 10, 5, 5, 3, 6, 0, 8, 3	1, 2, 4, 5, 6, 7, 8, 10, 11, 12	5	101	5	96
013.3.txt	a, b	11, 2, 12, 2, 9, 4, 6, 9, 0, 7, 8, 3, 0, 5, 4, 0, 11, 5, 1, 11, 2, 4, 6, 10, 9, 7	0, 1, 3, 5, 7, 9	4	101	4	74
013.4.txt	a, b	8, 1, 10, 8, 0, 12, 10, 0, 7, 9, 5, 5, 6, 8, 3, 2, 7, 0, 7, 12, 5, 11, 4, 6, 5, 4	0, 5, 6, 8, 9, 12	5	159	5	115
013.5.txt	a, b	10, 9, 5, 10, 10, 12, 6, 7, 11, 5, 8, 2, 12, 0, 6, 4, 1, 2, 5, 7, 0, 3, 1, 10, 2, 6	3, 7, 8, 10, 12	5	118	3	64
013.6.txt	a, b	2, 7, 8, 11, 2, 9, 10, 9, 6, 9, 12, 11, 6, 3, 5, 6, 12, 8, 3, 1, 11, 4, 12, 7, 5, 12	0, 1, 2, 4, 7, 9, 10, 11, 12	5	151	5	123
013.7.txt	a, b	11, 12, 8, 10, 12, 9, 7, 1, 7, 1, 3, 11, 1, 5, 0, 11, 2, 6, 9, 1, 11, 4, 0, 0, 1, 1	0, 1, 2, 3, 6, 7, 8, 10, 11	5	167	4	97
013.8.txt	a, b	8, 4, 6, 12, 4, 1, 12, 9, 2, 10, 8, 11, 8, 0, 0, 9, 6, 3, 1, 2, 11, 11, 1, 5, 7, 1	0, 1	5	189	3	47
013.9.txt	a, b	3, 0, 3, 8, 2, 3, 1, 11, 1, 2, 6, 10, 3, 10, 12, 5, 1, 9, 3, 6, 2, 4, 5, 7, 5, 10	11	6	263	3	47
013.10.txt	a, b	0, 5, 11, 4, 7, 7, 2, 7, 2, 8, 10, 3, 12, 3, 5, 8, 1, 3, 5, 9, 9, 6, 9, 10, 7, 7	3, 4, 5, 6, 9, 10, 11, 12	4	113	5	122
014.0.txt	a, b	11, 0, 12, 8, 0, 10, 8, 2, 10, 0, 11, 4, 7, 7, 5, 1, 3, 0, 6, 6, 4, 12, 4, 6, 13, 5, 9, 9	0, 1, 2, 4, 5, 6, 7, 8, 10, 11, 12, 13	4	208	4	204
014.1.txt	a, b	3, 12, 1, 1, 12, 5, 3, 7, 10, 11, 0, 8, 2, 1, 4, 10, 9, 8, 11, 6, 13, 9, 7, 11, 0, 0, 3, 13	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12	8	615	8	607
014.2.txt	a, b	8, 3, 4, 13, 0, 9, 6, 1, 1, 2, 3, 1, 11, 5, 10, 4, 7, 0, 11, 7, 4, 0, 3, 9, 12, 11, 12, 8	3, 5, 8	5	143	2	38
014.3.txt	a, b	5, 0, 3, 6, 0, 1, 8, 3, 8, 11, 4, 13, 7, 2, 9, 10, 6, 11, 5, 12, 9, 1, 12, 1, 6, 11, 3, 1	0, 4, 9, 12	6	209	5	106
014.4.txt	a, b	9, 3, 10, 8, 6, 4, 11, 0, 3, 6, 1, 13, 5, 12, 8, 4, 13, 1, 2, 13, 12, 11, 6, 0, 4, 8, 6, 7	2, 6, 7, 8	5	143	2	49
014.5.txt	a, b	8, 5, 6, 7, 2, 11, 10, 6, 2, 8, 11, 11, 2, 4, 3, 3, 13, 9, 5, 12, 9, 1, 6, 1, 9, 4, 10, 3	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12	6	337	6	334
014.6.txt	a, b	6, 10, 6, 6, 0, 8, 11, 8, 2, 3, 6, 13, 11, 11, 4, 2, 5, 7, 1, 1, 9, 12, 6, 2, 4, 3, 7, 0	1, 5, 10, 11, 13	5	199	3	55

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
014.7.txt	a, b	4, 11, 7, 7, 5, 9, 8, 5, 6, 4, 0, 13, 10, 0, 9, 1, 2, 6, 8, 12, 3, 0, 10, 4, 9, 3, 13, 7	0, 4, 7, 11	5	170	4	71
014.8.txt	a, b	5, 1, 7, 5, 4, 2, 5, 0, 1, 2, 3, 11, 1, 12, 11, 10, 9, 1, 1, 8, 3, 9, 13, 6, 2, 10, 8, 9	0, 2, 3, 5, 6, 7, 8, 12	5	188	4	82
014.9.txt	a, b	2, 10, 4, 8, 10, 13, 10, 2, 9, 11, 0, 13, 11, 4, 12, 1, 12, 10, 3, 5, 4, 4, 2, 13, 2, 6, 3, 7	3, 4, 11, 12, 13	6	167	4	81
014.10.txt	a, b	2, 0, 2, 6, 9, 10, 10, 10, 12, 11, 13, 0, 1, 5, 11, 6, 0, 7, 6, 1, 5, 4, 3, 8, 12, 5, 0, 0	6, 13	6	199	3	42
015.0.txt	a, b	3, 1, 14, 10, 8, 8, 4, 7, 12, 3, 0, 3, 1, 6, 9, 5, 6, 14, 7, 3, 3, 11, 13, 6, 2, 2, 14, 8, 11, 9	0, 3, 4, 5, 7, 8, 14	3	95	3	81
015.1.txt	a, b	6, 6, 5, 11, 6, 14, 12, 4, 7, 9, 12, 8, 3, 6, 13, 1, 4, 4, 0, 11, 7, 12, 6, 10, 14, 7, 2, 6, 2, 8	0, 3, 4, 5, 6, 7, 8, 9, 11, 12	6	219	6	182
015.2.txt	a, b	9, 1, 8, 5, 6, 14, 11, 14, 6, 7, 7, 12, 8, 10, 7, 8, 0, 3, 3, 2, 5, 13, 3, 4, 8, 8, 9, 1, 10, 6	0, 1, 2, 3, 4, 6, 8, 9, 10, 12, 14	8	319	8	293
015.3.txt	a, b	2, 11, 0, 14, 3, 10, 6, 12, 9, 3, 8, 2, 9, 12, 8, 0, 11, 11, 4, 4, 6, 7, 13, 1, 3, 11, 5, 8, 12, 13	1, 2, 4, 5, 7, 11, 13, 14	4	101	4	77
015.4.txt	a, b	9, 4, 3, 14, 13, 9, 12, 8, 3, 8, 1, 2, 11, 7, 11, 10, 1, 6, 4, 14, 9, 13, 12, 12, 3, 4, 6, 6, 6, 5	9	6	389	2	34
015.5.txt	a, b	11, 10, 12, 1, 1, 10, 4, 2, 7, 9, 11, 1, 10, 14, 11, 8, 13, 10, 14, 14, 6, 2, 5, 11, 8, 13, 10, 7, 5, 3	1, 3, 8	6	224	3	48
015.6.txt	a, b	7, 5, 4, 8, 10, 10, 2, 0, 13, 12, 7, 1, 7, 13, 8, 14, 4, 12, 11, 14, 6, 14, 7, 9, 9, 13, 1, 7, 3, 2	1, 2, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14	5	259	5	248
015.7.txt	a, b	5, 14, 0, 14, 12, 4, 0, 2, 8, 9, 10, 5, 7, 11, 8, 13, 6, 9, 9, 3, 3, 0, 1, 0, 3, 9, 7, 9, 7, 0	2, 3, 4, 5, 6, 7, 8, 9, 12	4	167	4	153
015.8.txt	a, b	1, 3, 14, 2, 13, 6, 7, 4, 9, 5, 0, 2, 14, 1, 9, 9, 4, 4, 5, 14, 4, 11, 1, 14, 1, 7, 12, 8, 2, 10	0, 7, 9, 12	4	135	5	88
015.9.txt	a, b	10, 2, 6, 0, 10, 5, 5, 10, 6, 13, 9, 4, 14, 11, 12, 5, 0, 14, 12, 3, 13, 1, 1, 3, 5, 8, 1, 7, 6, 2	0, 1, 2, 3, 5, 7, 8, 11, 12	5	167	4	80
015.10.txt	a, b	7, 11, 2, 9, 10, 14, 13, 2, 11, 1, 2, 5, 12, 6, 11, 3, 5, 6, 1, 0, 13, 1, 14, 8, 4, 3, 9, 4, 12, 13	0, 2, 7, 8, 11, 13, 14	6	183	5	100

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
016.0.txt	a, b	7, 3, 2, 11, 10, 9, 1, 1, 10, 10, 3, 0, 2, 15, 6, 7, 14, 13, 13, 4, 13, 0, 4, 11, 8, 11, 0, 12, 0, 13, 1, 5	5, 7, 9, 10, 15	7	383	6	204
016.1.txt	a, b	11, 1, 14, 2, 8, 13, 9, 4, 4, 3, 5, 1, 12, 9, 6, 14, 15, 3, 5, 15, 7, 4, 4, 12, 0, 6, 11, 10, 14, 12, 9, 1	0, 1, 11	7	285	4	62
016.2.txt	a, b	10, 11, 13, 8, 7, 1, 9, 6, 10, 8, 2, 1, 8, 14, 1, 5, 9, 3, 10, 13, 14, 1, 9, 4, 7, 2, 8, 11, 15, 2, 3, 12	2, 4, 6, 7, 8, 11, 12, 14	5	167	3	74
016.3.txt	a, b	14, 14, 9, 8, 8, 6, 7, 11, 13, 15, 5, 13, 3, 7, 6, 12, 10, 2, 8, 15, 3, 10, 4, 5, 8, 1, 15, 1, 7, 10, 15, 13	1, 3, 7, 8, 9, 10, 12, 14, 15	5	209	5	186
016.4.txt	a, b	1, 15, 4, 7, 3, 15, 6, 10, 14, 15, 7, 10, 5, 9, 13, 9, 3, 3, 8, 14, 5, 11, 4, 5, 2, 14, 15, 1, 8, 15, 12, 9	1, 6, 15	6	259	4	61
016.5.txt	a, b	8, 12, 7, 4, 14, 13, 14, 0, 14, 2, 9, 7, 7, 13, 1, 9, 10, 0, 0, 15, 3, 12, 10, 13, 2, 6, 2, 0, 5, 3, 11, 10	1, 3, 7, 8, 12, 14	5	242	3	85
016.6.txt	a, b	14, 15, 2, 7, 10, 7, 4, 2, 4, 0, 13, 4, 3, 5, 1, 12, 11, 13, 14, 2, 7, 3, 13, 13, 5, 2, 1, 8, 6, 7, 9, 8	12, 14	9	509	2	32
016.7.txt	a, b	10, 11, 3, 14, 5, 10, 2, 8, 3, 3, 10, 4, 13, 1, 6, 9, 2, 11, 0, 3, 11, 0, 7, 11, 15, 6, 3, 12, 11, 5, 8, 6	3, 13	9	415	4	53
016.8.txt	a, b	1, 14, 14, 7, 15, 5, 8, 9, 0, 9, 3, 5, 13, 13, 10, 8, 9, 2, 2, 10, 6, 10, 0, 4, 3, 11, 0, 14, 12, 13, 13, 7	0, 1, 2, 3, 4, 5, 8, 10, 15	6	167	4	99
016.9.txt	a, b	2, 3, 14, 12, 7, 7, 11, 5, 7, 2, 13, 13, 1, 4, 0, 8, 13, 2, 0, 6, 8, 13, 13, 1, 9, 8, 4, 15, 0, 1, 8, 10	0, 1, 3, 5, 6, 8, 11, 13, 14, 15	6	206	7	175
016.10.txt	a, b	13, 9, 7, 12, 15, 15, 9, 7, 15, 5, 10, 2, 8, 2, 10, 14, 11, 10, 1, 3, 13, 12, 12, 6, 15, 7, 11, 9, 3, 10, 4, 9	0, 2, 3, 4, 5, 6, 7, 8, 9, 11, 14	4	215	4	147
017.0.txt	a, b	8, 13, 14, 5, 5, 13, 9, 2, 15, 11, 5, 13, 10, 2, 15, 4, 3, 7, 12, 3, 10, 0, 6, 16, 3, 1, 15, 12, 12, 16, 15, 8, 12, 8	1, 2, 3, 4, 7, 8, 9, 10, 11, 12, 13, 15, 16	6	247	5	182
017.1.txt	a, b	8, 5, 16, 13, 16, 11, 11, 11, 6, 0, 15, 1, 8, 10, 6, 10, 9, 12, 9, 3, 4, 0, 12, 1, 16, 3, 8, 2, 5, 0, 2, 14, 7, 6	7	7	506	3	58
017.2.txt	a, b	14, 12, 6, 11, 9, 16, 11, 5, 15, 4, 10, 11, 15, 1, 7, 12, 16, 4, 12, 16, 6, 8, 1, 7, 3, 4, 4, 13, 1, 6, 9, 13, 15, 2	0, 1, 2, 4, 5, 6, 7, 8, 9, 13, 14, 15, 16	8	383	8	313

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
017.3.txt	a, b	3, 16, 12, 7, 5, 14, 1, 10, 11, 12, 8, 8, 16, 5, 8, 6, 4, 15, 16, 10, 2, 0, 9, 12, 3, 13, 6, 1, 14, 6, 1, 2, 2, 7	2, 7, 15	8	351	3	46
017.4.txt	a, b	10, 16, 15, 12, 2, 8, 4, 15, 12, 9, 9, 14, 7, 3, 8, 5, 11, 14, 13, 1, 6, 2, 10, 15, 10, 16, 15, 8, 0, 11, 14, 13, 16, 16	0, 4, 5, 10	7	285	4	61
017.5.txt	a, b	16, 12, 15, 6, 8, 13, 9, 4, 11, 15, 11, 9, 6, 13, 10, 14, 5, 10, 13, 7, 1, 10, 14, 14, 7, 7, 6, 11, 2, 7, 3, 6, 2, 8	0, 2, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16	7	351	7	329
017.6.txt	a, b	10, 3, 14, 6, 15, 14, 5, 5, 14, 12, 13, 11, 15, 3, 12, 13, 5, 2, 6, 7, 0, 4, 8, 0, 7, 12, 9, 1, 13, 16, 2, 12, 9, 5	1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15	5	239	5	224
017.7.txt	a, b	1, 1, 14, 16, 11, 12, 14, 8, 0, 1, 2, 16, 4, 11, 15, 9, 5, 7, 2, 8, 0, 8, 2, 11, 14, 1, 8, 3, 10, 9, 14, 6, 8, 13	4, 5, 7, 15	5	274	3	54
017.8.txt	a, b	14, 10, 5, 3, 7, 6, 9, 7, 9, 6, 0, 14, 9, 14, 0, 16, 2, 12, 0, 5, 16, 15, 1, 6, 8, 11, 4, 5, 3, 11, 12, 7, 16, 13	1, 5, 6, 7, 10, 11, 12, 13, 16	6	206	4	105
017.9.txt	a, b	2, 1, 9, 8, 6, 11, 16, 15, 14, 4, 7, 14, 10, 16, 4, 2, 9, 13, 5, 6, 13, 2, 10, 2, 6, 8, 11, 4, 4, 13, 12, 3, 3, 3	0, 1, 4, 5, 8, 9, 10, 11, 13, 15, 16	5	175	4	103
017.10.txt	a, b	6, 15, 15, 4, 7, 11, 6, 5, 10, 2, 7, 1, 5, 8, 5, 0, 3, 16, 11, 0, 12, 13, 9, 5, 4, 0, 8, 8, 16, 1, 14, 14, 4, 0	0, 4, 6, 7, 12, 13, 15	6	233	4	78
018.0.txt	a, b	4, 16, 14, 3, 8, 10, 12, 11, 6, 0, 11, 15, 1, 3, 16, 4, 16, 2, 10, 9, 5, 8, 17, 7, 11, 12, 9, 6, 14, 1, 16, 6, 3, 13, 16, 9	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 16, 17	10	629	10	608
018.1.txt	a, b	0, 1, 10, 0, 2, 4, 15, 9, 17, 1, 8, 3, 0, 14, 6, 2, 12, 11, 7, 5, 16, 9, 16, 13, 12, 7, 2, 5, 16, 14, 6, 15, 7, 11, 16, 11	0, 1, 2, 3, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15	7	359	7	305
018.2.txt	a, b	10, 8, 1, 2, 11, 10, 5, 12, 14, 17, 15, 8, 2, 17, 8, 11, 13, 6, 14, 1, 6, 4, 0, 4, 2, 2, 0, 14, 16, 3, 17, 1, 13, 11, 7, 9	10, 16	7	402	3	64
018.3.txt	a, b	14, 5, 14, 16, 2, 8, 13, 10, 4, 1, 0, 17, 8, 8, 15, 8, 3, 14, 10, 11, 10, 3, 14, 6, 16, 17, 4, 12, 7, 4, 0, 2, 9, 3, 1, 4	0, 2, 3, 4, 5, 6, 7, 8, 9, 14, 15, 16, 17	7	362	7	351
018.4.txt	a, b	11, 5, 17, 12, 7, 7, 9, 7, 1, 14, 16, 11, 8, 13, 10, 14, 16, 2, 7, 13, 13, 0, 12, 2, 9, 6, 11, 4, 8, 6, 13, 9, 16, 14, 15, 3	3, 5, 6, 7, 10, 11, 12, 14, 17	6	231	6	120

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
018.5.txt	a, b	4, 4, 17, 3, 3, 14, 15, 6, 11, 17, 4, 15, 9, 13, 14, 12, 8, 16, 10, 8, 11, 0, 7, 5, 6, 1, 2, 3, 8, 3, 0, 16, 4, 5, 6, 12	0, 2, 4, 6, 7	7	278	4	101
018.6.txt	a, b	2, 16, 14, 0, 15, 11, 6, 2, 12, 14, 16, 9, 10, 9, 8, 0, 10, 1, 11, 17, 5, 2, 4, 3, 16, 7, 10, 6, 10, 13, 16, 9, 6, 1, 2, 7	2, 4, 5, 7, 12, 13, 17	6	269	3	51
018.7.txt	a, b	11, 10, 1, 8, 11, 13, 13, 13, 8, 3, 4, 6, 14, 8, 15, 6, 1, 2, 15, 7, 1, 9, 12, 11, 3, 5, 13, 17, 5, 3, 2, 3, 11, 9, 11, 16	1, 10, 14	8	601	2	39
018.8.txt	a, b	0, 9, 4, 13, 14, 5, 17, 3, 13, 8, 15, 10, 5, 12, 14, 2, 6, 16, 2, 3, 8, 12, 7, 14, 11, 11, 10, 11, 13, 4, 8, 14, 7, 7, 10, 1	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 14, 15, 17	5	239	5	214
018.9.txt	a, b	14, 14, 13, 1, 9, 1, 17, 12, 7, 9, 15, 6, 8, 10, 6, 11, 2, 2, 10, 16, 12, 12, 13, 12, 15, 2, 15, 1, 15, 7, 3, 9, 4, 5, 3, 15	0, 4, 8, 12, 13, 14, 15, 17	7	323	4	107
018.10.txt	a, b	2, 3, 11, 9, 10, 17, 8, 9, 8, 17, 5, 14, 12, 15, 16, 2, 15, 4, 3, 1, 12, 6, 12, 12, 15, 17, 1, 10, 9, 7, 5, 13, 14, 12, 15, 12	0, 3, 4, 8, 9, 11, 17	6	233	4	75
019.0.txt	a, b	11, 15, 11, 3, 14, 16, 17, 3, 15, 18, 12, 2, 9, 12, 14, 3, 1, 15, 7, 16, 13, 4, 10, 8, 18, 13, 12, 16, 6, 5, 1, 8, 8, 11, 17, 14, 18, 2	3, 15	6	405	2	51
019.1.txt	a, b	5, 4, 11, 10, 11, 12, 9, 10, 2, 16, 6, 14, 11, 16, 12, 15, 5, 3, 5, 8, 13, 1, 14, 12, 17, 7, 18, 10, 10, 12, 9, 10, 17, 16, 16, 9, 6, 17	3, 5, 7, 10, 14	5	241	4	79
019.2.txt	a, b	10, 12, 13, 6, 14, 2, 8, 10, 5, 5, 11, 7, 18, 1, 7, 17, 10, 9, 15, 15, 11, 16, 14, 11, 6, 14, 4, 18, 6, 15, 17, 8, 16, 2, 1, 0, 3, 18	0, 3, 4, 6, 7, 9, 10, 11, 13, 14, 15, 17, 18	7	287	6	231
019.3.txt	a, b	3, 6, 17, 15, 7, 14, 2, 3, 1, 11, 10, 8, 10, 14, 1, 13, 1, 16, 4, 12, 7, 12, 14, 15, 5, 5, 16, 9, 18, 10, 1, 10, 0, 1, 0, 5, 10, 2	0, 3, 4, 5, 7, 8, 11, 12, 13, 15, 16, 17	6	263	6	212
019.4.txt	a, b	14, 7, 6, 3, 10, 12, 7, 9, 5, 10, 7, 3, 8, 18, 18, 12, 4, 11, 2, 0, 14, 17, 0, 14, 9, 1, 16, 7, 13, 11, 4, 9, 10, 2, 15, 3, 18, 15	0, 2, 5, 6, 8, 9, 10, 11, 15, 18	4	207	4	103
019.5.txt	a, b	17, 9, 16, 2, 17, 7, 17, 13, 5, 15, 1, 5, 14, 0, 0, 10, 18, 5, 10, 0, 6, 1, 17, 12, 7, 11, 13, 5, 8, 2, 15, 3, 12, 4, 5, 7, 4, 16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 15, 17	6	319	6	268

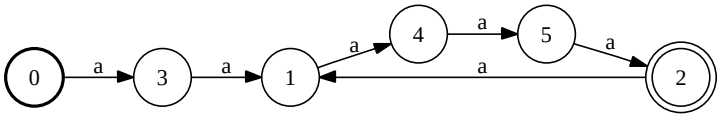
– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
019.6.txt	a, b	11, 18, 1, 4, 5, 16, 5, 15, 10, 6, 14, 0, 10, 2, 2, 13, 3, 9, 12, 4, 10, 7, 10, 13, 4, 16, 15, 6, 14, 0, 17, 15, 8, 11, 7, 1, 18, 9	1, 5	6	371	2	36
019.7.txt	a, b	1, 9, 6, 5, 11, 17, 16, 3, 4, 9, 9, 14, 14, 15, 18, 4, 12, 4, 8, 18, 4, 7, 0, 8, 6, 10, 2, 9, 6, 18, 4, 13, 9, 18, 14, 12, 0, 3	1, 2, 3, 6, 7, 8, 11, 12, 16	7	260	5	128
019.8.txt	a, b	1, 13, 8, 6, 18, 13, 0, 4, 11, 7, 11, 8, 10, 9, 14, 17, 5, 3, 9, 5, 17, 2, 8, 12, 4, 14, 8, 15, 10, 13, 3, 11, 6, 0, 16, 4, 18, 8	0, 1, 2, 4, 5, 6, 10, 11, 12, 13, 14, 15, 18	5	183	6	157
019.9.txt	a, b	2, 18, 4, 18, 8, 14, 17, 6, 15, 4, 14, 8, 10, 4, 9, 16, 5, 7, 14, 13, 3, 1, 5, 11, 6, 0, 5, 10, 5, 12, 9, 0, 9, 0, 3, 11, 13, 0	1, 3, 4, 5, 7, 8, 9, 10, 12, 13, 14, 15, 18	6	296	6	243
019.10.txt	a, b	14, 9, 13, 8, 2, 15, 11, 6, 12, 10, 5, 15, 5, 0, 3, 2, 17, 16, 18, 0, 1, 5, 4, 2, 4, 11, 15, 3, 3, 13, 3, 6, 7, 7, 1, 1, 17, 11	0, 1, 2, 4, 6, 8, 9, 10, 14, 15, 17, 18	7	233	5	115
020.0.txt	a, b	2, 4, 18, 16, 8, 19, 14, 17, 7, 18, 15, 13, 6, 1, 12, 12, 5, 0, 3, 16, 18, 3, 7, 7, 18, 11, 10, 6, 9, 19, 15, 16, 12, 15, 13, 0, 5, 2, 8, 3	2, 5, 6, 7, 8, 11, 12, 14, 15, 16	5	242	6	251
020.1.txt	a, b	4, 9, 12, 10, 16, 9, 13, 6, 4, 17, 0, 17, 13, 5, 2, 7, 17, 6, 5, 8, 15, 1, 10, 17, 19, 8, 7, 17, 4, 11, 18, 4, 14, 12, 17, 9, 13, 14, 11, 3	4	7	467	3	69
020.2.txt	a, b	17, 5, 19, 12, 11, 13, 18, 2, 5, 18, 10, 1, 13, 2, 3, 0, 7, 15, 6, 5, 19, 16, 7, 9, 14, 19, 16, 2, 4, 10, 7, 0, 6, 8, 14, 5, 18, 0, 16, 9	0, 1, 3, 4, 6, 7, 8, 10, 13, 14, 15, 18, 19	5	224	6	230
020.3.txt	a, b	15, 18, 7, 11, 9, 5, 19, 1, 13, 19, 3, 1, 2, 14, 12, 10, 15, 5, 4, 2, 11, 16, 11, 1, 18, 9, 5, 7, 10, 14, 2, 0, 16, 10, 11, 8, 6, 12, 17, 16	1, 2, 7, 9, 13, 15, 17	5	279	2	56
020.4.txt	a, b	10, 7, 9, 14, 8, 0, 16, 0, 8, 4, 8, 7, 18, 15, 15, 5, 19, 1, 12, 10, 5, 6, 9, 1, 17, 15, 12, 16, 2, 9, 4, 14, 11, 13, 1, 16, 7, 3, 14, 8	0, 1, 3, 5, 7, 10, 11, 12, 13, 16, 18, 19	6	269	5	157
020.5.txt	a, b	19, 16, 6, 13, 14, 9, 1, 10, 19, 15, 2, 1, 3, 1, 6, 0, 11, 7, 19, 12, 5, 8, 5, 15, 16, 17, 5, 1, 10, 11, 14, 7, 7, 13, 4, 13, 11, 19, 18, 3	11	9	854	2	46

– continued from previous page

Language description				L^*		LCA	
Id	Alphabet	Linear transition table	Final states	EQs	MQs	EQs	CQs
020.6.txt	a, b	13, 3, 7, 10, 0, 19, 8, 11, 17, 9, 16, 7, 9, 12, 17, 17, 14, 16, 11, 1, 13, 14, 10, 4, 15, 18, 5, 10, 2, 8, 0, 16, 15, 12, 12, 17, 15, 2, 6, 2	0, 1, 3, 4, 7, 8, 12, 14, 15, 16, 17, 19	6	215	5	184
020.7.txt	a, b	2, 8, 6, 16, 15, 7, 10, 12, 15, 18, 19, 2, 8, 3, 9, 1, 5, 10, 0, 11, 17, 14, 0, 18, 4, 19, 15, 7, 8, 8, 5, 7, 6, 4, 5, 18, 17, 7, 13, 11	0, 3, 4, 5, 6, 9, 10, 11, 12, 13, 14, 16, 17, 18, 19	7	314	5	182
020.8.txt	a, b	1, 9, 3, 4, 19, 8, 0, 9, 15, 16, 14, 8, 12, 7, 17, 14, 2, 6, 14, 12, 13, 18, 10, 1, 5, 0, 5, 8, 4, 14, 11, 14, 9, 13, 18, 10, 16, 17, 17, 17	0, 1, 4, 5, 9, 12, 13, 15, 19	6	231	6	167
020.9.txt	a, b	19, 16, 11, 17, 2, 11, 11, 10, 4, 7, 13, 4, 4, 9, 11, 4, 15, 19, 15, 12, 1, 14, 18, 18, 16, 3, 18, 2, 5, 1, 1, 6, 3, 17, 9, 11, 2, 12, 8, 7	3, 6, 7, 8, 9, 12	6	274	4	89
020.10.txt	a, b	8, 3, 3, 2, 13, 10, 18, 16, 15, 3, 3, 0, 7, 4, 19, 12, 11, 6, 14, 14, 3, 5, 1, 7, 12, 15, 18, 4, 10, 6, 13, 4, 15, 12, 19, 9, 11, 3, 18, 17	1, 2, 3, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19	5	269	4	189

Test 1. DFA **test set**

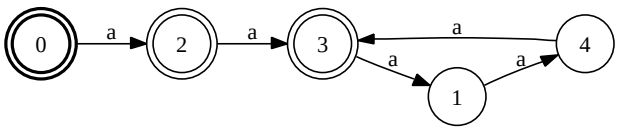


001M.fsm

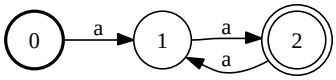
002M.fsm

003M.fsm

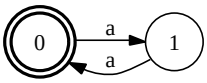
004M.fsm



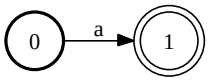
005M.fsm



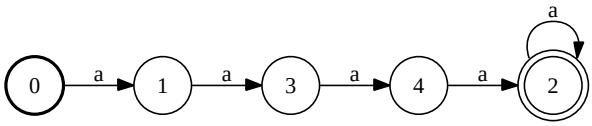
006M.fsm



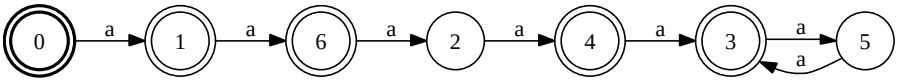
007M.fsm



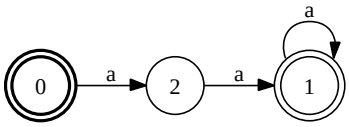
008M.fsm



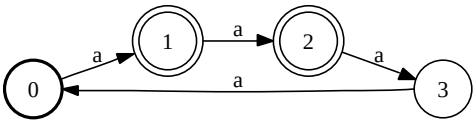
009M.fsm



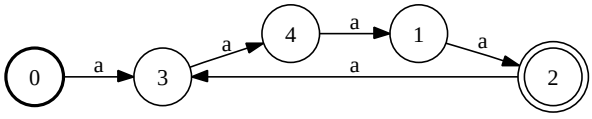
010M.fsm



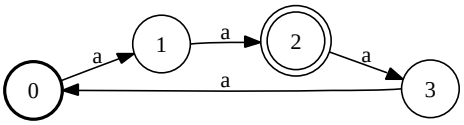
011M.fsm



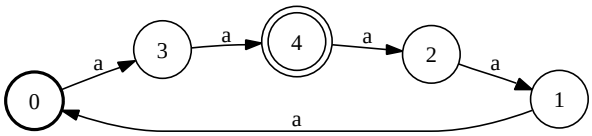
012M.fsm



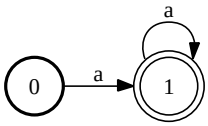
013M.fsm



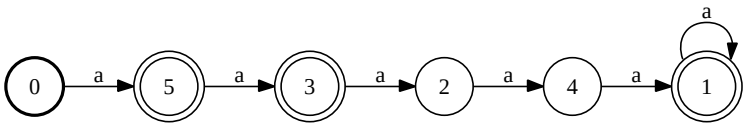
014M.fsm



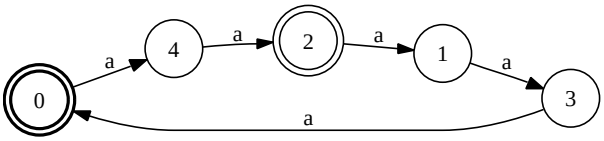
015M.fsm



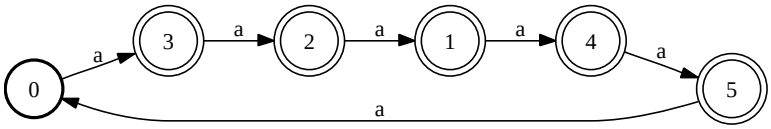
016M.fsm



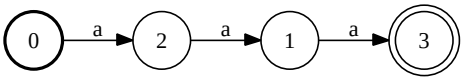
017M.fsm



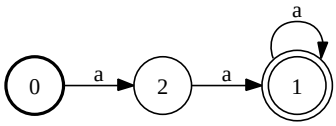
018M.fsm



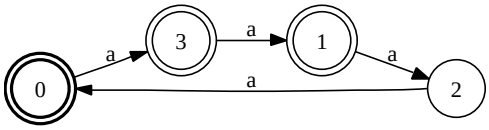
019M.fsm



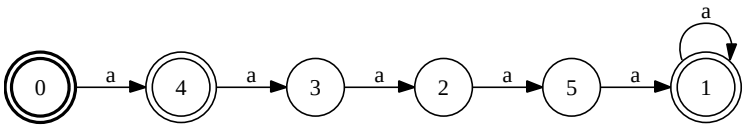
020M.fsm



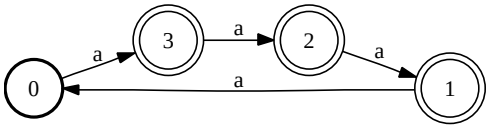
021M.fsm



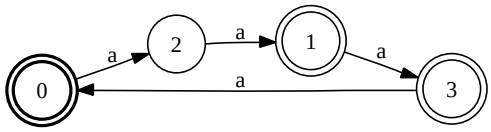
022M.fsm



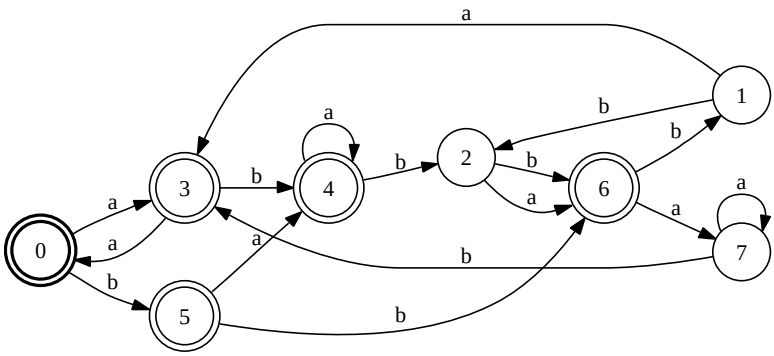
023M.fsm



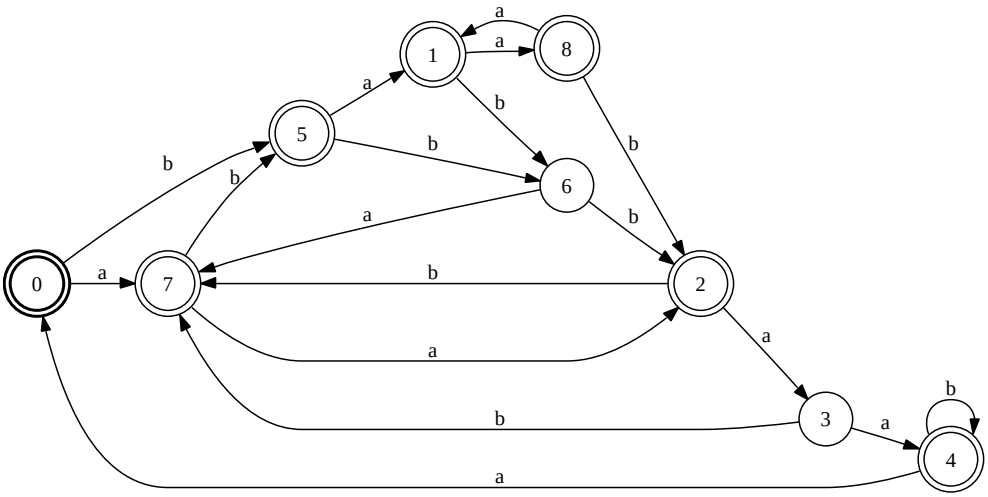
024M.fsm



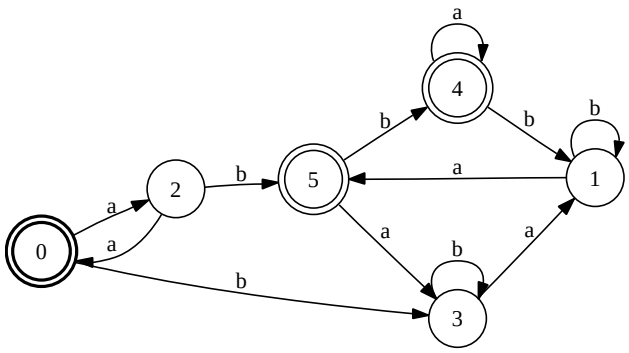
025M.fsm



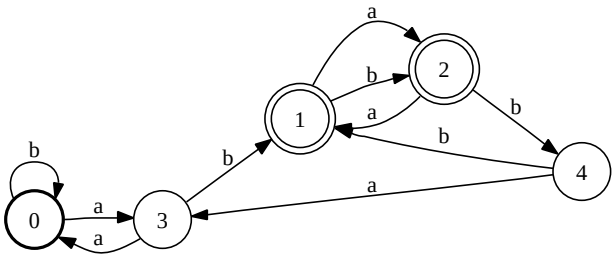
026M.fsm



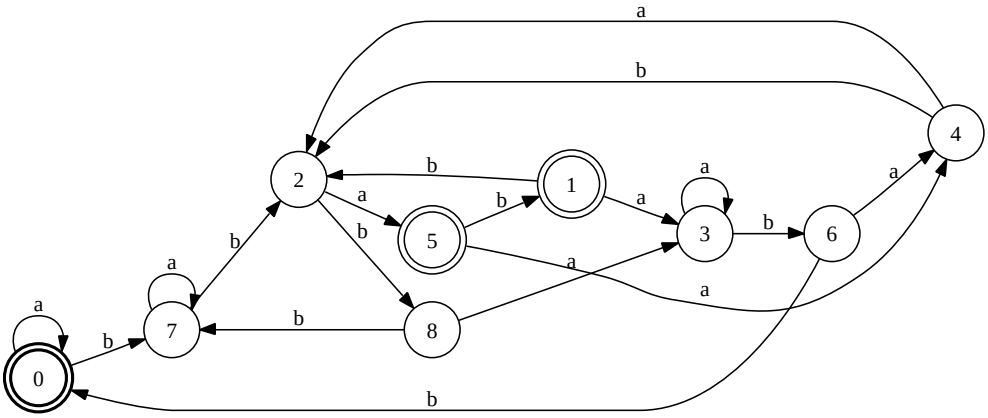
027M.fsm



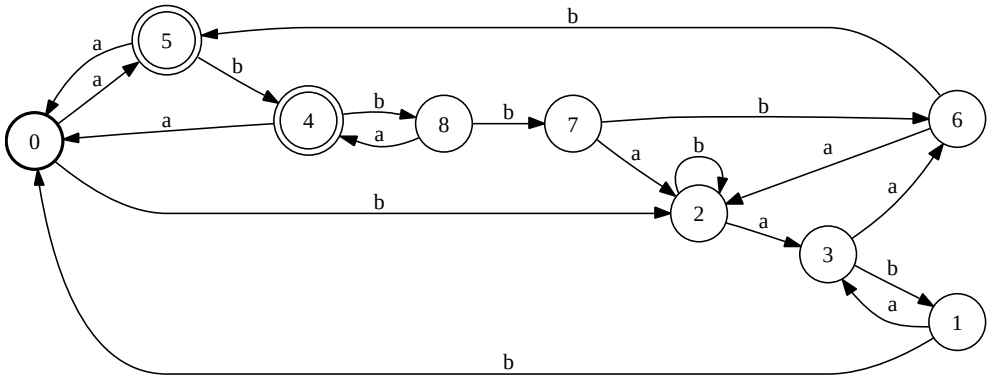
028M.fsm



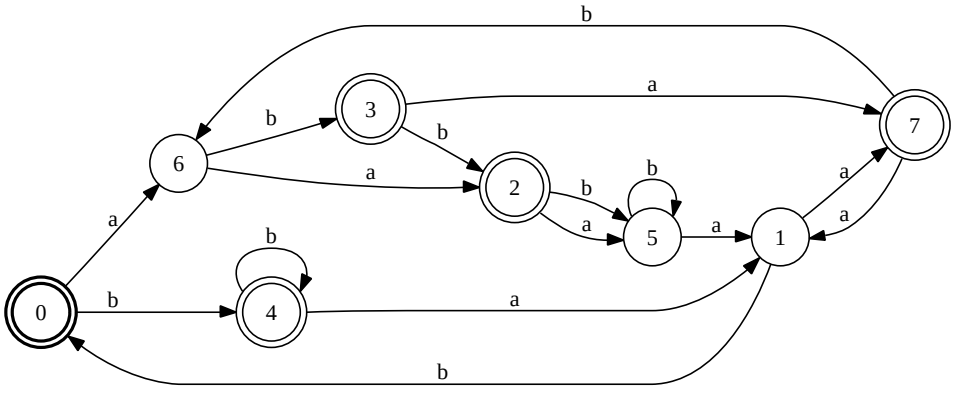
029M.fsm



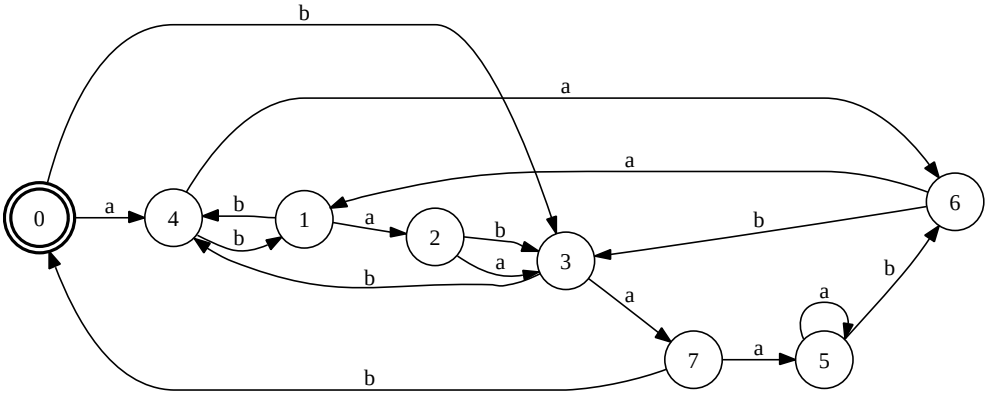
030M.fsm



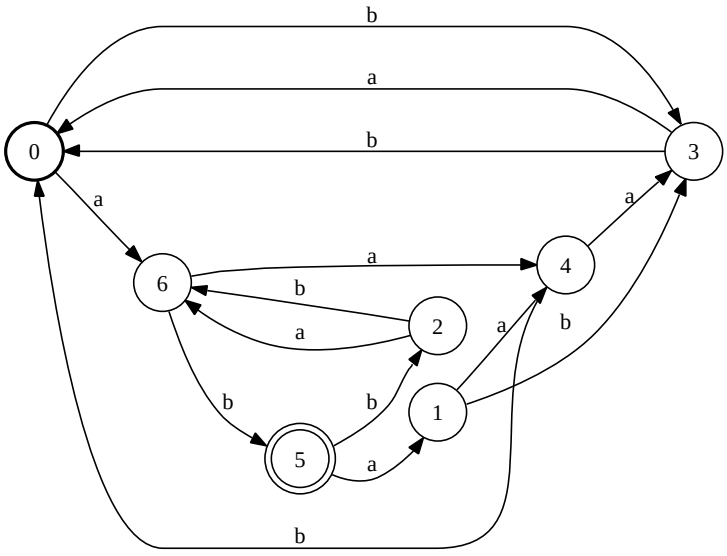
031M.fsm



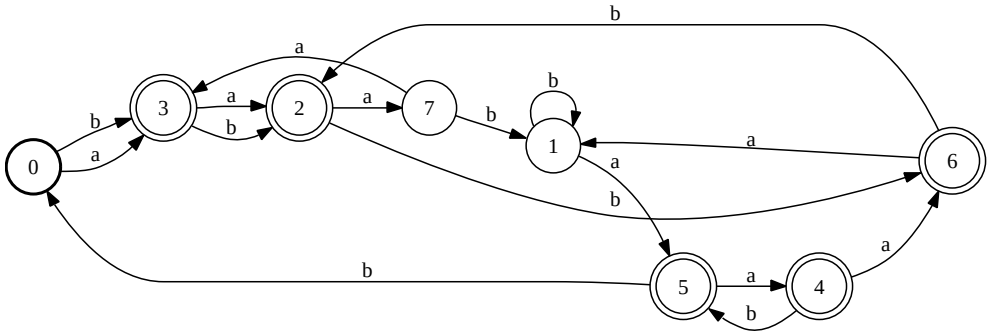
032M.fsm



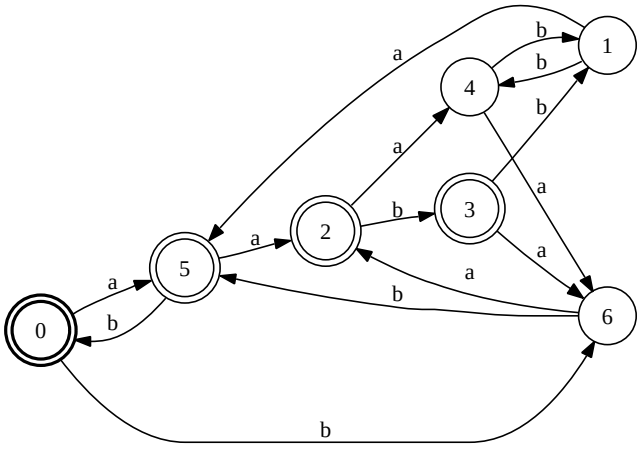
033M.fsm



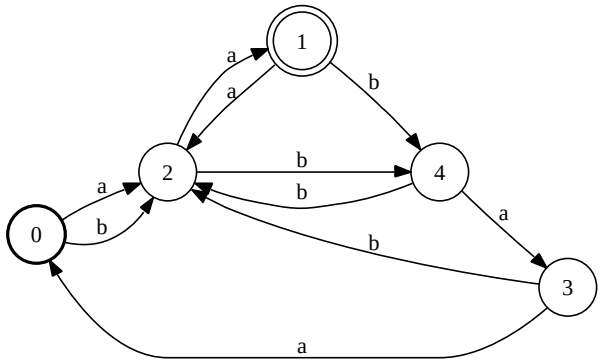
034M.fsm



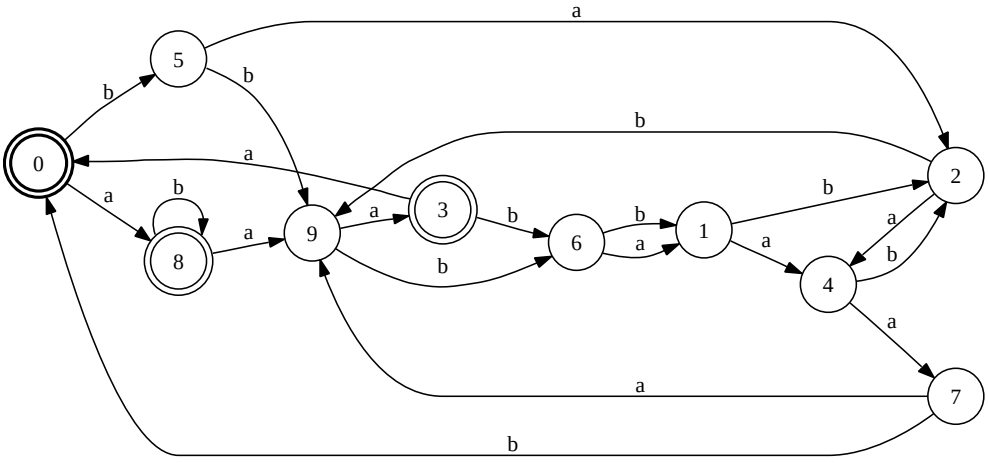
035M.fsm



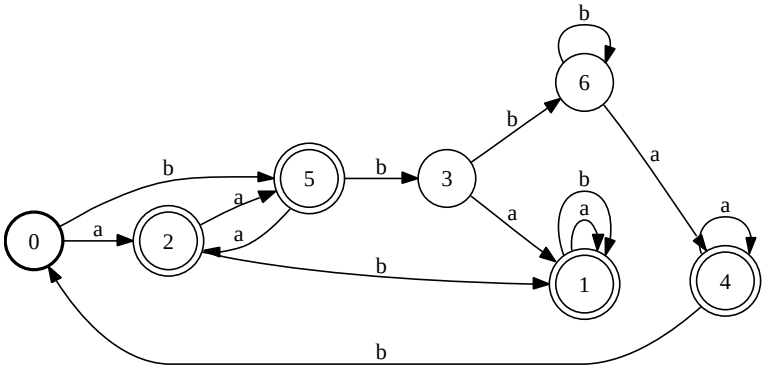
036M.fsm



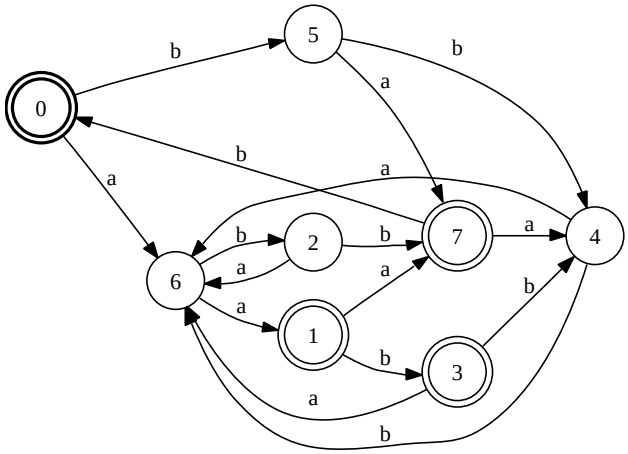
037M.fsm



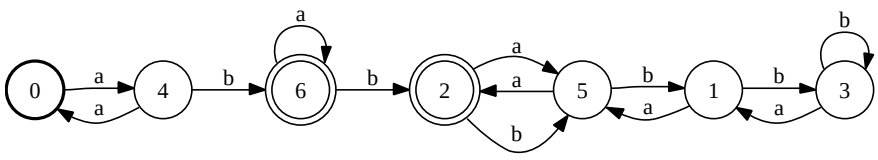
038M.fsm



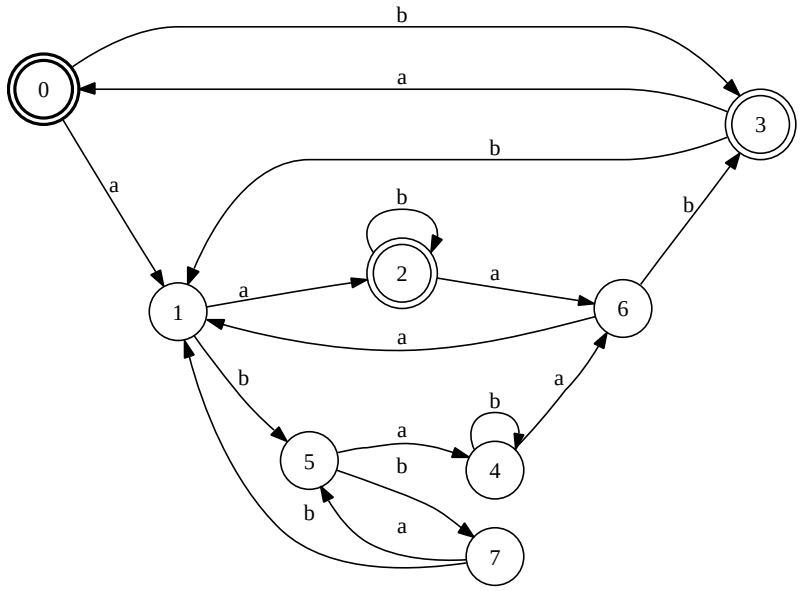
039M.fsm



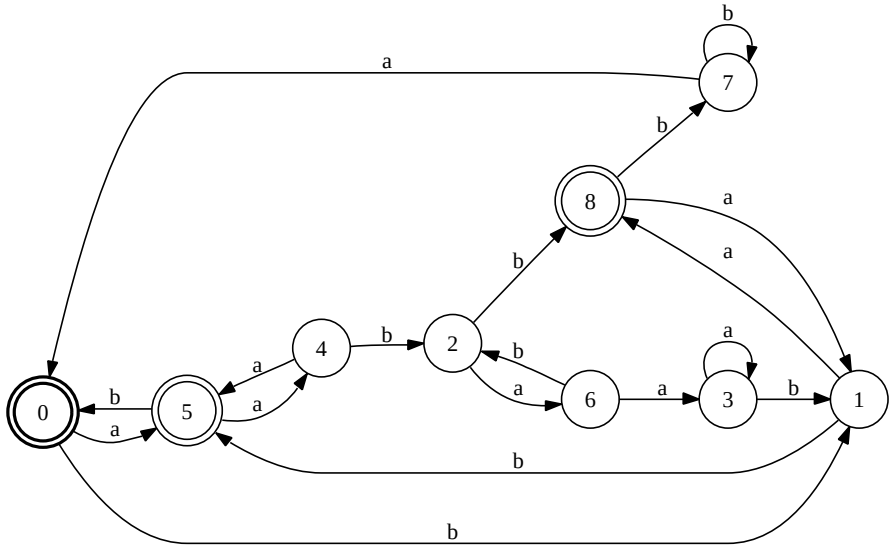
040M.fsm



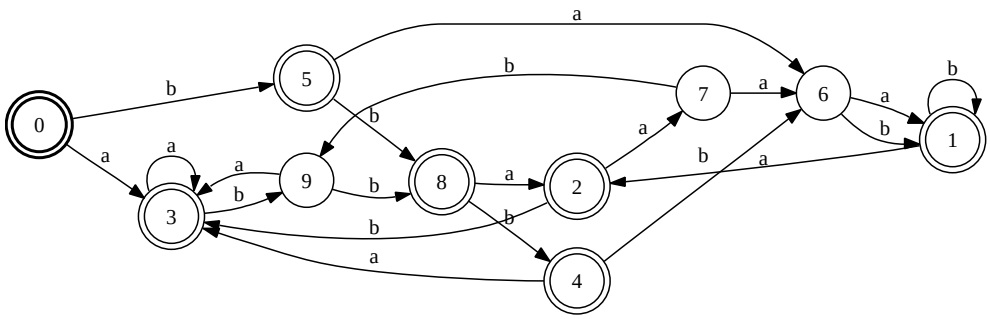
041M.fsm



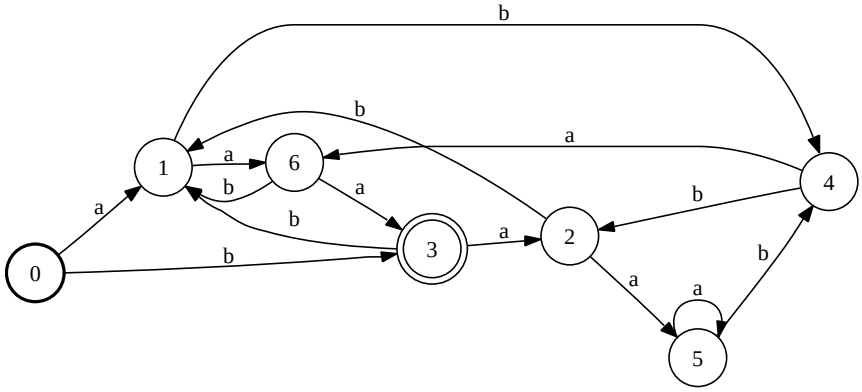
042M.fsm



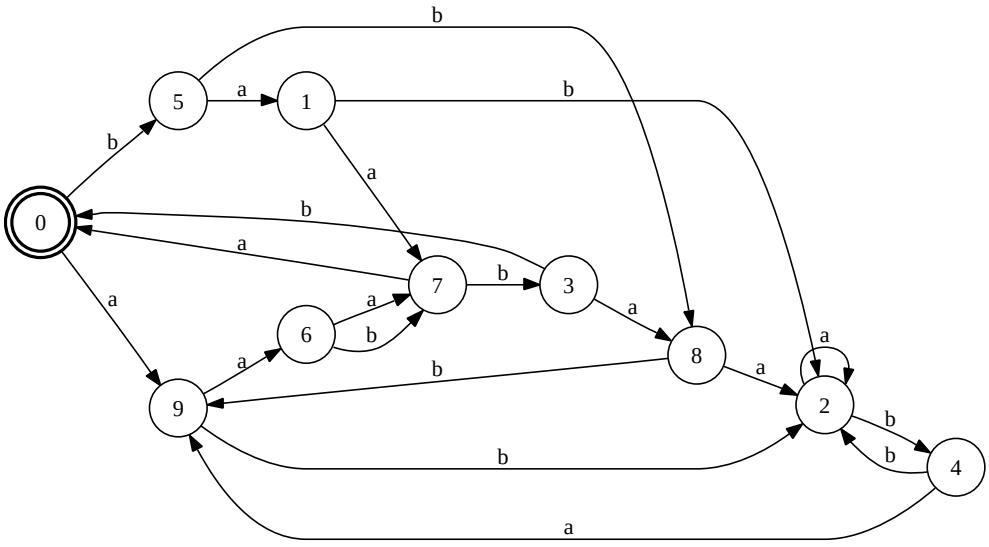
043M.fsm



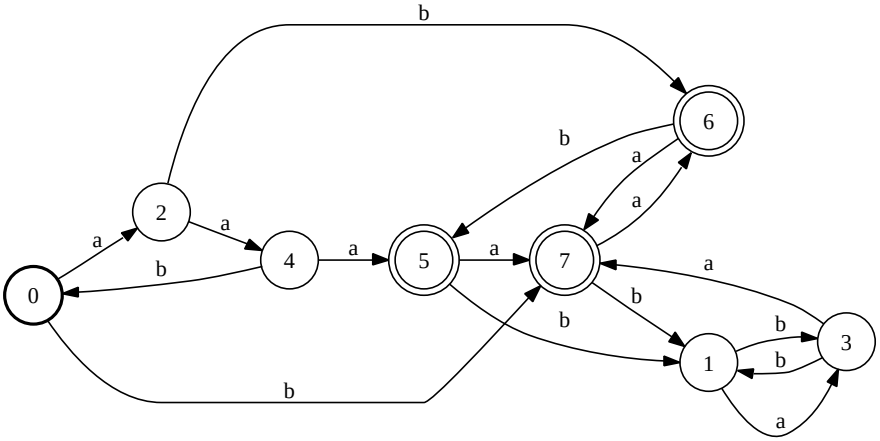
044M.fsm



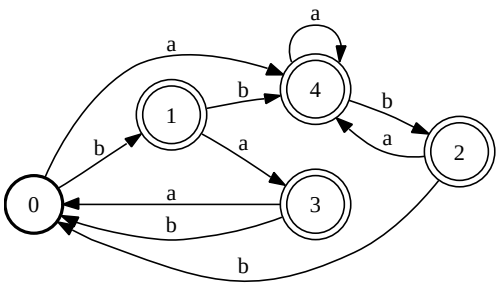
045M.fsm



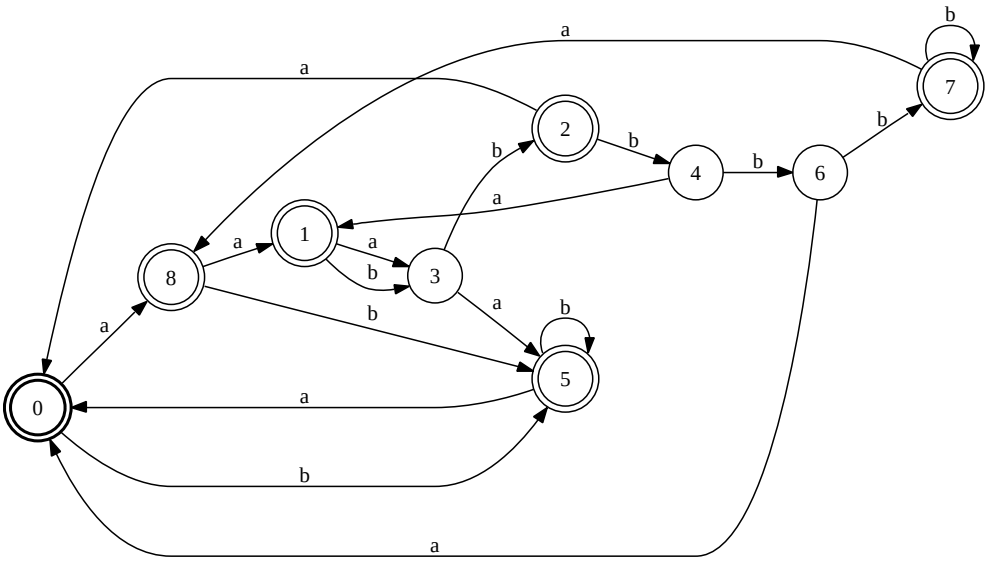
046M.fsm



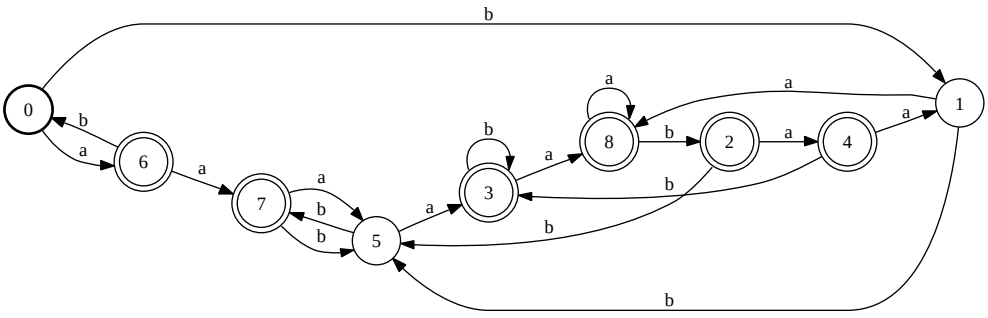
047M.fsm



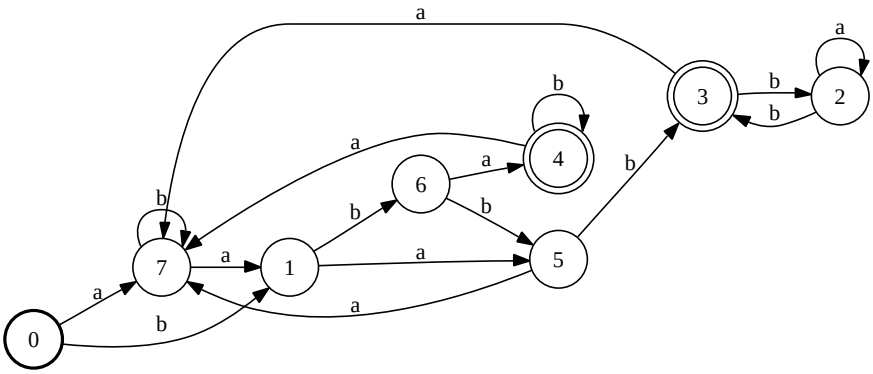
048M.fsm



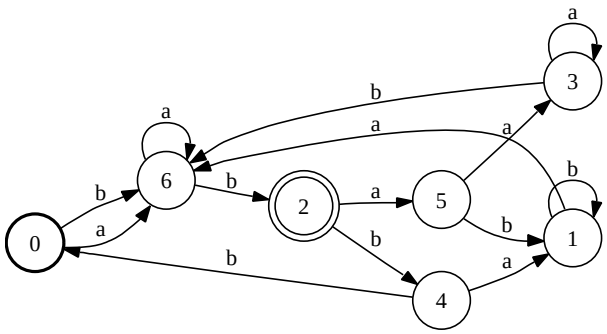
049M.fsm



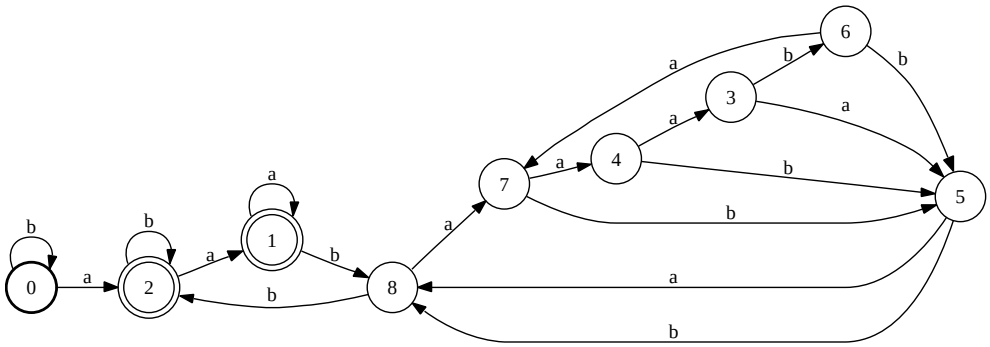
050M.fsm



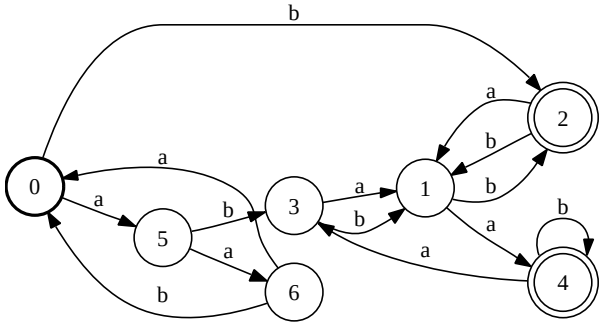
051M.fsm



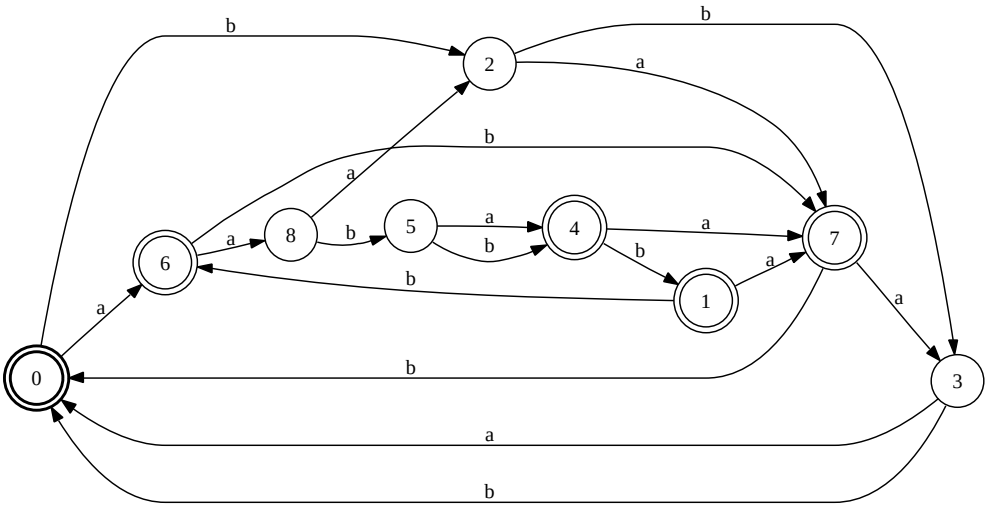
052M.fsm



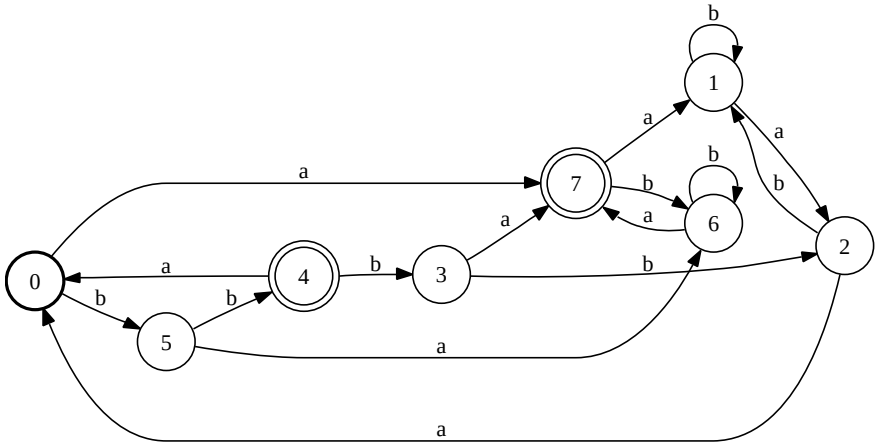
053M.fsm



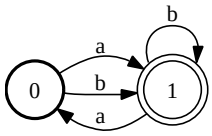
054M.fsm



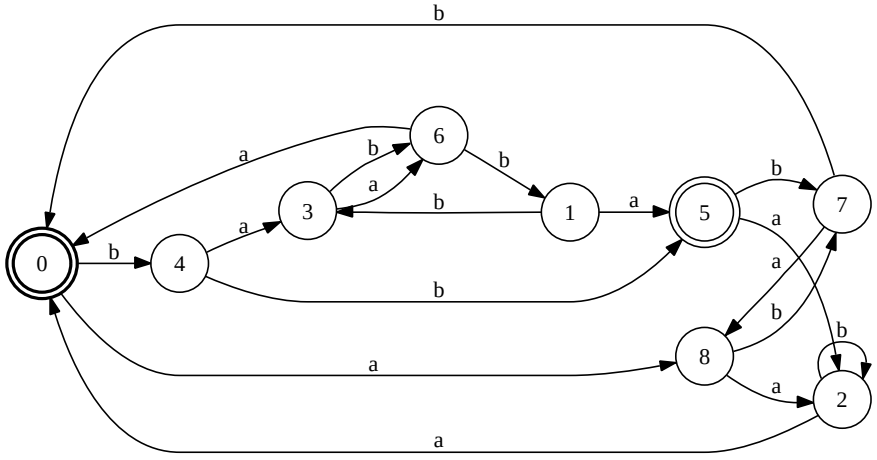
055M.fsm



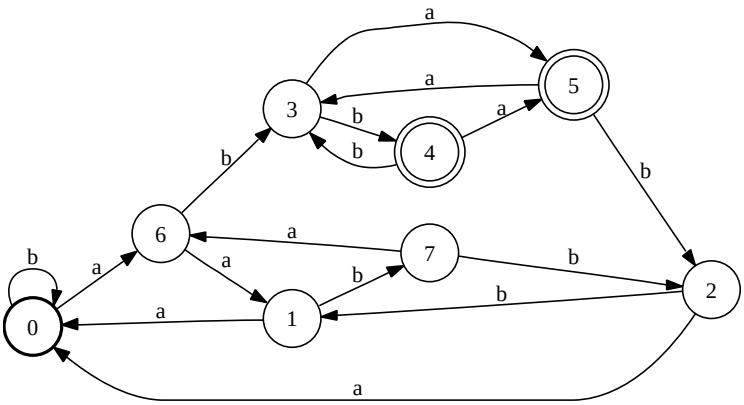
056M.fsm



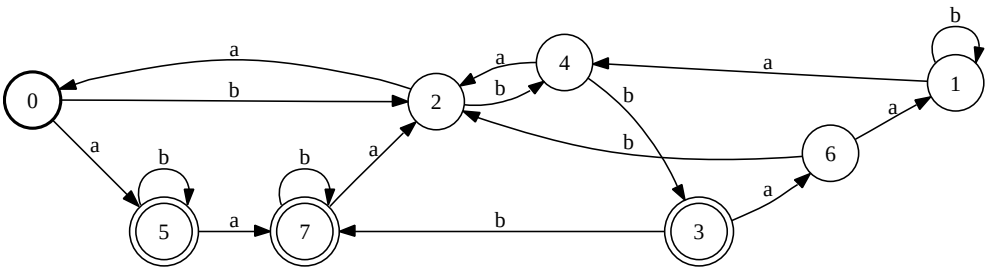
057M.fsm



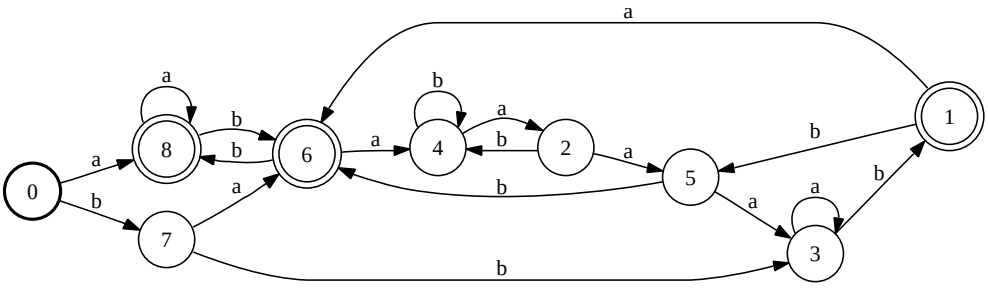
058M.fsm



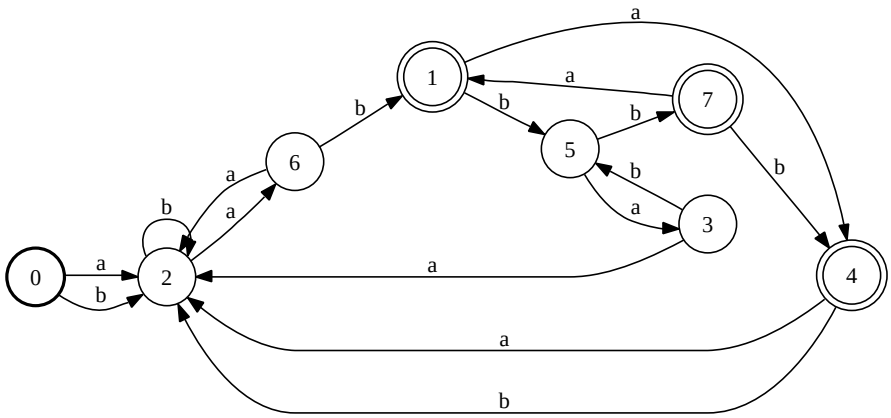
059M.fsm



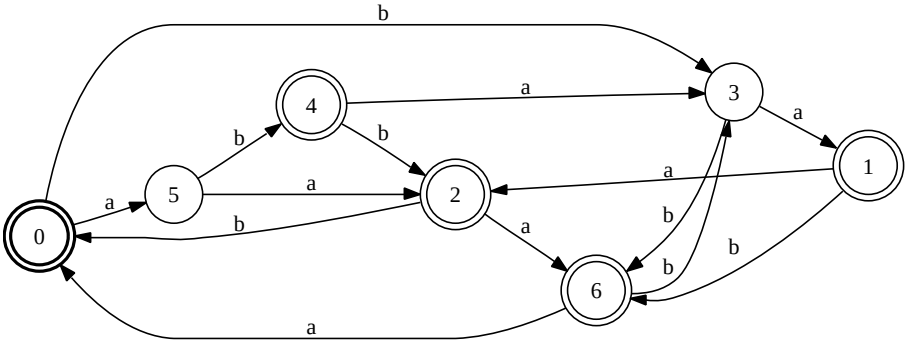
060M.fsm



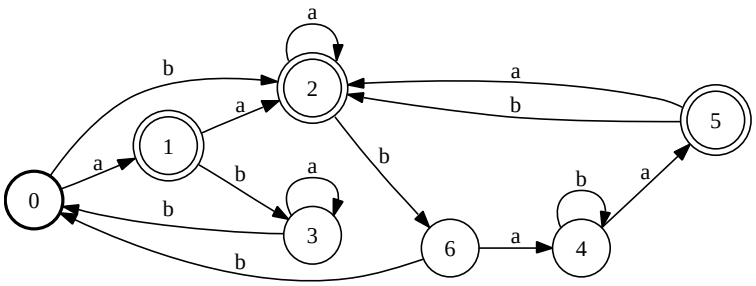
061M.fsm



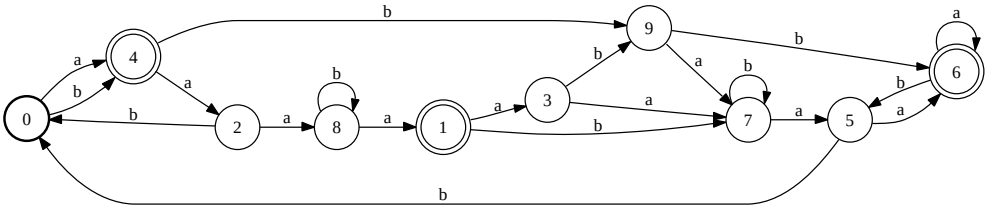
062M.fsm



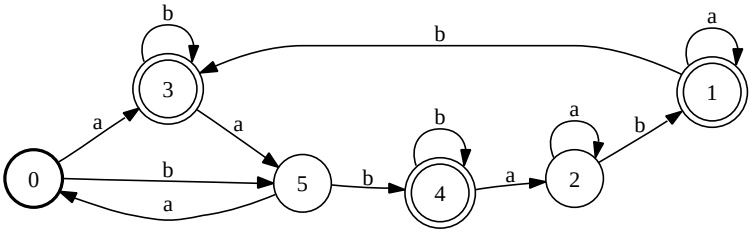
063M.fsm



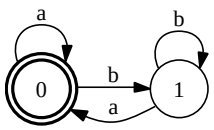
064M.fsm



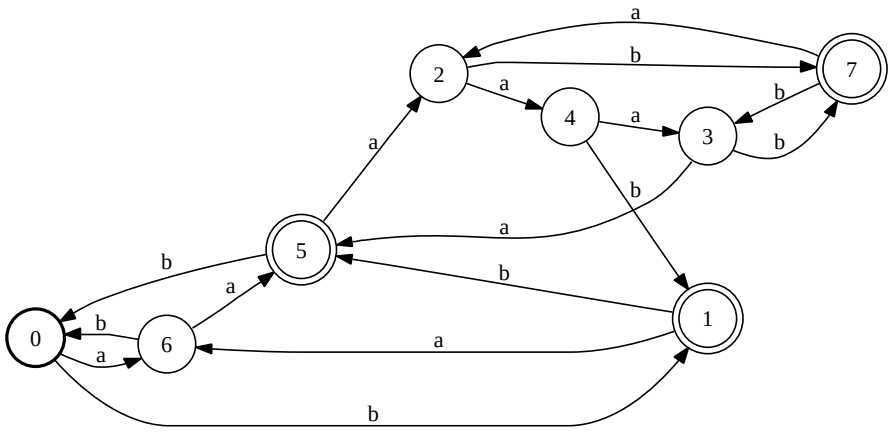
065M.fsm



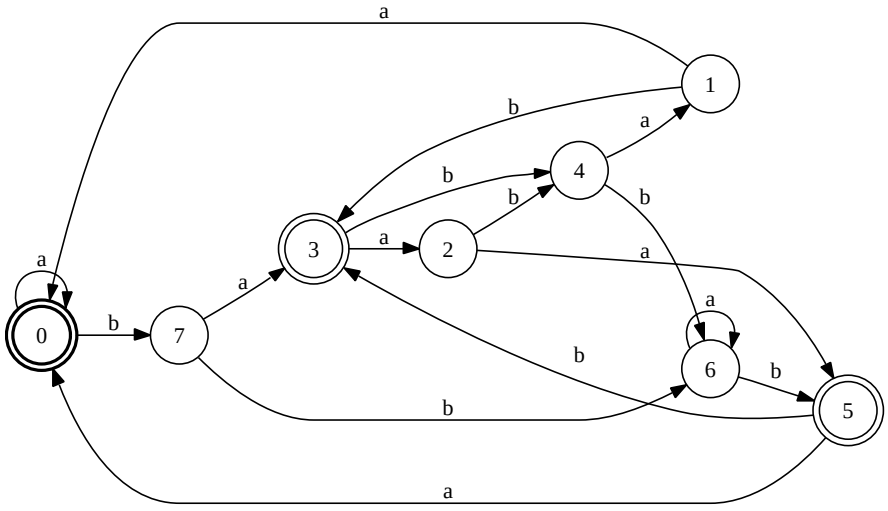
066M.fsm



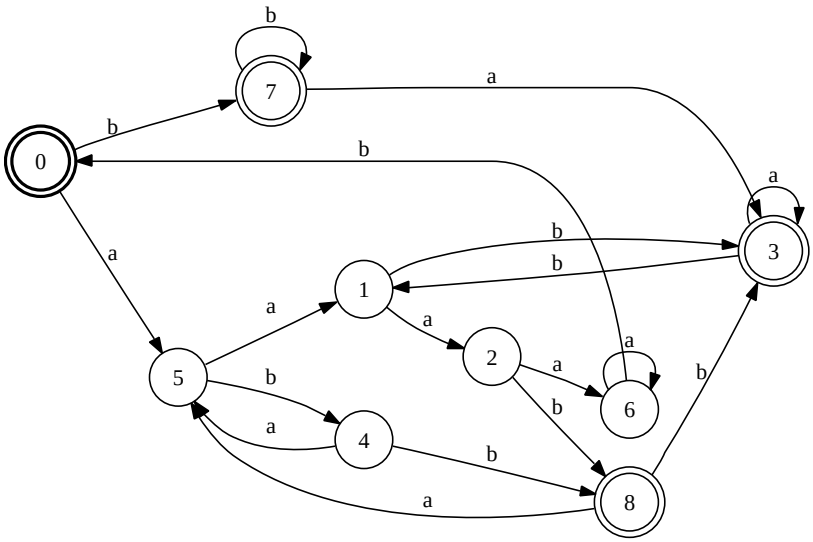
067M.fsm



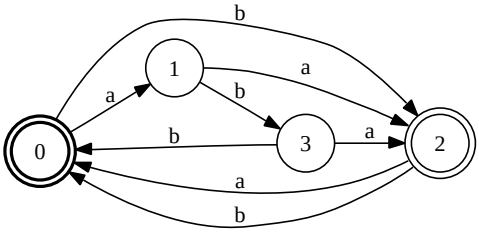
068M.fsm



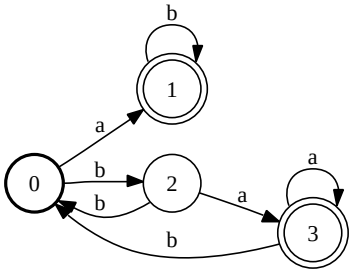
069M.fsm



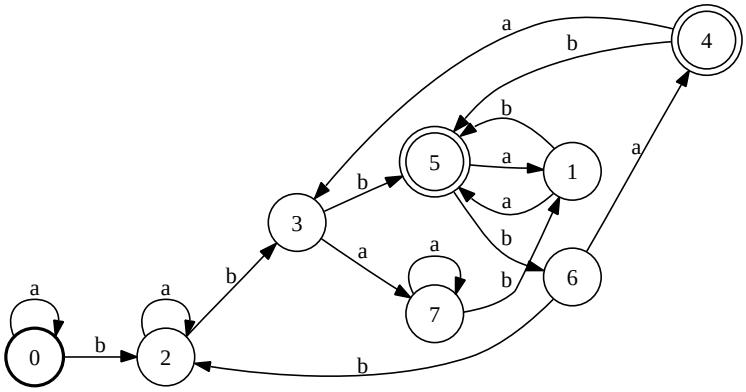
070M.fsm



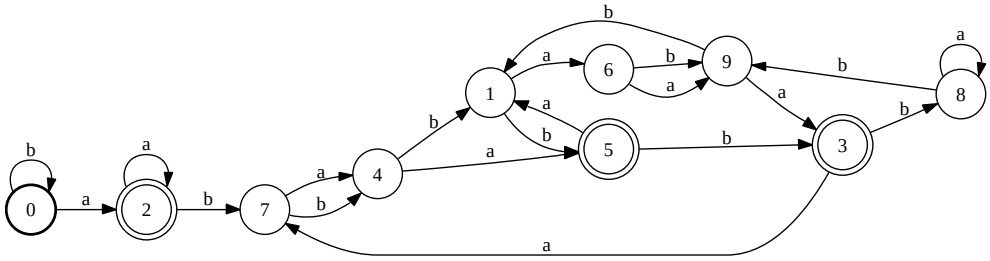
071M.fsm



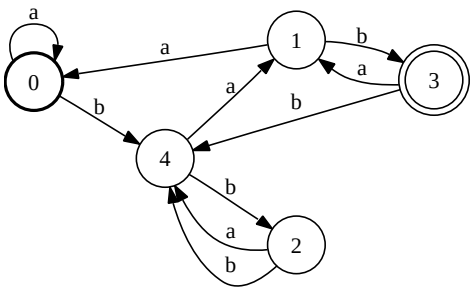
072M.fsm



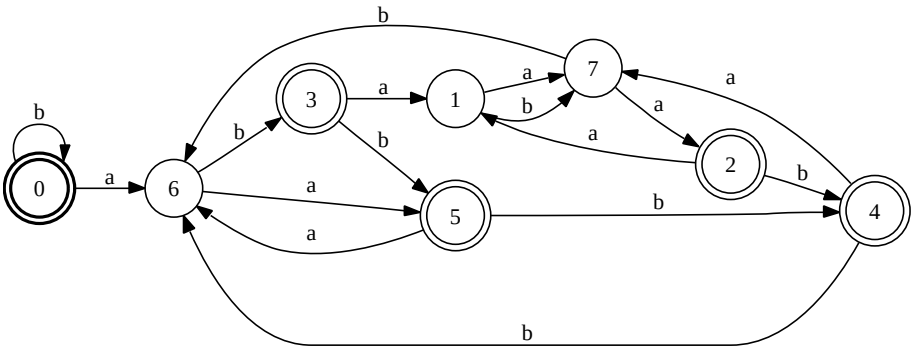
073M.fsm



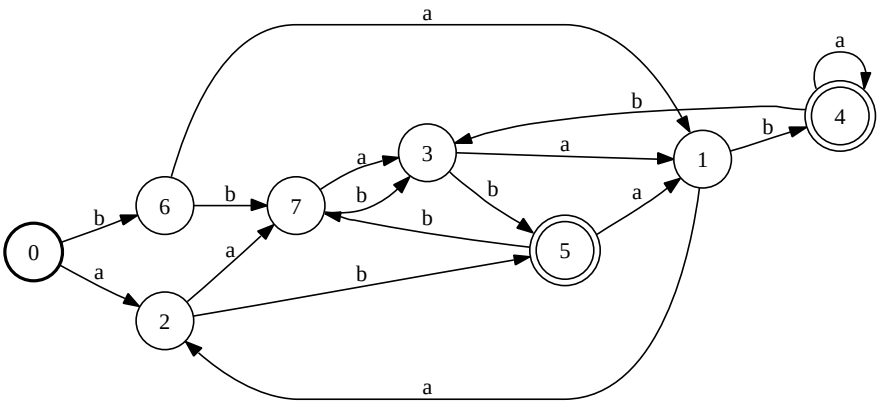
074M.fsm



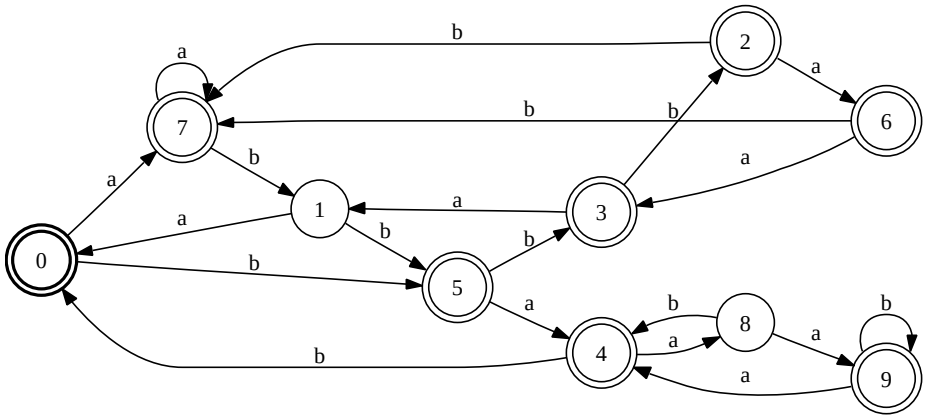
075M.fsm



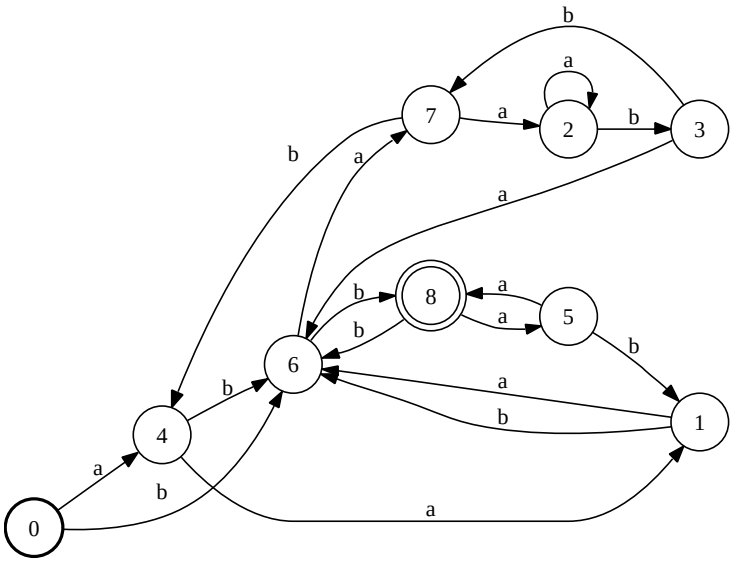
076M.fsm



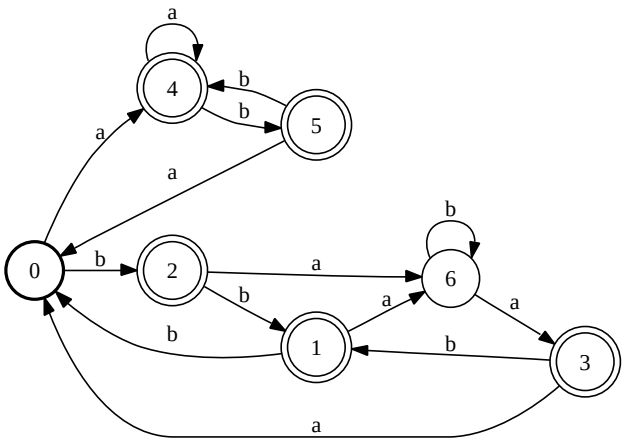
077M.fsm



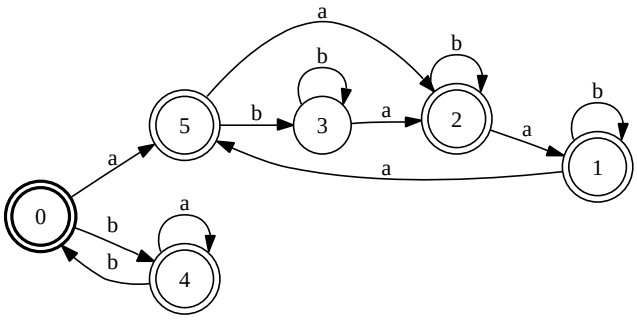
078M.fsm



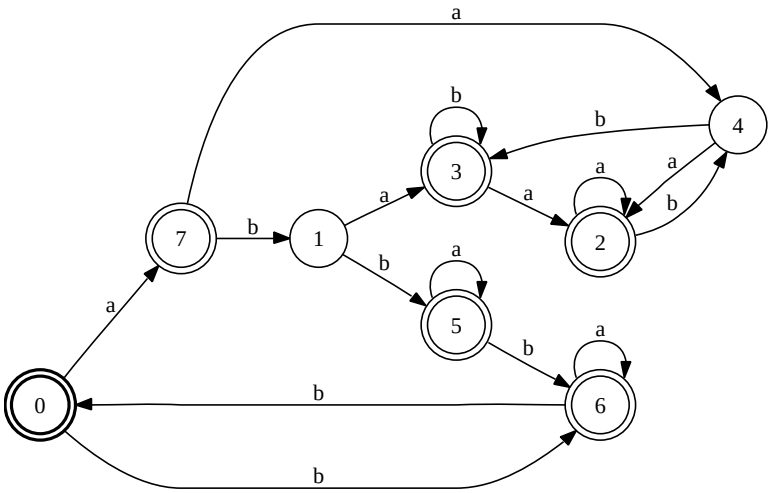
079M.fsm



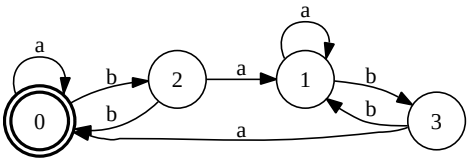
080M.fsm



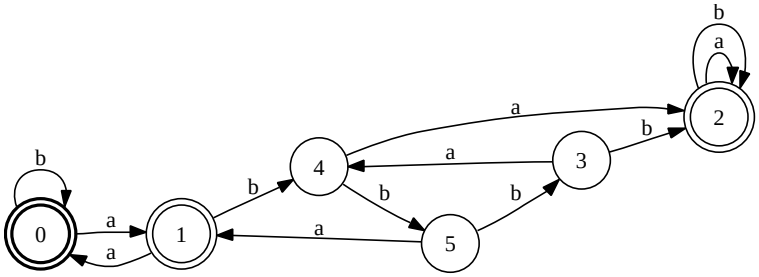
081M.fsm



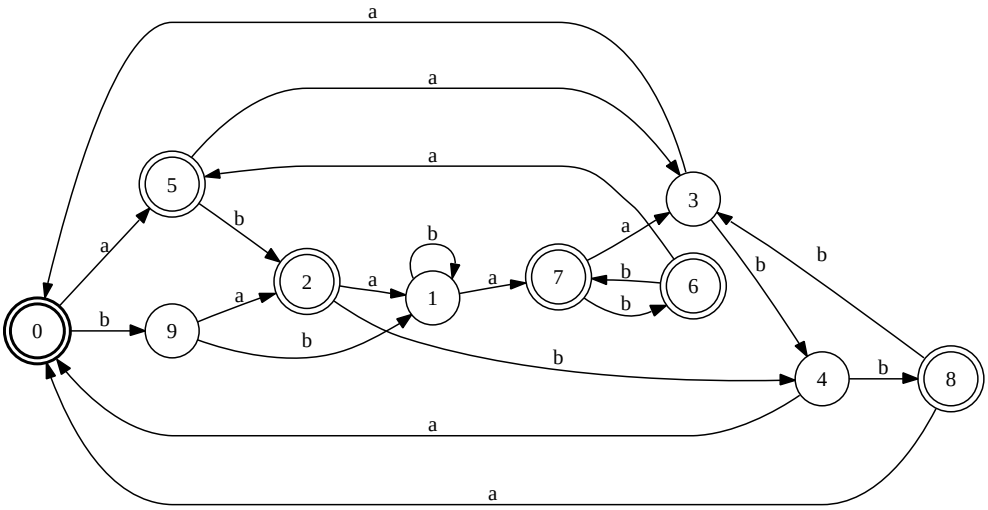
082M.fsm



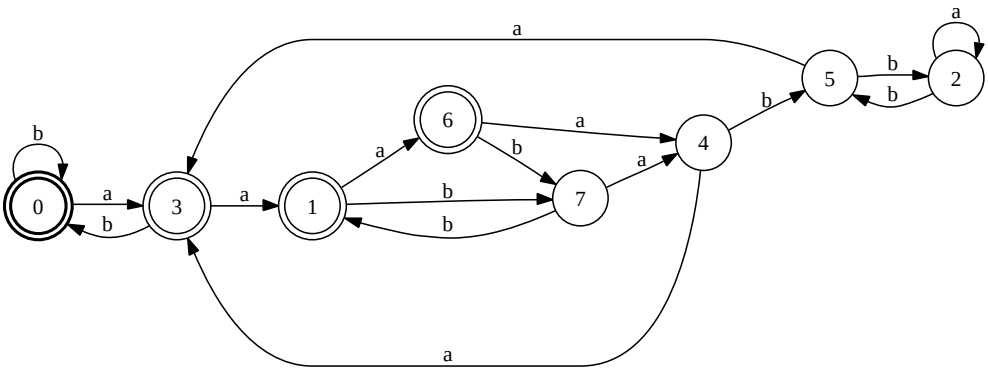
083M.fsm



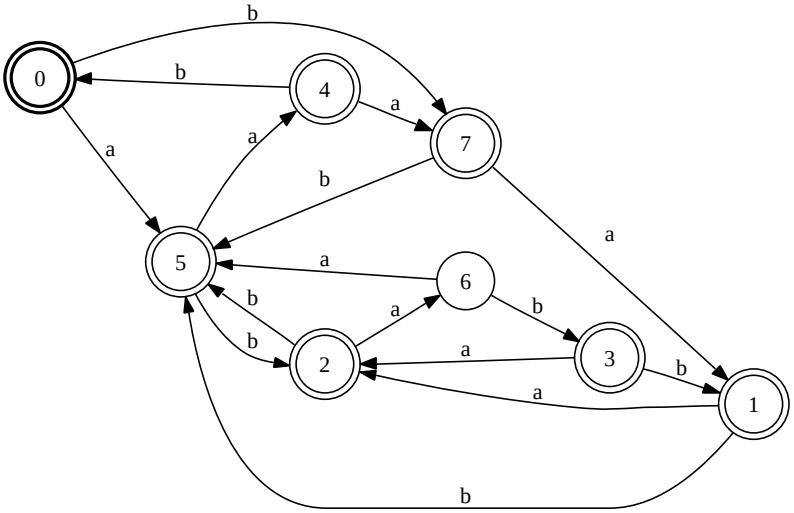
084M.fsm



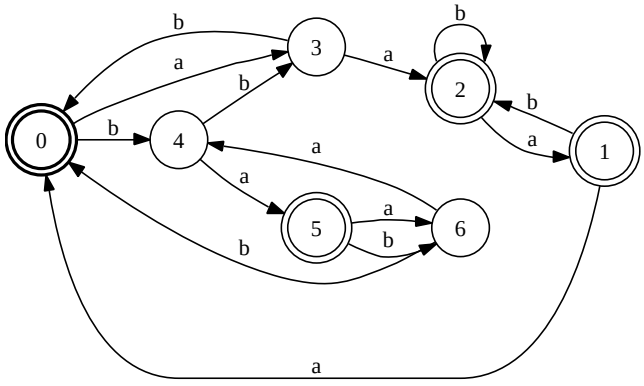
085M.fsm



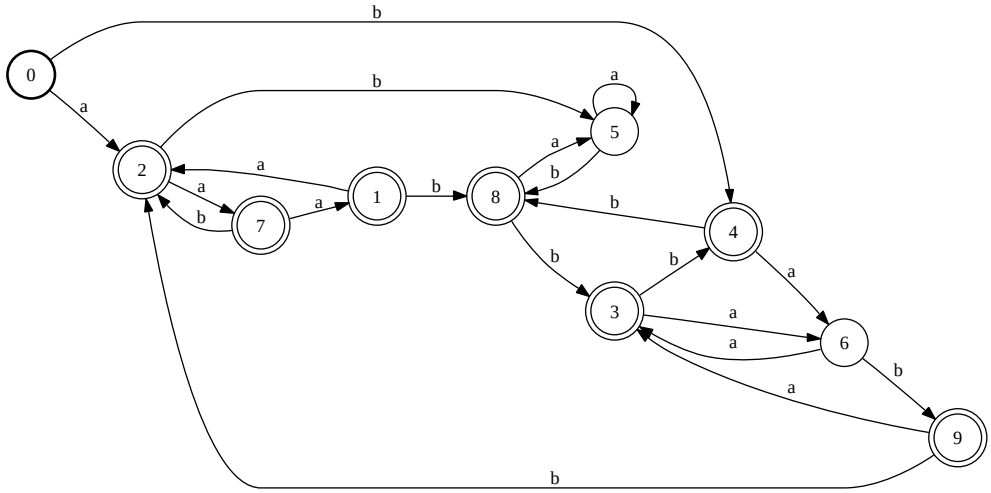
086M.fsm



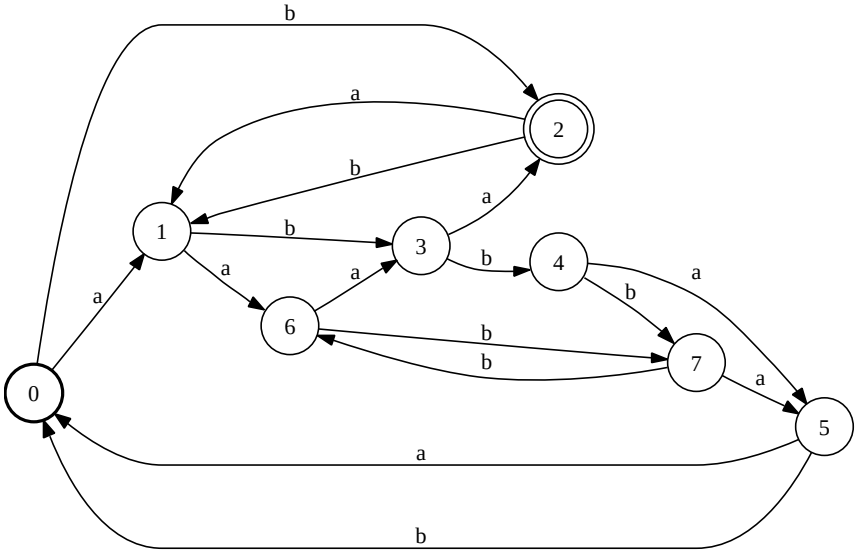
087M.fsm



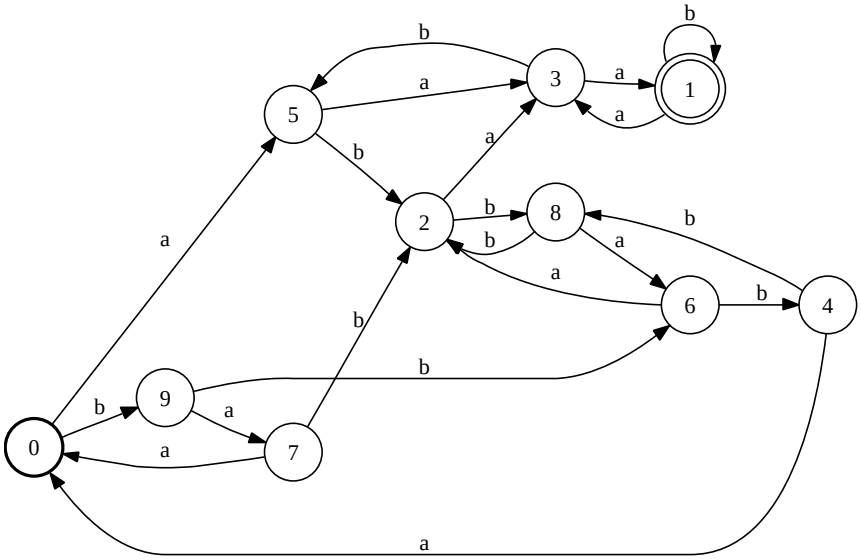
088M.fsm



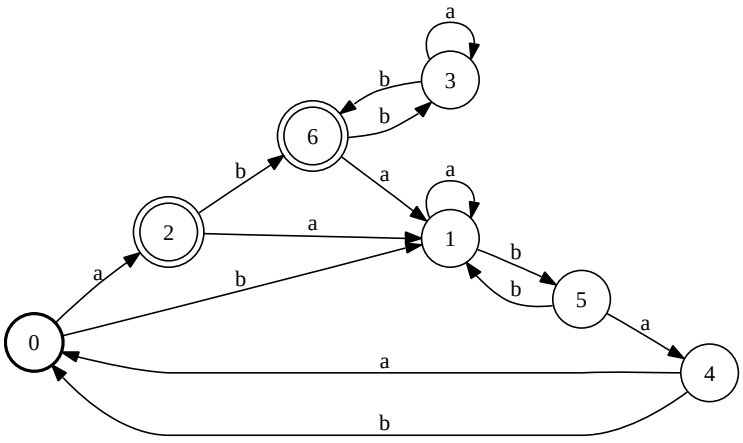
089M.fsm



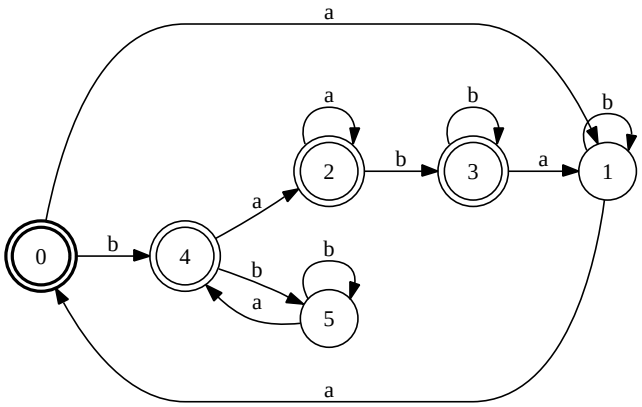
090M.fsm



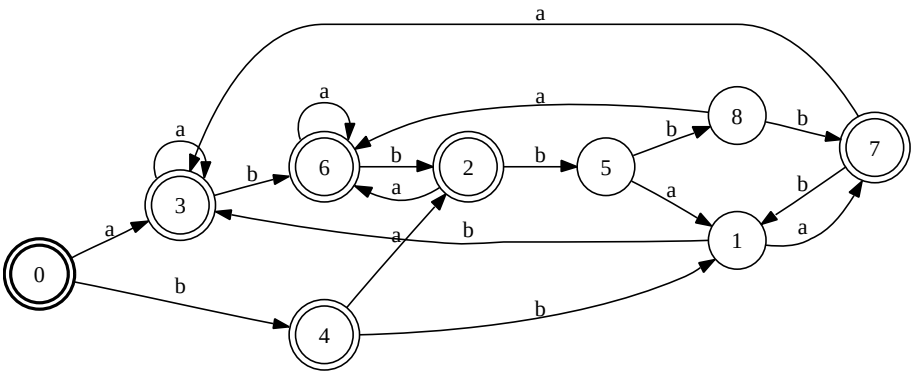
091M.fsm



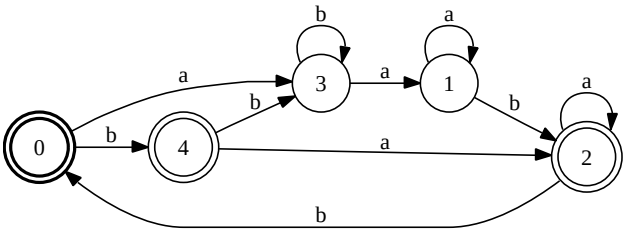
092M.fsm



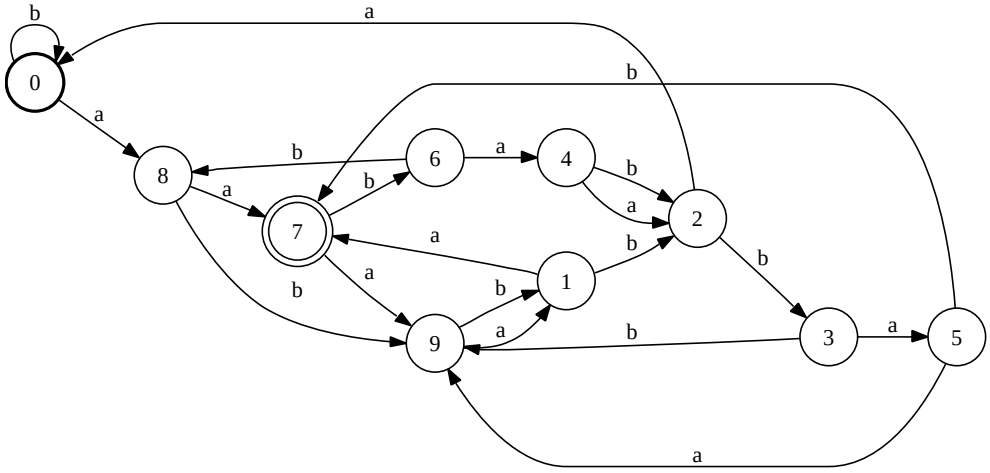
093M.fsm



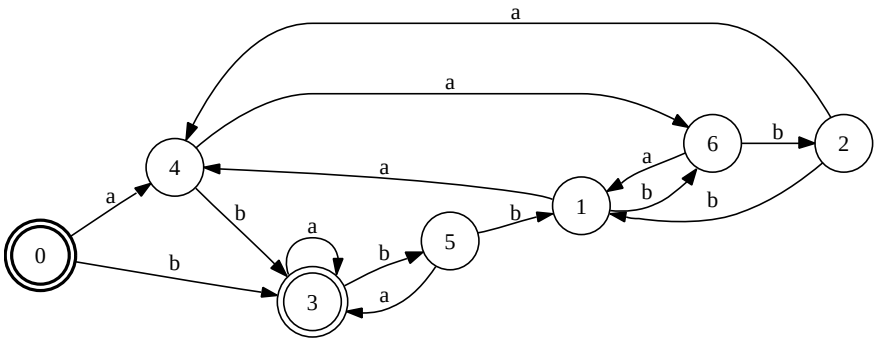
094M.fsm



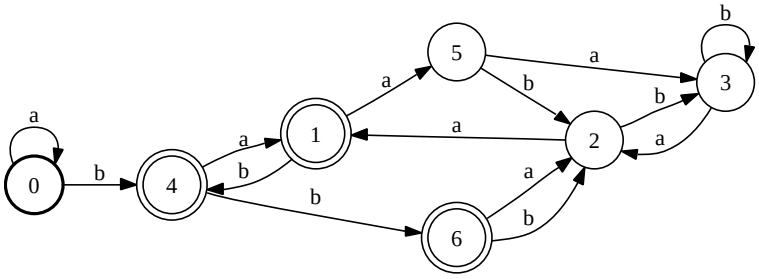
095M.fsm



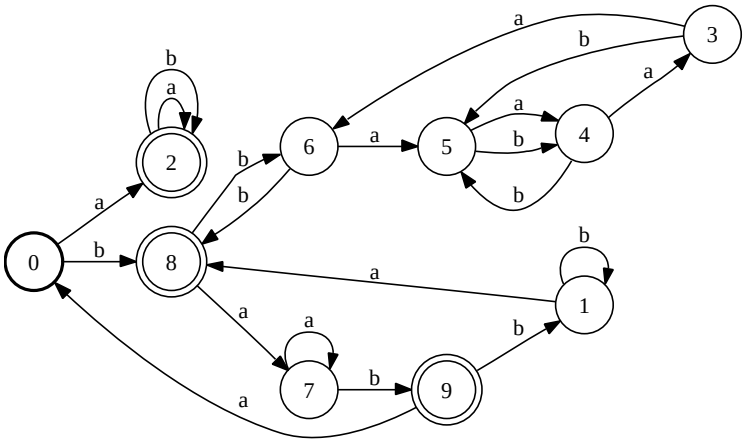
096M.fsm



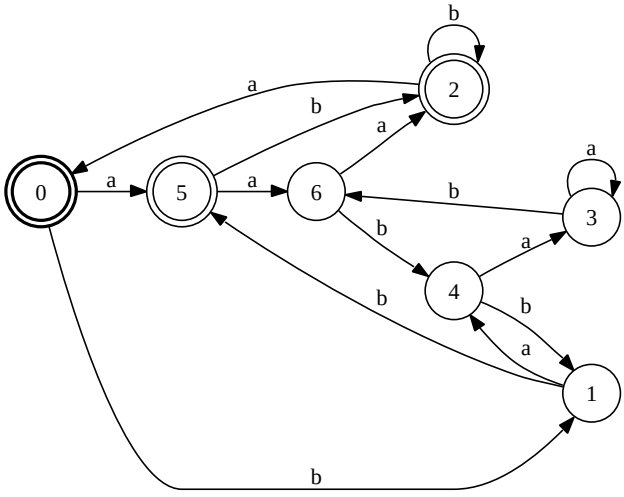
097M.fsm



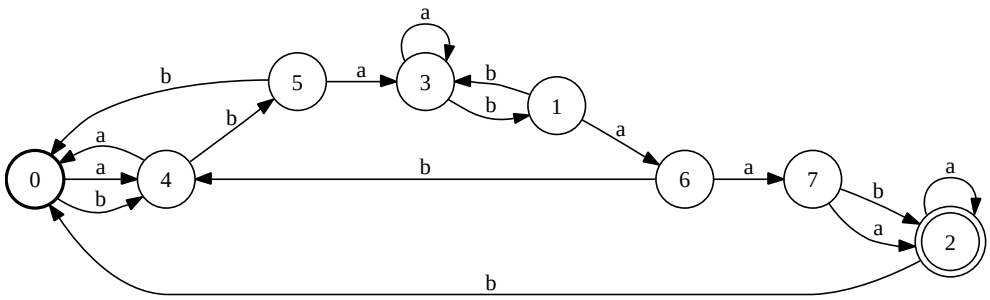
098M.fsm



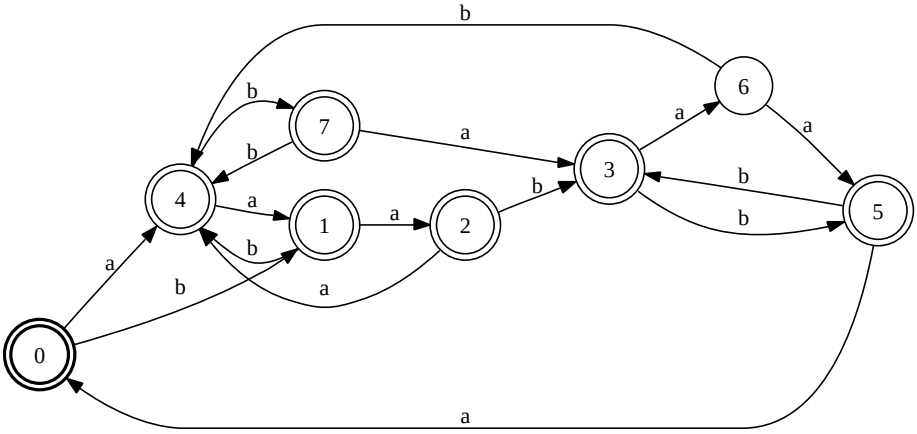
099M.fsm



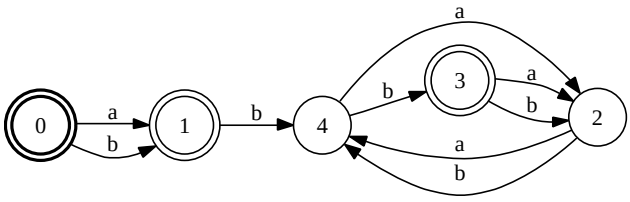
100M.fsm



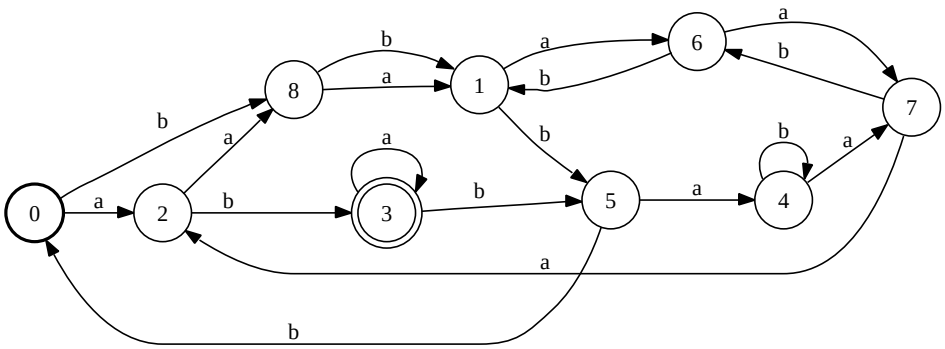
101M.fsm



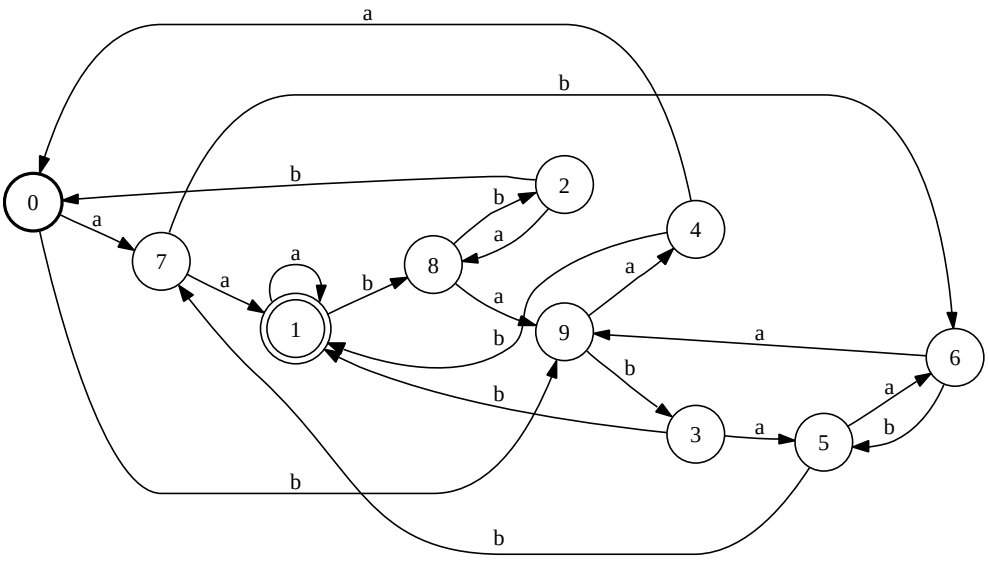
102M.fsm



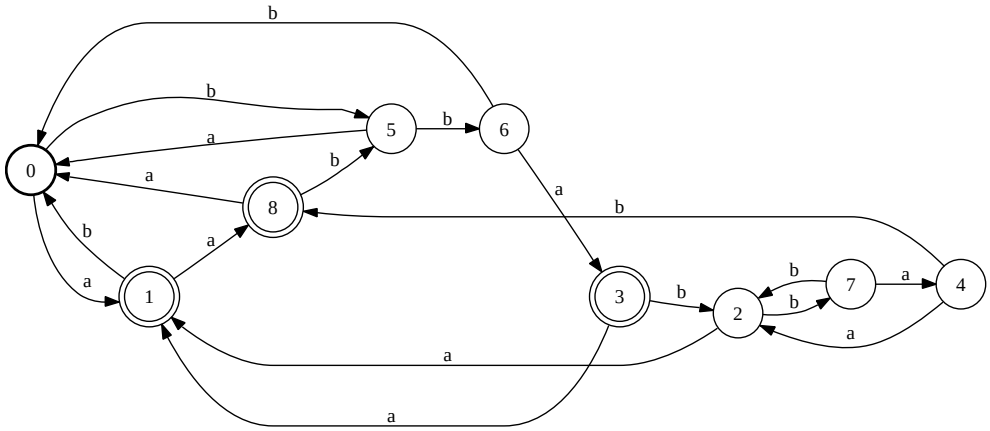
103M.fsm



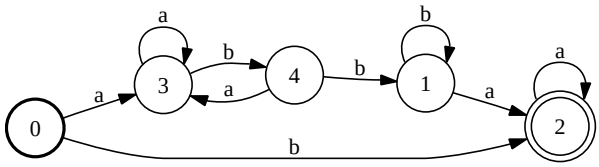
104M.fsm



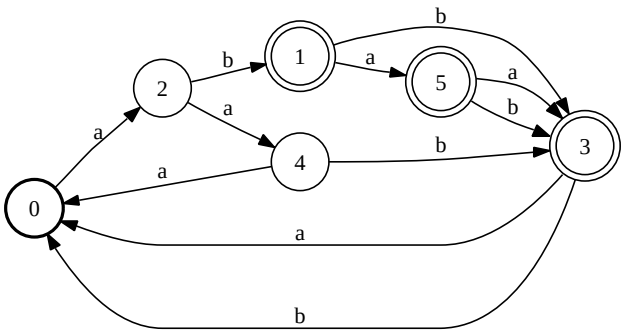
105M.fsm



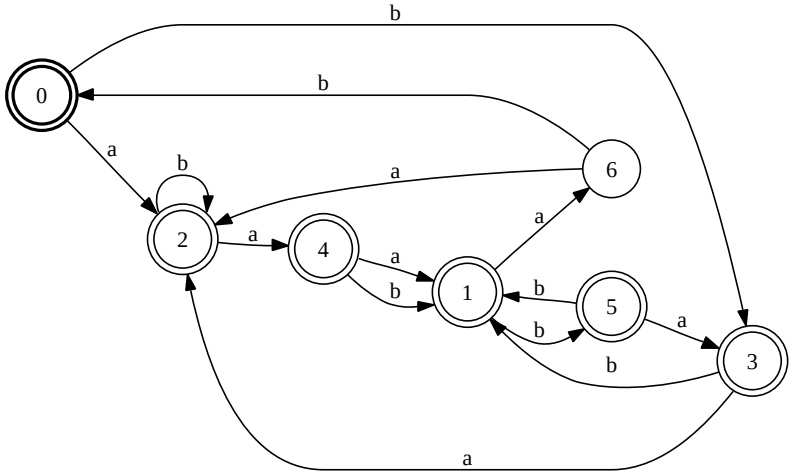
106M.fsm



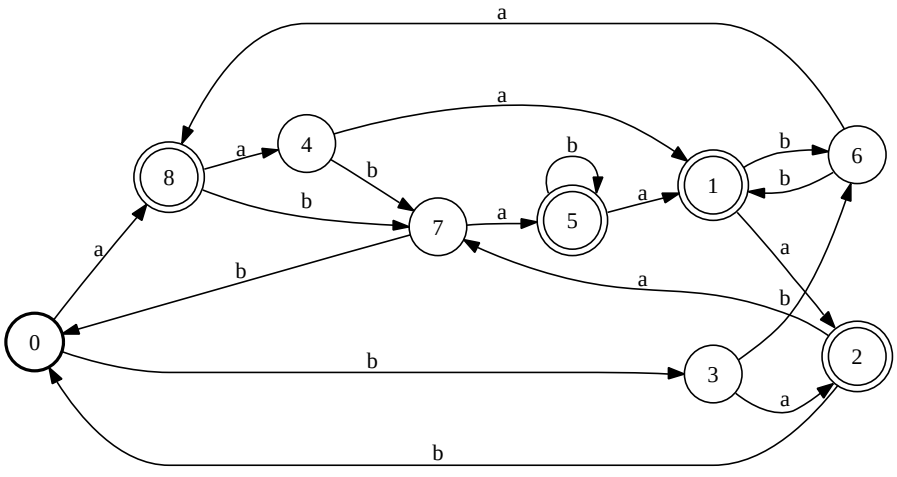
107M.fsm



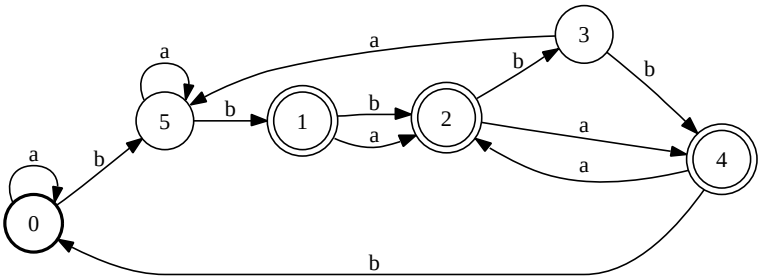
108M.fsm



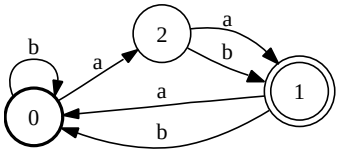
109M.fsm



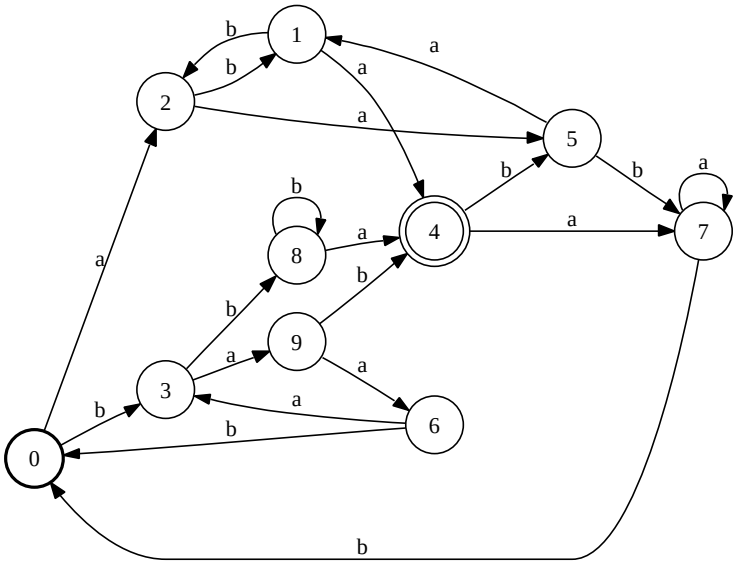
110M.fsm



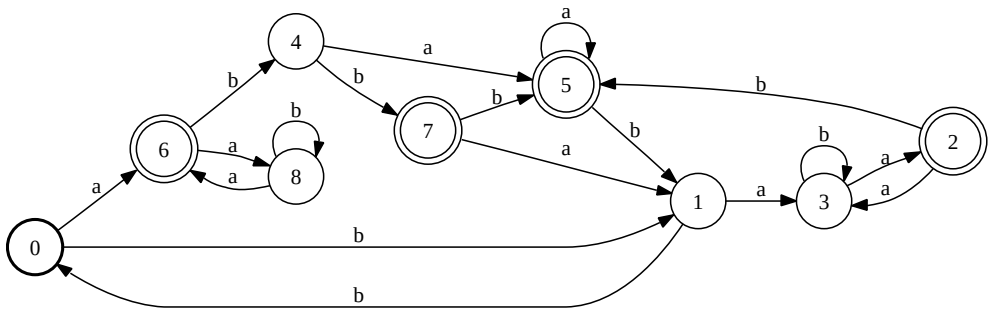
111M.fsm



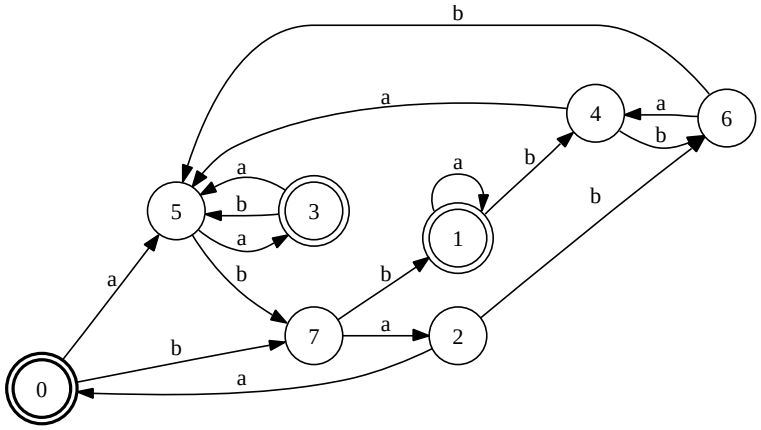
112M.fsm



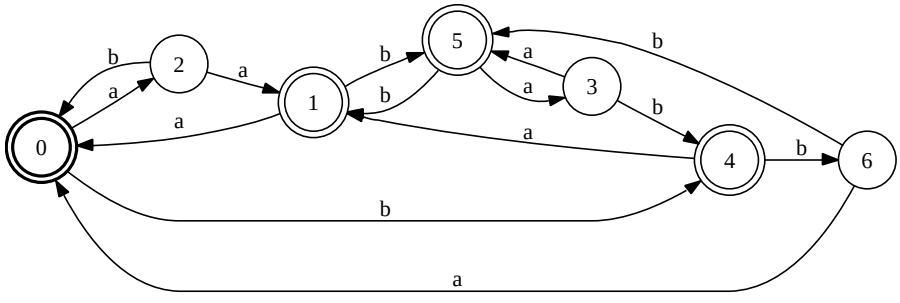
113M.fsm



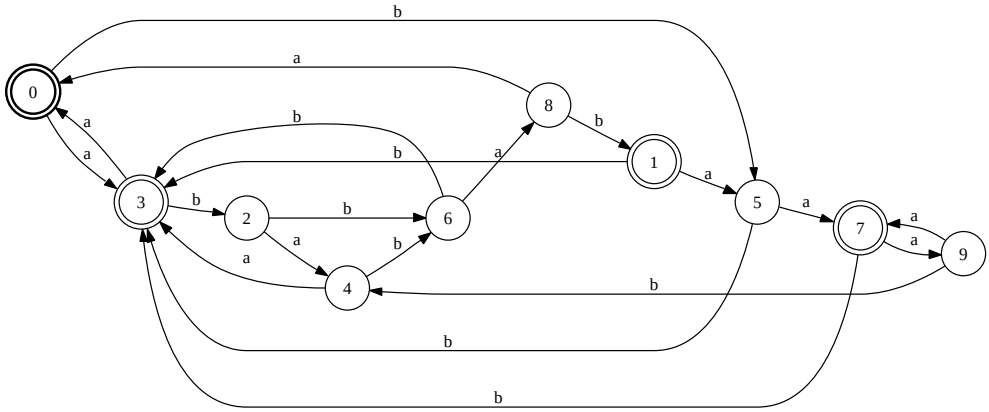
114M.fsm



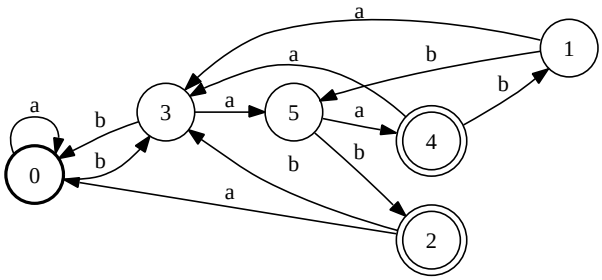
115M.fsm



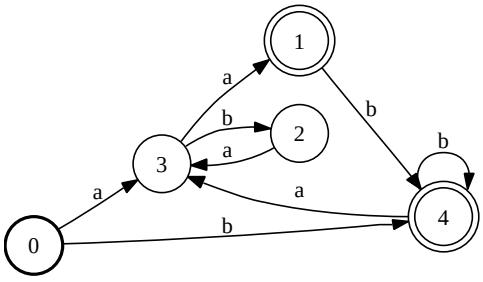
116M.fsm



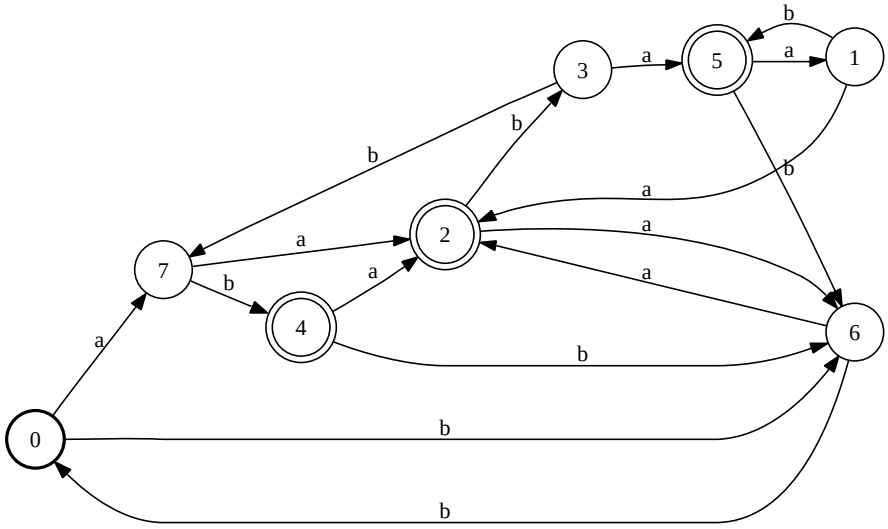
117M.fsm



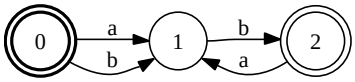
118M.fsm



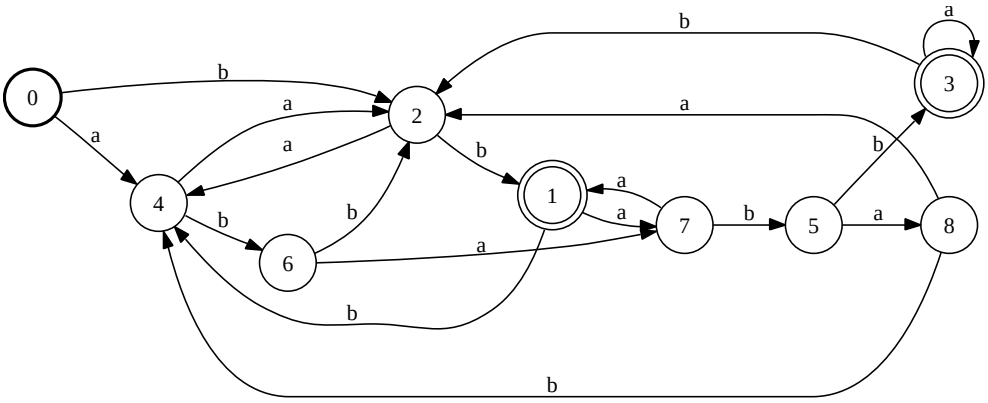
119M.fsm



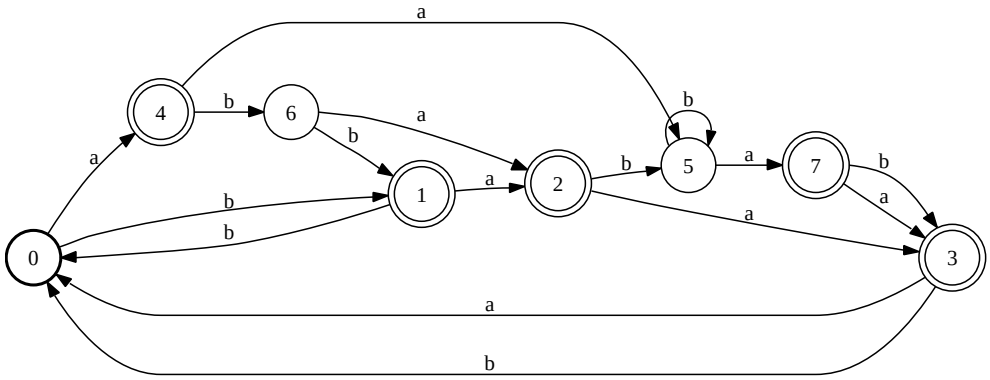
120M.fsm



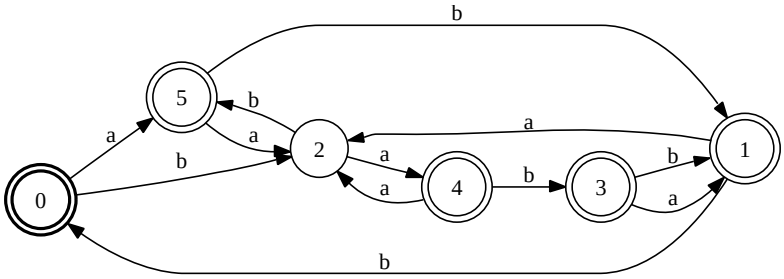
121M.fsm



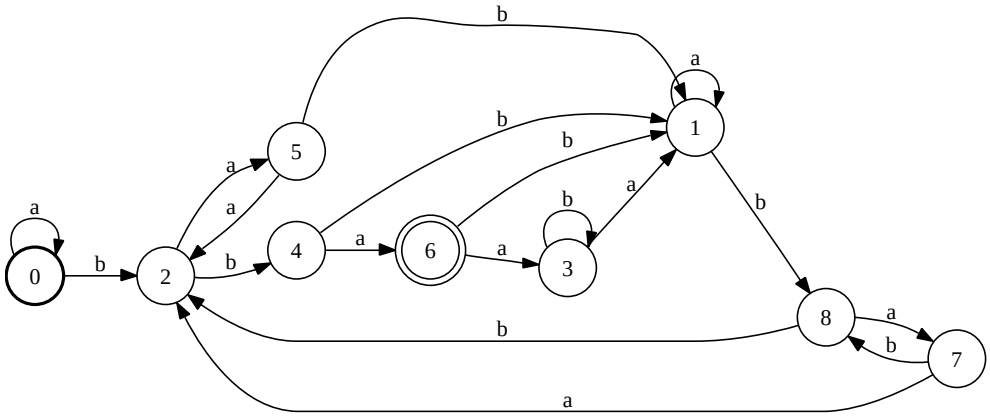
122M.fsm



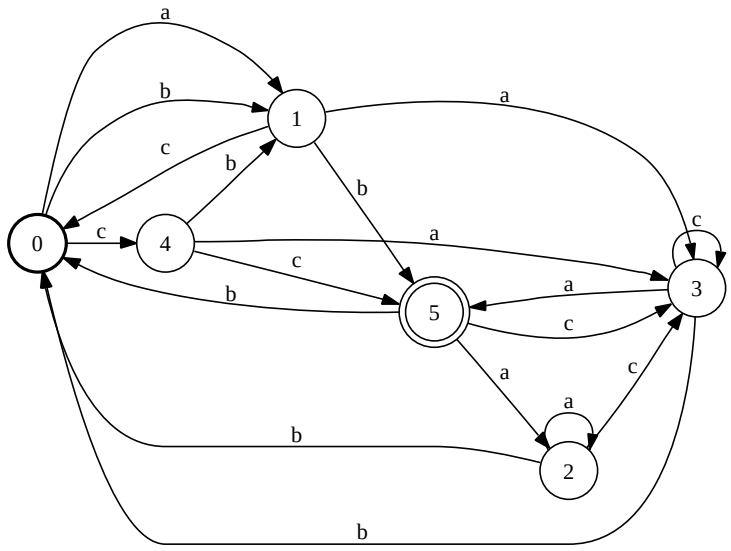
123M.fsm



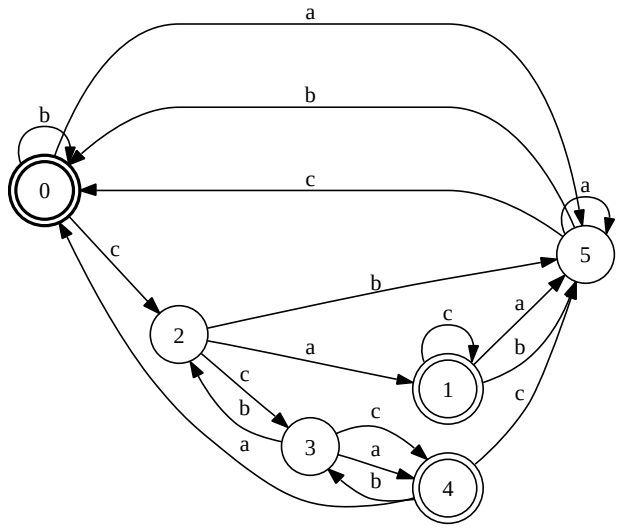
124M.fsm



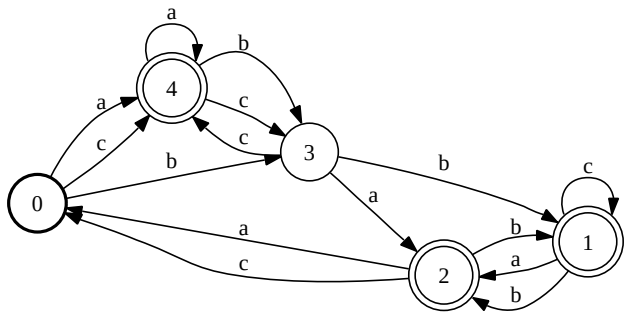
125M.fsm



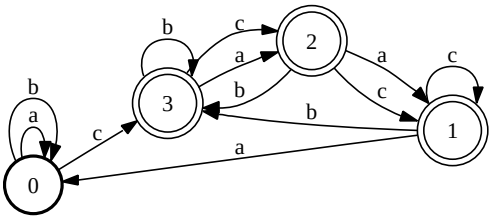
126M.fsm



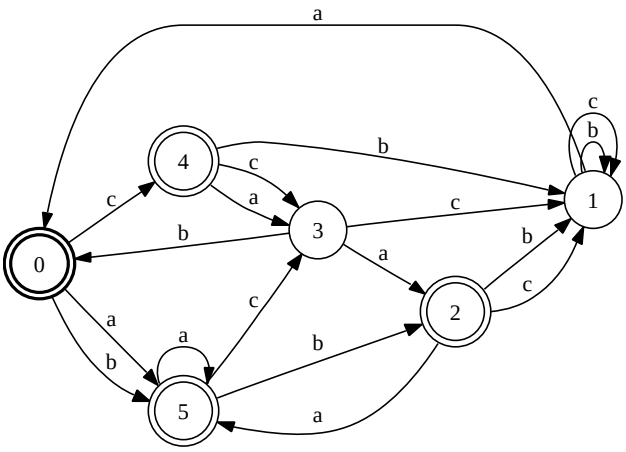
127M.fsm



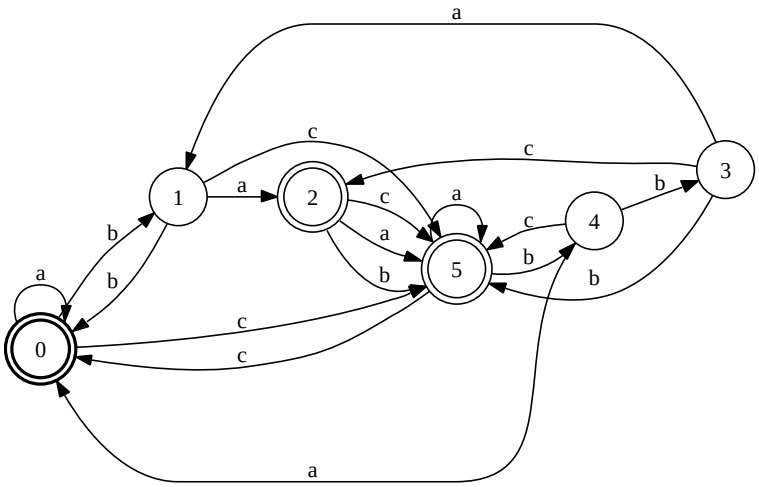
128M.fsm



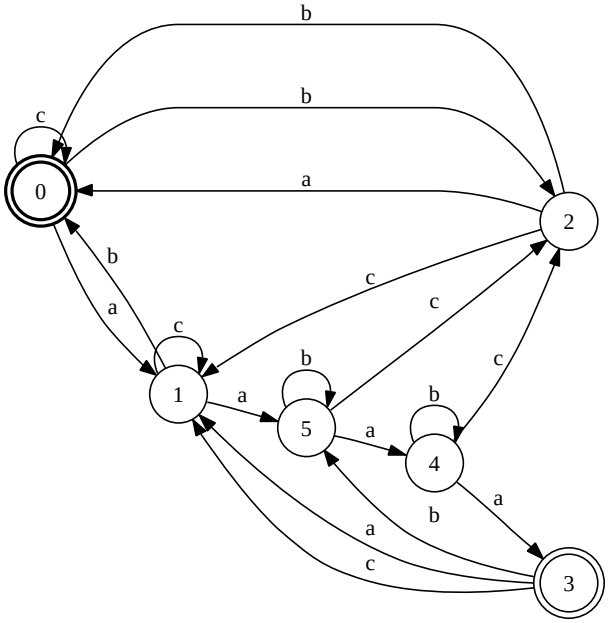
129M.fsm



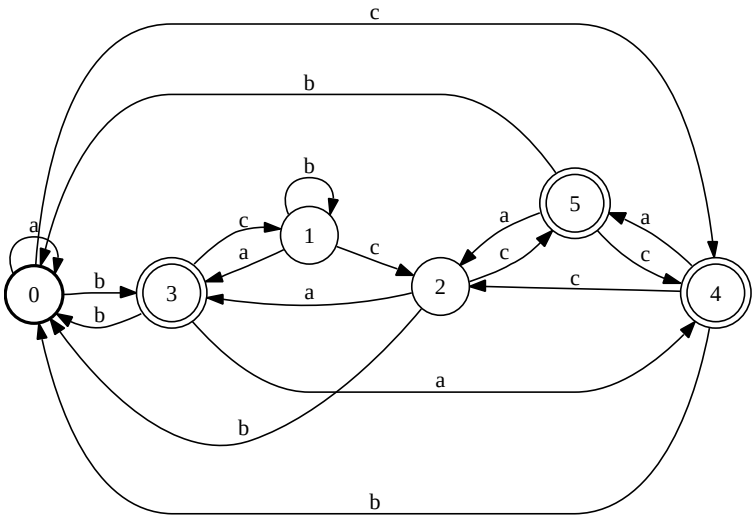
130M.fsm



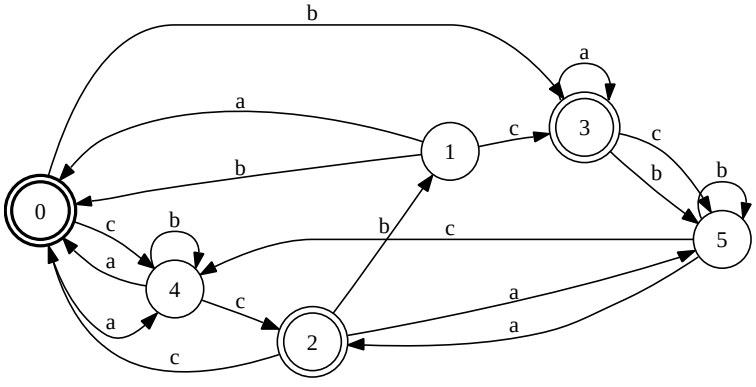
131M.fsm



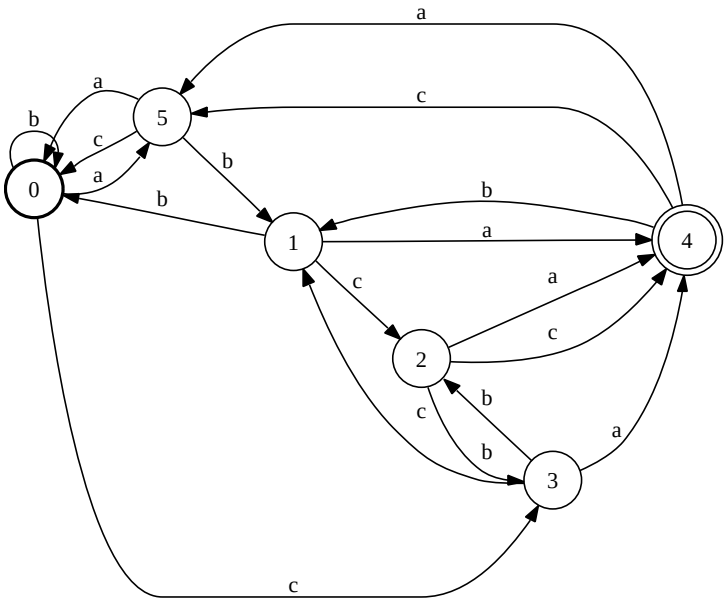
132M.fsm



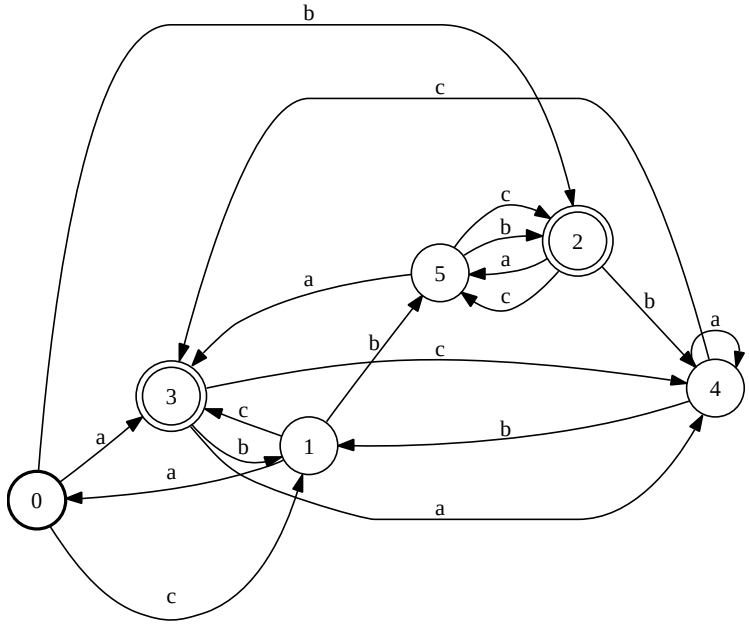
133M.fsm



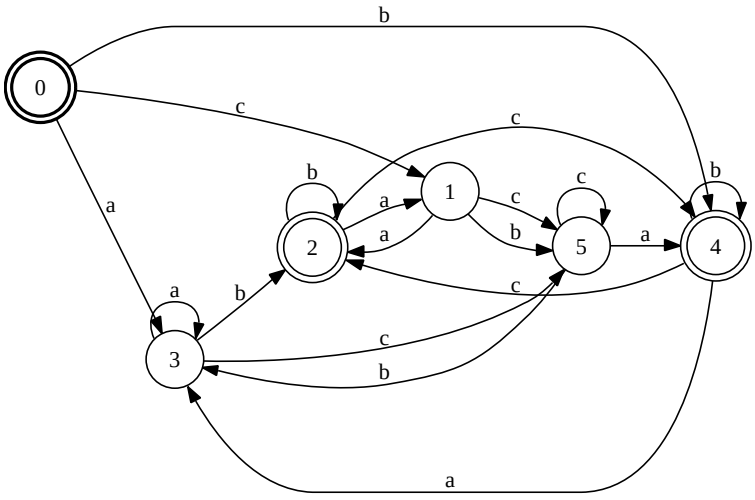
134M.fsm



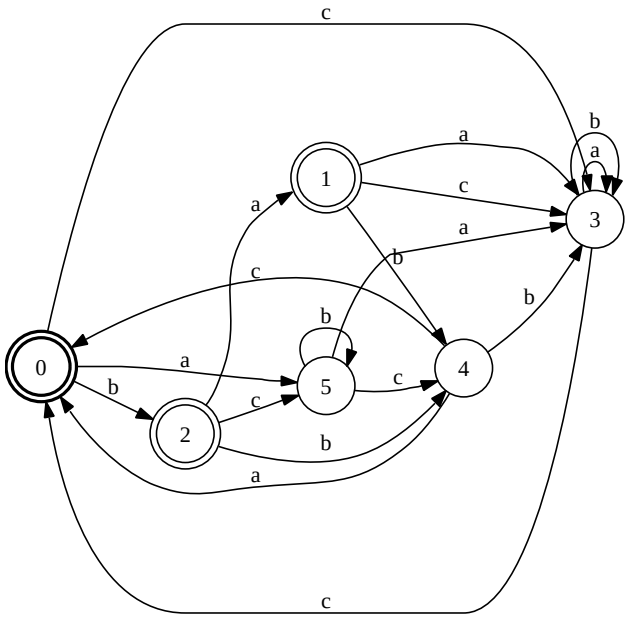
135M.fsm



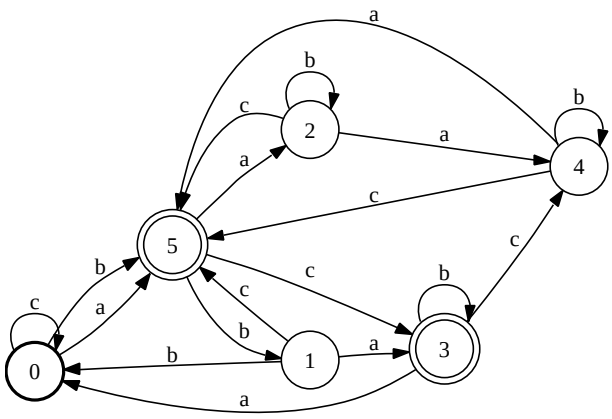
136M.fsm



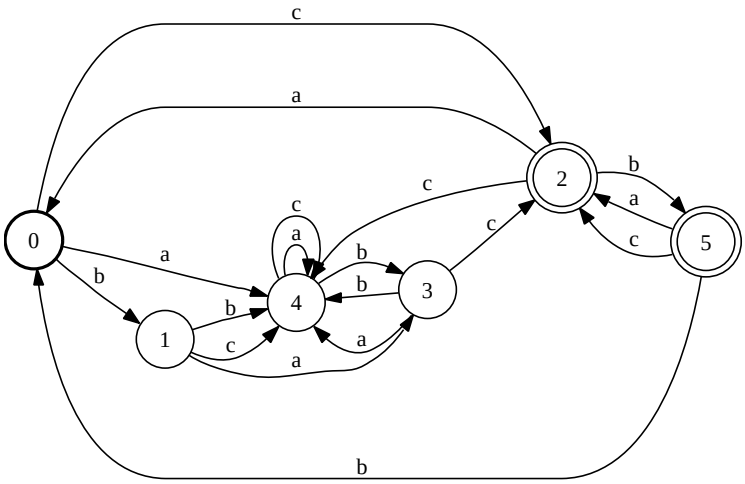
137M.fsm



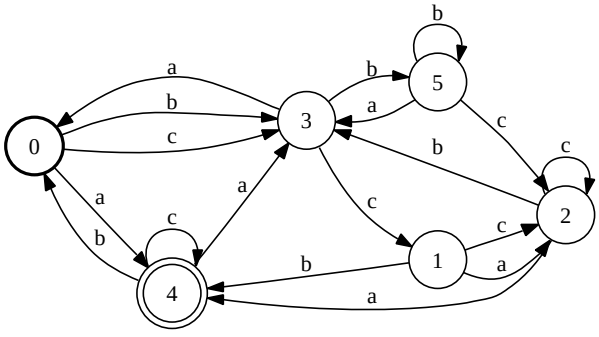
138M.fsm



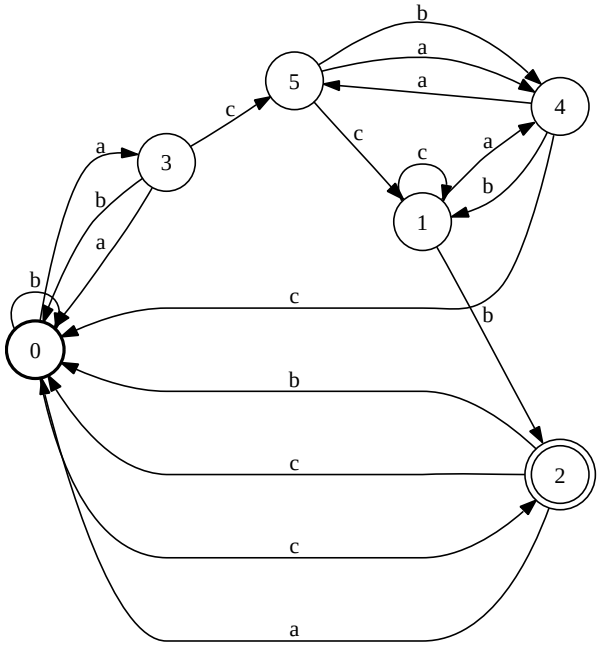
139M.fsm



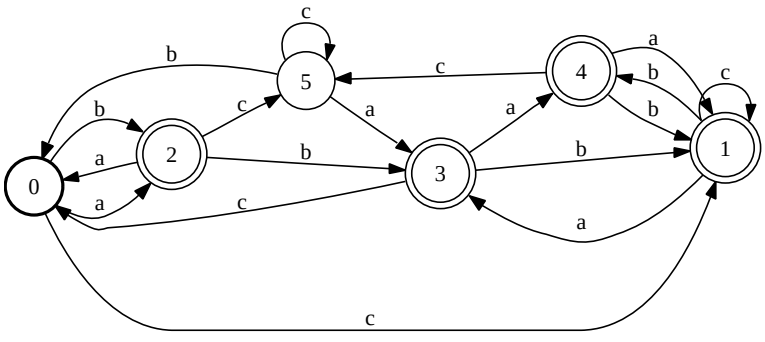
140M.fsm



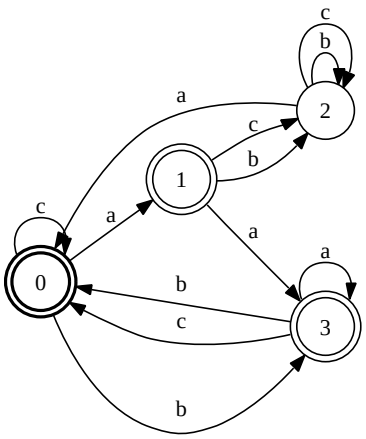
141M.fsm



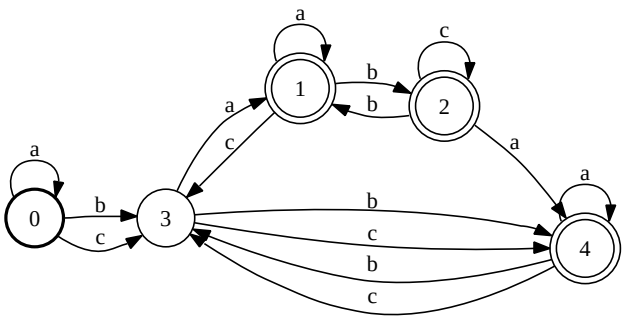
142M.fsm



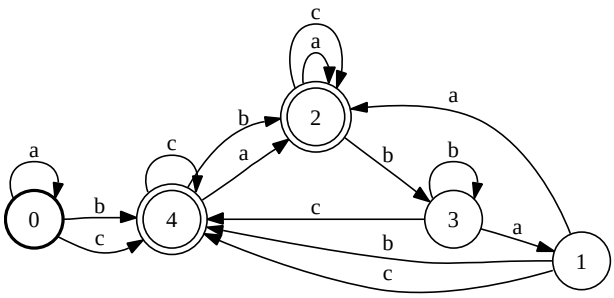
143M.fsm



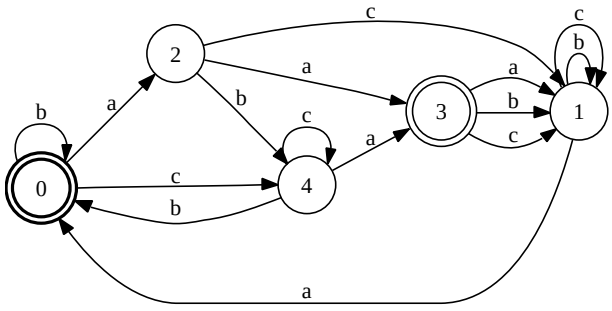
144M.fsm



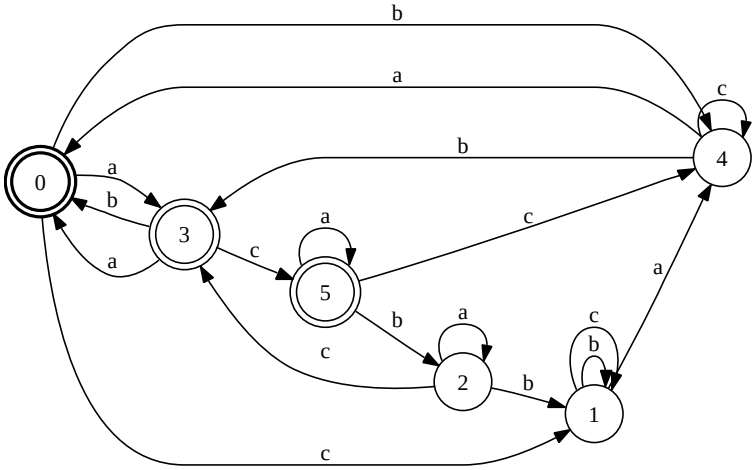
145M.fsm



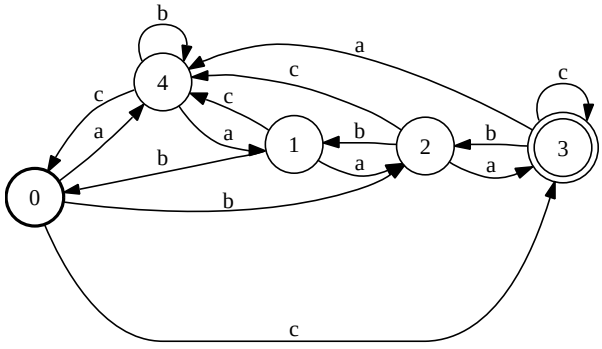
146M.fsm



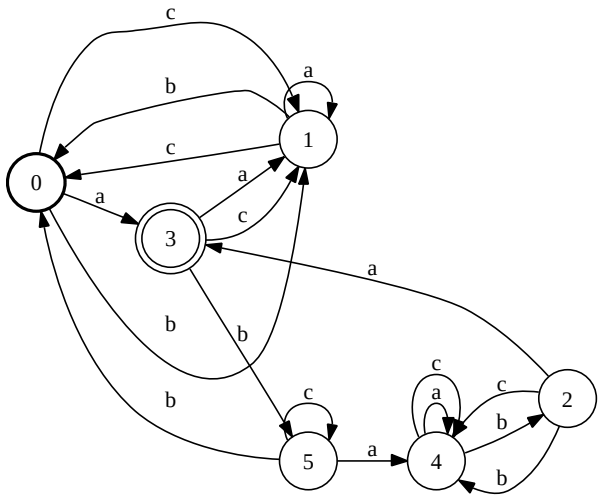
147M.fsm



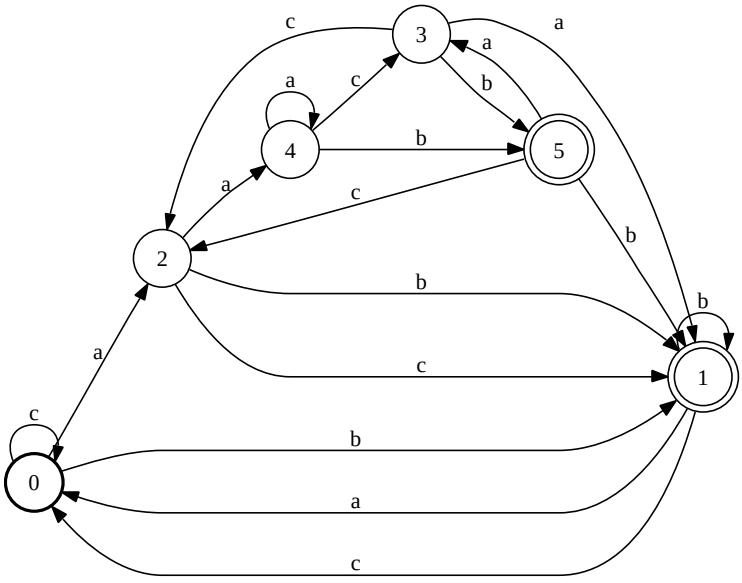
148M.fsm



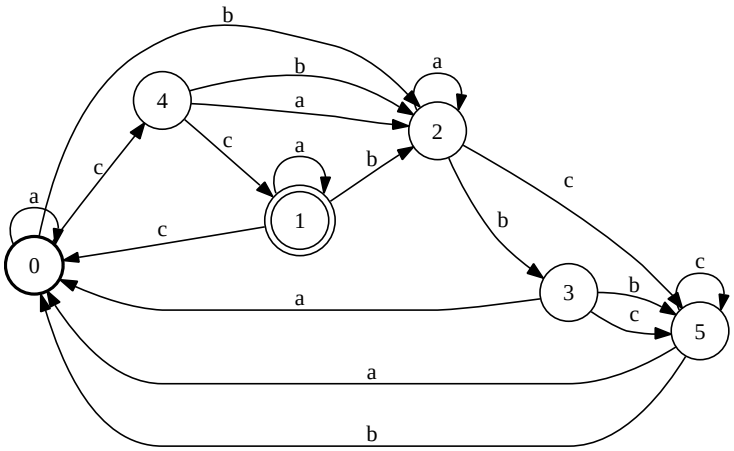
149M.fsm



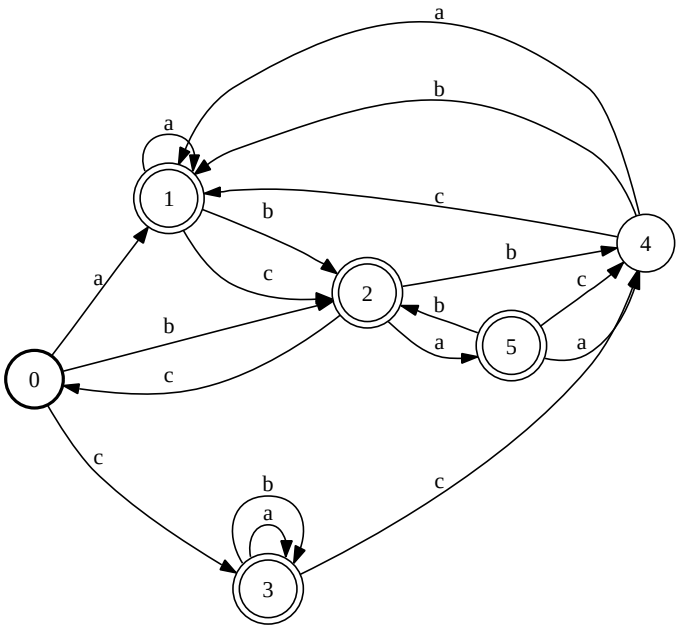
150M.fsm



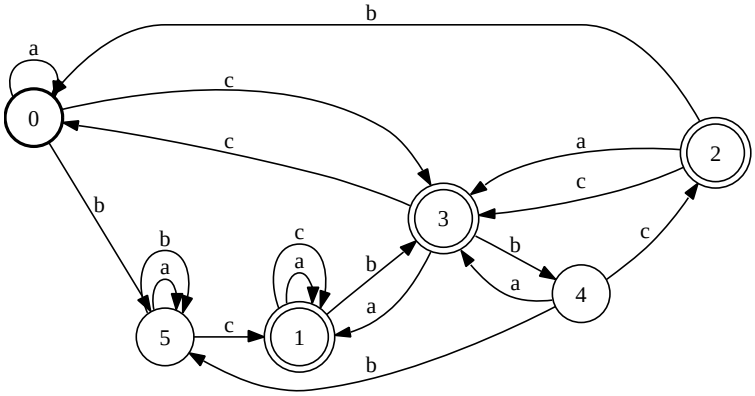
151M.fsm



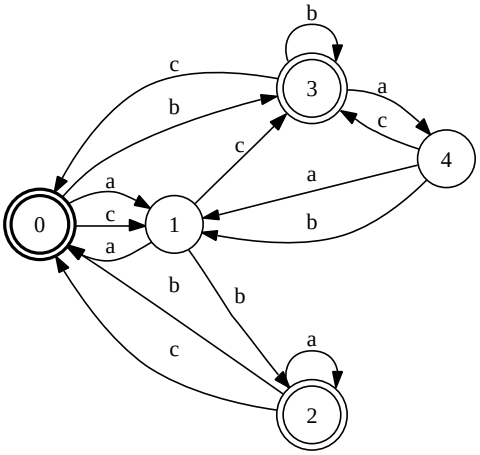
152M.fsm



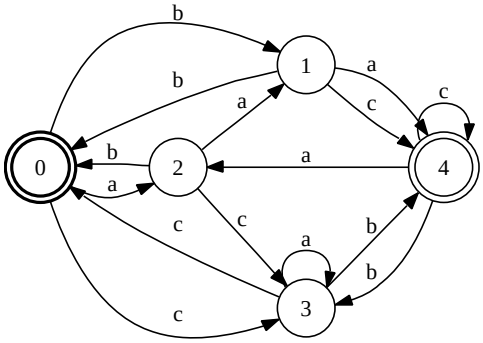
153M.fsm



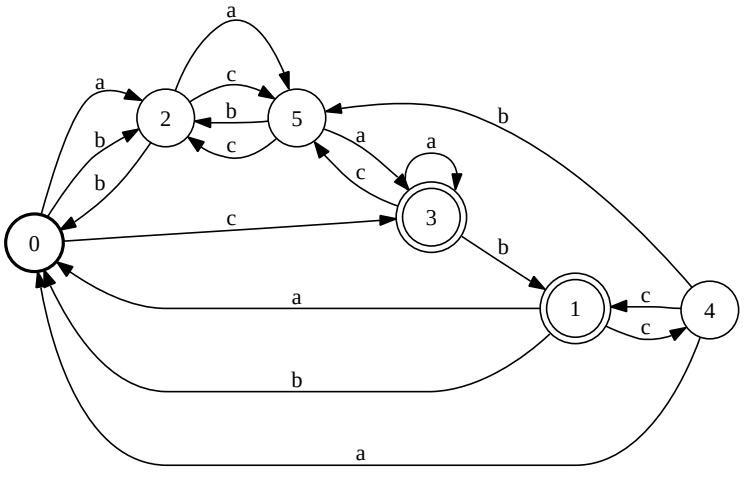
154M.fsm



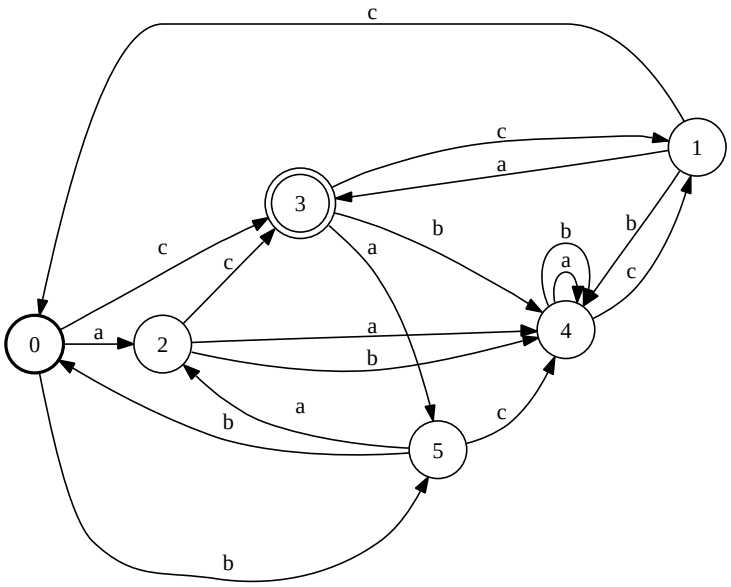
155M.fsm



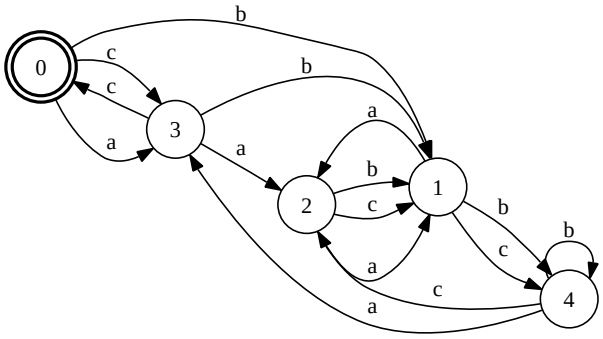
156M.fsm



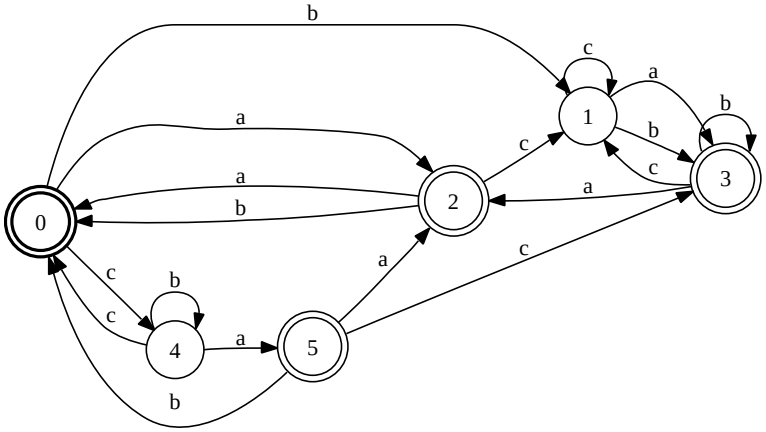
157M.fsm



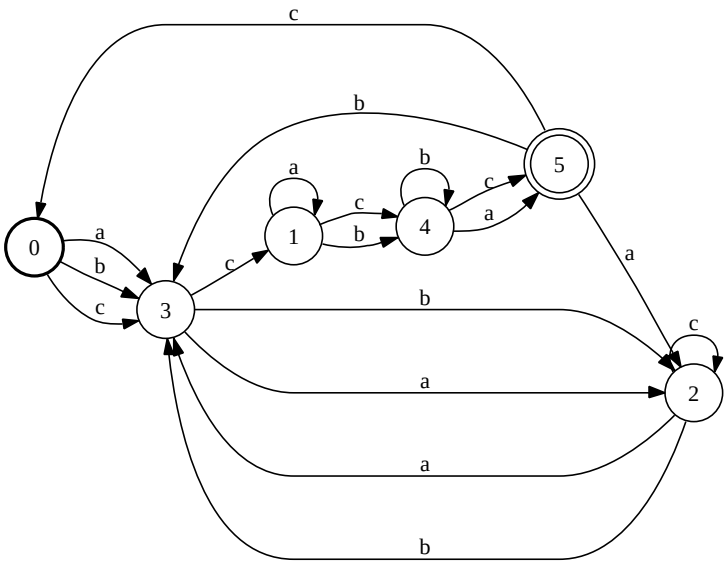
158M.fsm



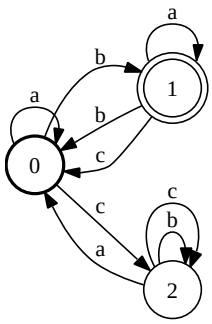
159M.fsm



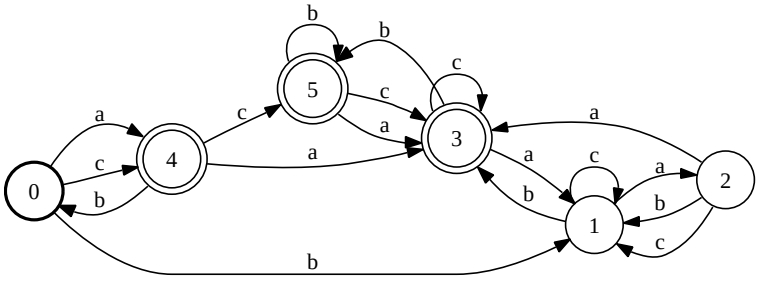
160M.fsm



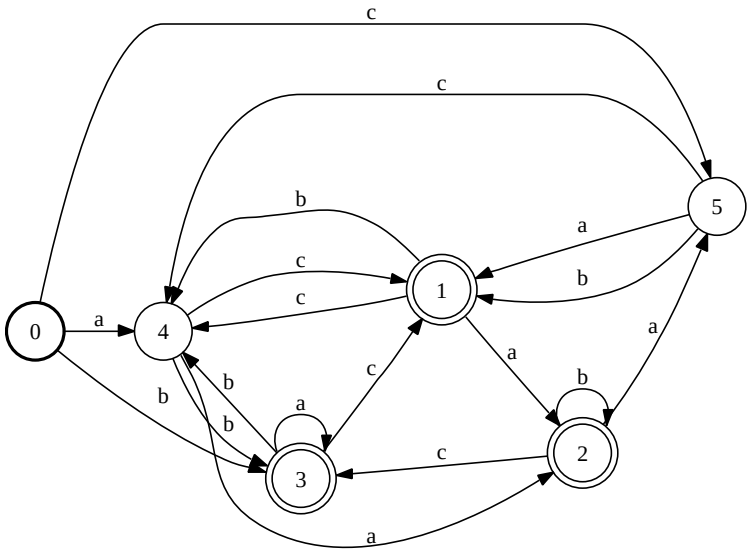
161M.fsm



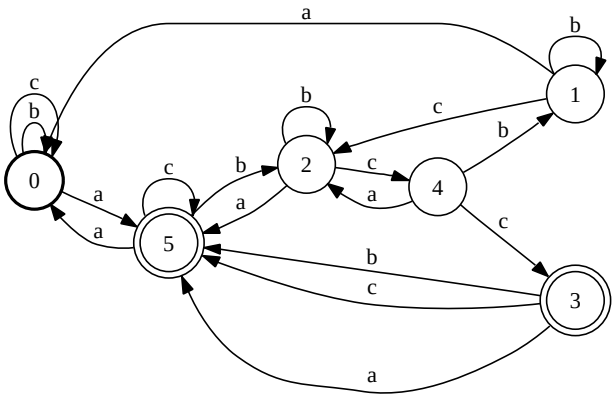
162M.fsm



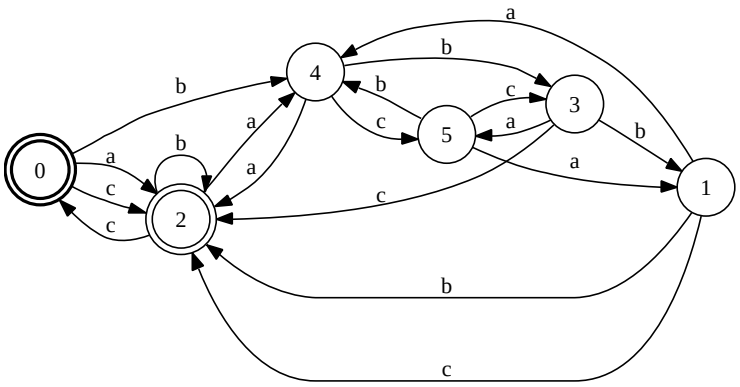
163M.fsm



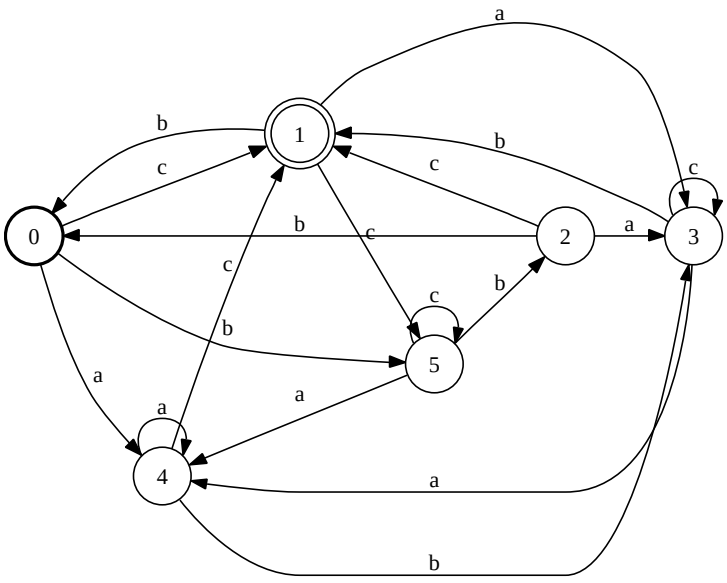
164M.fsm



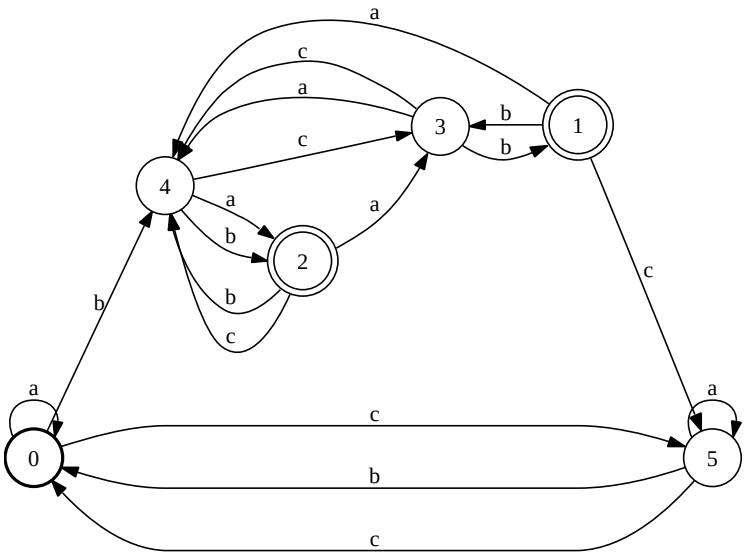
165M.fsm



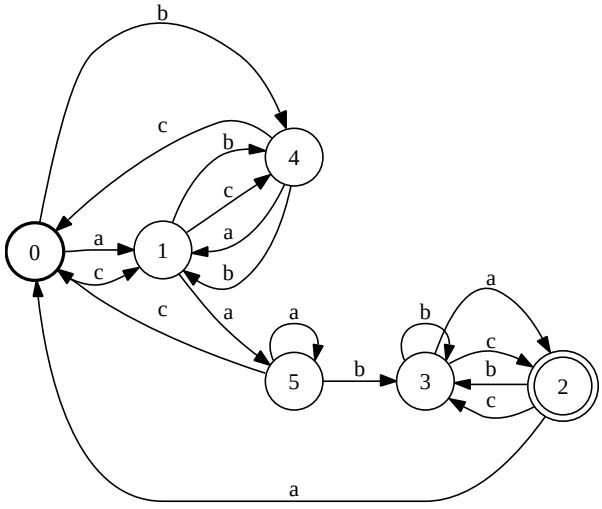
166M.fsm



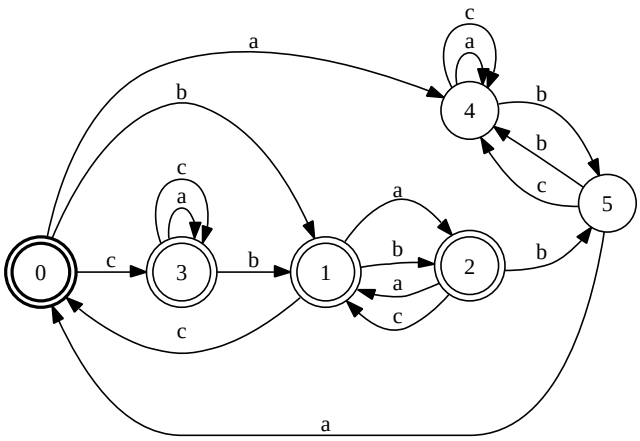
167M.fsm



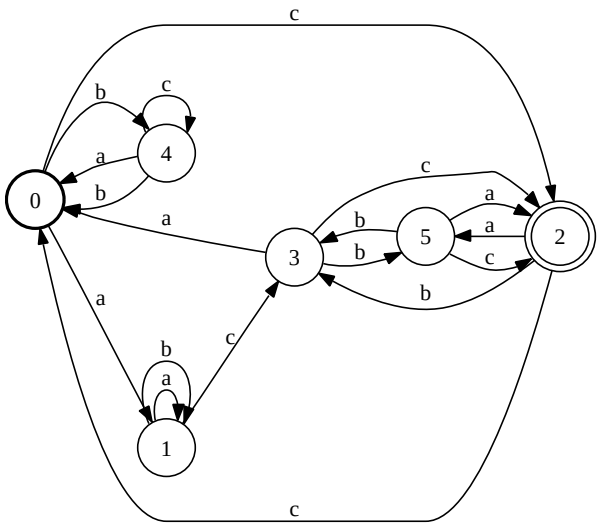
168M.fsm



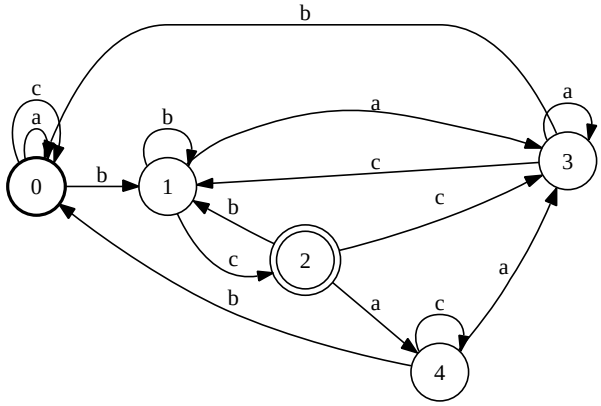
169M.fsm



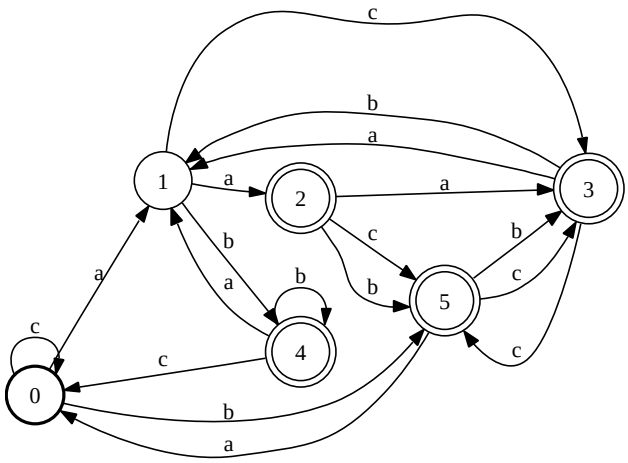
170M.fsm



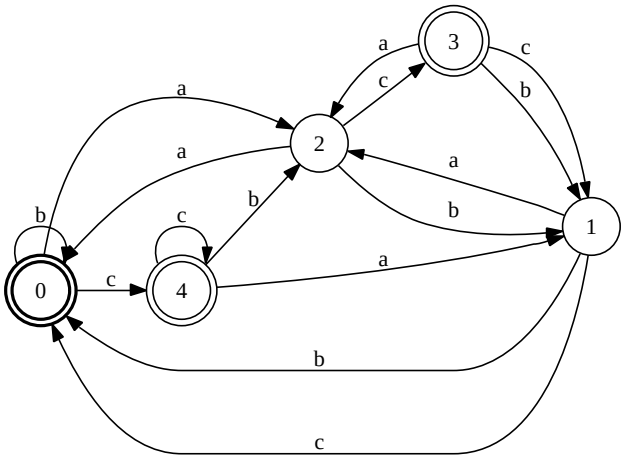
171M.fsm



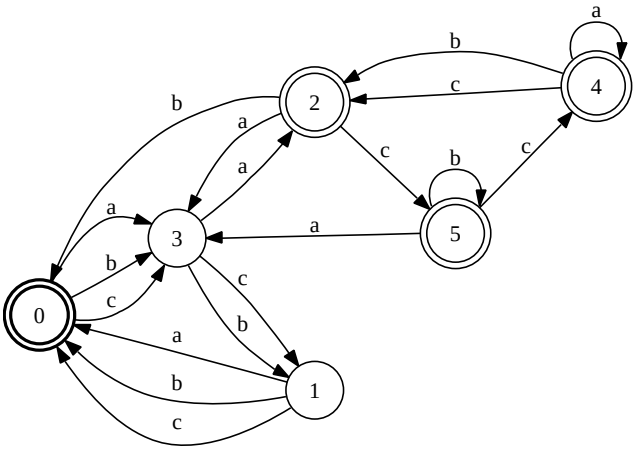
172M.fsm



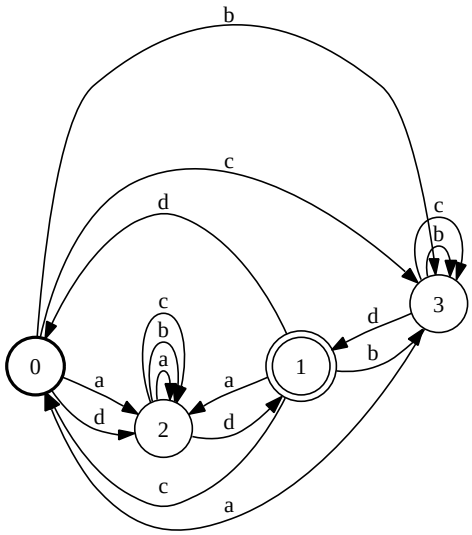
173M.fsm



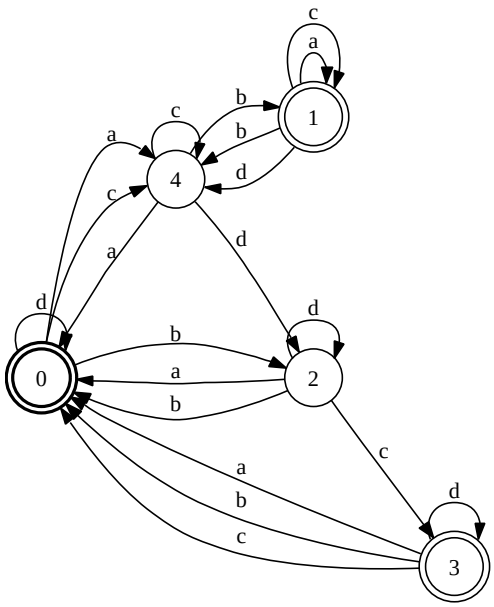
174M.fsm



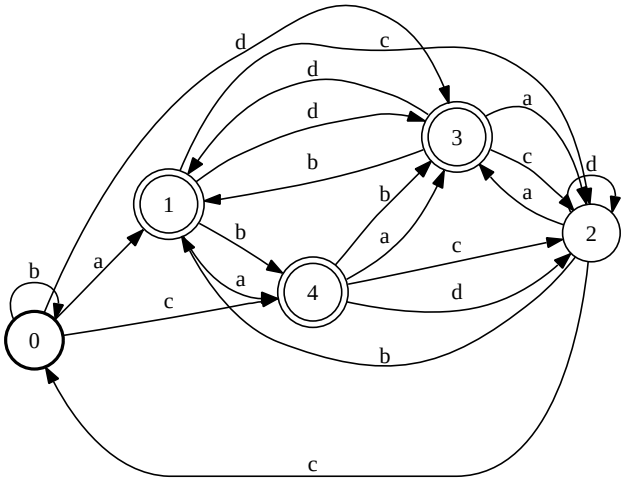
175M.fsm



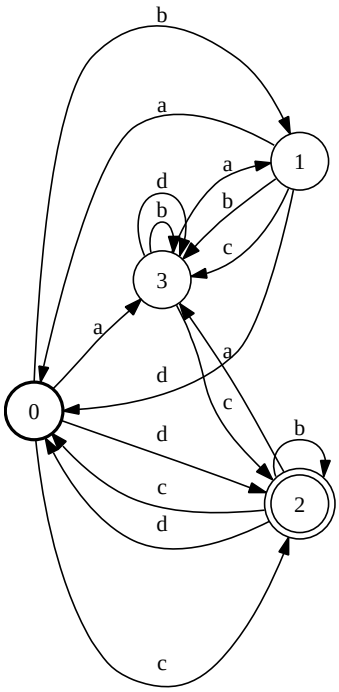
176M.fsm



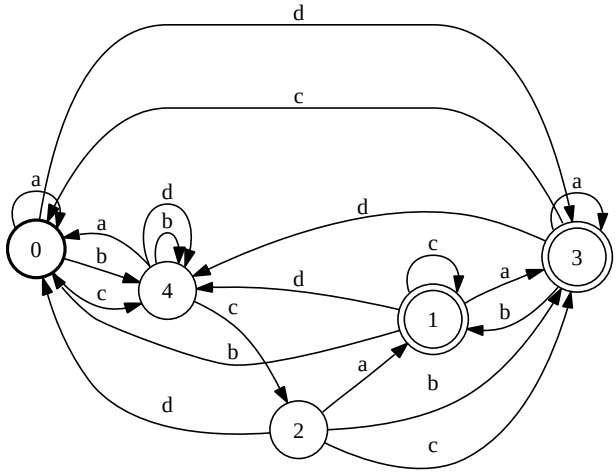
177M.fsm



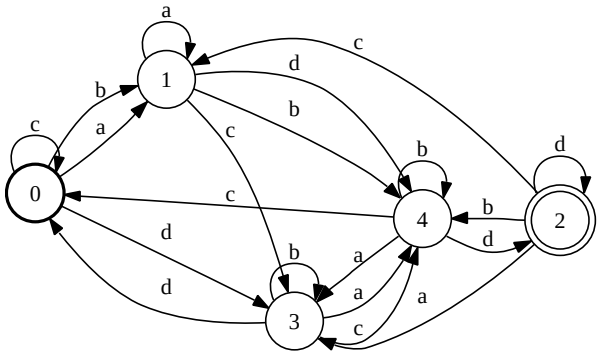
178M.fsm



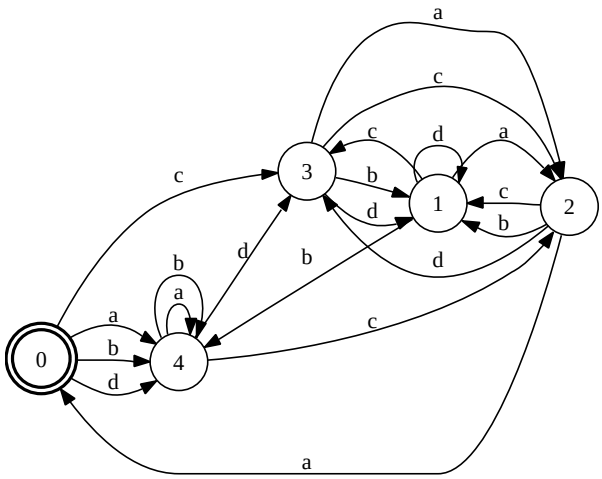
179M.fsm



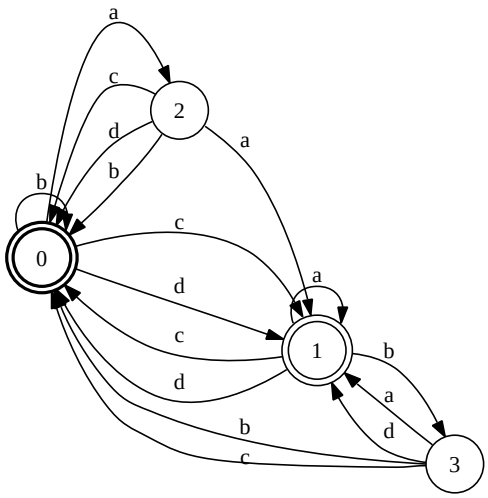
180M.fsm



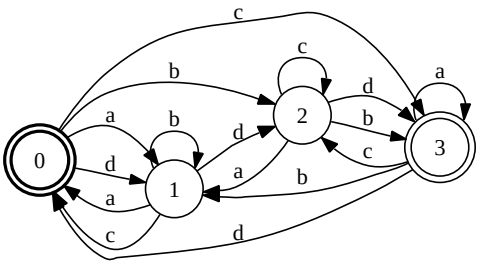
181M.fsm



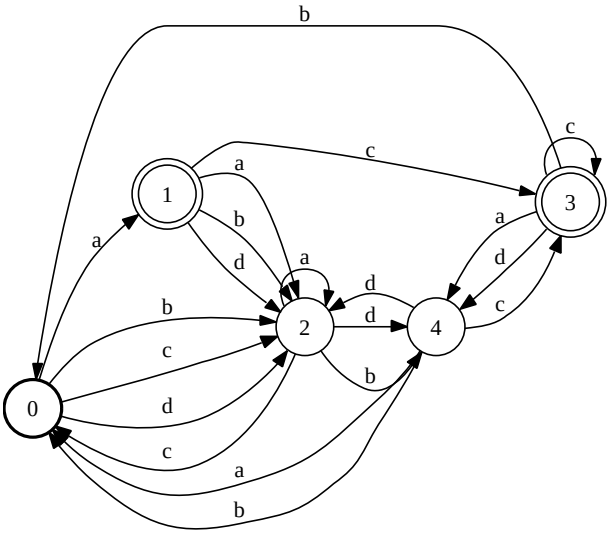
182M.fsm



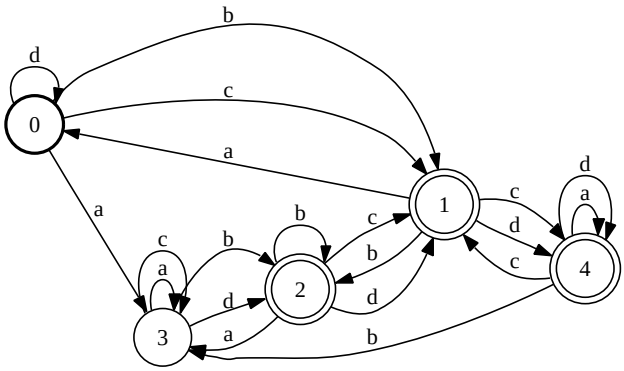
183M.fsm



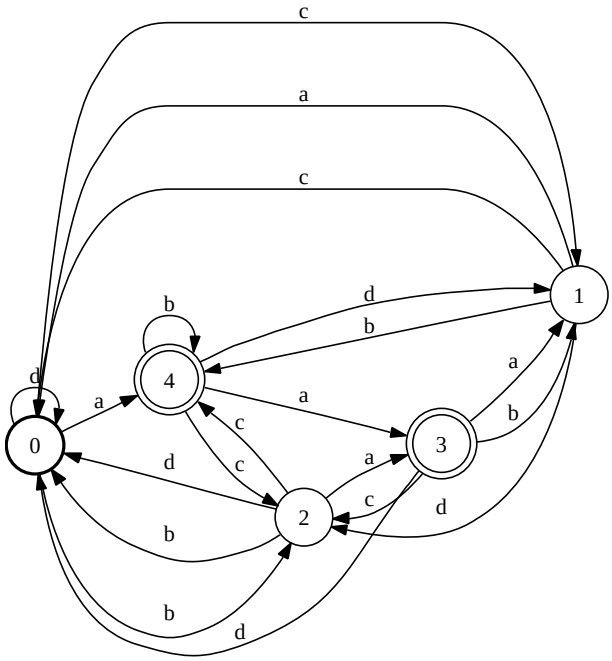
184M.fsm



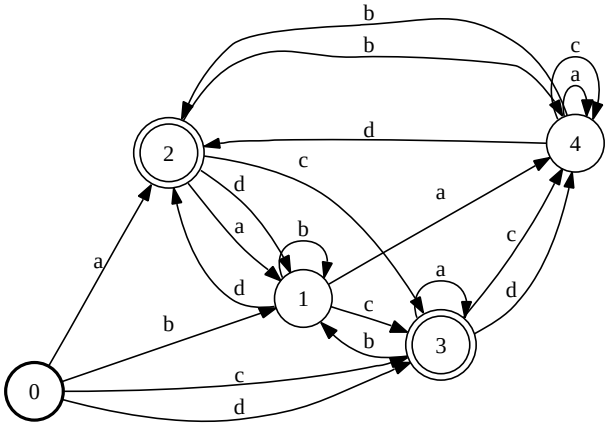
185M.fsm



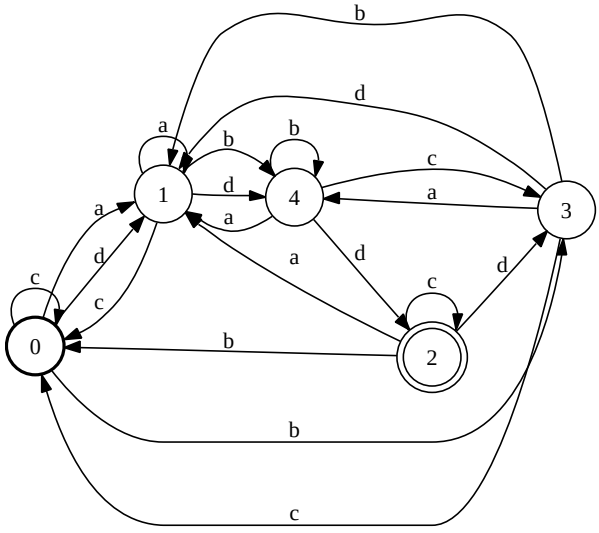
186M.fsm



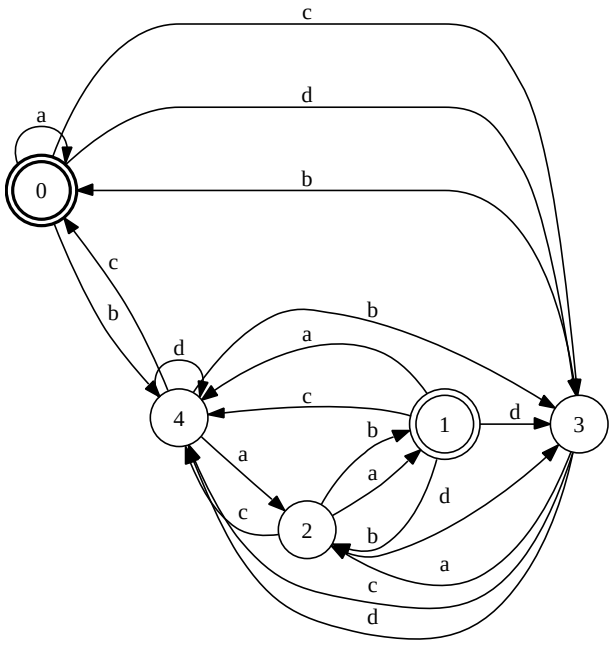
187M.fsm



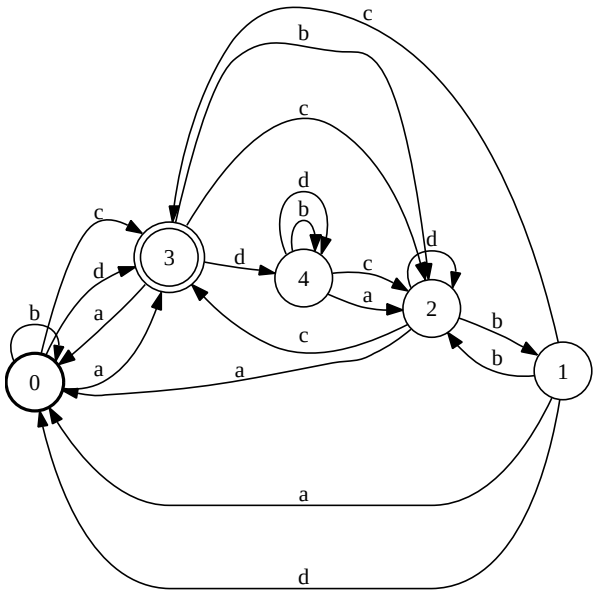
188M.fsm



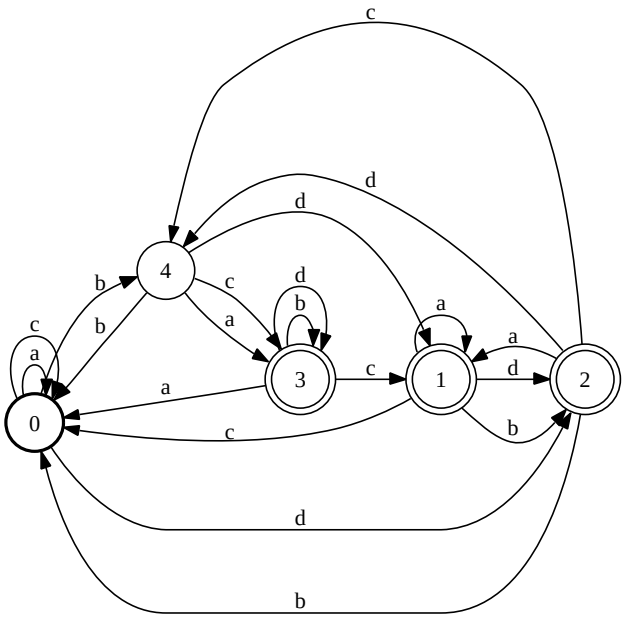
189M.fsm



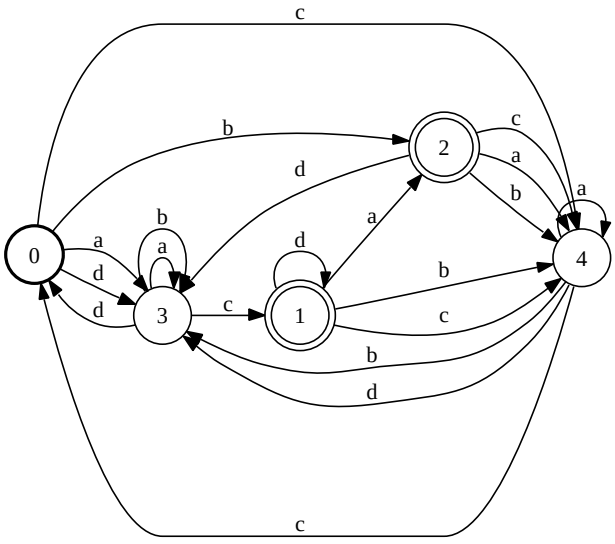
190M.fsm



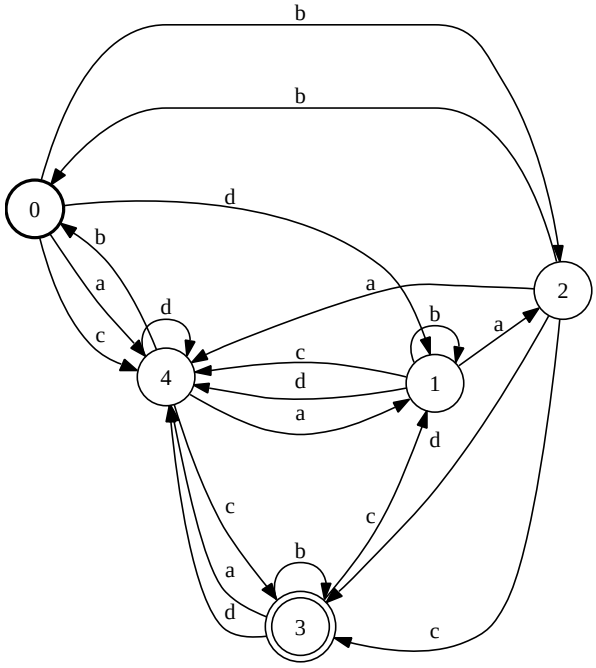
191M.fsm



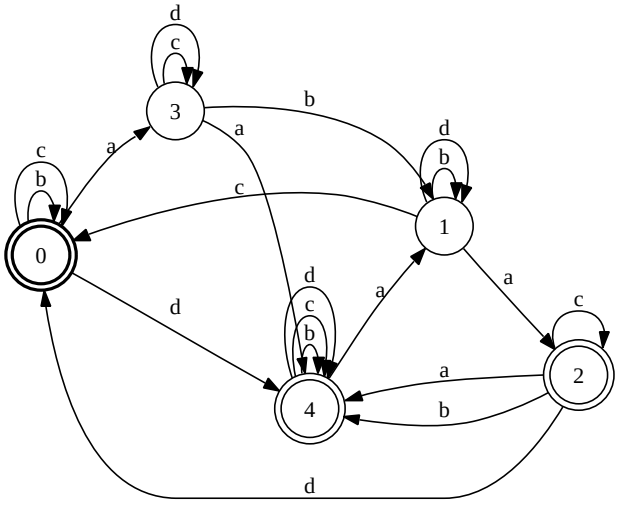
192M.fsm



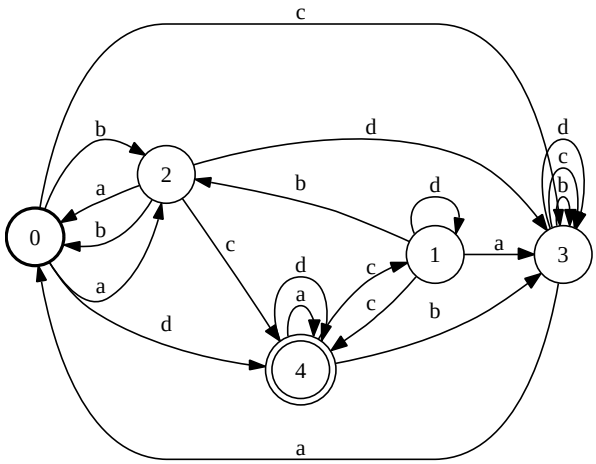
193M.fsm



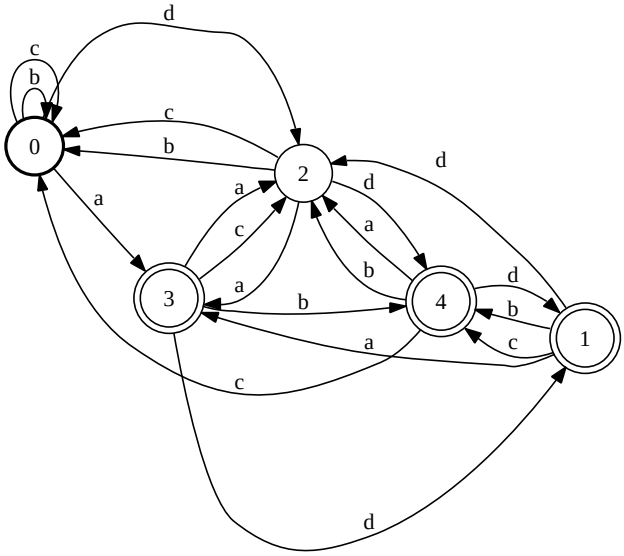
194M.fsm



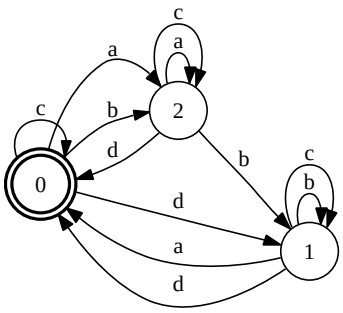
195M.fsm



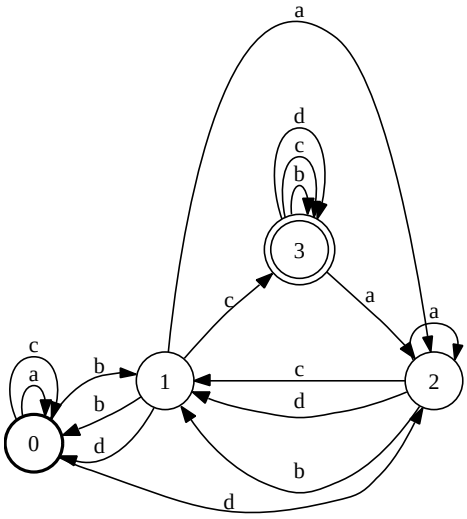
196M.fsm



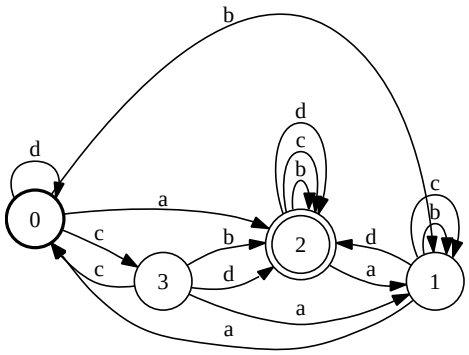
197M.fsm



198M.fsm

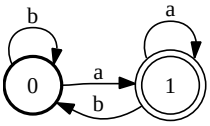


199M.fsm

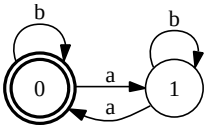


200M.fsm

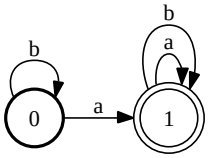
Test 2. DFA **test set**



002.0.fsm

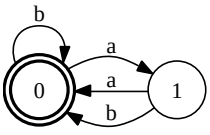


002.1.fsm

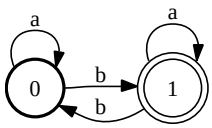


002.2.fsm

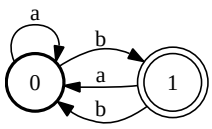
002.3.fsm



002.4.fsm

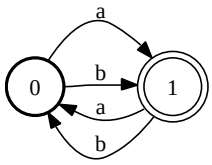


002.5.fsm

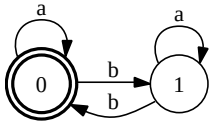


002.6.fsm

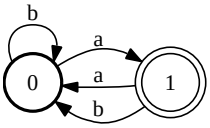
002.7.fsm



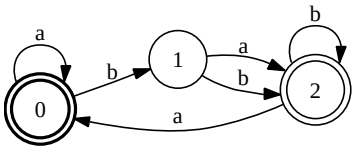
002.8.fsm



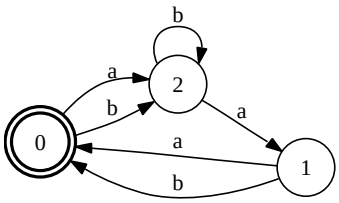
002.9.fsm



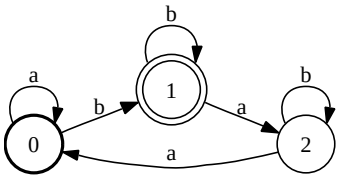
002.10.fsm



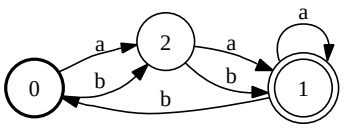
003.0.fsm



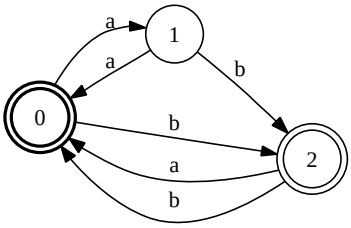
003.1.fsm



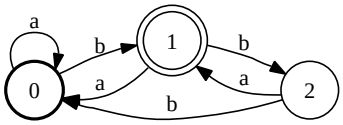
003.2.fsm



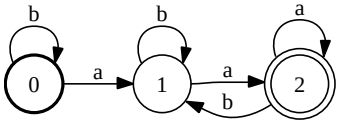
003.3.fsm



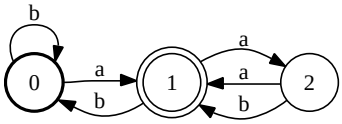
003.4.fsm



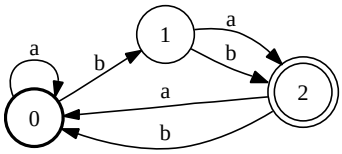
003.5.fsm



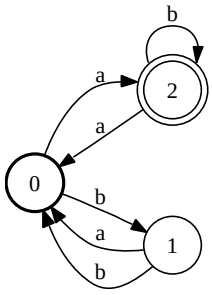
003.6.fsm



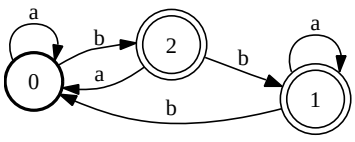
003.7.fsm



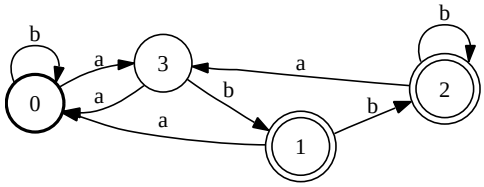
003.8.fsm



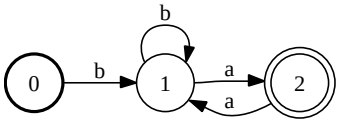
003.9.fsm



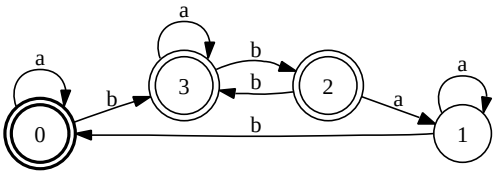
003.10.fsm



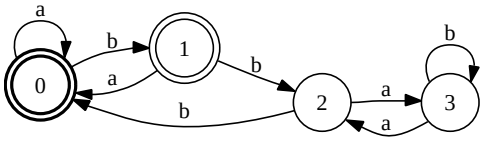
004.0.fsm



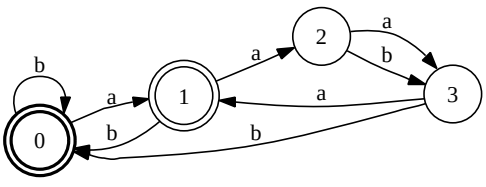
004.1.fsm



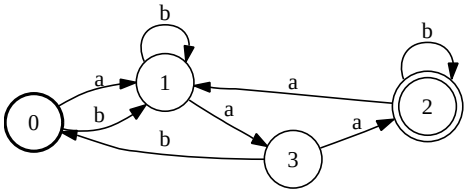
004.2.fsm



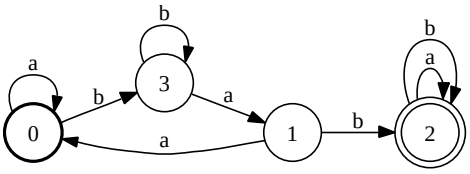
004.3.fsm



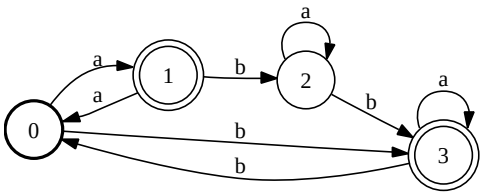
004.4.fsm



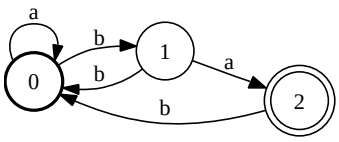
004.5.fsm



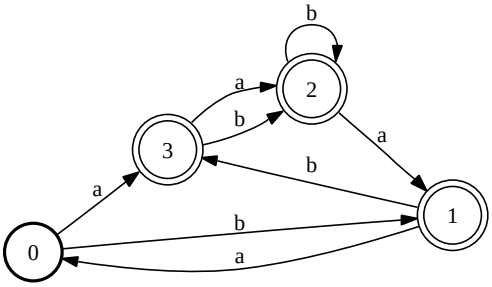
004.6.fsm



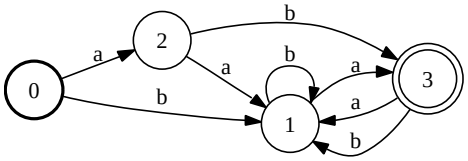
004.7.fsm



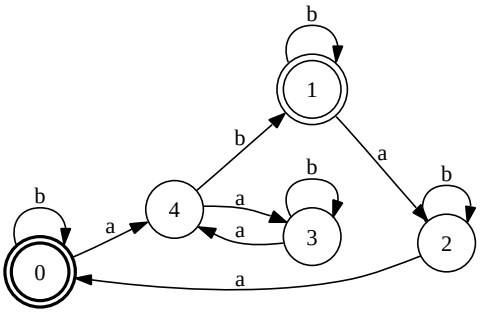
004.8.fsm



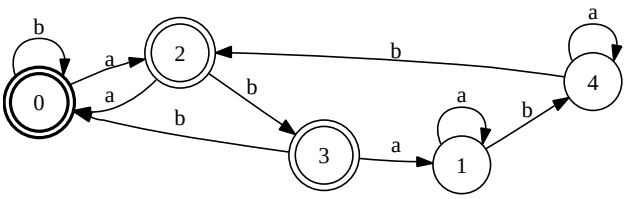
004.9.fsm



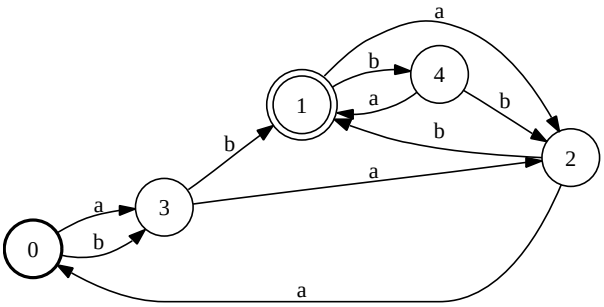
004.10.fsm



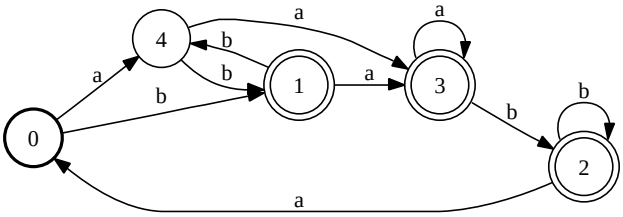
005.0.fsm



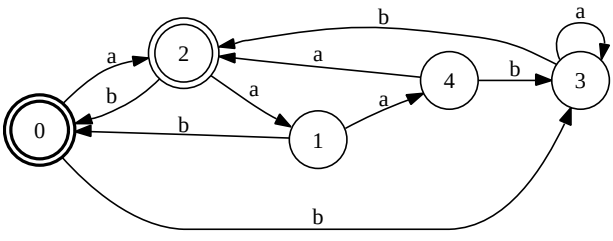
005.1.fsm



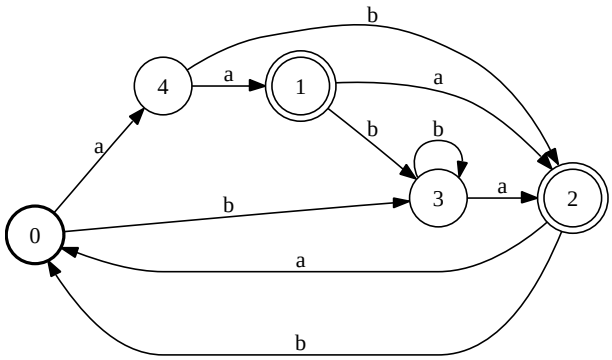
005.2.fsm



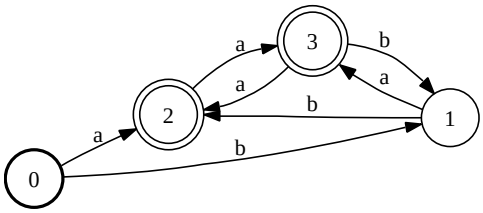
005.3.fsm



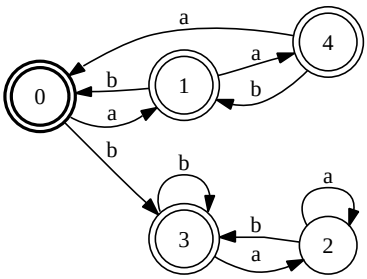
005.4.fsm



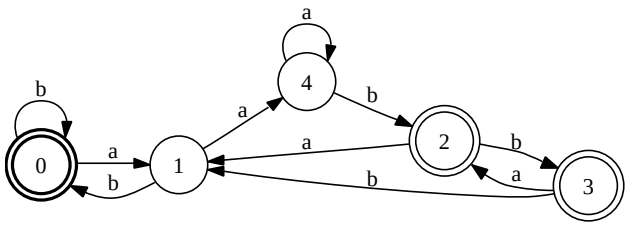
005.5.fsm



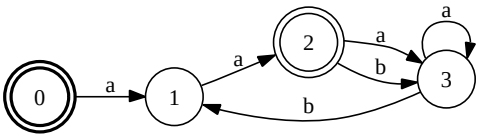
005.6.fsm



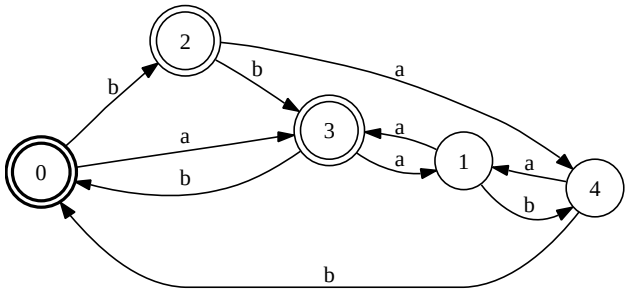
005.7.fsm



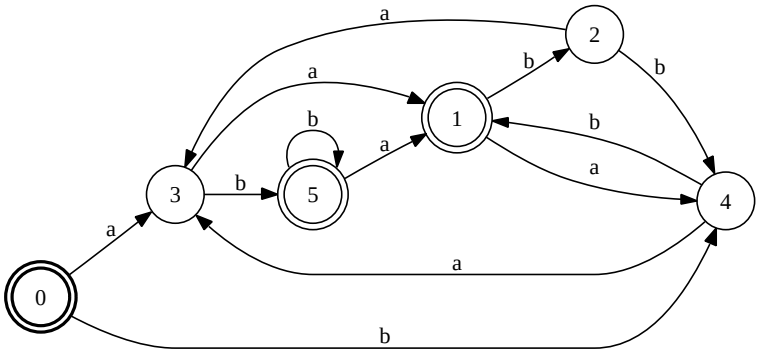
005.8.fsm



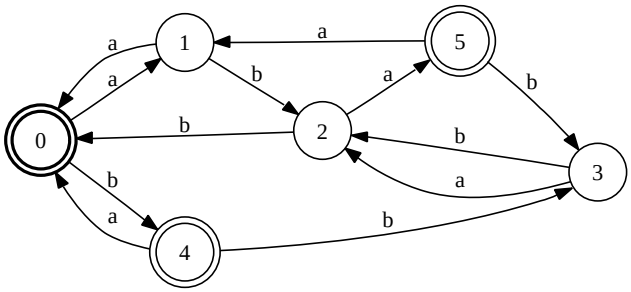
005.9.fsm



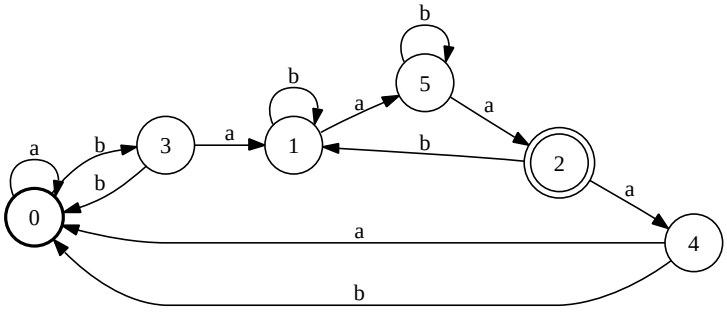
005.10.fsm



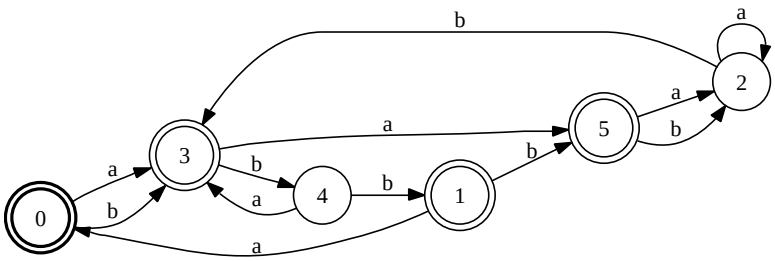
006.0.fsm



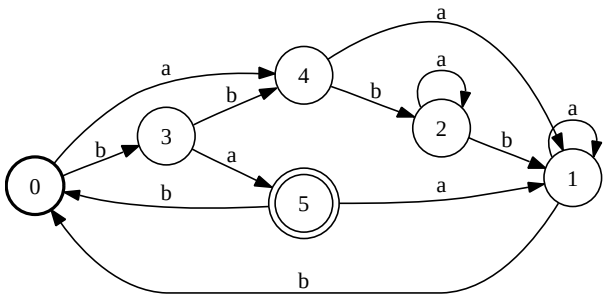
006.1.fsm



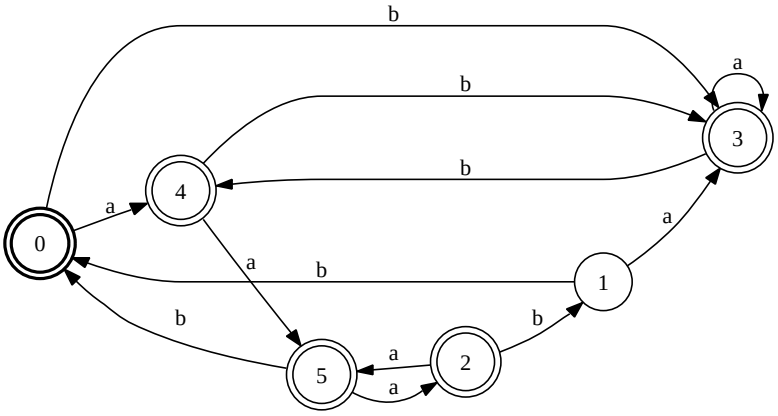
006.2.fsm



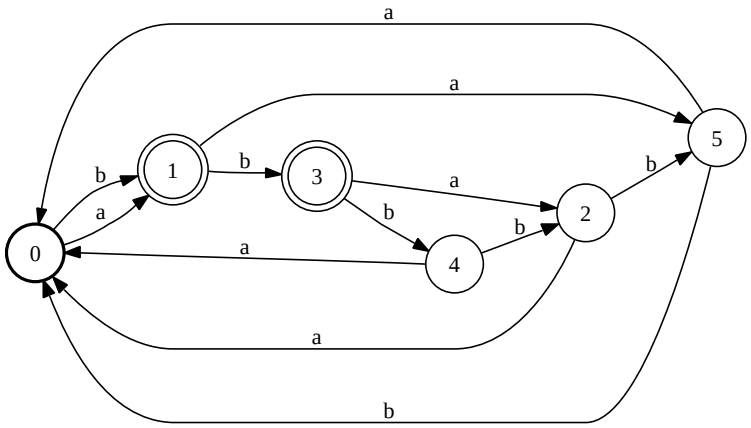
006.3.fsm



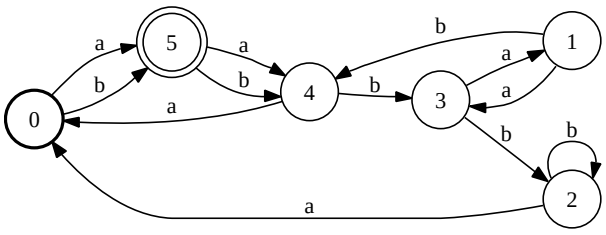
006.4.fsm



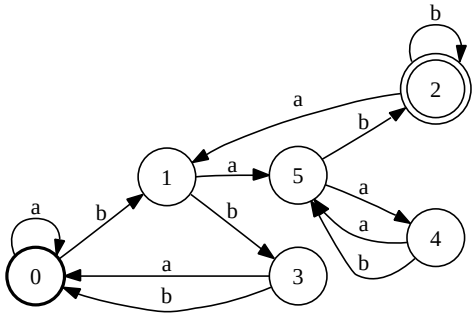
006.5.fsm



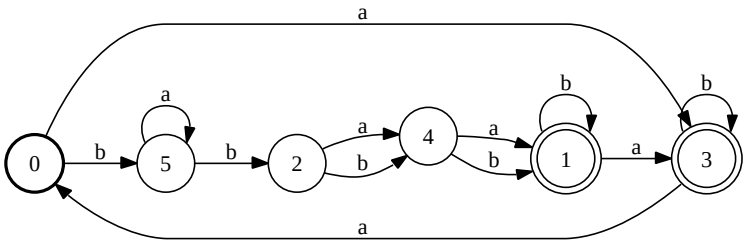
006.6.fsm



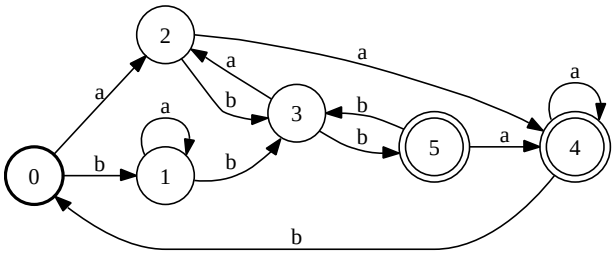
006.7.fsm



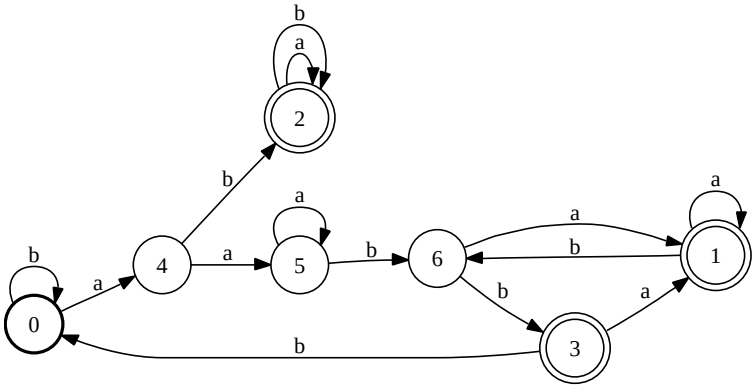
006.8.fsm



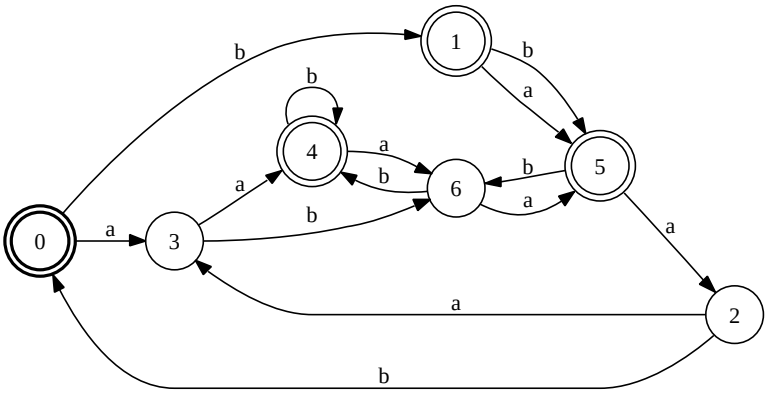
006.9.fsm



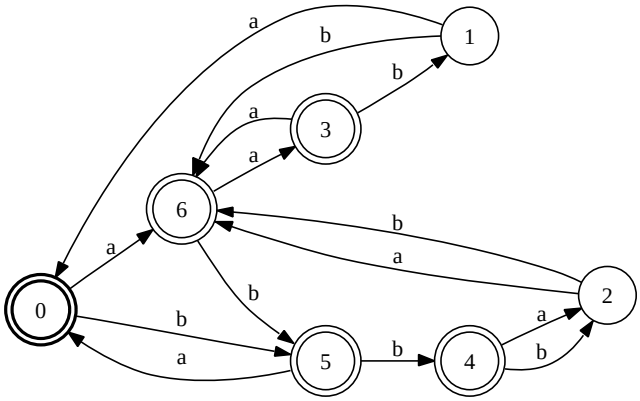
006.10.fsm



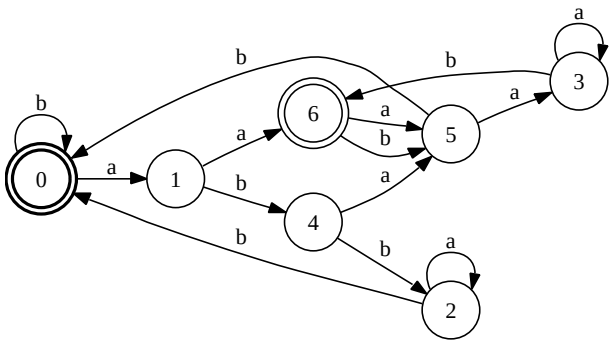
007.0.fsm



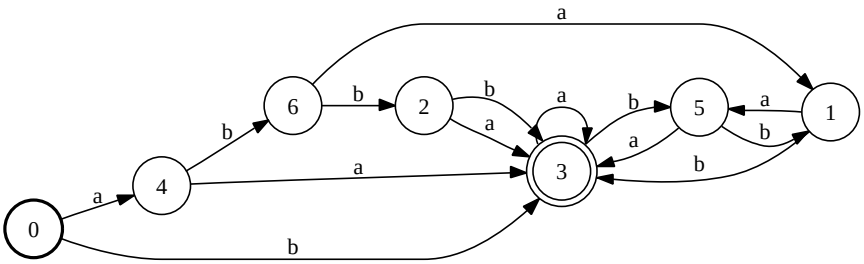
007.1.fsm



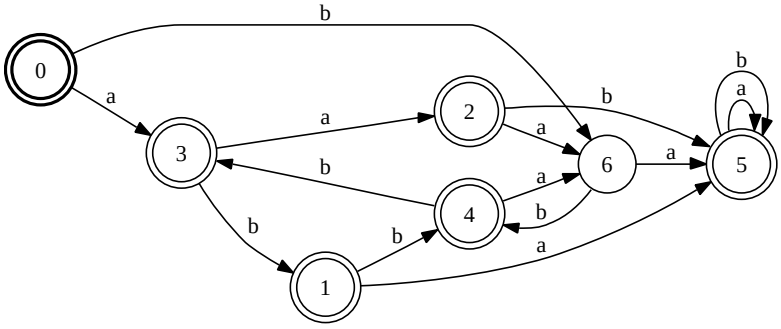
007.2.fsm



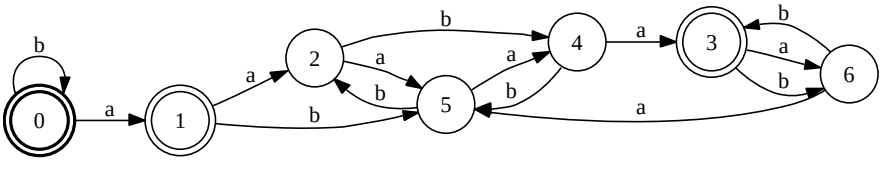
007.3.fsm



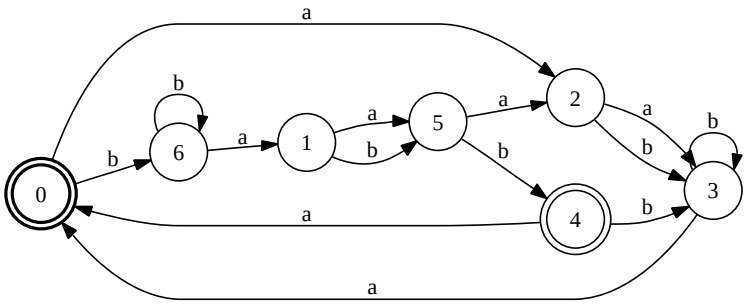
007.4.fsm



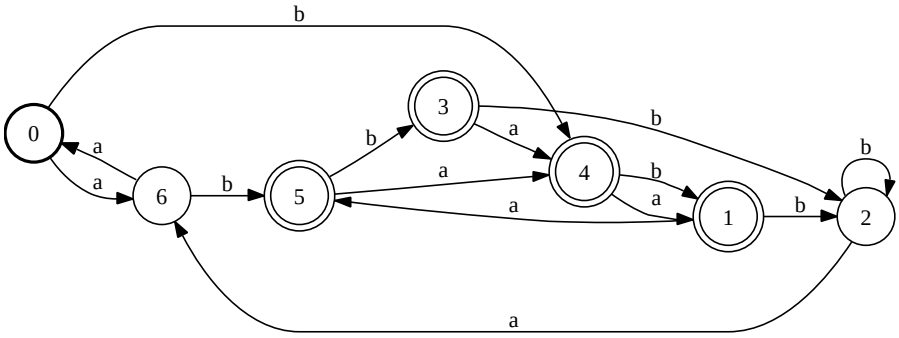
007.5.fsm



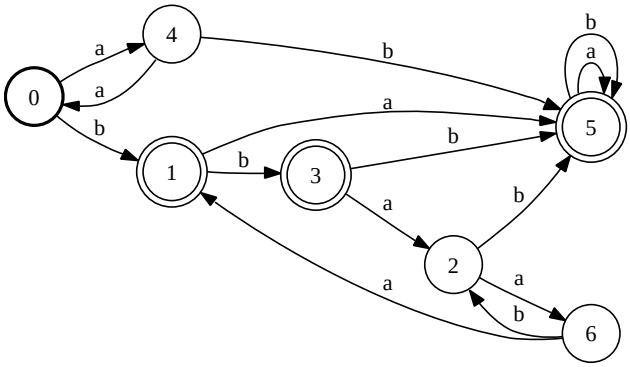
007.6.fsm



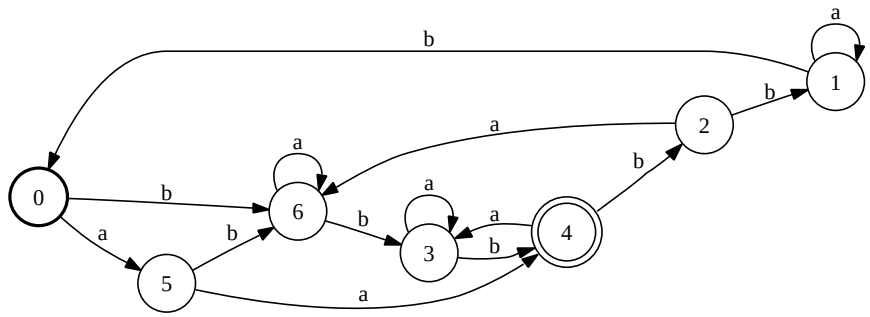
007.7.fsm



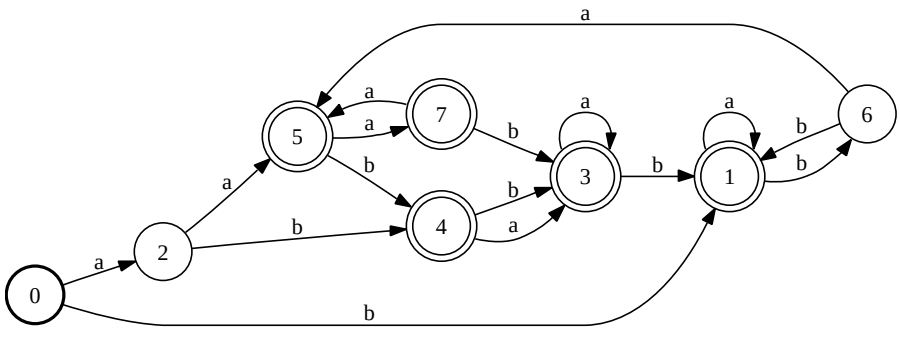
007.8.fsm



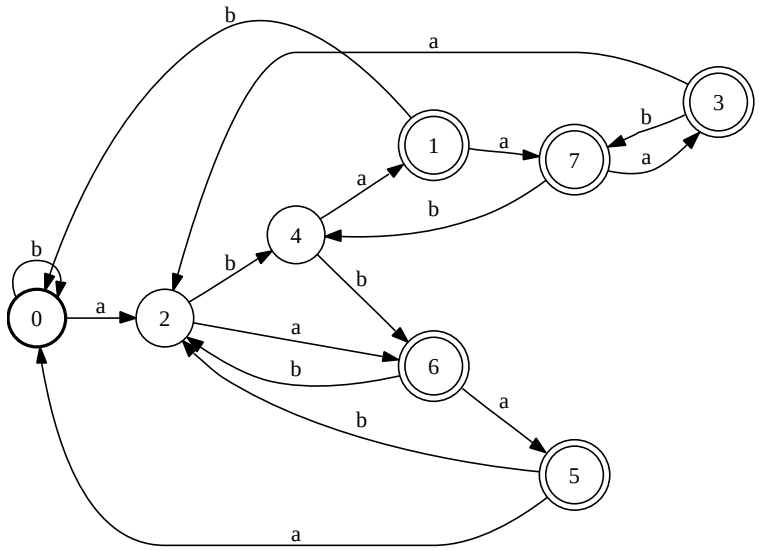
007.9.fsm



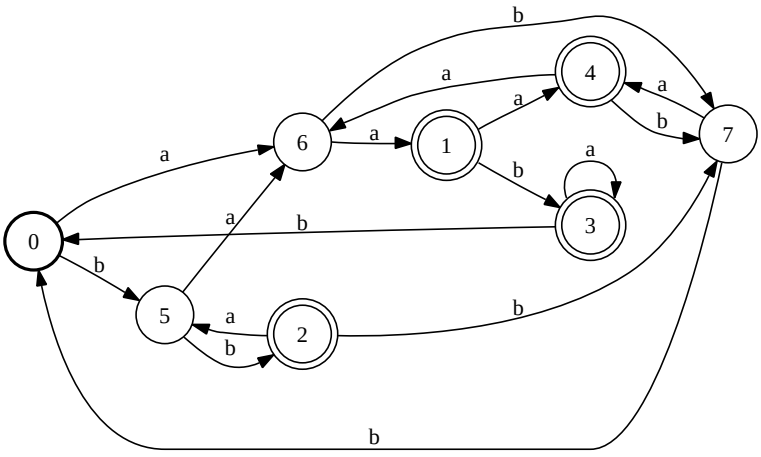
007.10.fsm



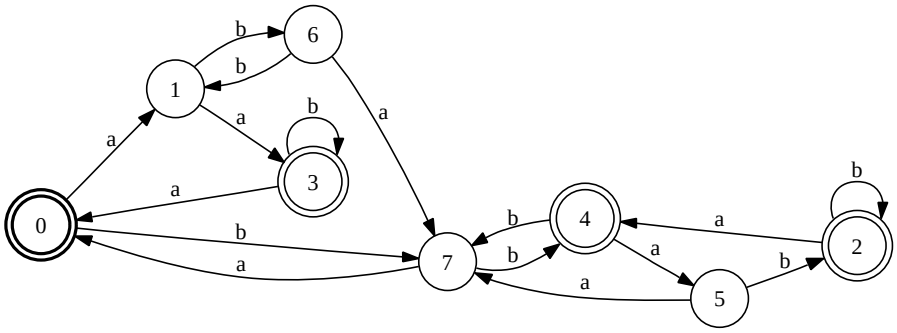
008.0.fsm



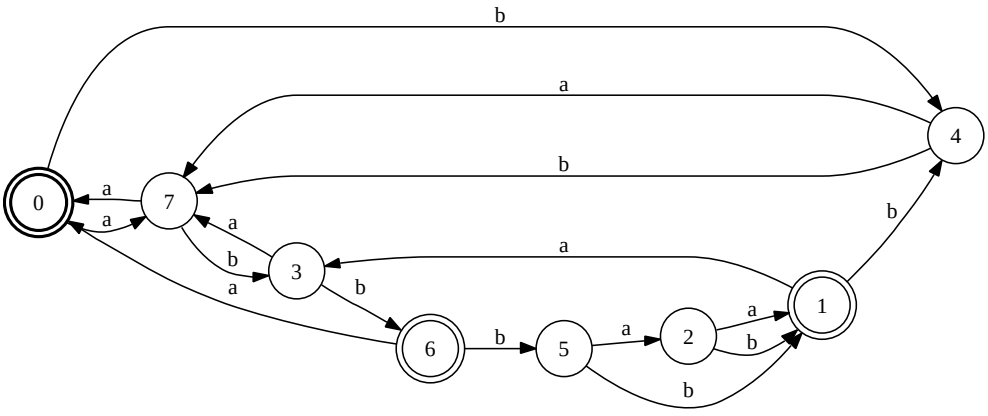
008.1.fsm



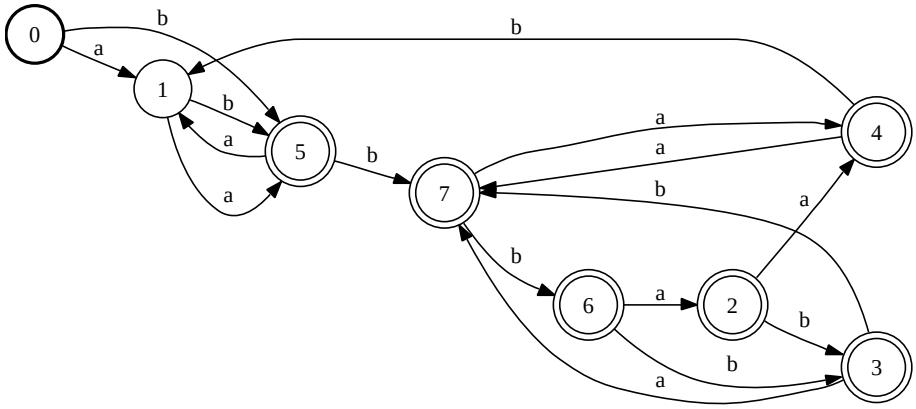
008.2.fsm



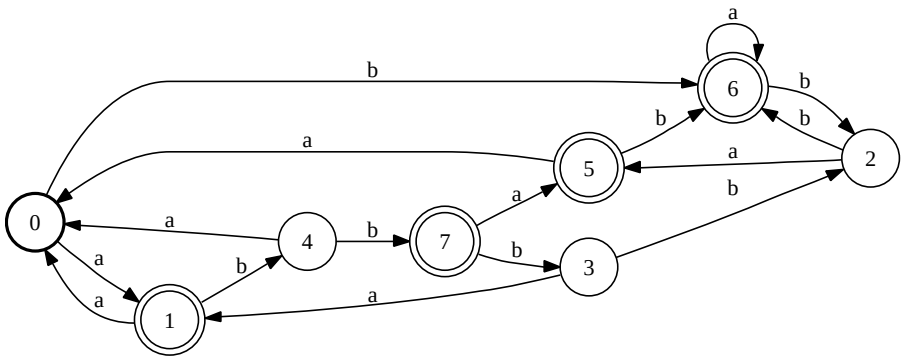
008.3.fsm



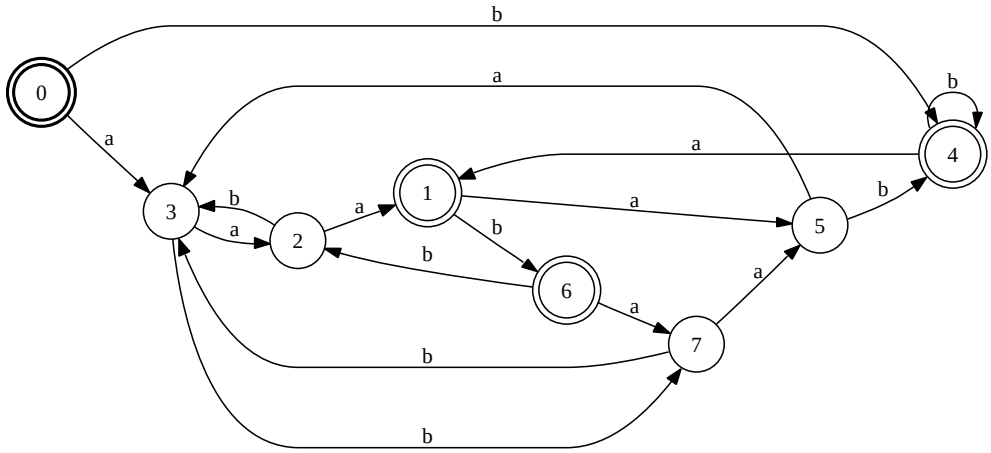
008.4.fsm



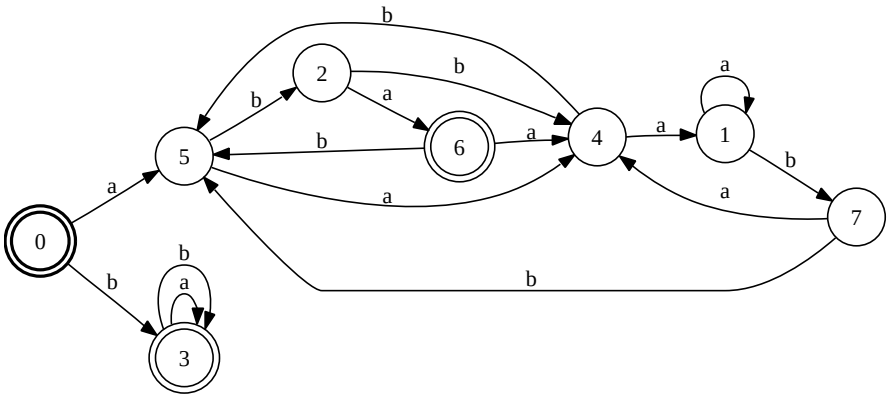
008.5.fsm



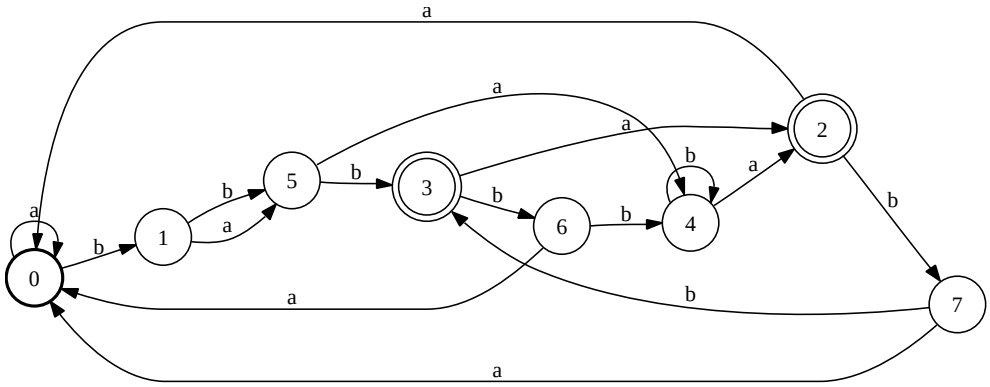
008.6.fsm



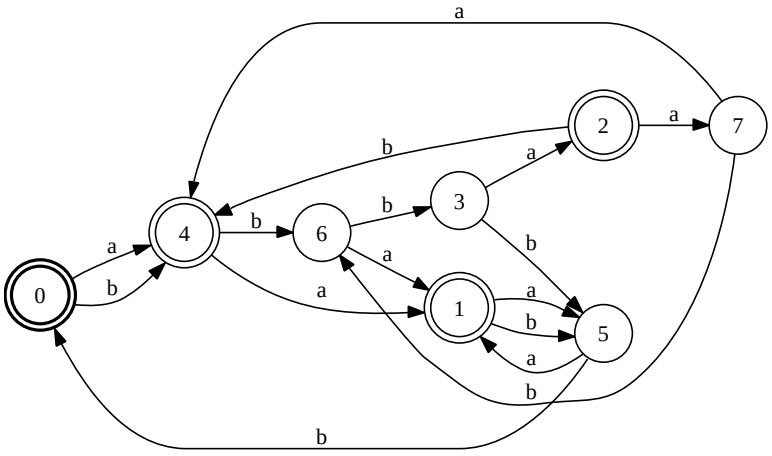
008.7.fsm



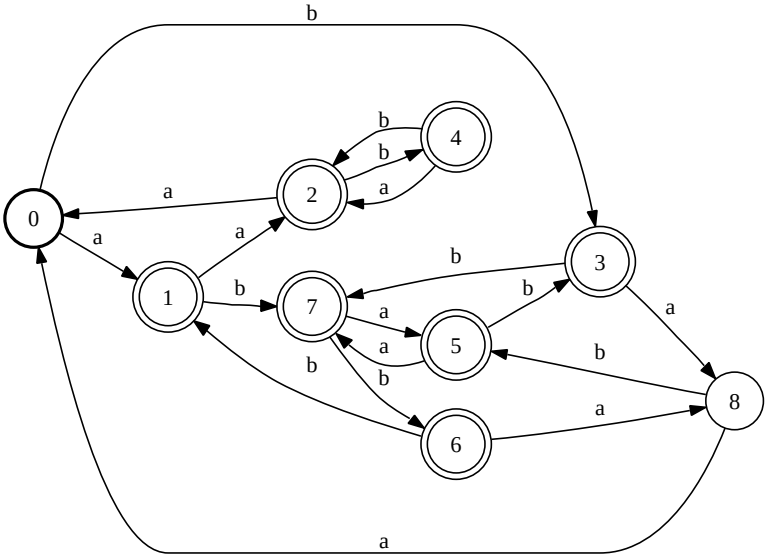
008.8.fsm



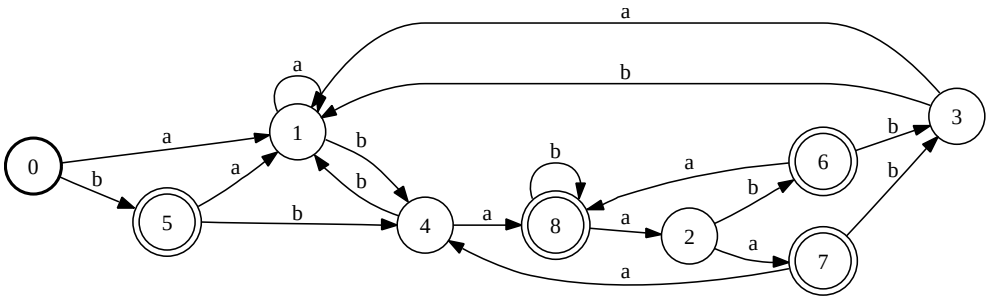
008.9.fsm



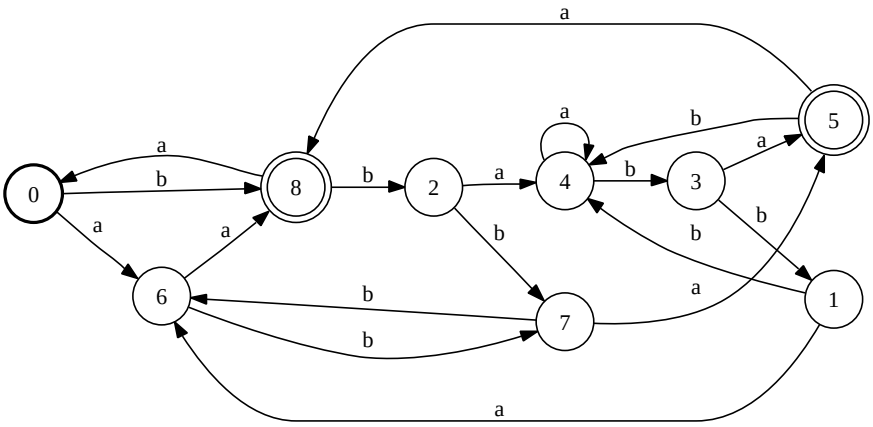
008.10.fsm



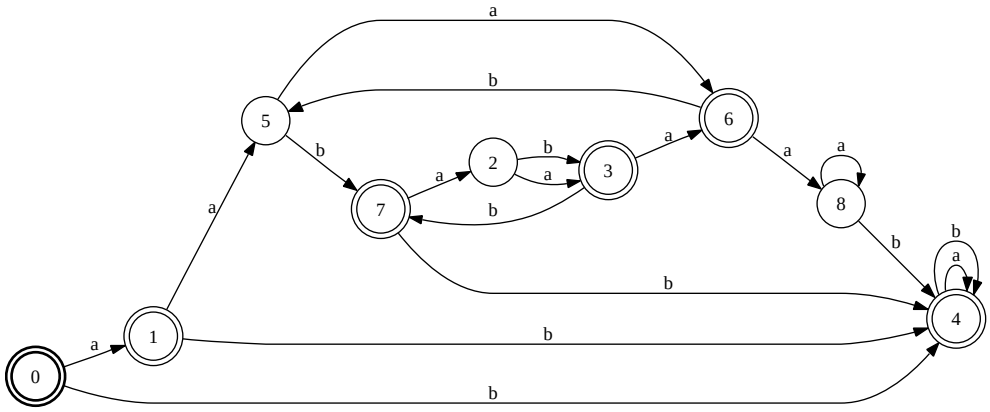
009.0.fsm



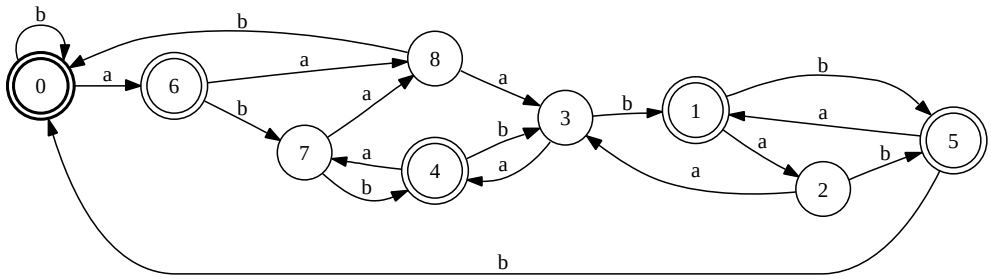
009.1.fsm



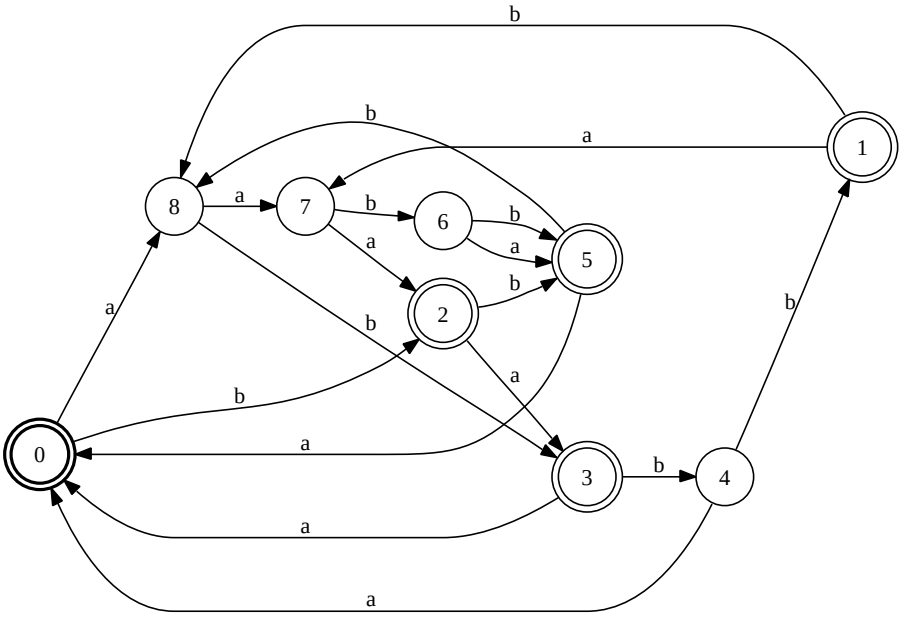
009.2.fsm



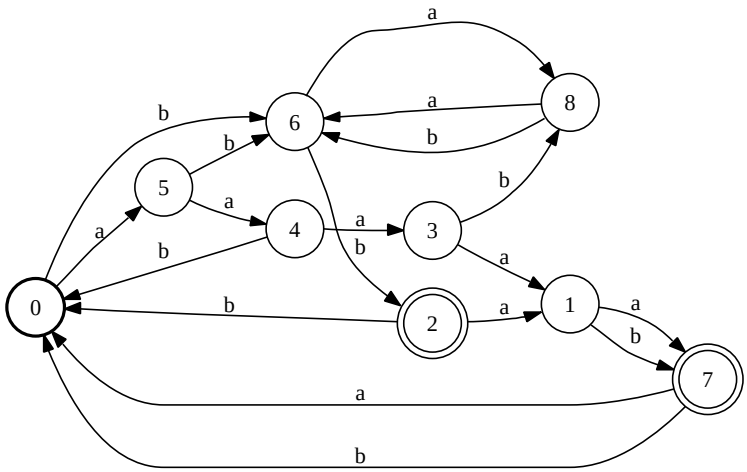
009.3.fsm



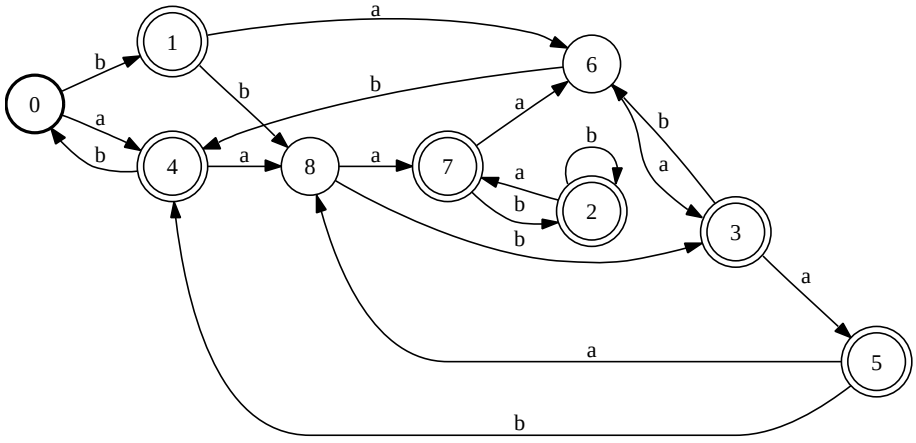
009.4.fsm



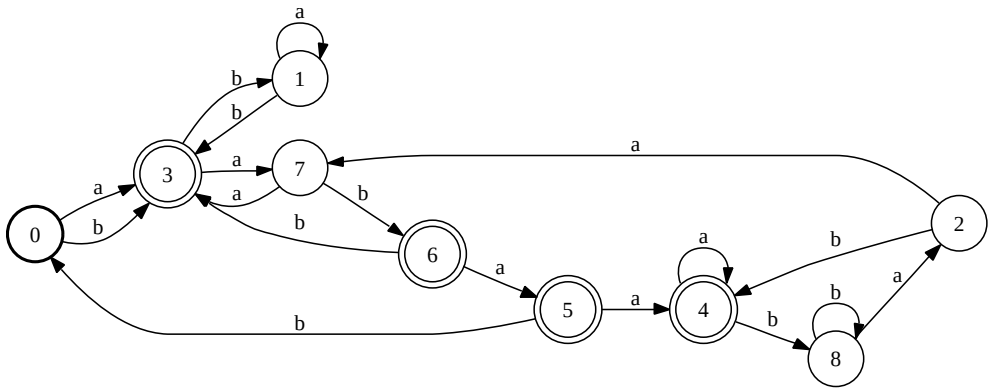
009.5.fsm



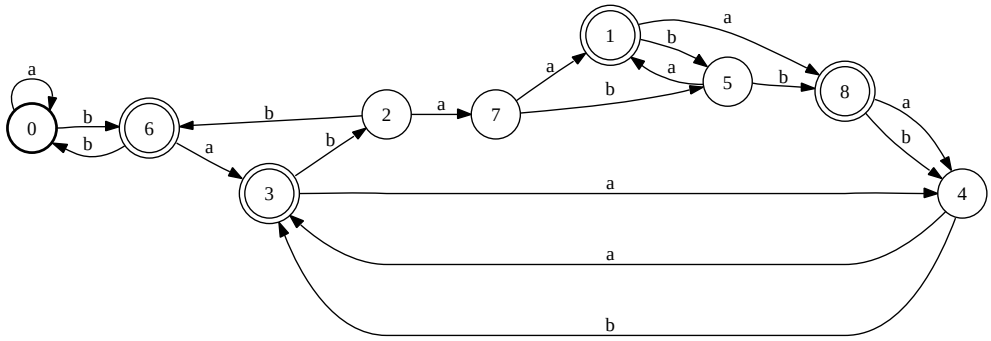
009.6.fsm



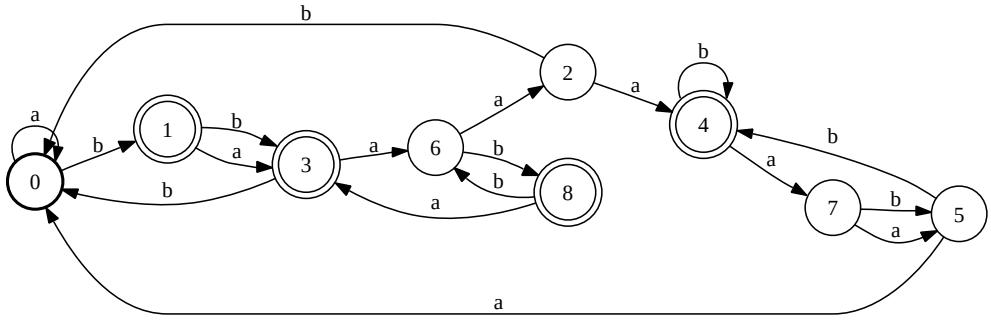
009.7.fsm



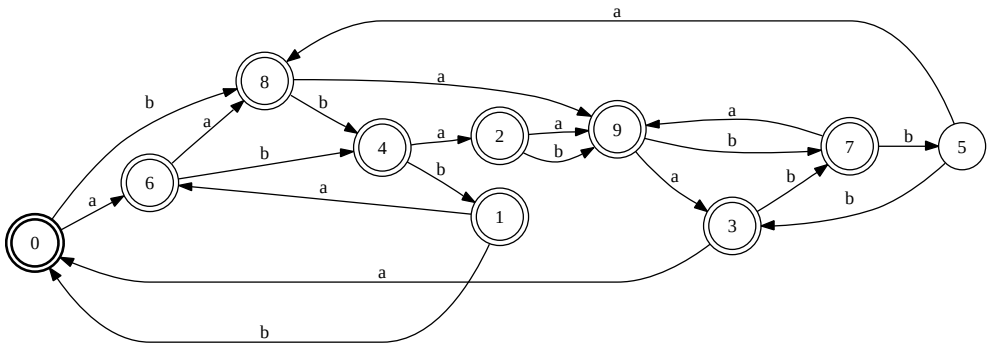
009.8.fsm



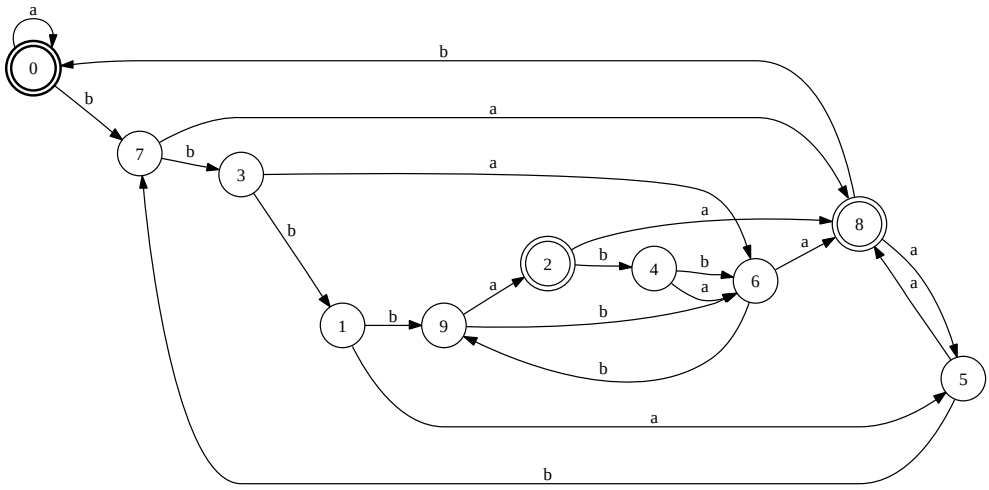
009.9.fsm



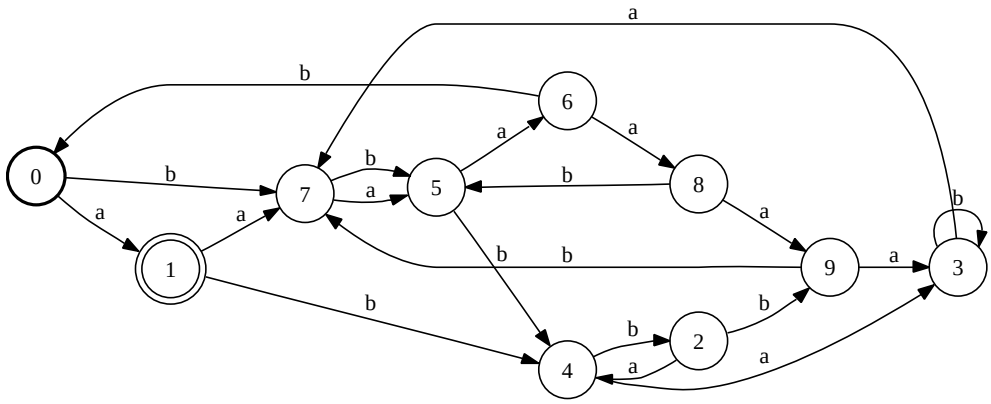
009.10.fsm



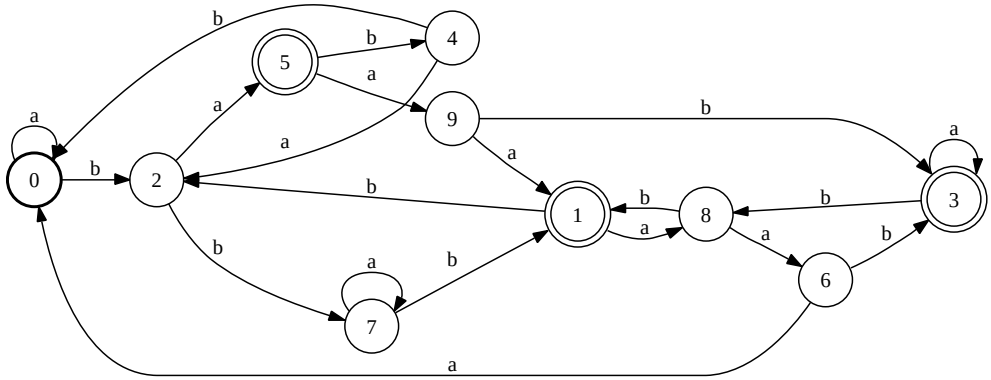
010.0.fsm



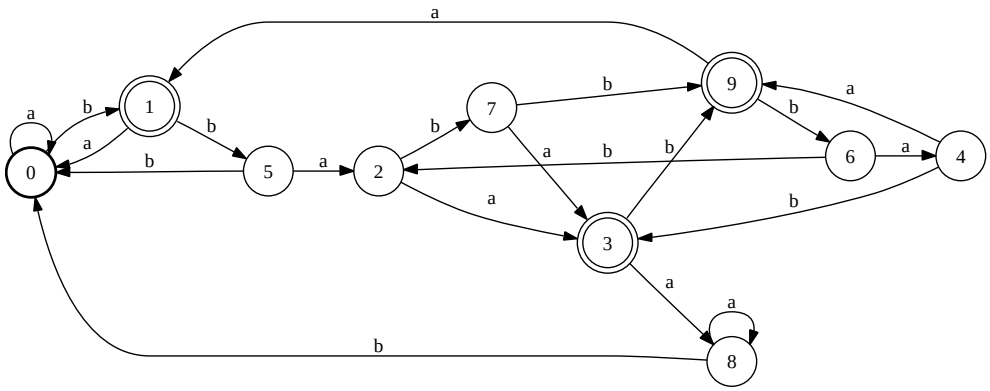
010.1.fsm



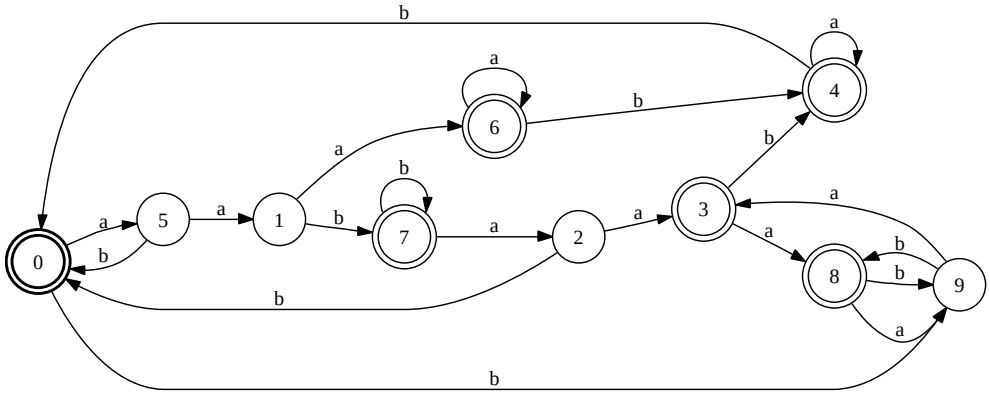
010.2.fsm



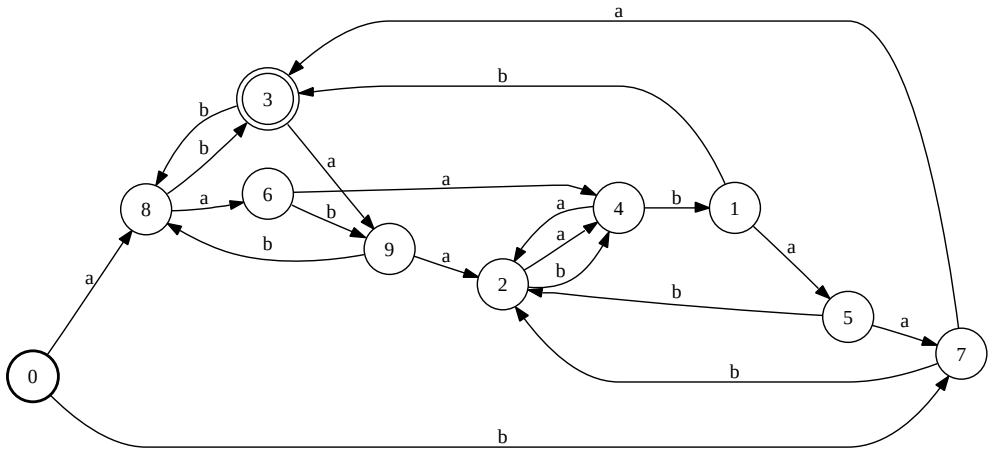
010.3.fsm



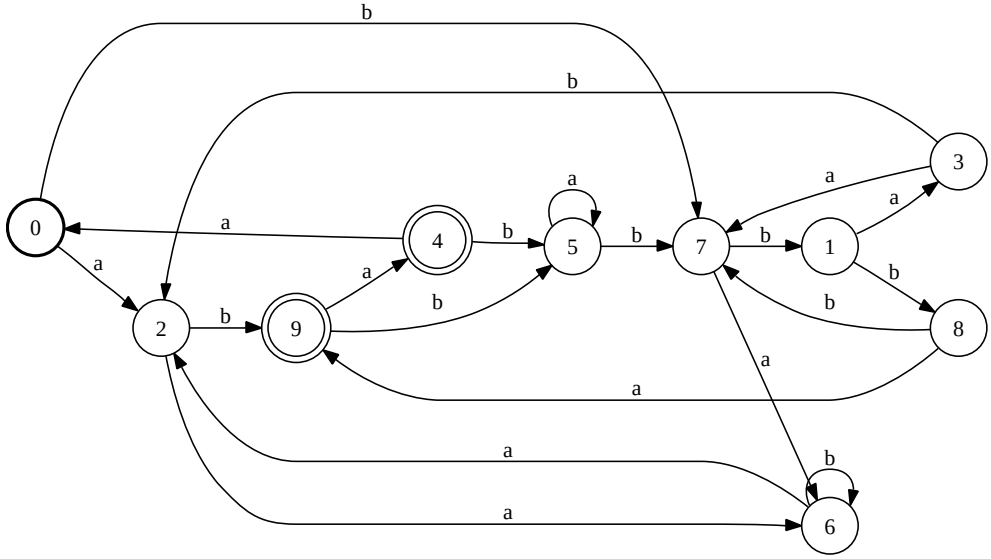
010.4.fsm



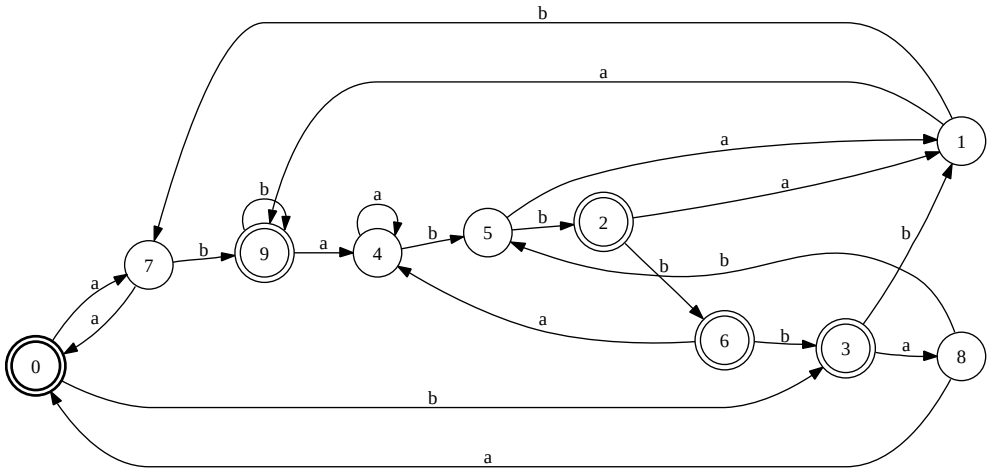
010.5.fsm



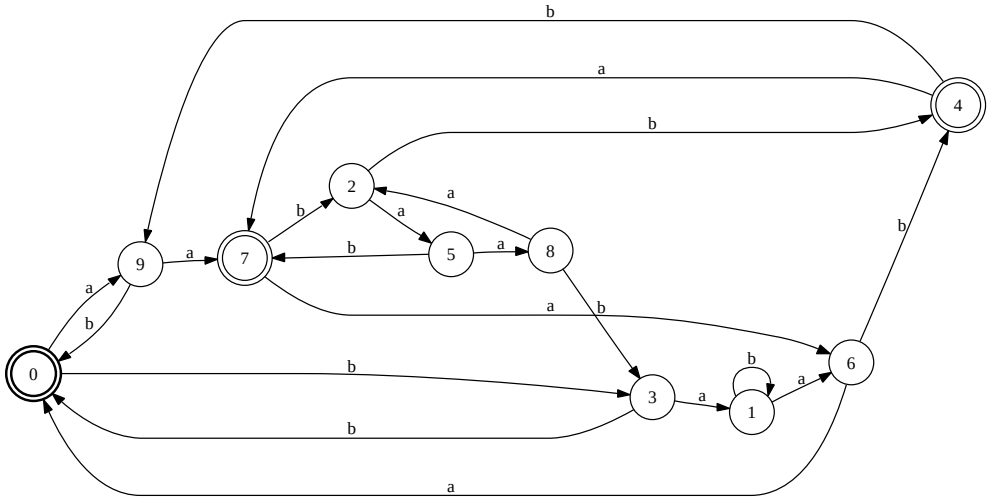
010.6.fsm



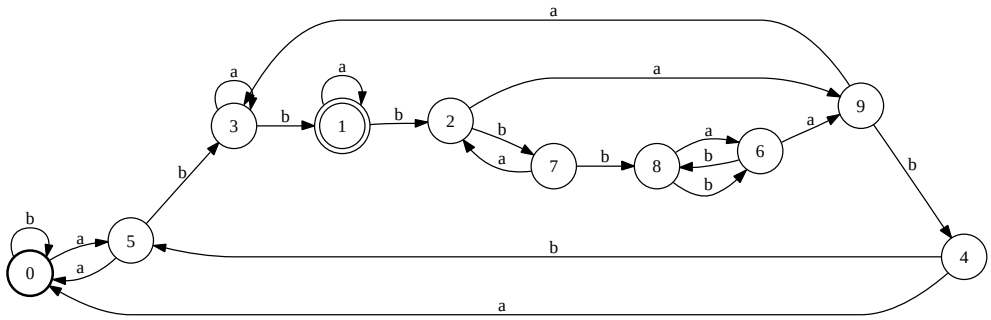
010.7.fsm



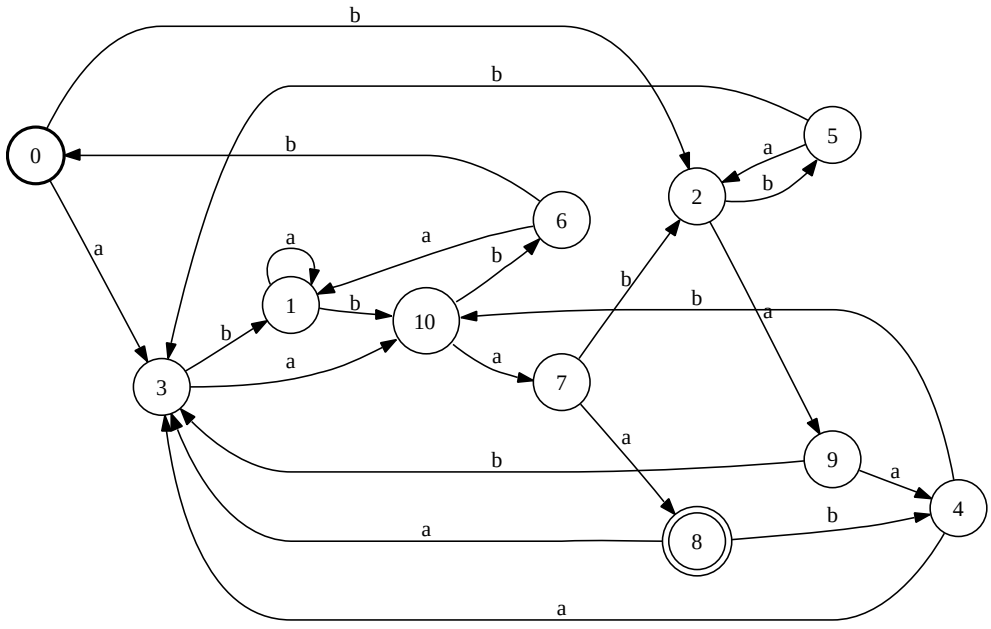
010.8.fsm



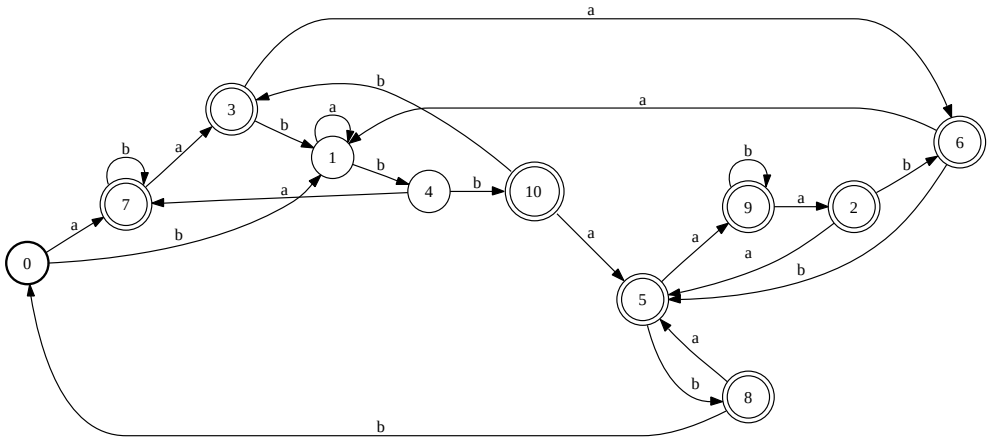
010.9.fsm



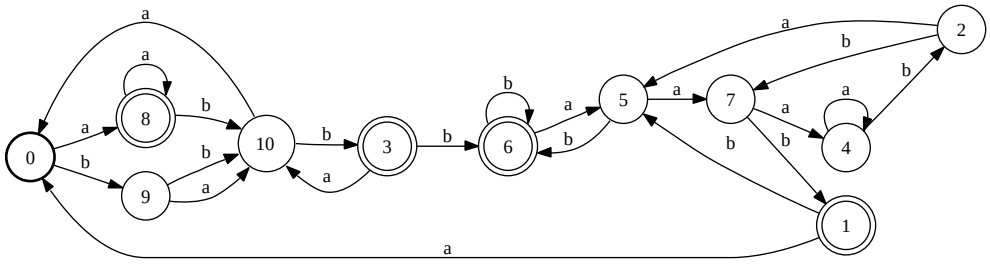
010.10.fsm



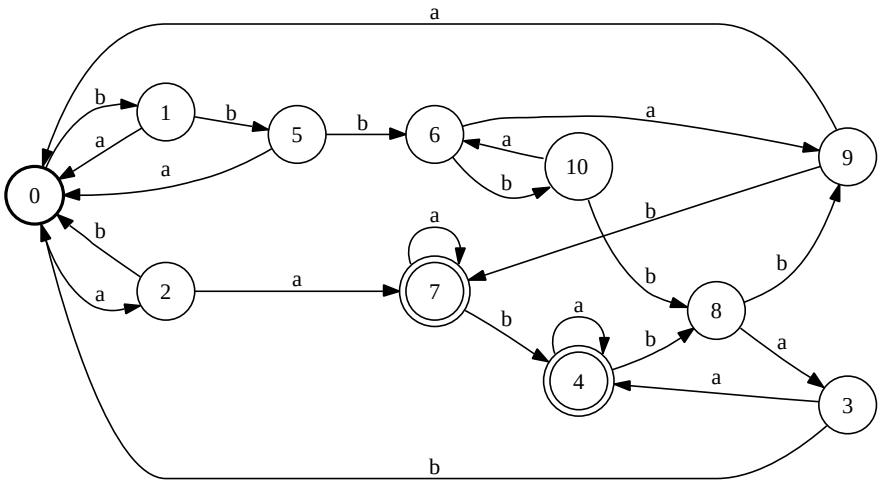
011.0.fsm



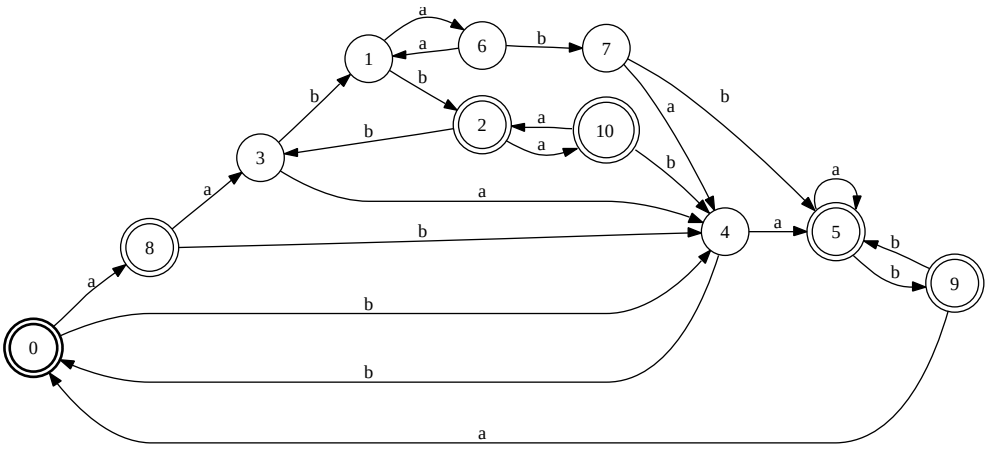
011.1.fsm



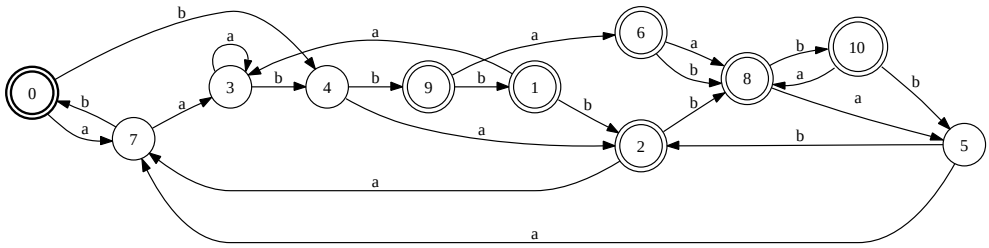
011.2.fsm



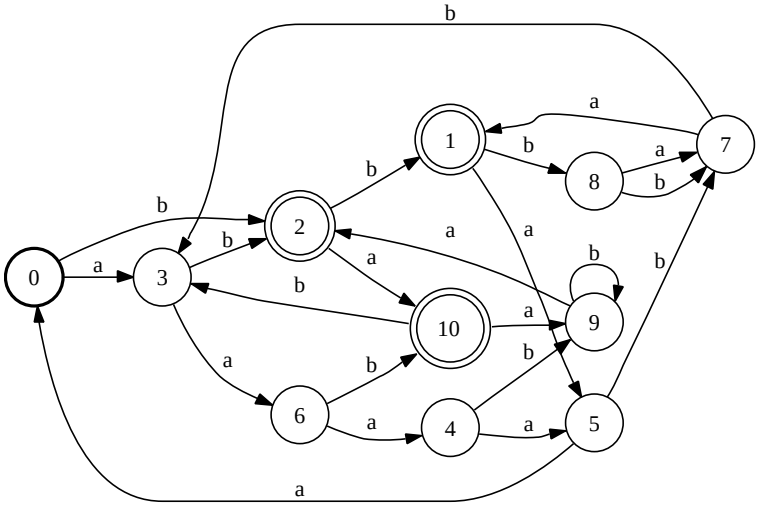
011.3.fsm



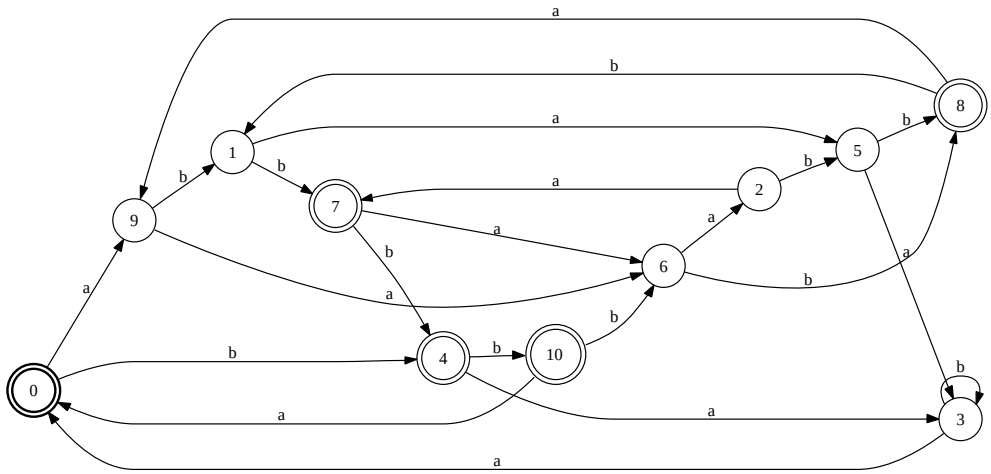
011.4.fsm



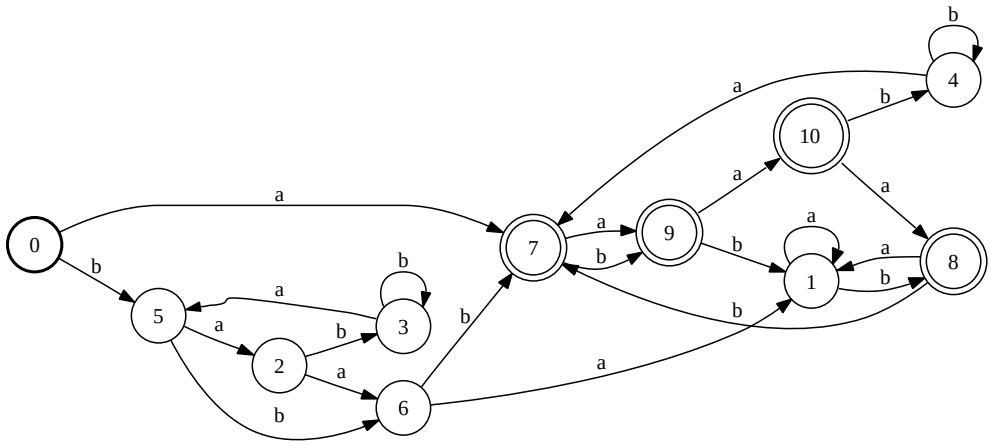
011.5.fsm



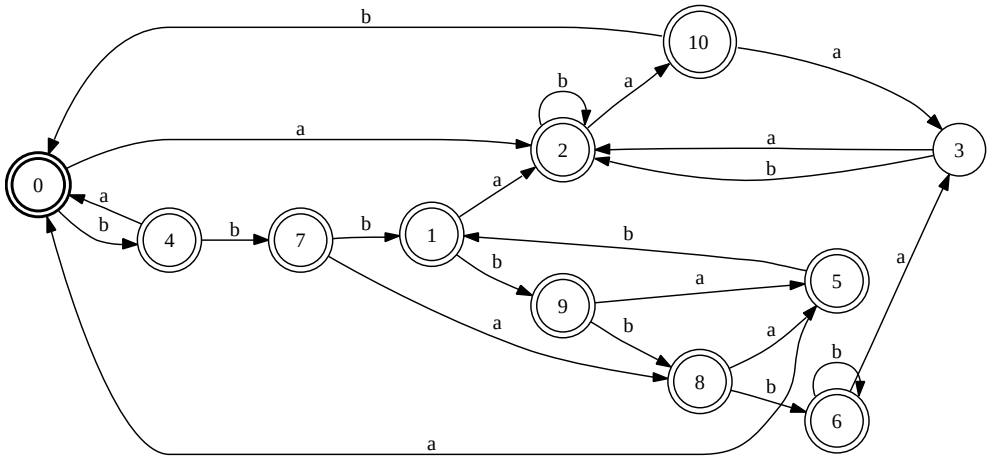
011.6.fsm



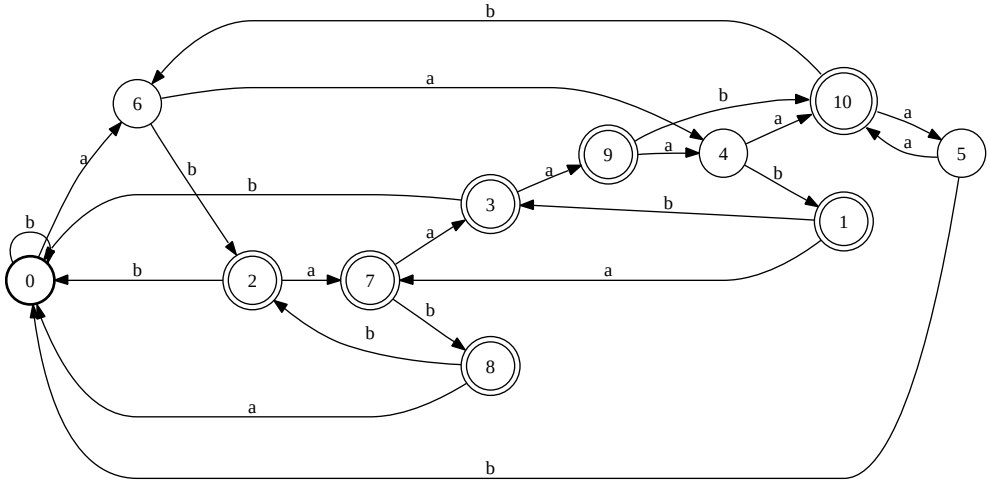
011.7.fsm



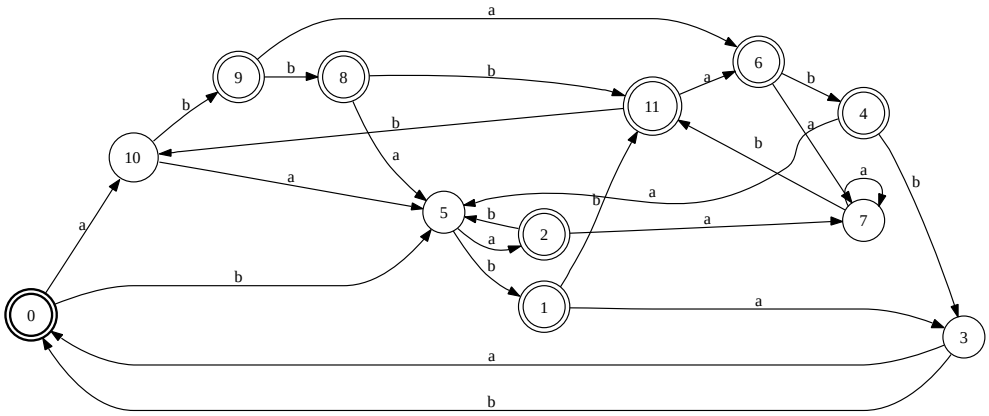
011.8.fsm



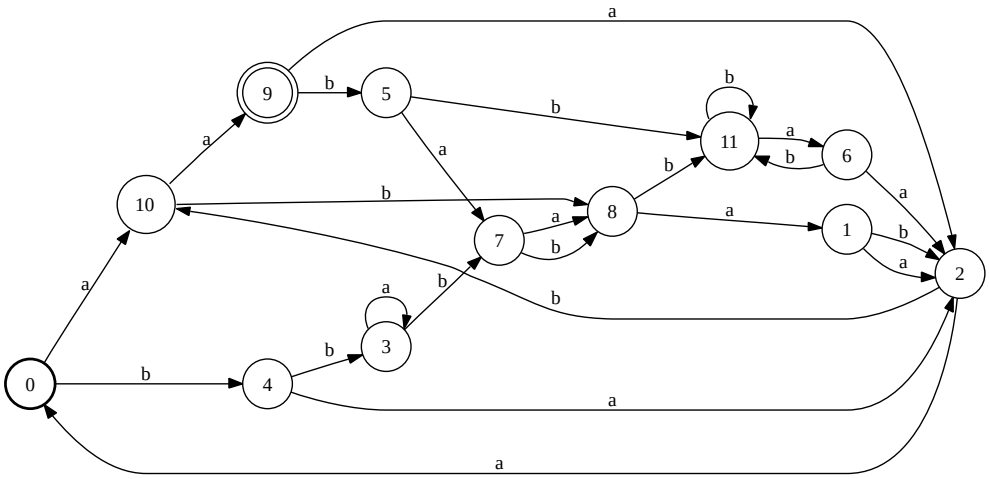
011.9.fsm



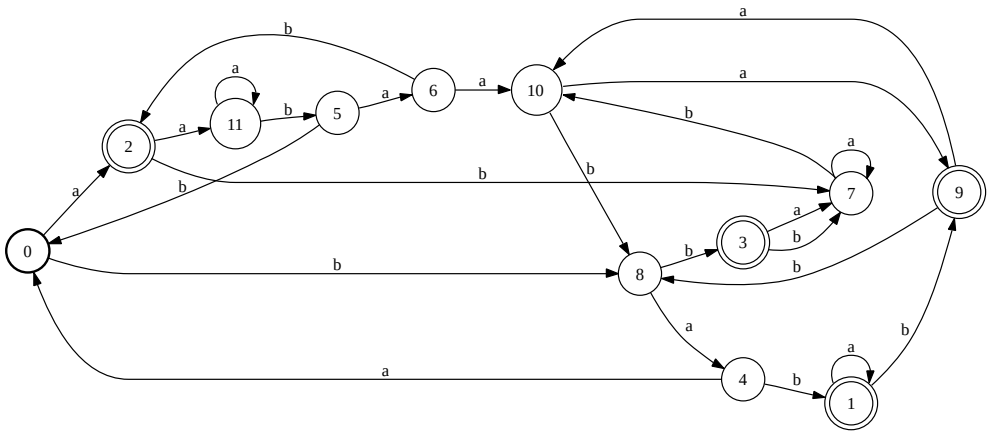
011.10.fsm



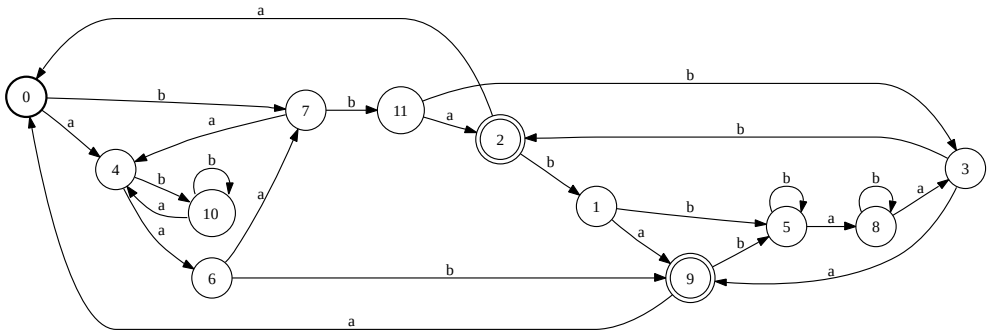
012.0.fsm



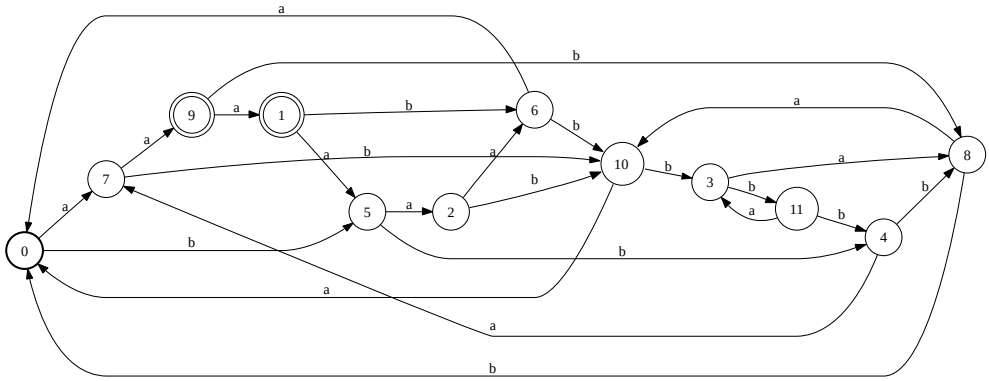
012.1.fsm



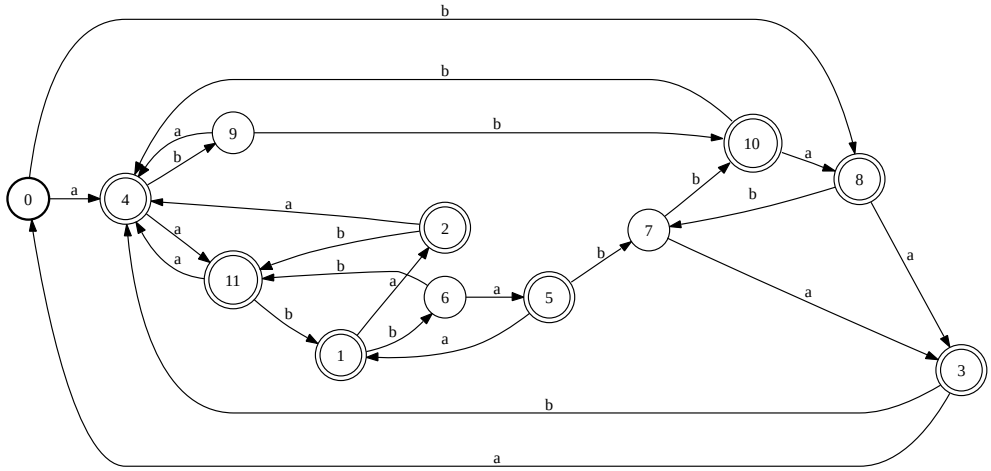
012.2.fsm



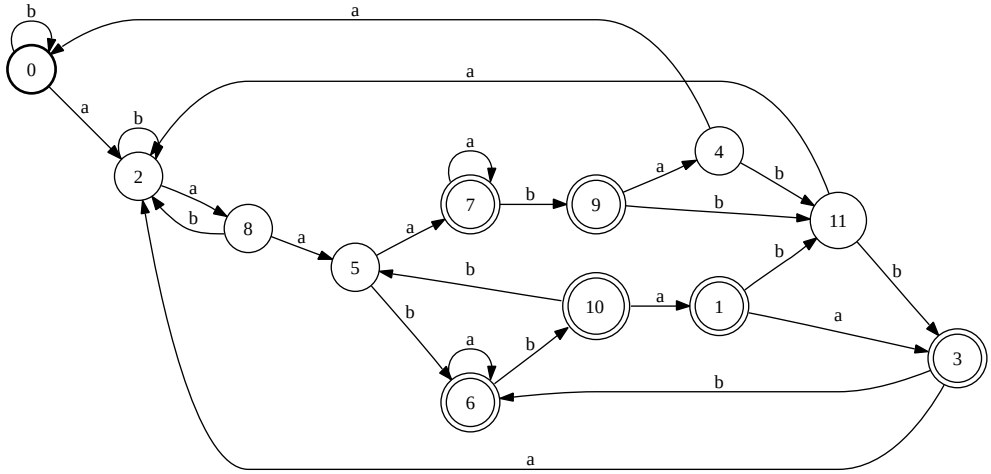
012.3.fsm



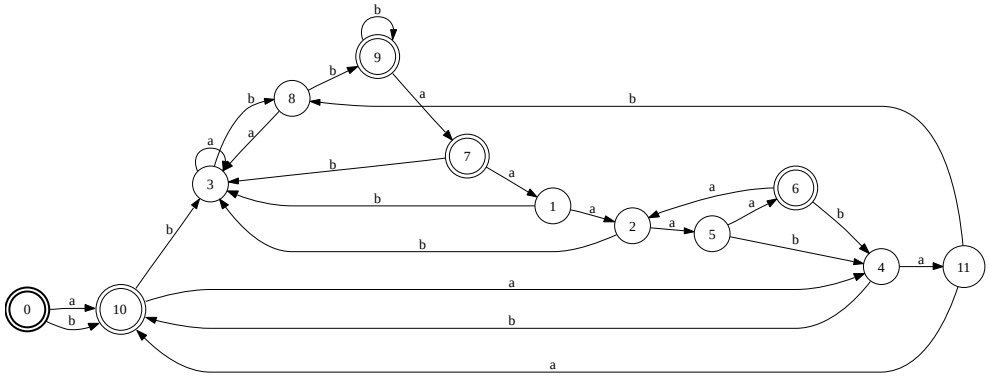
012.4.fsm



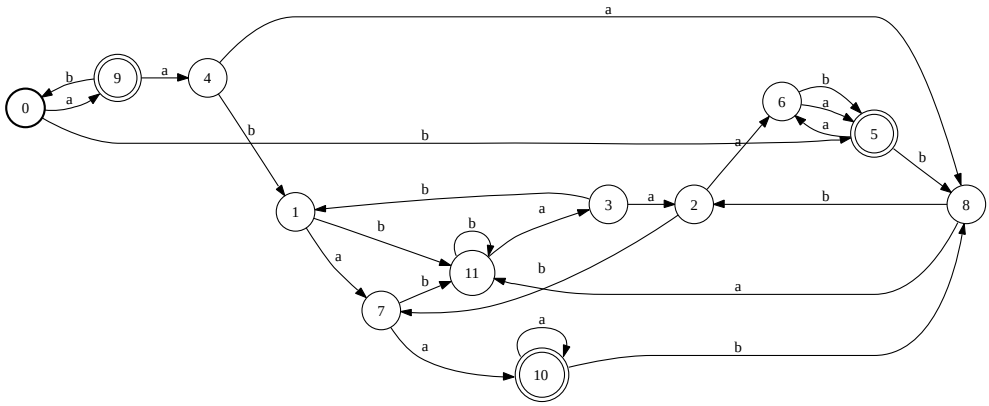
012.5.fsm



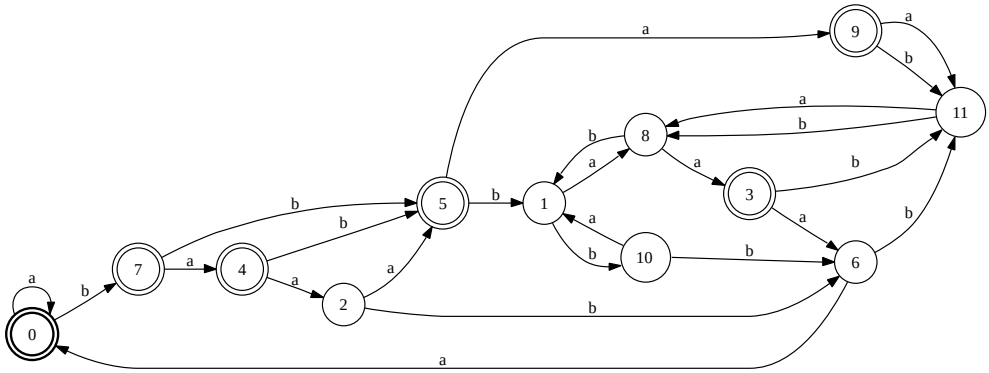
012.6.fsm



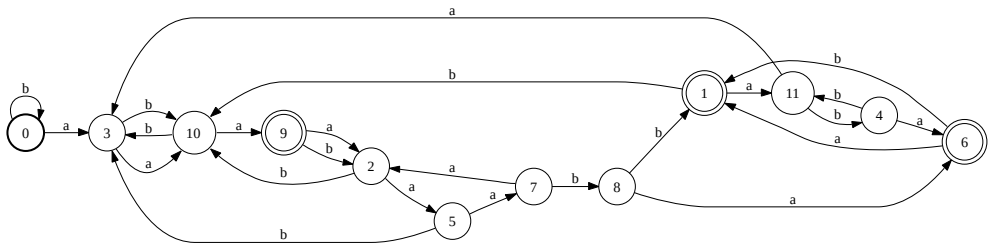
012.7.fsm



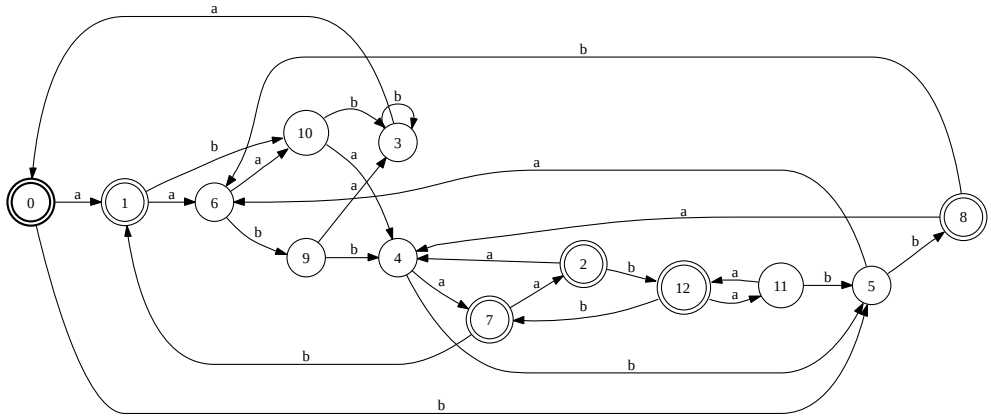
012.8.fsm



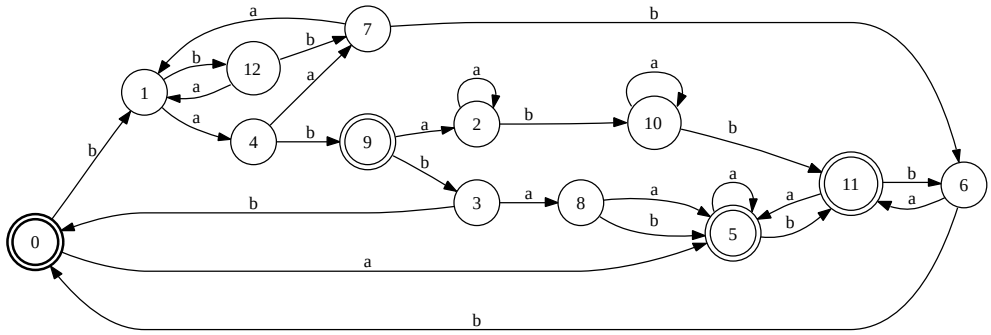
012.9.fsm



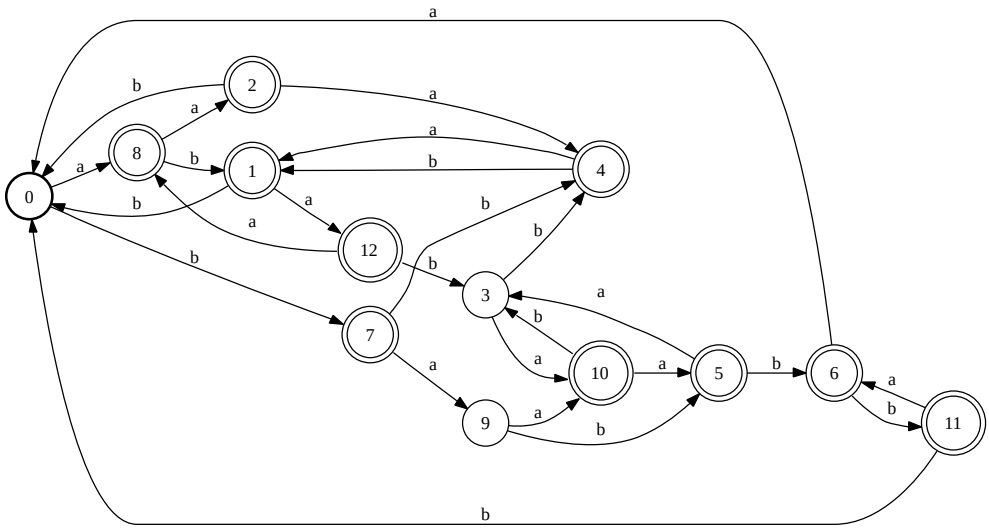
012.10.fsm



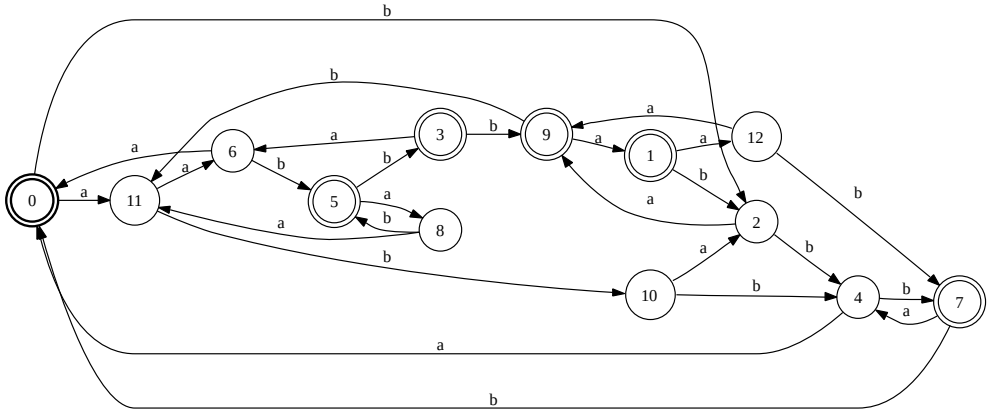
013.0.fsm



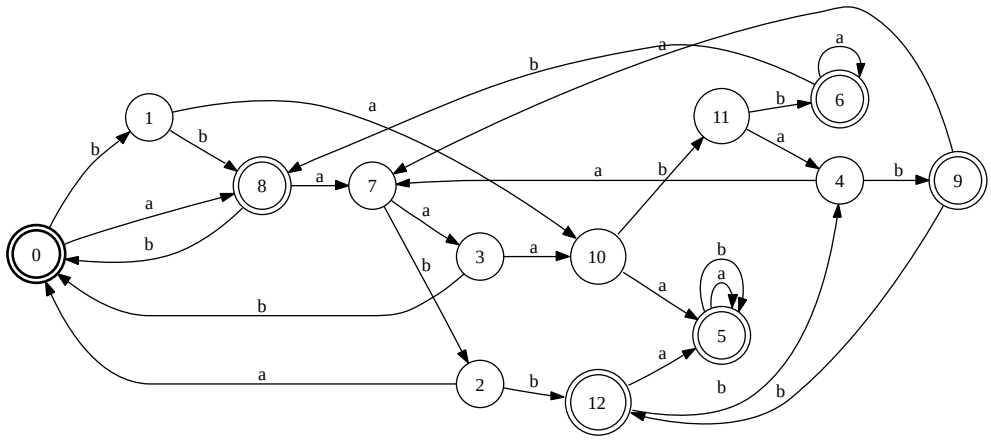
013.1.fsm



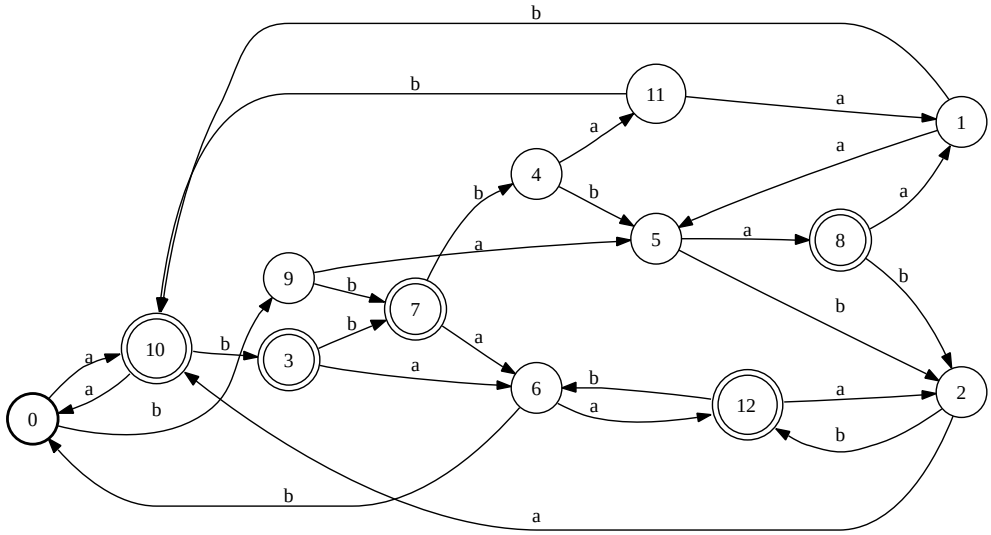
013.2.fsm



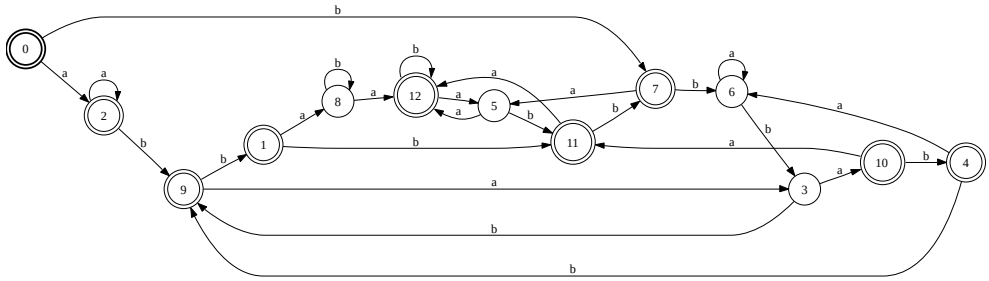
013.3.fsm



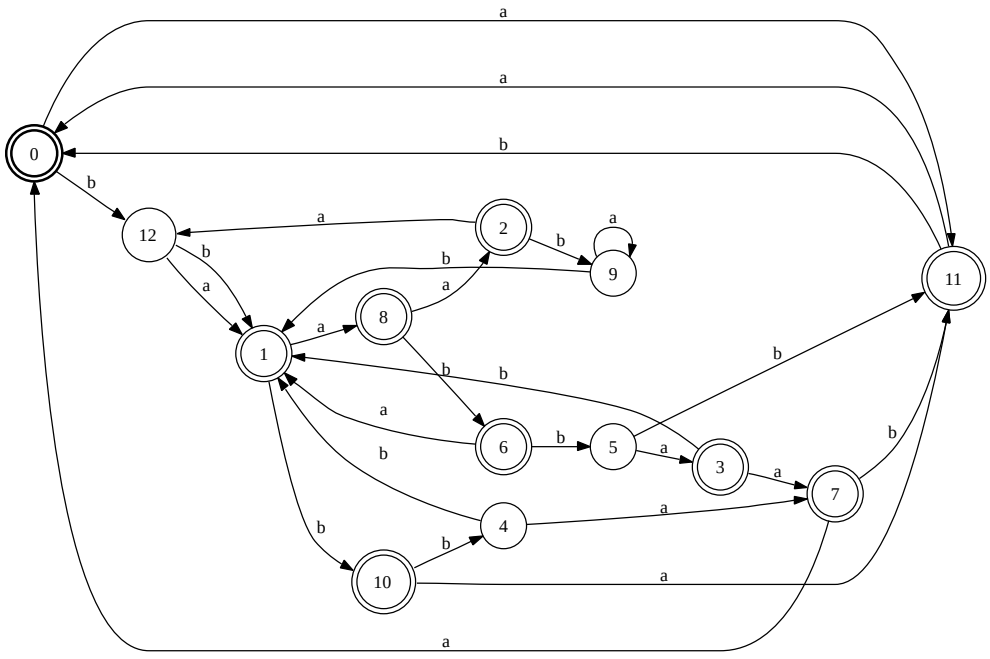
013.4.fsm



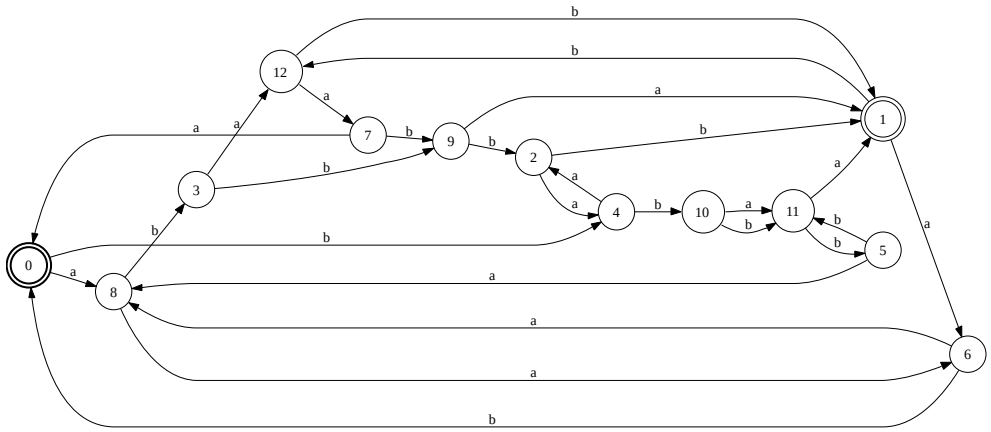
013.5.fsm



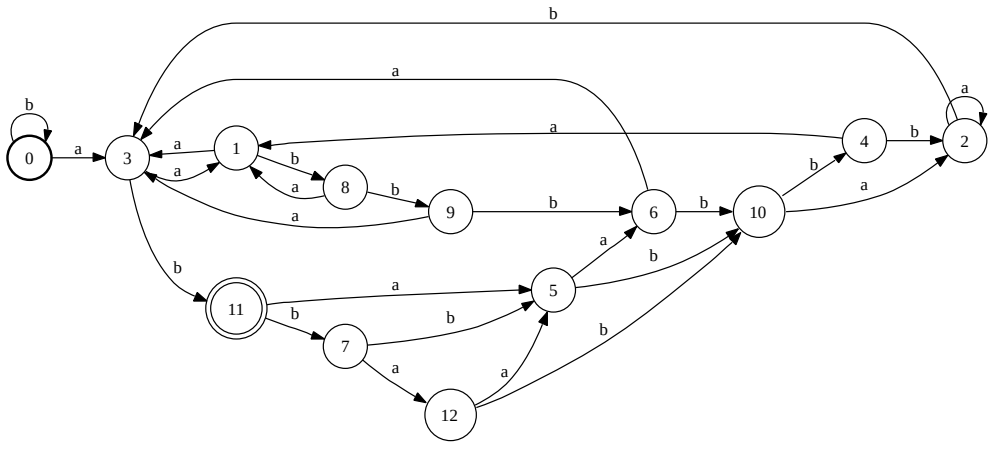
013.6.fsm



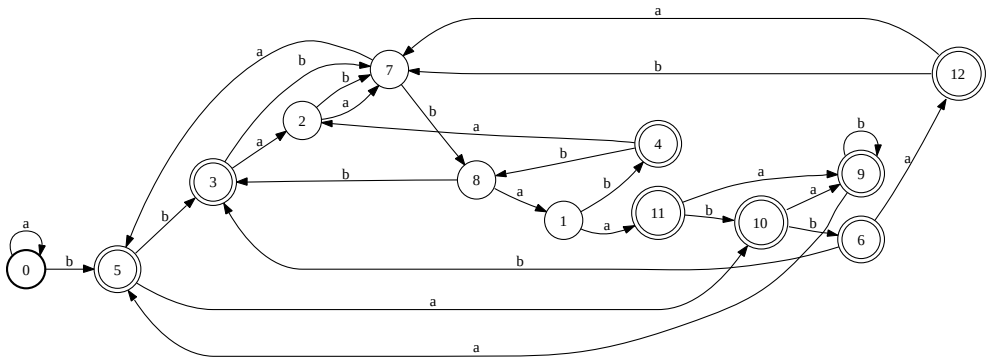
013.7.fsm



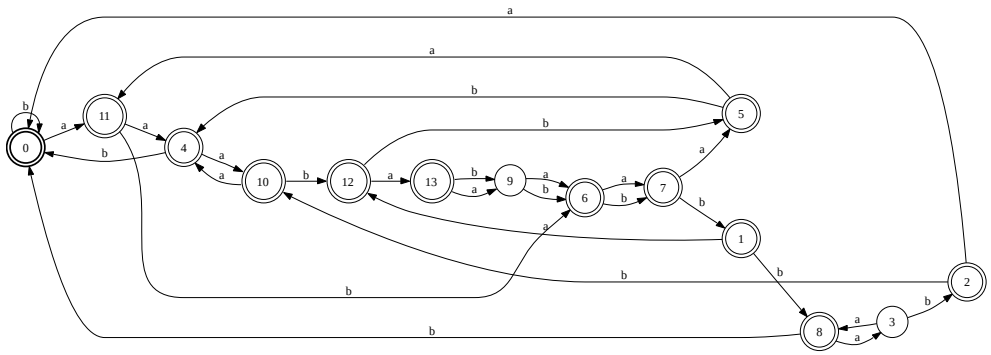
013.8.fsm



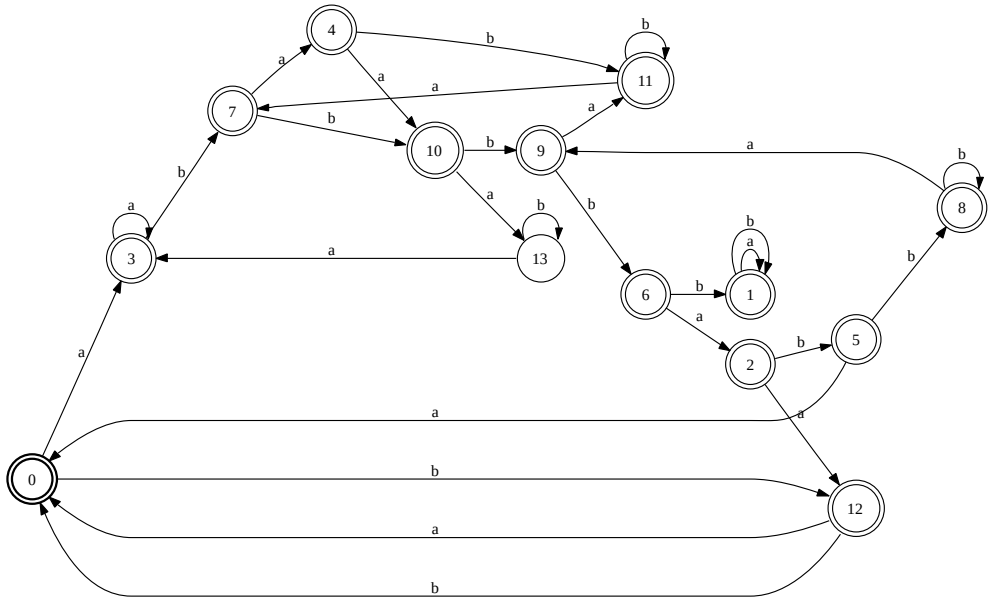
013.9.fsm



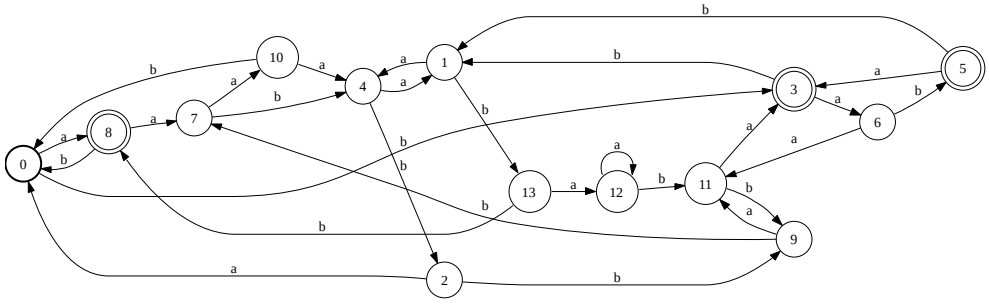
013.10.fsm



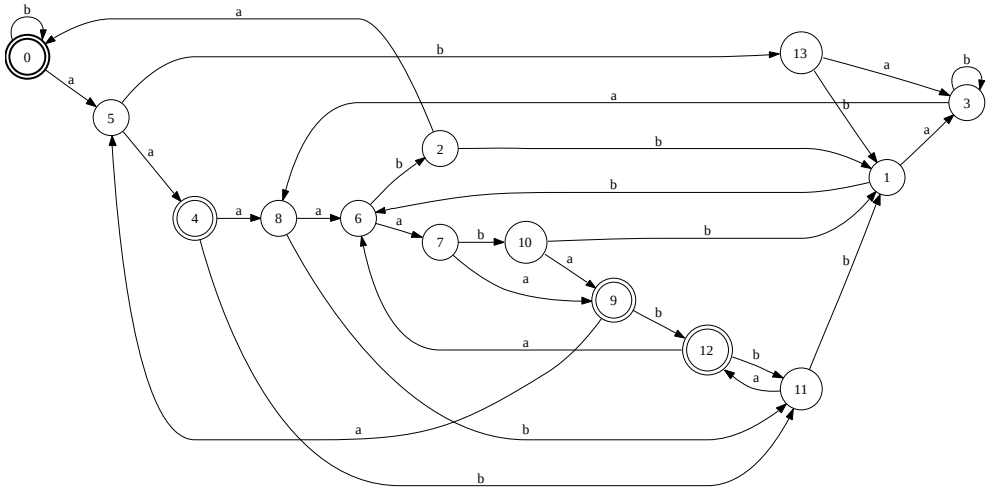
014.0.fsm



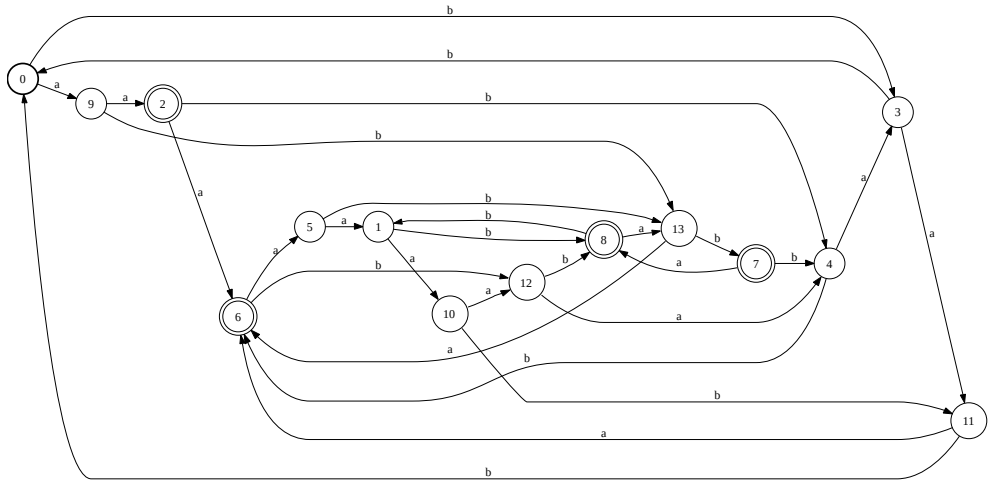
014.1.fsm



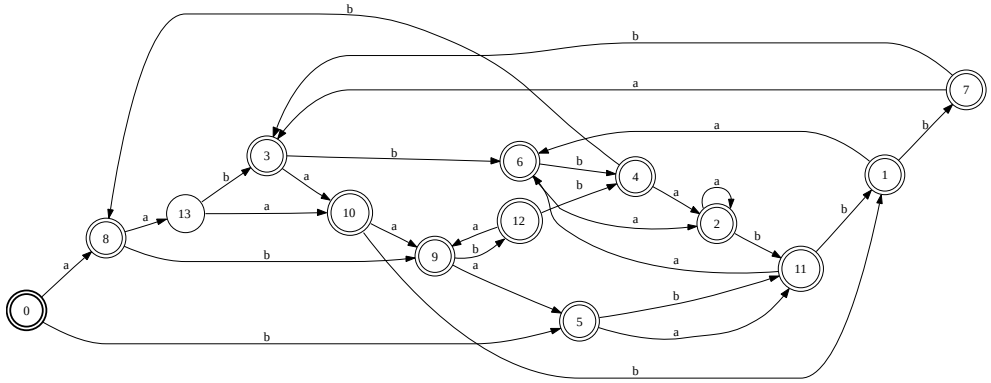
014.2.fsm



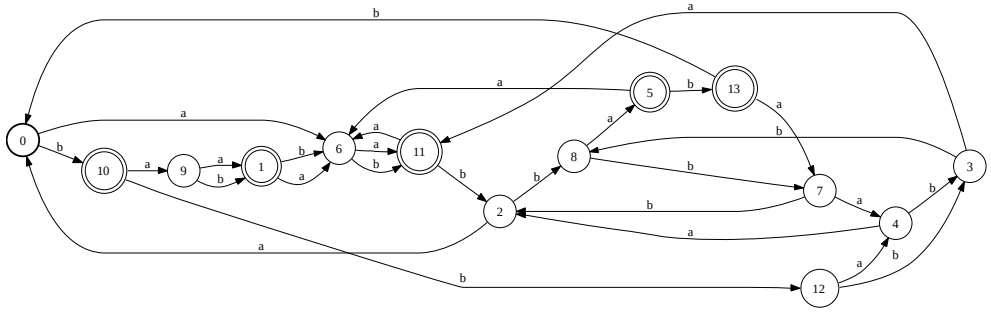
014.3.fsm



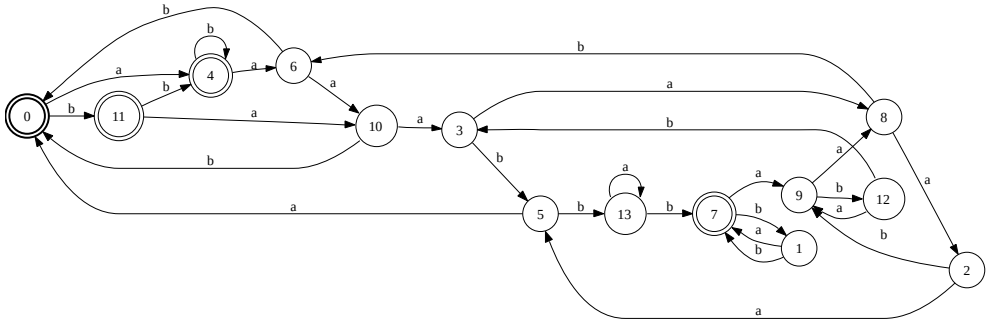
014.4.fsm



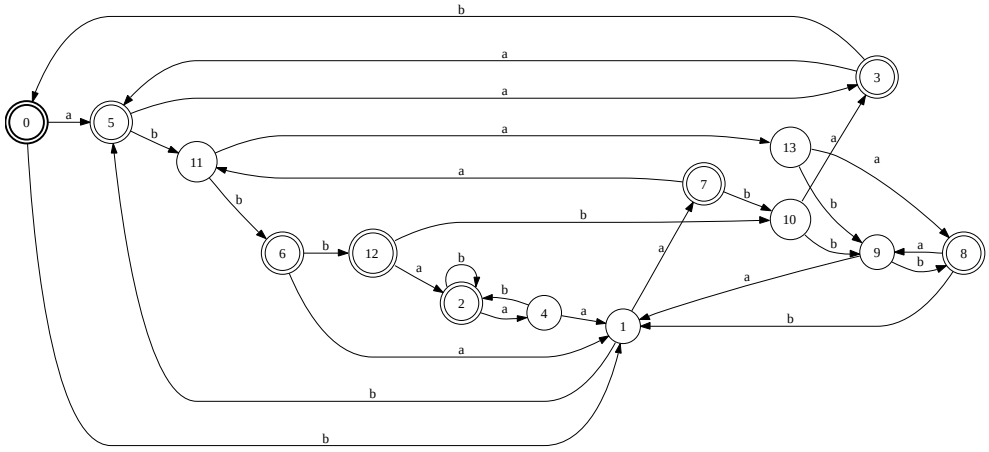
014.5.fsm



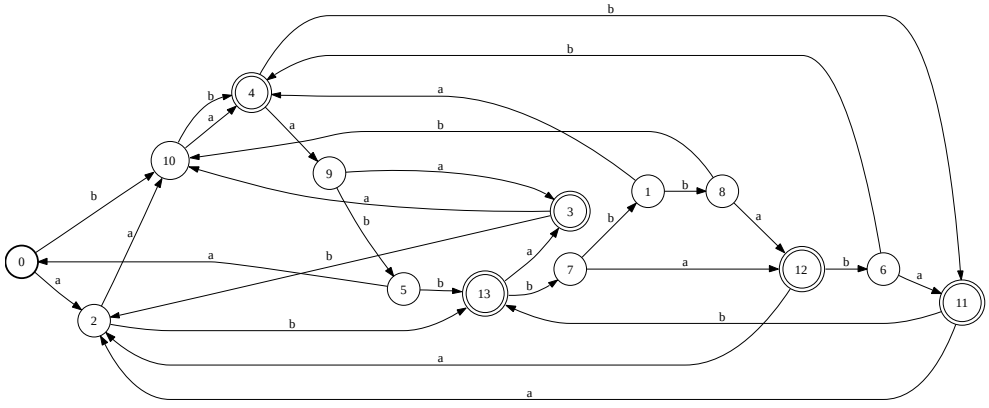
014.6.fsm



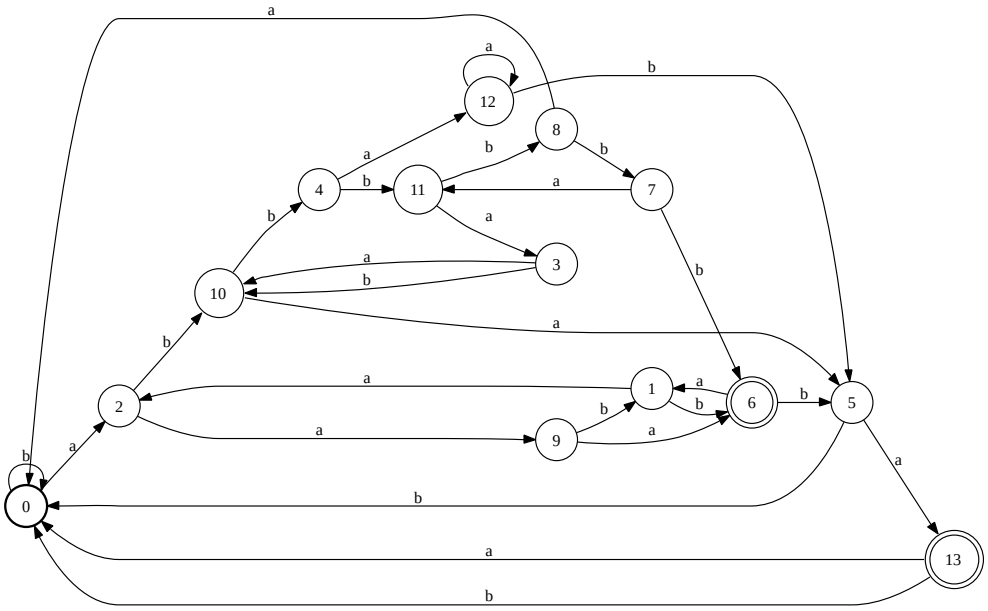
014.7.fsm



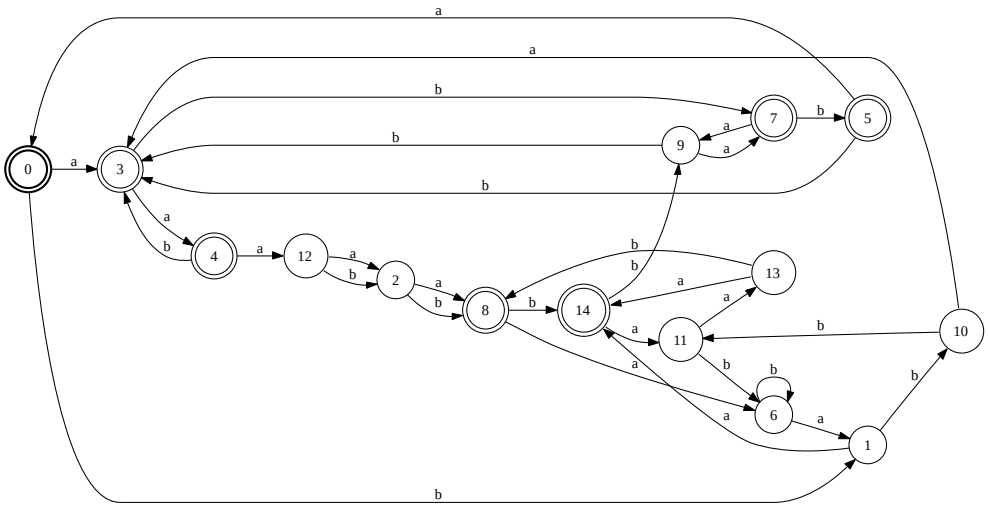
014.8.fsm



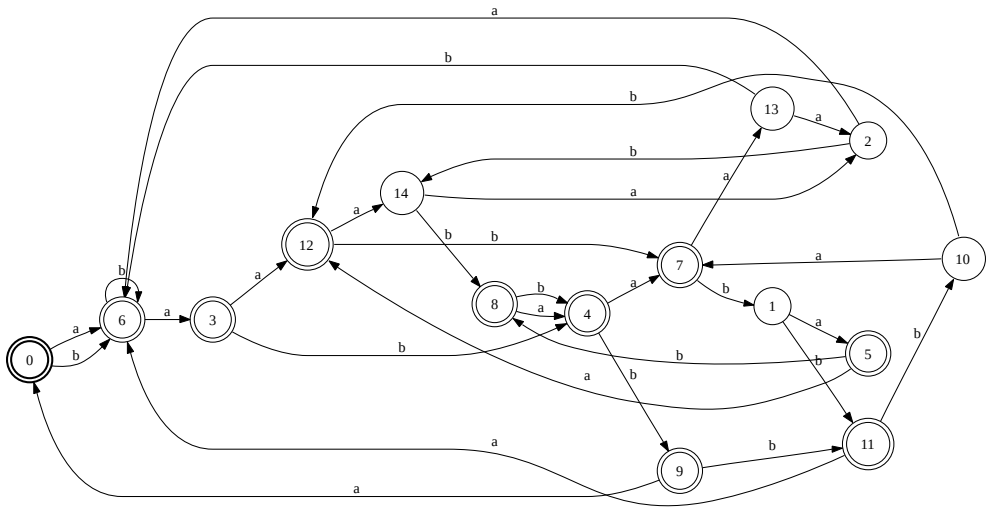
014.9.fsm



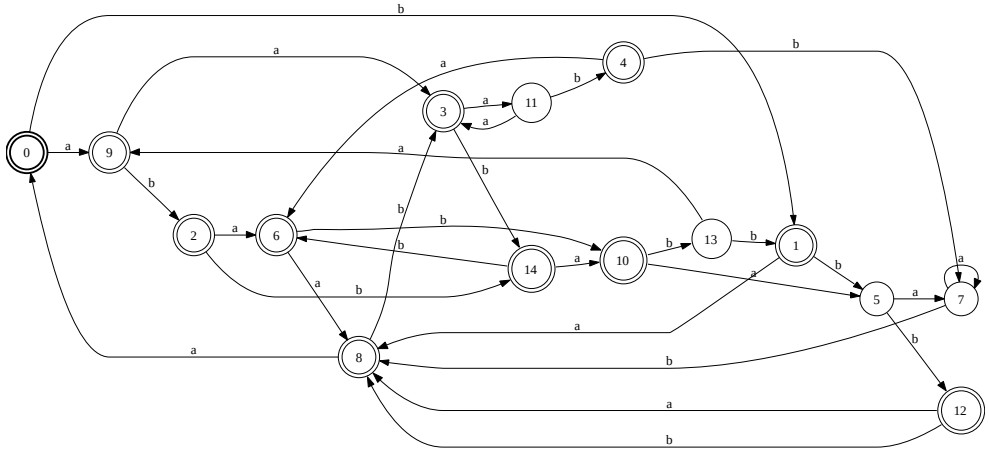
014.10.fsm



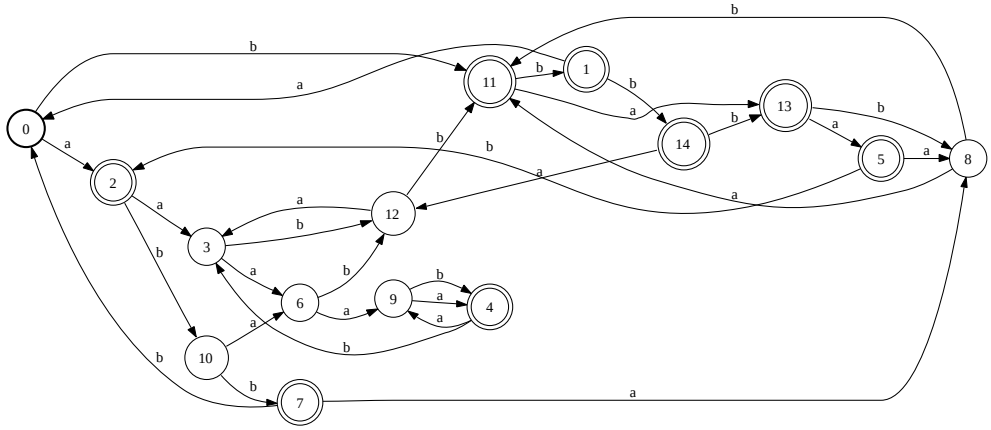
015.0.fsm



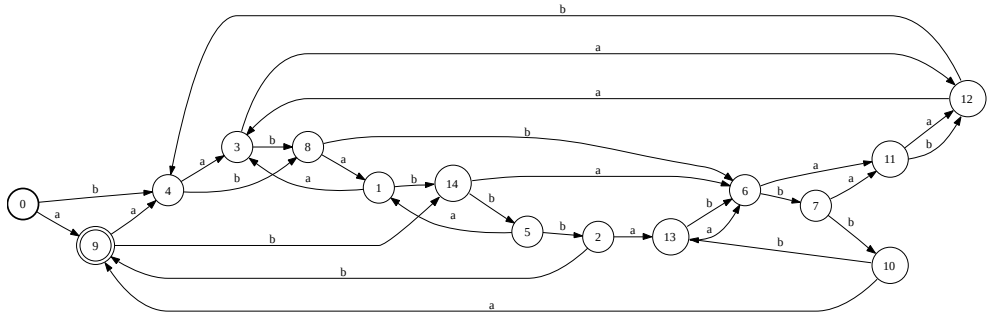
015.1.fsm



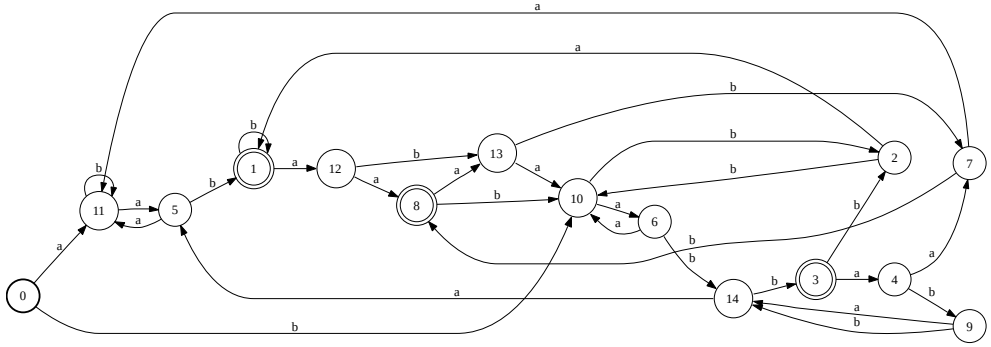
015.2.fsm



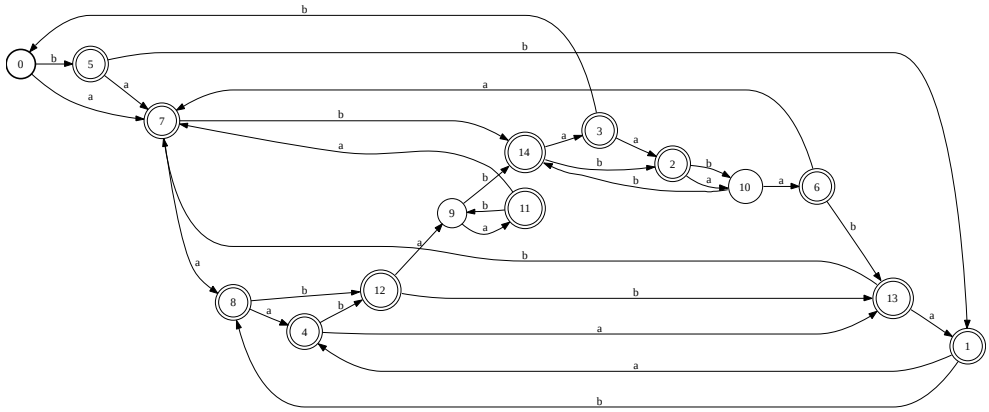
015.3.fsm



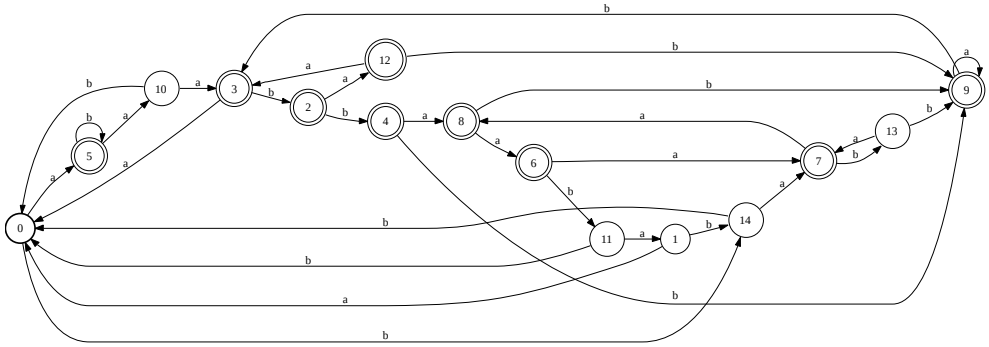
015.4.fsm



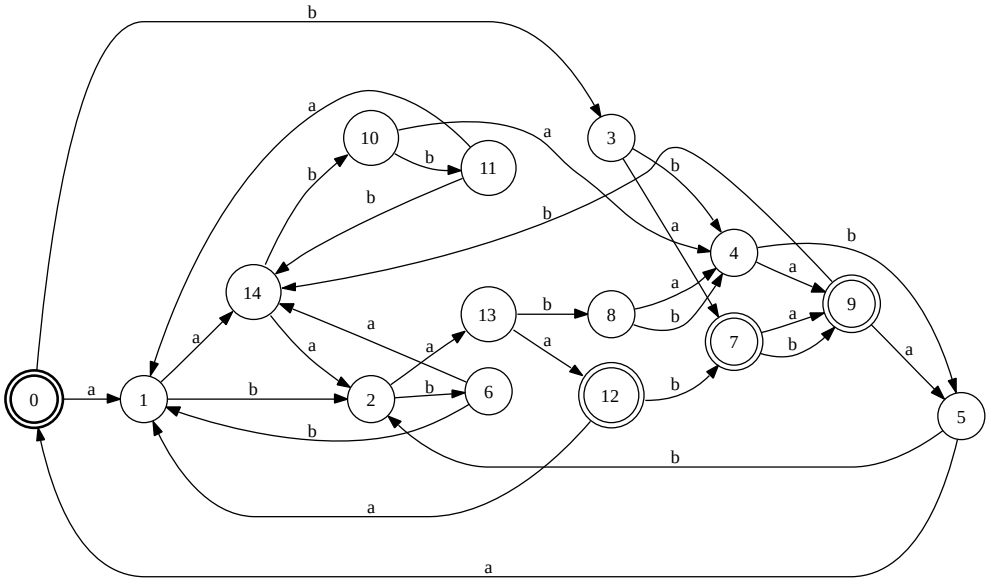
015.5.fsm



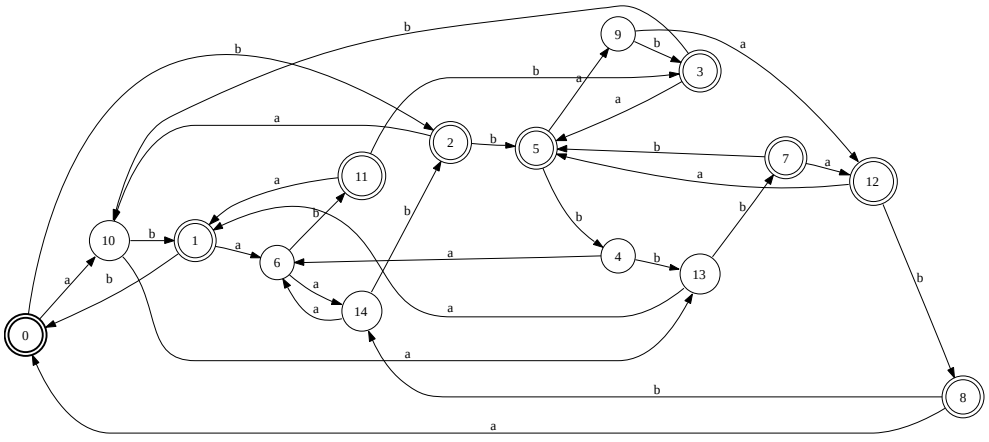
015.6.fsm



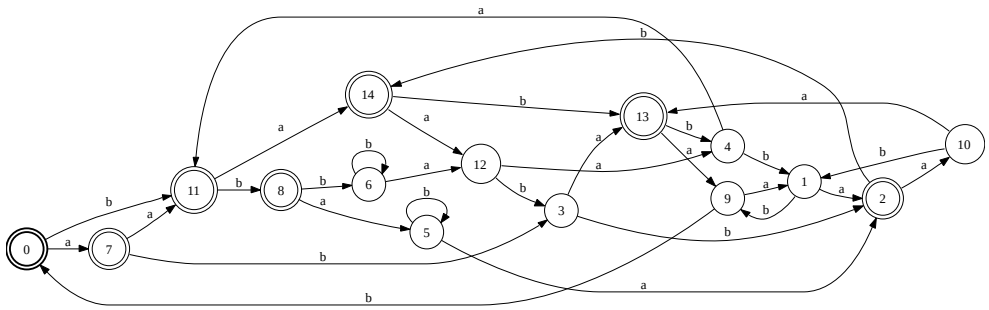
015.7.fsm



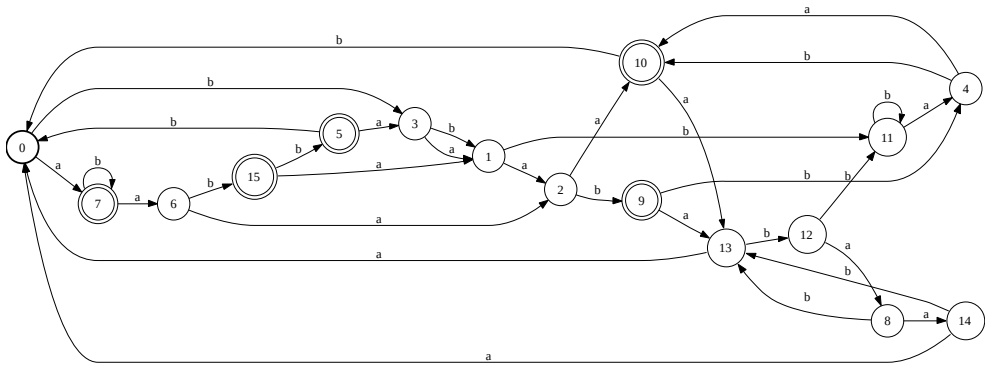
015.8.fsm



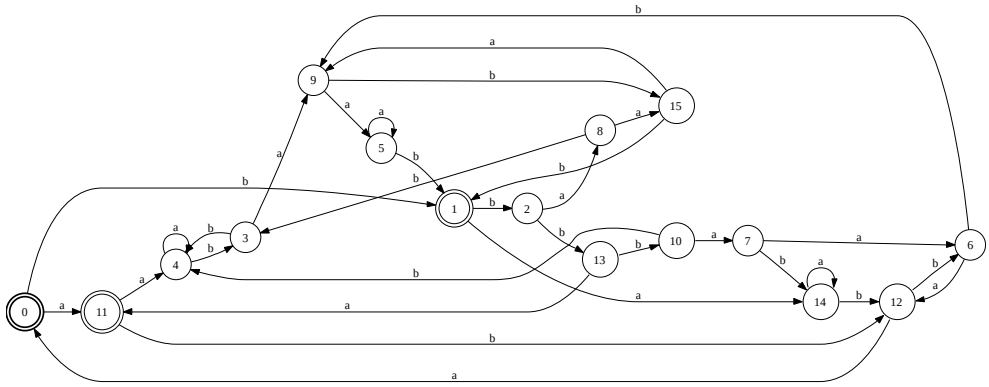
015.9.fsm



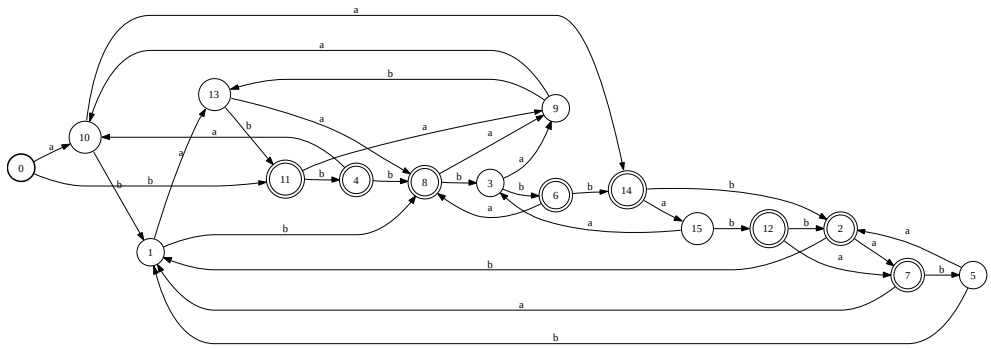
015.10.fsm



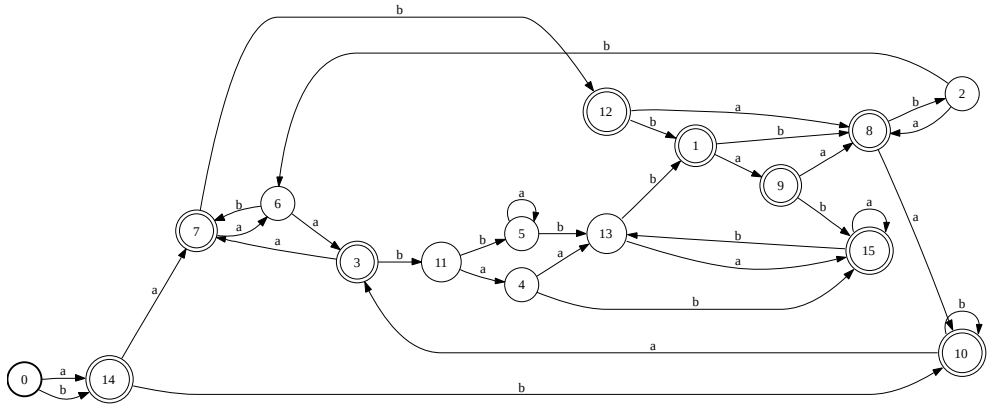
016.0.fsm



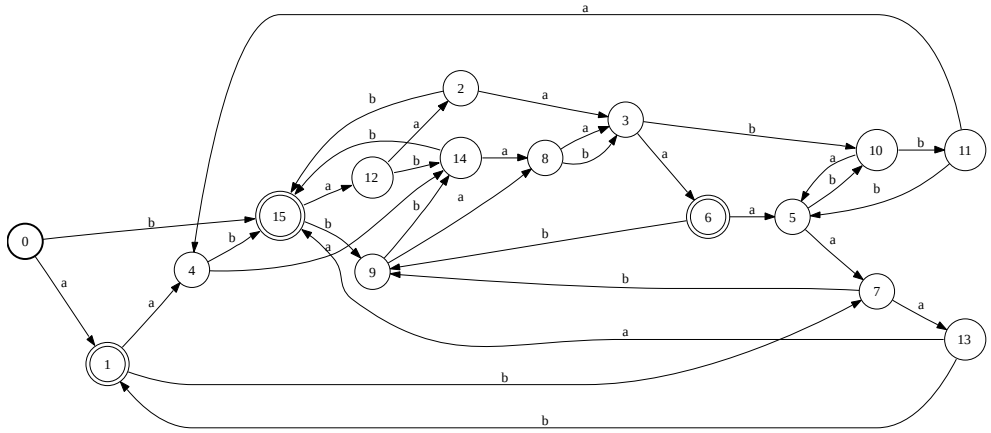
016.1.fsm



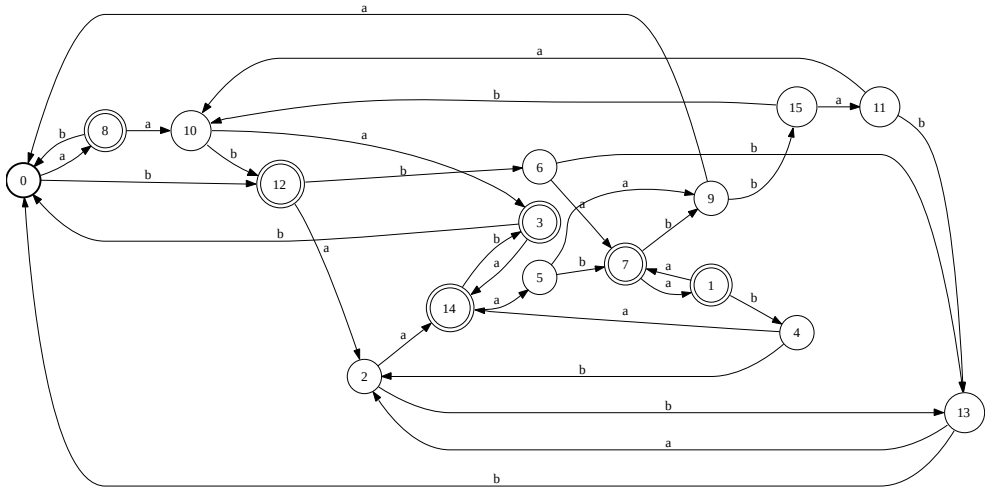
016.2.fsm



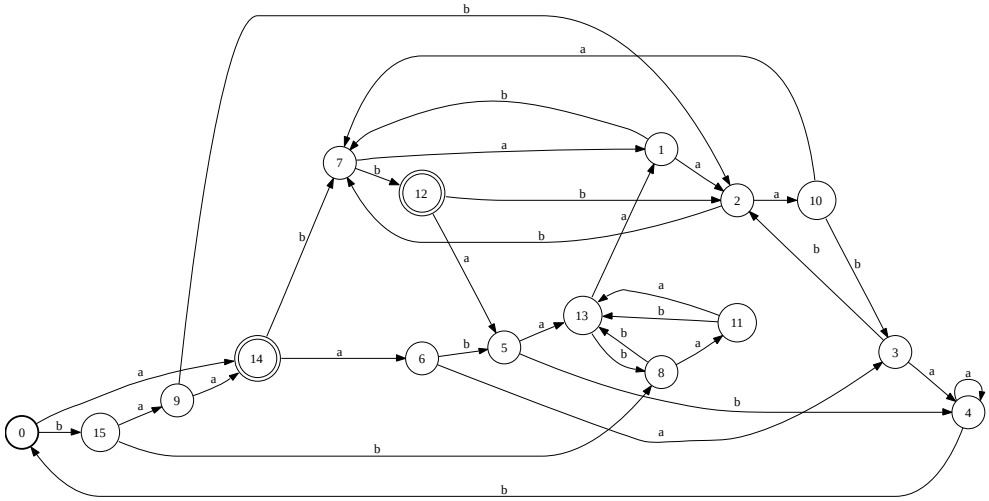
016.3.fsm



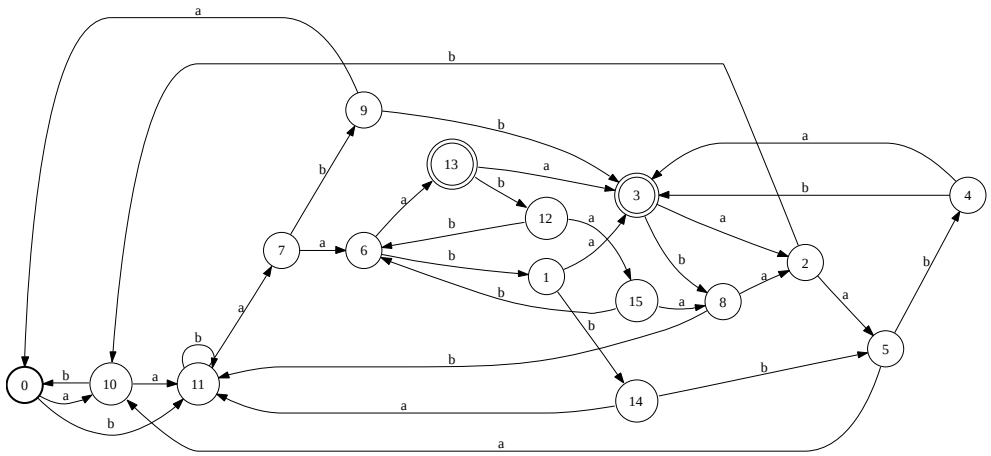
016.4.fsm



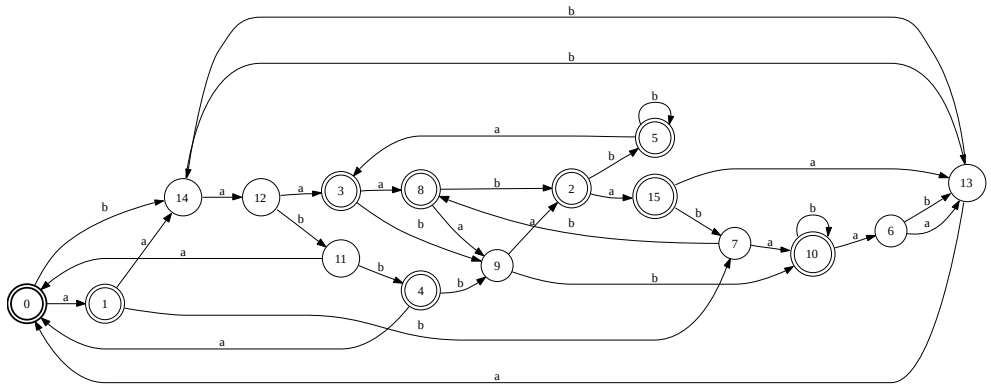
016.5.fsm



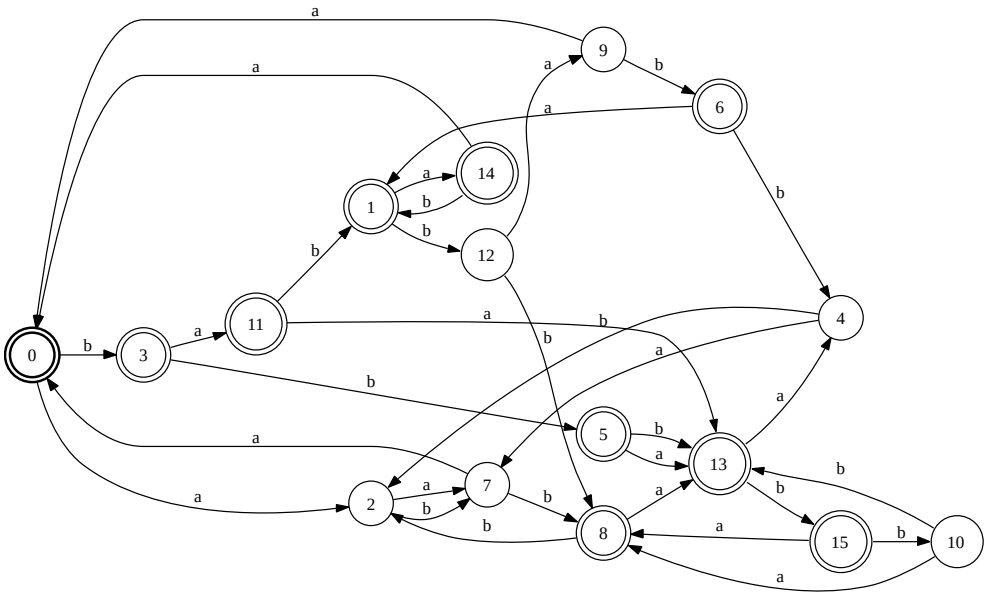
016.6.fsm



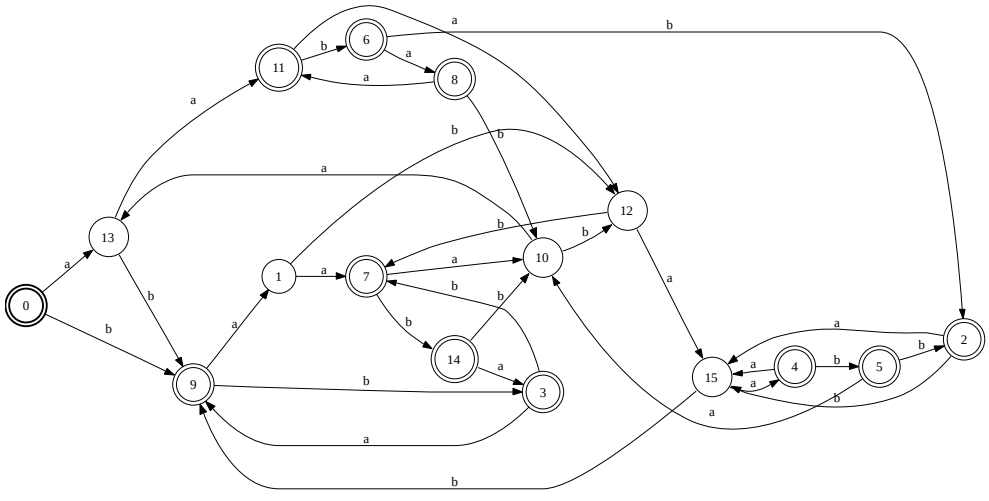
016.7.fsm



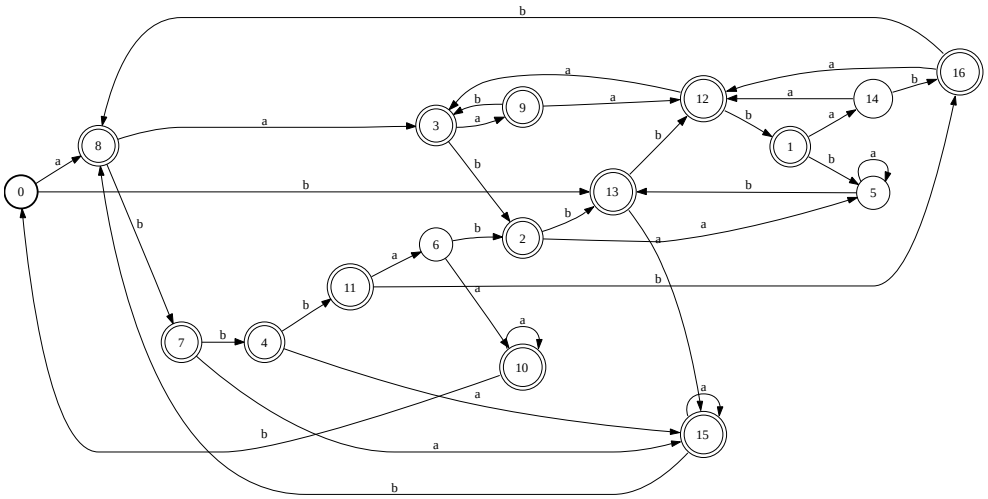
016.8.fsm



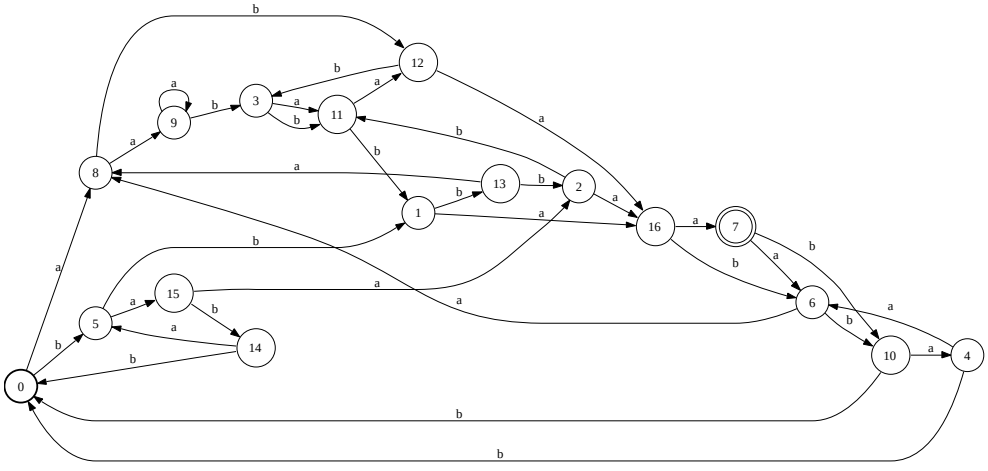
016.9.fsm



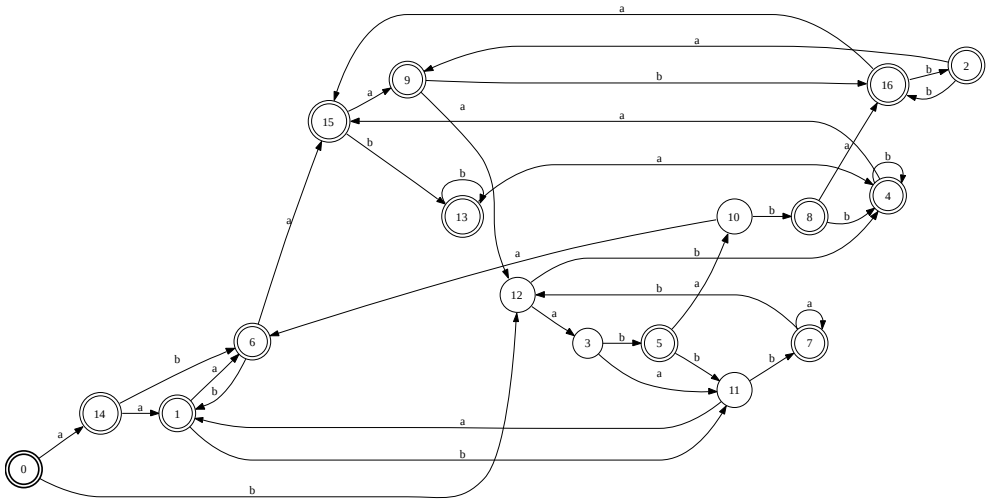
016.10.fsm



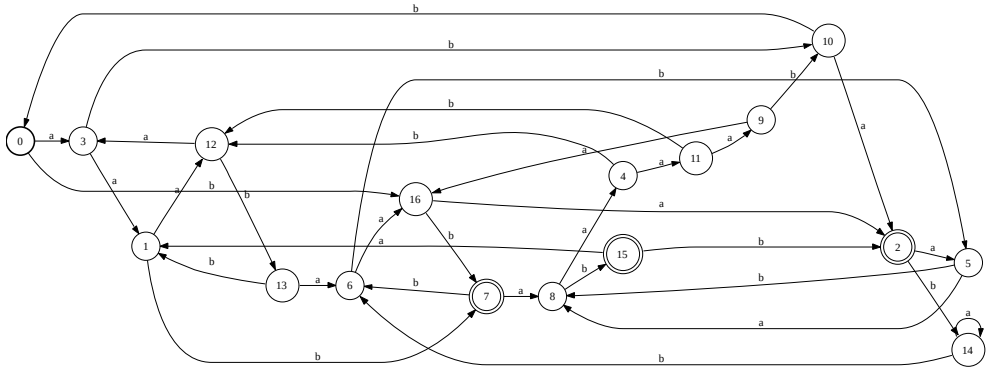
017.0.fsm



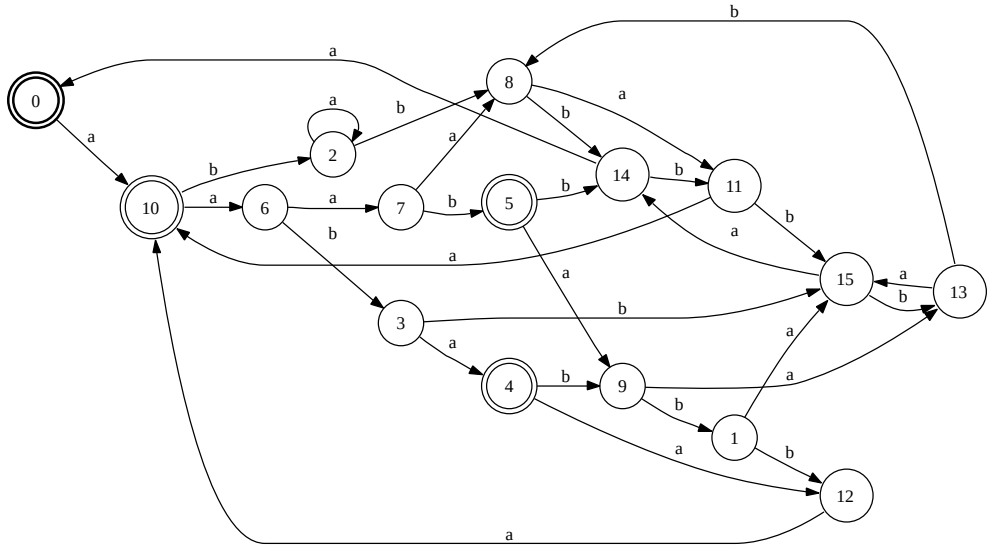
017.1.fsm



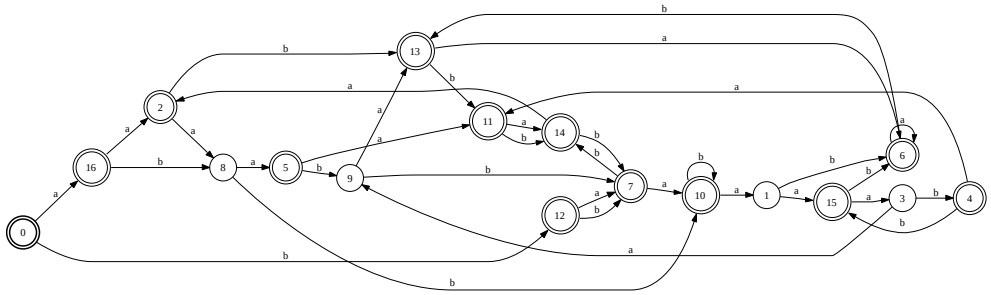
017.2.fsm



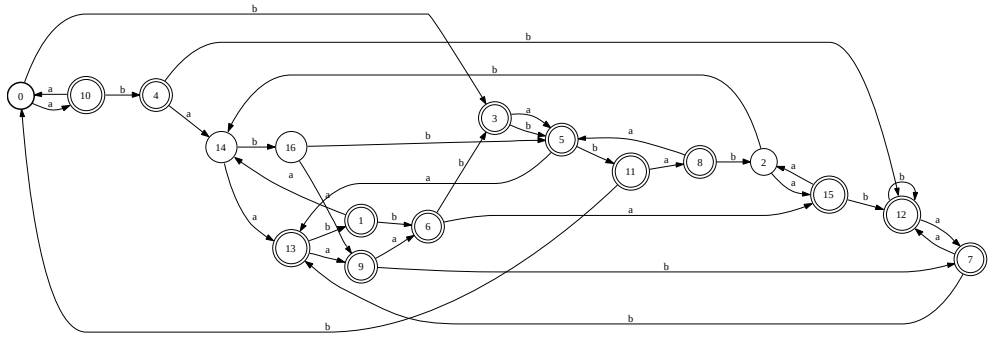
017.3.fsm



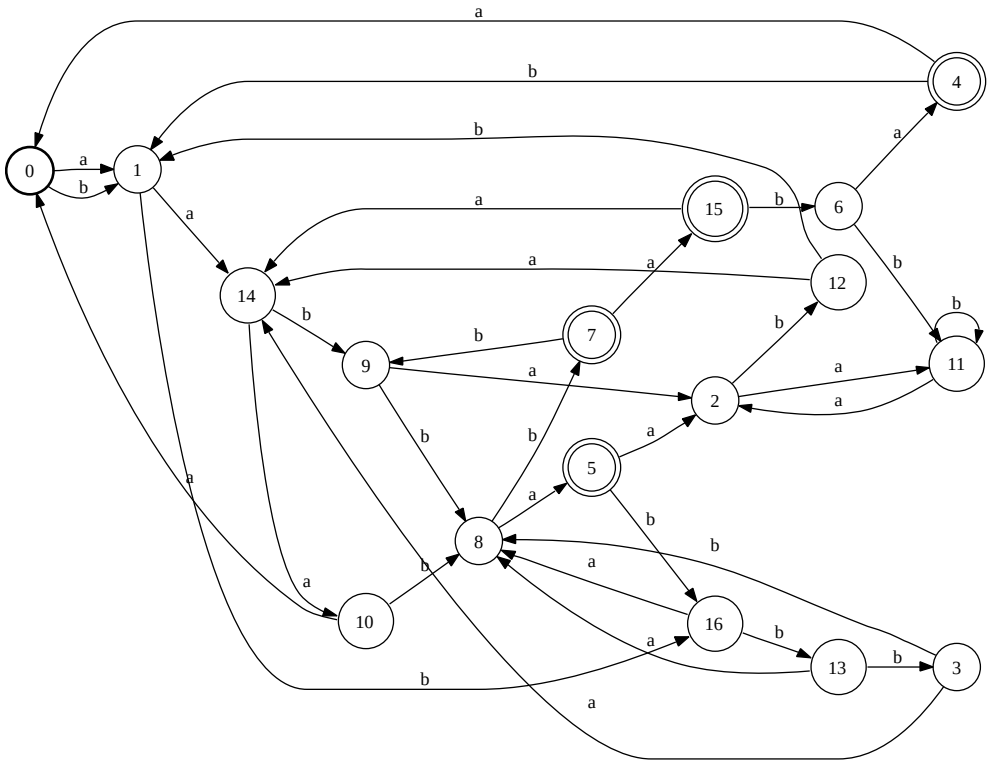
017.4.fsm



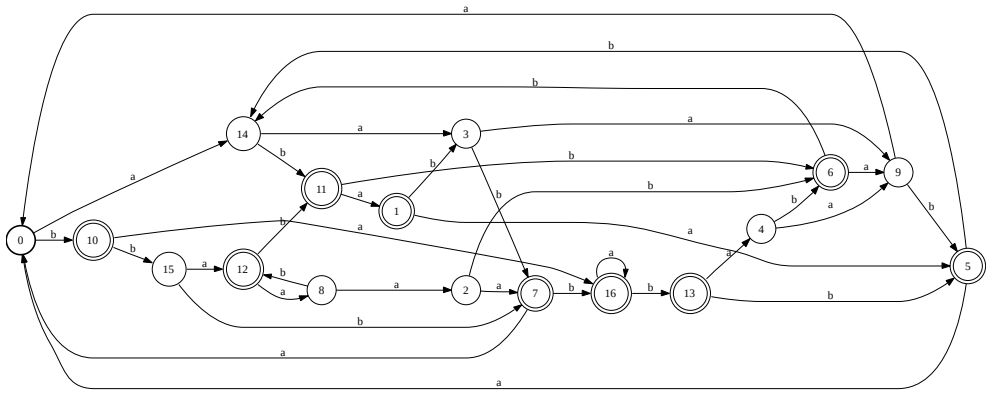
017.5.fsm



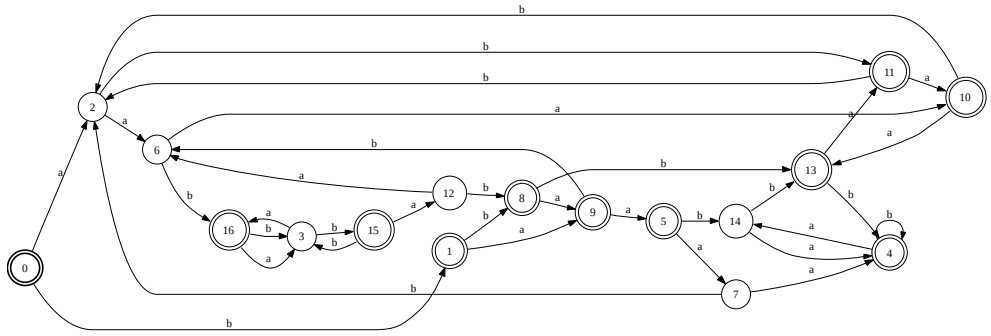
017.6.fsm



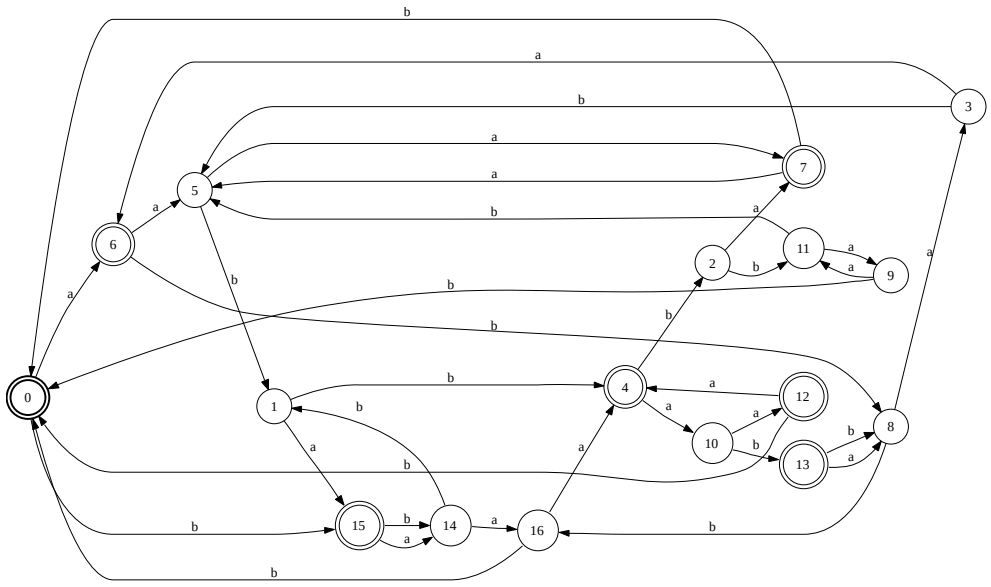
017.7.fsm



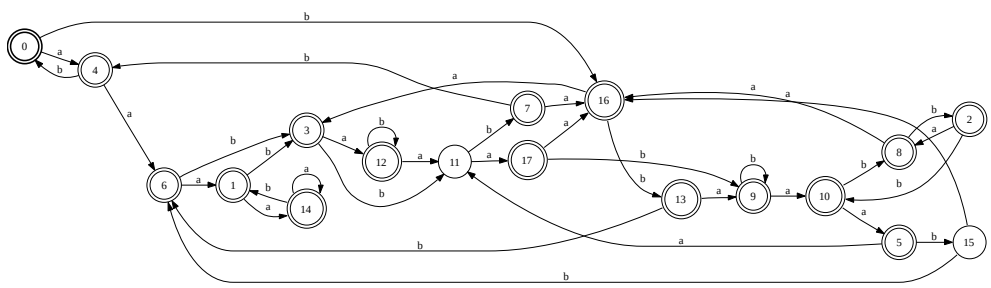
017.8.fsm



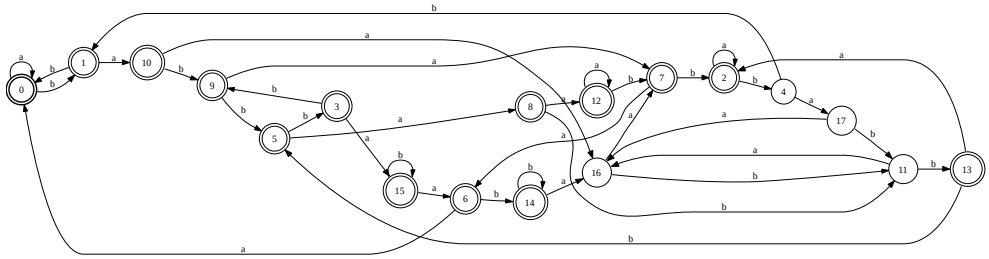
017.9.fsm



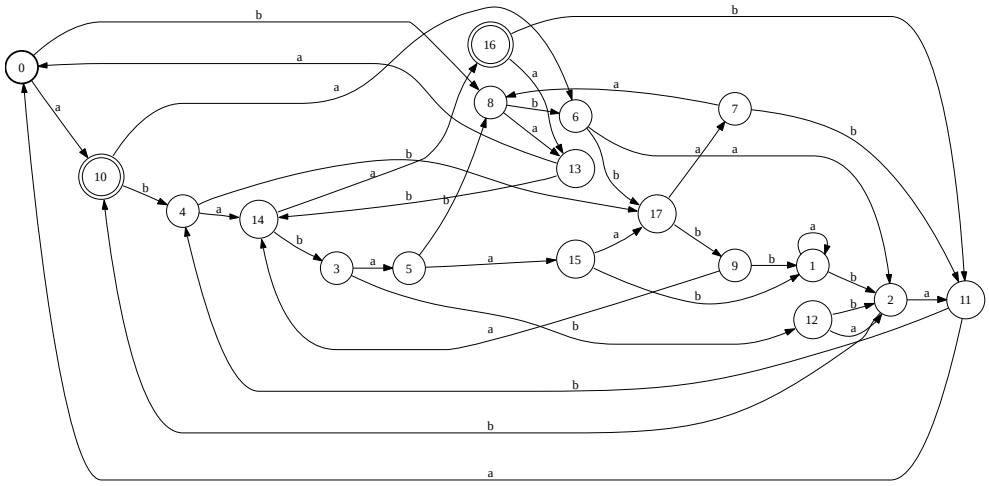
017.10.fsm



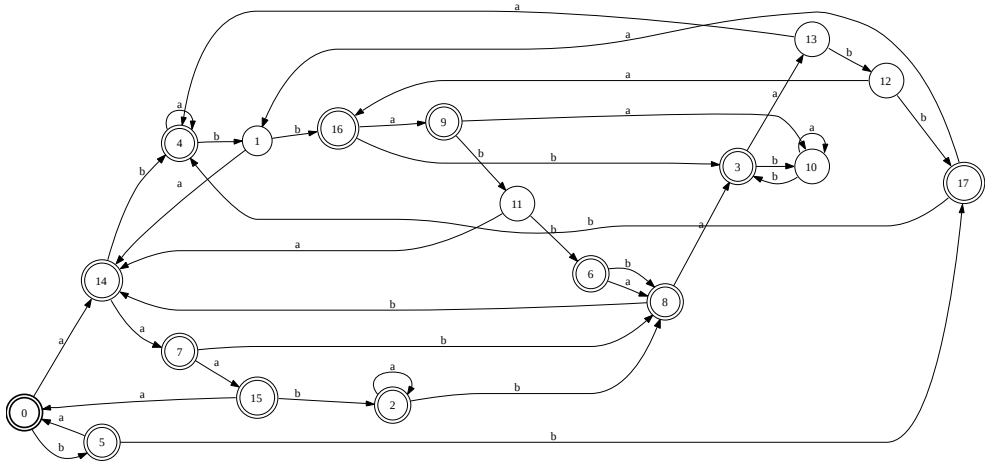
018.0.fsm



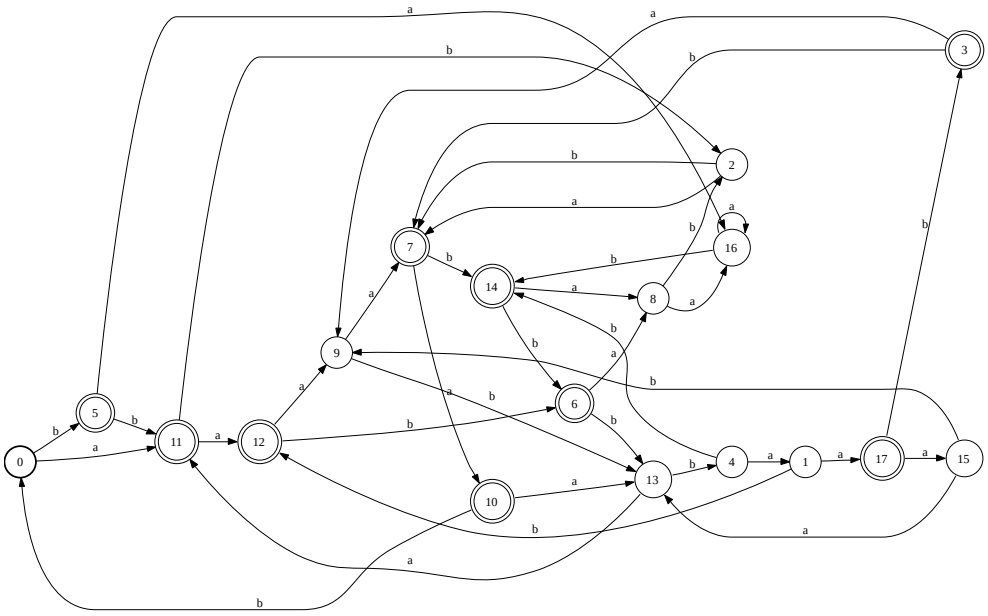
018.1.fsm



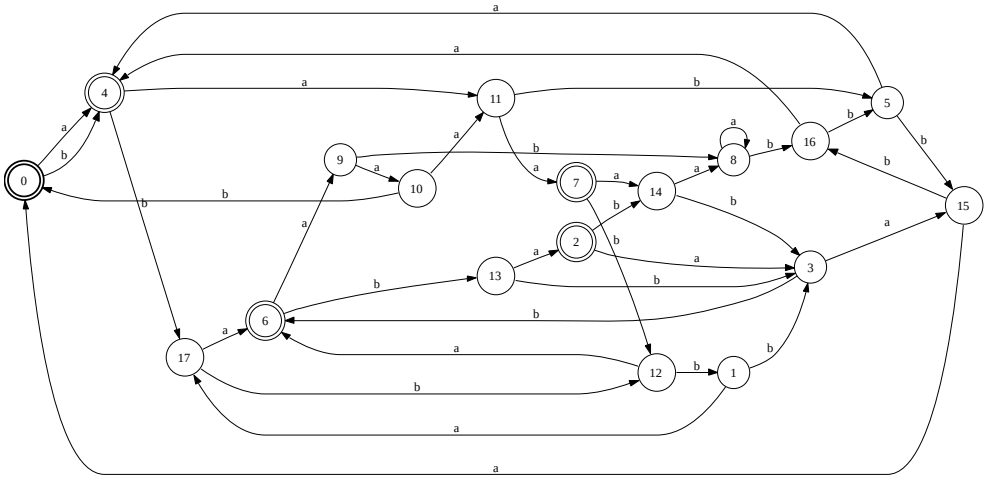
018.2.fsm



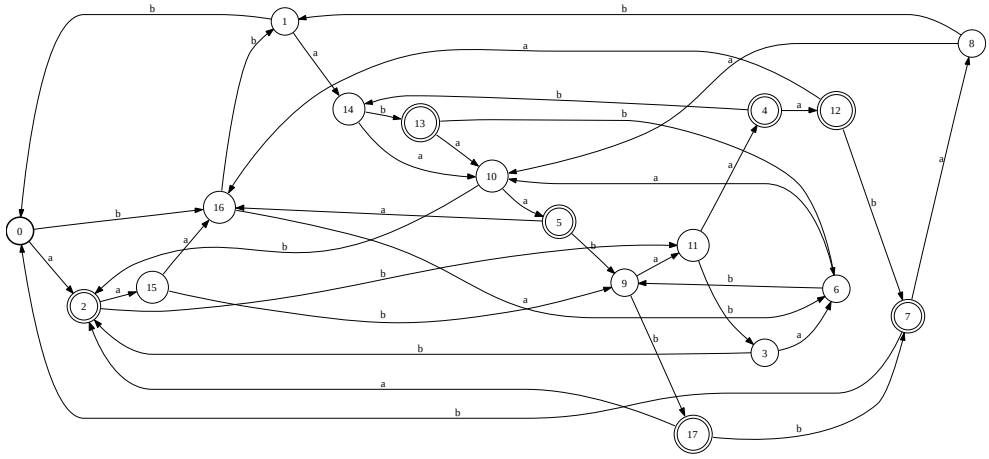
018.3.fsm



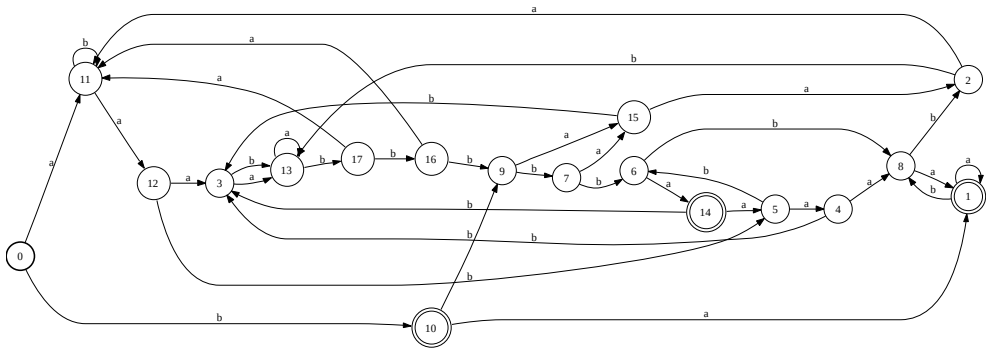
018.4.fsm



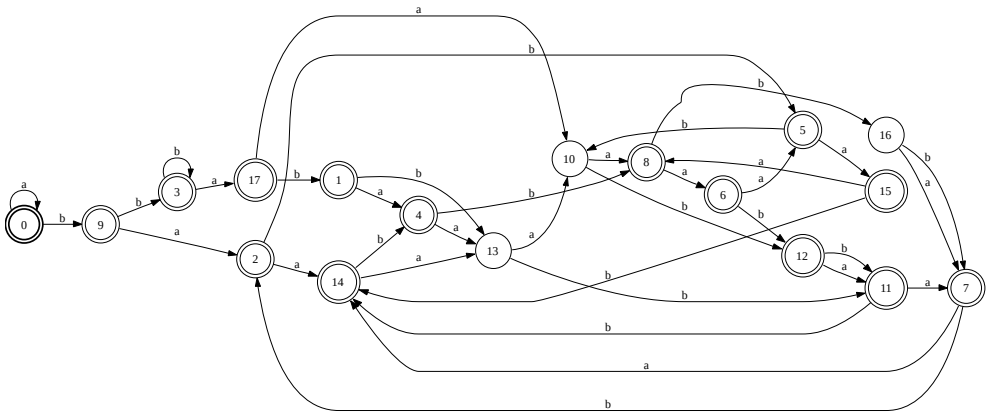
018.5.fsm



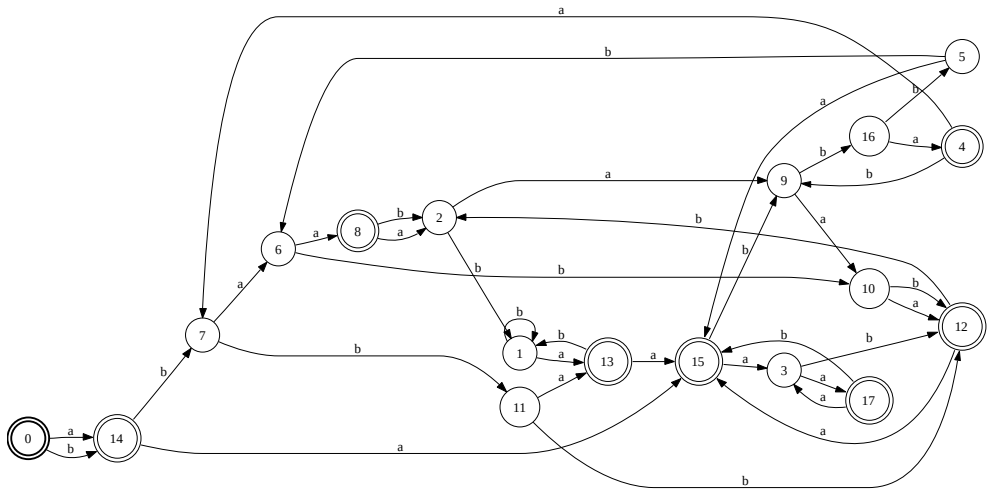
018.6.fsm



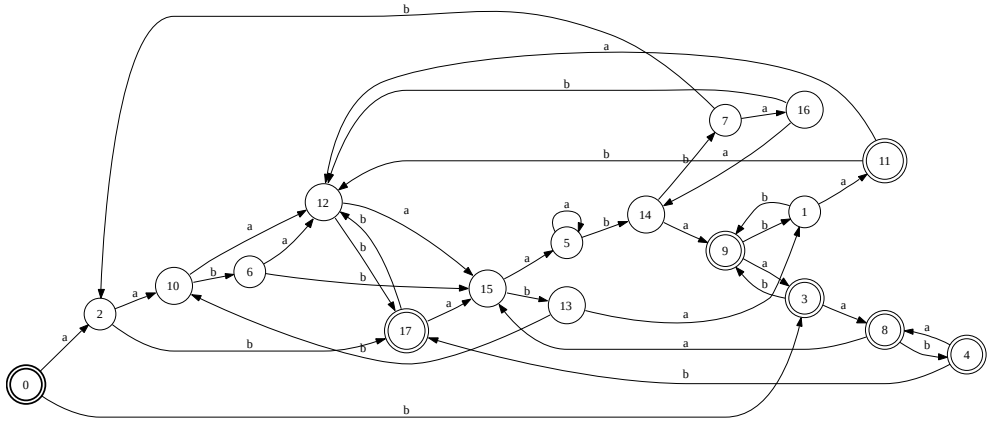
018.7.fsm



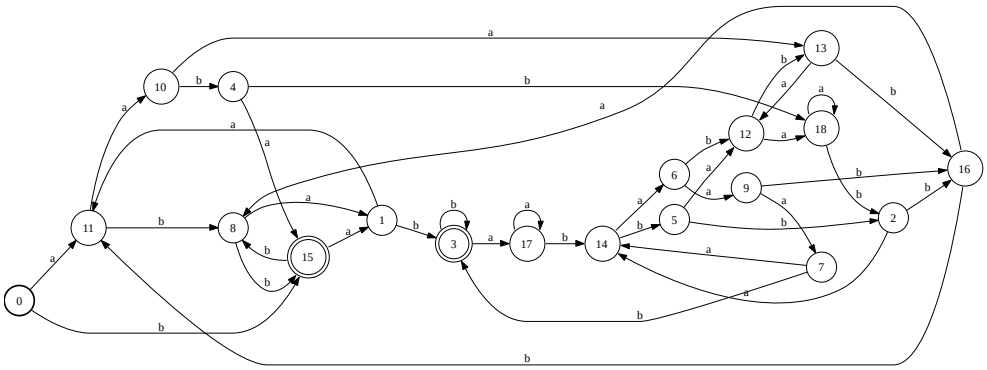
018.8.fsm



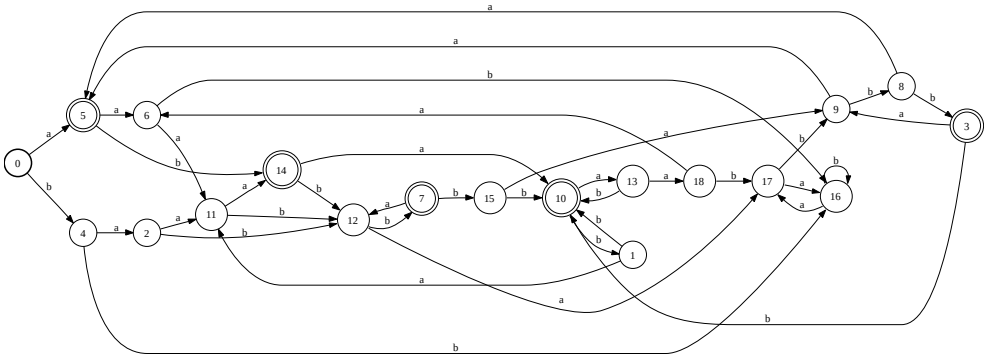
018.9.fsm



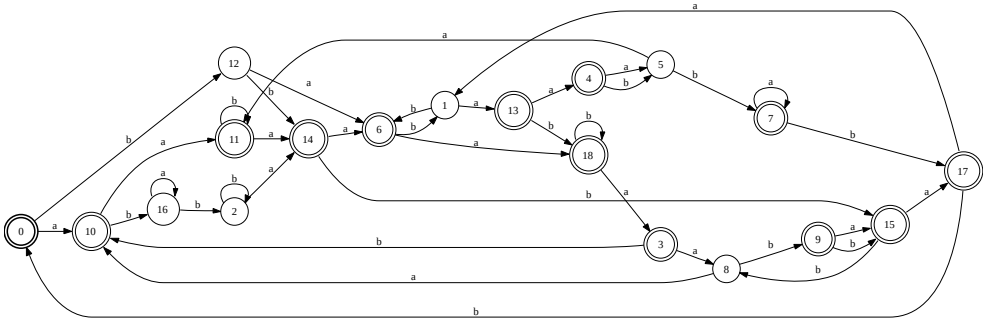
018.10.fsm



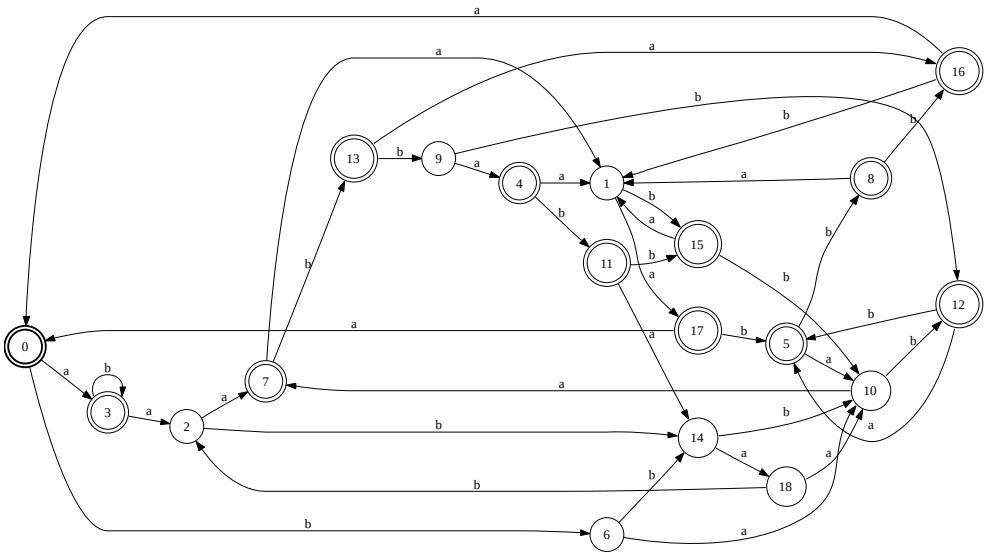
019.0.fsm



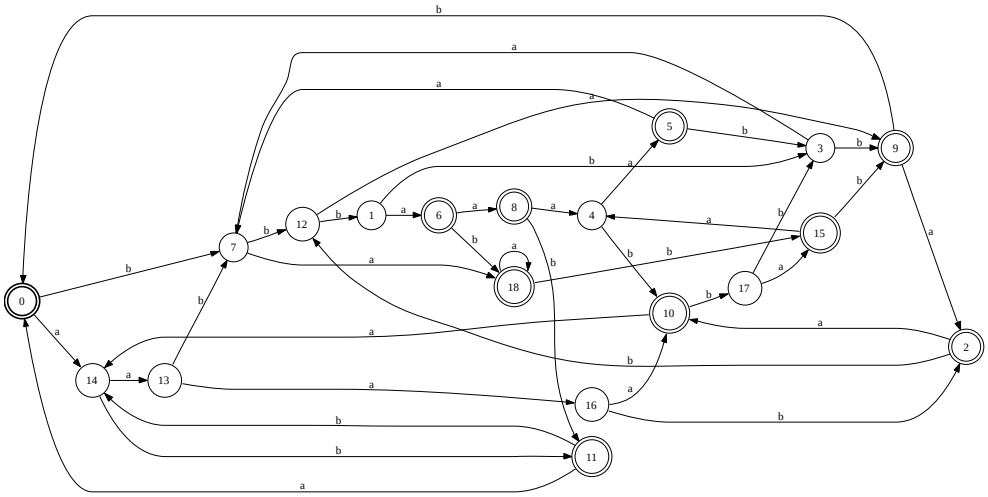
019.1.fsm



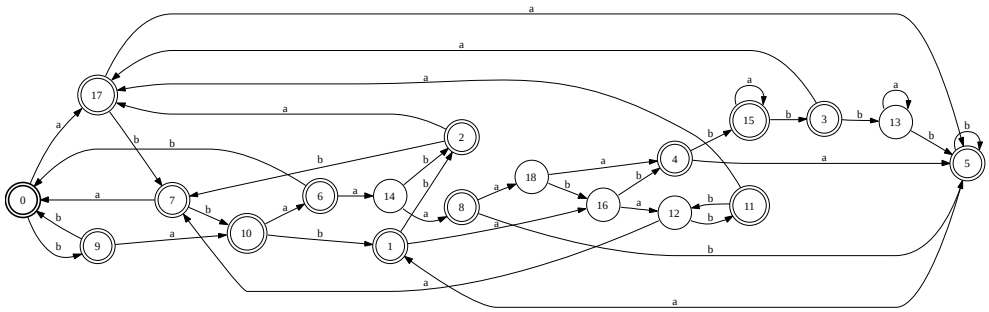
019.2.fsm



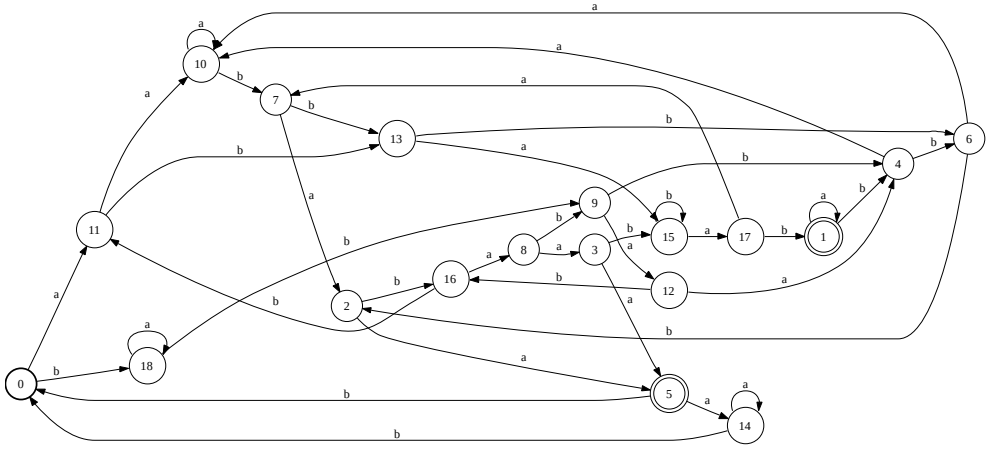
019.3.fsm



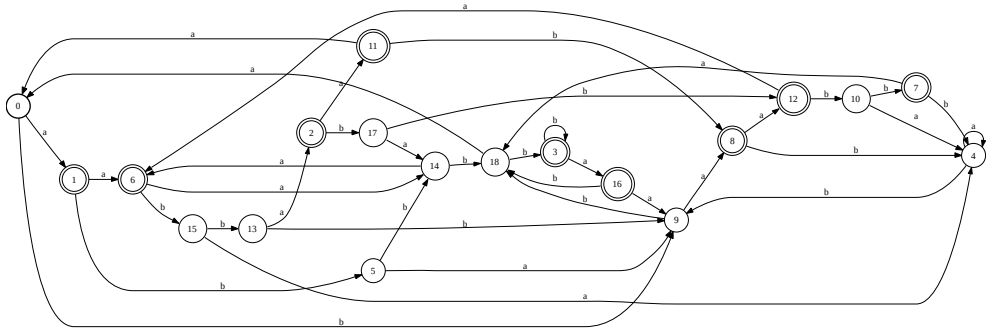
019.4.fsm



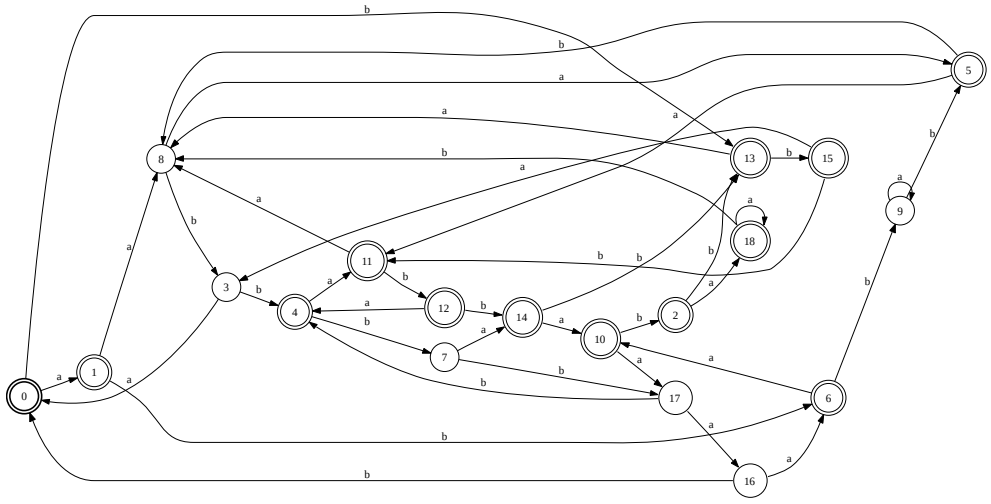
019.5.fsm



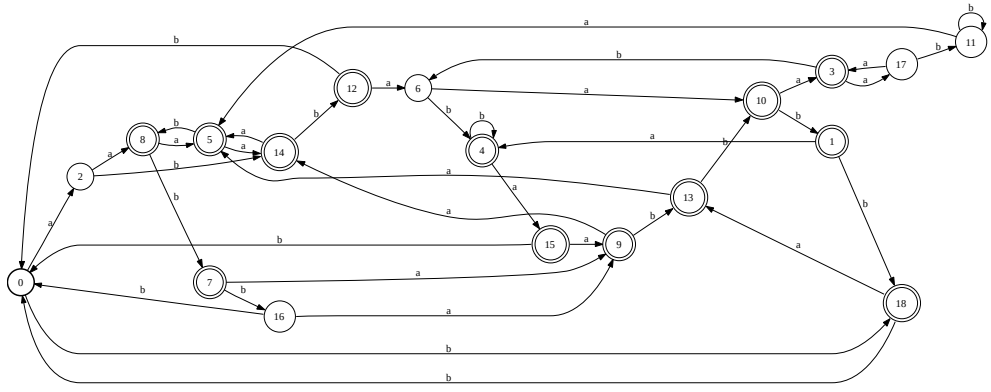
019.6.fsm



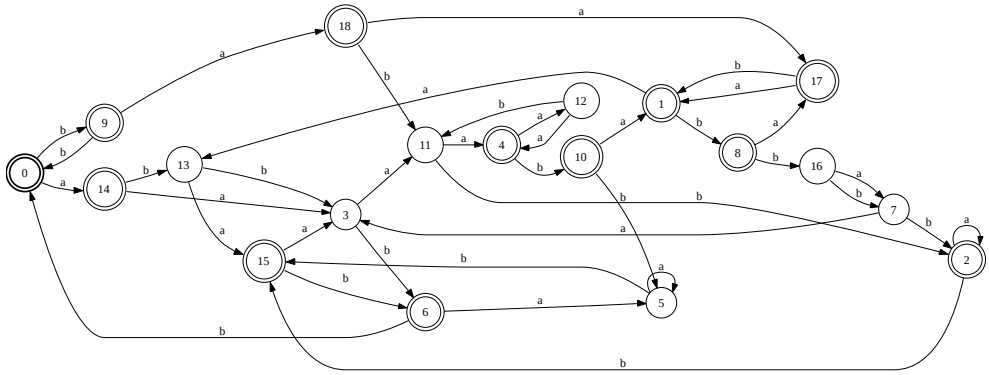
019.7.fsm



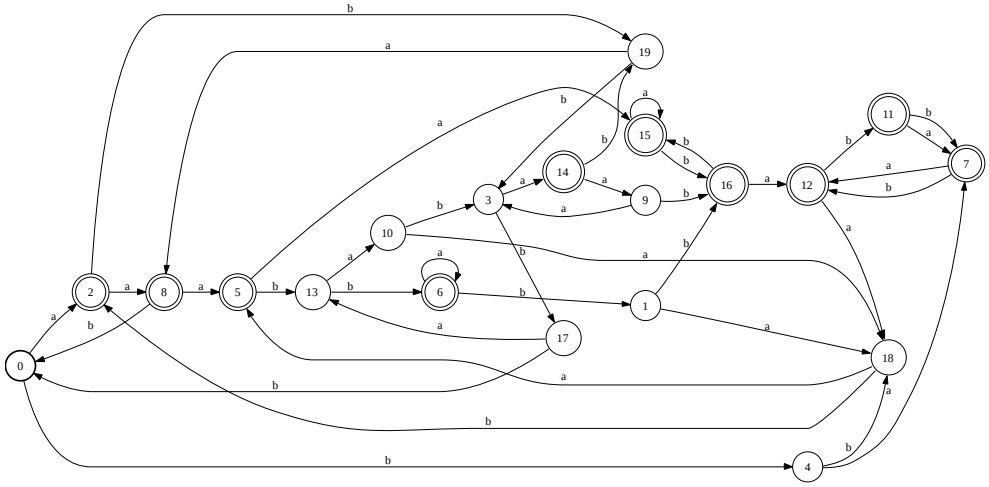
019.8.fsm



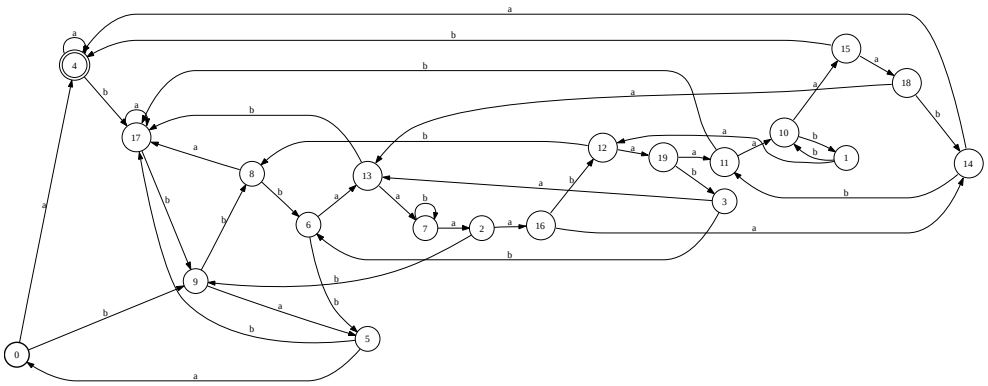
019.9.fsm



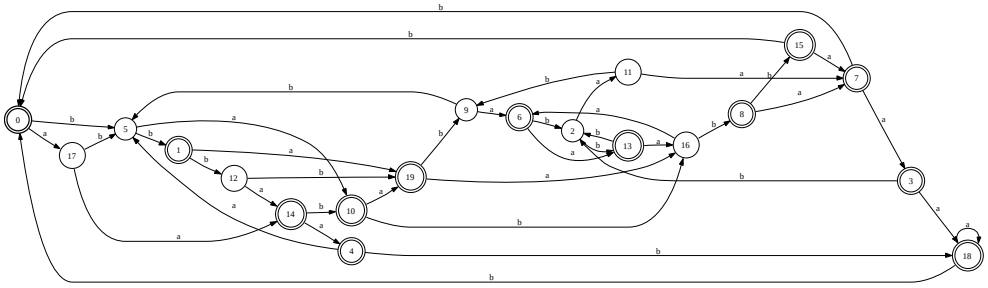
019.10.fsm



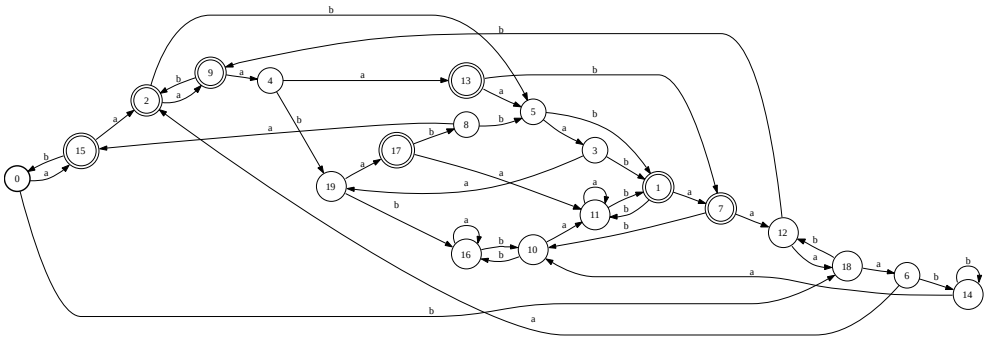
020.0.fsm



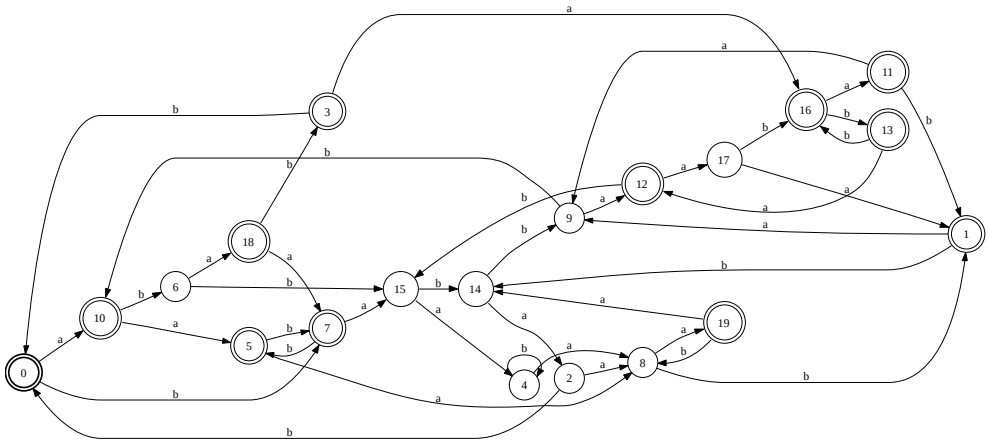
020.1.fsm



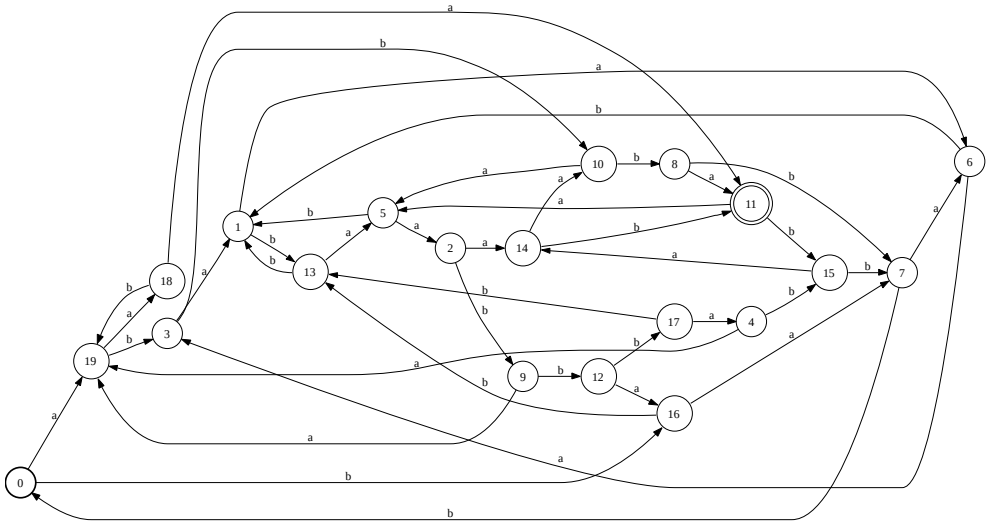
020.2.fsm



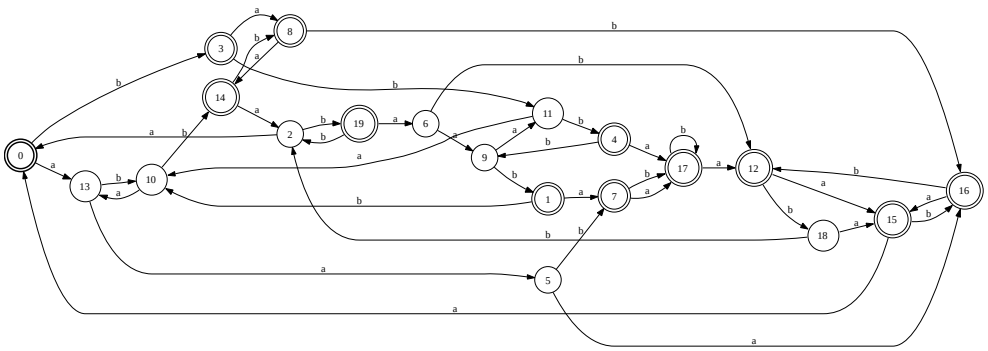
020.3.fsm



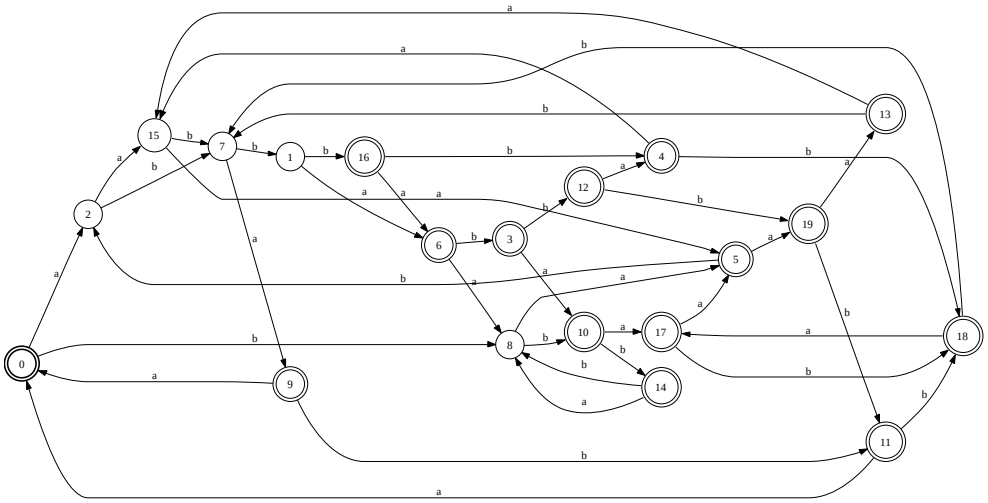
020.4.fsm



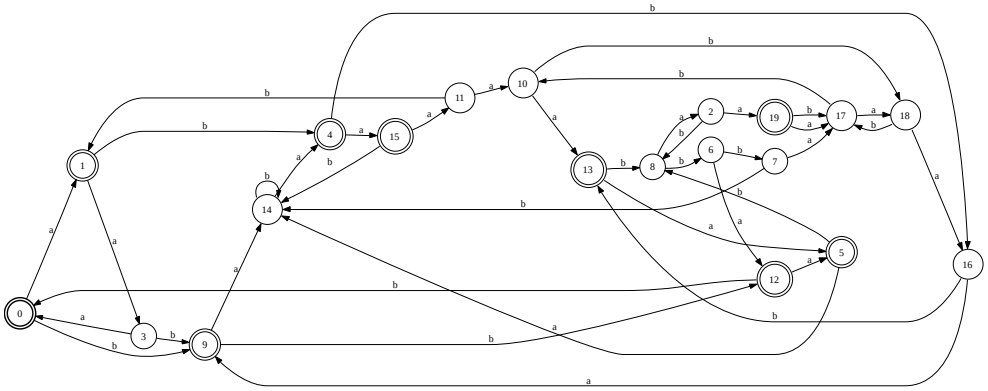
020.5.fsm



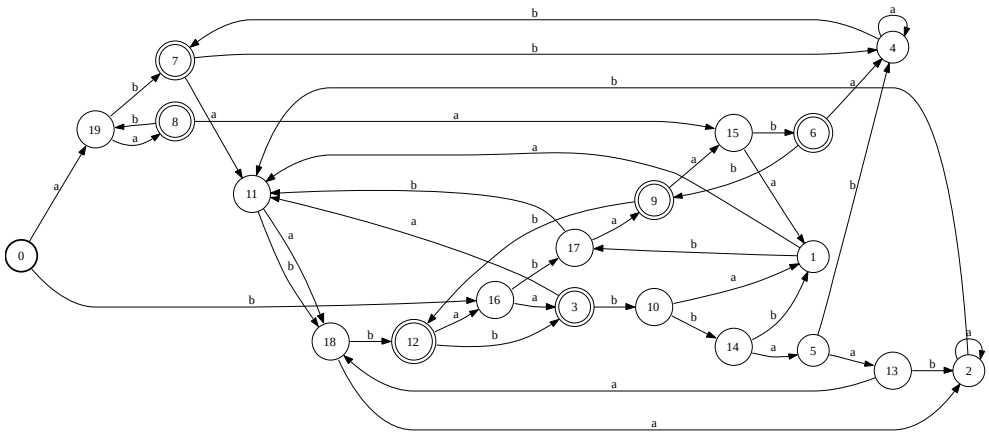
020.6.fsm



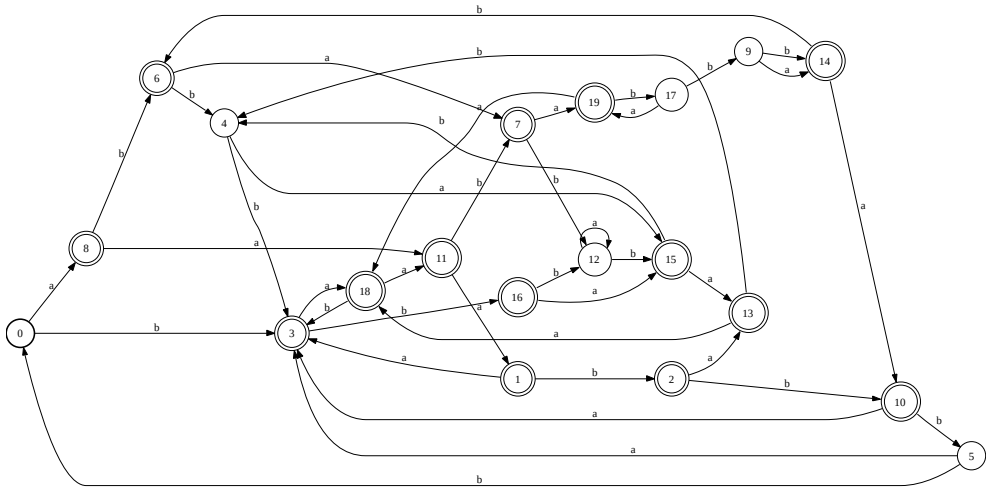
020.7.fsm



020.8.fsm



020.9.fsm



020.10.fsm

References

[Angluin, 1979] Angluin, D. (1979). Finding patterns common to a set of strings. In *STOC'79: Proceedings of the 11th annual ACM Symposium on Theory of Computing*, pages 130–141. ACM Press, New York, NY.

[Angluin, 1980] Angluin, D. (1980). Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135.

[Angluin, 1982] Angluin, D. (1982). Inference of reversible languages. *Journal of the Association for Computing Machinery*, 29(3):741–765.

[Angluin, 1987] Angluin, D. (1987). Learning regular sets from queries and counterexamples. *Information and Computation*, 75:87–106.

[Angluin, 1988] Angluin, D. (1988). Queries and concept learning. *Machine Learning*, 2:319–342.

[Angluin, 1990] Angluin, D. (1990). Negative results for equivalence queries. *Machine Learning*, 5:121–150.

[Angluin and Kharitonov, 1991] Angluin, D. and Kharitonov, M. (1991). When won't membership queries help? In *STOC'91: Proceedings of the 24th annual ACM Symposium on Theory of Computing*, pages 444–454. ACM Press, New York, NY.

[Angluin and Smith, 1983] Angluin, D. and Smith, A. H. (1983). Inductive inference: Theory and methods. *ACM Computing Surveys*, 15(3):237–269.

[Arikawa et al., 1989] Arikawa, S., Shinohara, T., and Yamamoto, A. (1989). Elementary formal system as a unifying framework for language learning. In *COLT '89: Proceedings of the second annual workshop on Computational learning theory*, pages 312–327. Morgan Kaufmann, San Francisco, CA.

[Balcázar et al., 1997] Balcázar, J. L., Díaz, J., Gavaldà, R., and Watanabe, O. (1997). Algorithms for learning finite automata from queries: A unified view. In Du, D. Z. and Ko, K. I., editors, *Chapter in Advances in Algorithms, Languages, and Complexity*, pages 73–91. Kluwer Academic Publishers, Dordrech.

[Bertolo, 2001] Bertolo, S. (2001). *Language Acquisition and Learnability.* Cambridge University Press, Cambridge, MA.

[Bohannon and Stanowicz, 1988] Bohannon, J. and Stanowicz, L. (1988). The issue of negative evidence: Adult responses to children's language errors. *Developmental Psychilogy*, 24:684–689.

[Bresnan et al., 1987] Bresnan, J., Kaplan, R., Peters, S., and Zaenen, A. (1987). Cross-serial dependencies in dutch. In Savitch, W., Bach, E., Marsh, W., and Safran-Naveh, G., editors, *The Formal Complexity of Natural Language*, pages 286–319. D. Reidel, Dordrecht.

[Brown and Hanlon, 1970] Brown, R. and Hanlon, C. (1970). Derivational complexity and order of acquisition in child speech. In Hayes, J., editor, *Cognition and the Development of Language*, pages 11–54. Wiley, New York, NY.

[Chomsky, 1956] Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, 3:113–124.

[Chomsky, 1957] Chomsky, N. (1957). *Syntactic Structures.* Mouton, The Hague.

[Chomsky, 1959] Chomsky, N. (1959). On certain formal properties of grammars. *Information and Control*, (2):137–167.

[Chomsky, 1965] Chomsky, N. (1965). *Aspects of the theory of syntax*. MIT Press, Cambridge, MA.

[Chomsky, 1975] Chomsky, N. (1975). *The Logical Structure of Linguistic Theory*. Plenum Press, New York, NY.

[Chomsky, 1980] Chomsky, N. (1980). *Rules and Representations*. Columbia University Press, New York, NY.

[Chomsky, 1981] Chomsky, N. (1981). *Lectures on Government and Binding*. Foris, Dordrecht.

[Chomsky, 1986] Chomsky, N. (1986). *Knowledge of language. Its Nature, Origin, and Use*. Praeger, Westport, CT.

[Chomsky, 1988] Chomsky, N. (1988). *Language and Problems of Knowledge*. MIT Press, Cambridge, MA.

[Chomsky, 1995] Chomsky, N. (1995). *The Minimalist Program*. MIT Press, Cambridge, MA.

[Clark, 2004] Clark, A. (2004). Grammatical inference and first language acquisition. In *Workshop on Psychocomputational Models of Human Language Acquisition*, pages 25–32. Geneva.

[Costa-Florêncio, 2003] Costa-Florêncio, C. (2003). *Learning Categorial Grammars*. PhD thesis, Proefschrift Universiteit Utrecht.

[Culy, 1987] Culy, C. D. Reidel, D. (1987). The complexity of the vocabulary of bambara. In Savitch, W., Bach, E., Marsh, W., and Safran-Naveh, G., editors, *The Formal Complexity of Natural Language*, pages 349–357.

[Dale and Christiansen, 2004] Dale, R. and Christiansen, M. (2004). Active and passive statistical learning: Exploring the role of feedback in artificial grammar learning and language. In *Proceedings of the 26th Annual Conferenceof the Cognitive Science Society*, pages 262–267. Lawrence Erlbaum, Mahwah, NJ.

[Dassow and Păun, 1989] Dassow, J. and Păun, G. (1989). *Regulated Rewriting in Formal Language Theory*. Springer-Verlag, Berlin.

[de Boysson-Bardies, 1999] de Boysson-Bardies, B. (1999). *How Language Comes to Children*. MIT Press, Cambridge, MA.

[de la Higuera, 1997] de la Higuera, C. (1997). Characteristic sets for polynomial grammatical inference. *Machine Learning Journal*, 27:125–138.

[de la Higuera, 2005] de la Higuera, C. (2005). A bibliographical study of grammatical inference. *Pattern Recognition*, 38:1332–1348.

[Denis, 2001] Denis, F. (2001). Learning regular languages from simple positive examples. *Machine Learning*, 44:37–66.

[Denis et al., 2002] Denis, F., Lemay, A., and Terlutte, A. (2002). Some classes of regular languages identifiable in the limit from positive data. In Adriaans, P., Fernau, H., and van Zaannen, M., editors, *Grammatical Inference: Algorithms and Applications, Proceedings of ICGI'00*, volume 2484 of *Lecture Notes in Computer Science*, pages 63–76. Springer-Verlag, Berlin.

[Farrar, 1992] Farrar, M. (1992). Negative evidence and grammatical morpheme acquisition. *Developmental Psychilogy*, 28:90–98.

[Fernau, 1999] Fernau, H. (1999). Efficient learning of some linear simple matrix languages. In Asano, T., Imai, H., Lee, D. T., Nakano, S.-I., and Tokuyama, T., editors, *Proc. of Computing and Combinatorics, 5th Annual International Conference, COCOON'99*, volume 1627 of *Lecture Notes in Computer Science*, pages 221–230. Springer-Verlag, Berlin.

[Fernau, 2000] Fernau, H. (2000). Identification of function distinguishable languages. In Arimura, H., Jain, S., and Sharma, A., editors, *Proceedings of the 11th International Conference on Algorithmic Learning Theory (ALT 2000)*, volume 1968 of *Lecture Notes in Computer Science*, pages 116–130. Springer-Verlag, Berlin.

[García and Vidal, 1990] García, P. and Vidal, E. (1990). Inference of k-testable languages in the strict sense and application to syntactic pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):920–925.

[Gavaldà, 1993] Gavaldà, R. (1993). On the power of equivalence queries. In *Euro-COLT '93: Proceedings of the first European conference on Computational learning theory*, volume 53, pages 193–203. Oxford University Press, New York, NY.

[Gazdar and Pullum, 1985] Gazdar, G. and Pullum, G. K. (1985). Computationally relevant properties of natural languages and their grammars. *New Generation Computing*, 3:273–306.

[Gold, 1967] Gold, E. (1967). Language identification in the limit. *Information and Control*, 10:447–474.

[Gold, 1978] Gold, E. (1978). Complexity of automaton identification from given data. *Information and Control*, 37:302–320.

[Goldman and Kearns, 1995] Goldman, S. and Kearns, M. (1995). On the complexity of teaching. *Journal of Computer and System Sciences*, 50(1):20–31.

[Goldman and Mathias, 1996] Goldman, S. and Mathias, H. (1996). Teaching a smarter learner. *Journal of Computer and System Sciences*, 52(2):255–267.

[Goodluck, 1991] Goodluck, H. (1991). *Language acquisition: A linguistic introduction*. Blackwell, Cambridge, MA.

[Gordon, 1990] Gordon, P. (1990). Learnability and feedback. *Developmental Psychology*, 26:217–220.

[Hellerstein et al., 1995] Hellerstein, L., Pillaipakkamnatt, K., Raghavan, V., and Wilkins, D. (1995). How many queries are needed to learn? In *Pro-*

ceedings of the Twenty-Seventh Annual ACM Symposium on the Theory of Computing, pages 190–199. ACM Press.

[Hopcroft et al., 2001] Hopcroft, J., Motwani, R., and Ullman, J. (2001). *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, Reading, MA.

[Ibarra, 1970] Ibarra, O. (1970). Simple matrix languages. *Information and Control*, 17:359–394.

[Ishizaka, 1990] Ishizaka, H. (1990). Polynomial time learnability of simple deterministic languages. *Machine Learning*, 5:151–164.

[Jain and Kinber, 2004] Jain, S. and Kinber, E. (2004). Learning languages from positive data and negative counterexamples. In *Proceedings of Algorithmic Learning Theory, 15th International Conference, ALT*, volume 3244 of *Lecture Notes in Computer Science*, pages 54–68. Springer-Verlag.

[Jain et al., 1999] Jain, S., Osherson, D., Royer, J., and Sharma, A. (1999). *Systems that Learn*. MIT Press, Cambridge, MA.

[Joshi, 1985] Joshi, A. K. (1985). How much context-sensitivity is required to provide reasonable structural descriptions: Tree adjoining grammars. In Dowty, D., Karttunen, L., and Zwicky, A., editors, *Natural Language Parsing: Psychological, Computational and Theoretical Perspectives*, pages 206–250. Cambridge University Press, New York, NY.

[Joshi and Schabes, 1997] Joshi, A. K. and Schabes, Y. (1997). Tree-adjoining grammars. In Rozenberg, G. and Salomaa, A., editors, *Handbook of Formal Languages*, volume 3, pages 69–123. Springer-Verlag, Berlin.

[Joshi et al., 1991] Joshi, A. K., Vijay-Shanker, K., and Weir, D. (1991). The convergence of mildly context-sensitive grammar formalisms. In Sells, P., Shieber, S. M., and Wasow, T., editors, *Foundational Issues in Natural Language Processing*, pages 31–81. MIT Press, Cambridge, MA.

[Kanazawa, 1998] Kanazawa, M. (1998). *Learnable Classes of Categorial Grammars*. Cambridge University Press, New York, NY.

[Kudlek et al., 2002] Kudlek, M., Martín-Vide, C., Mateescu, A., and Mitrana, V. (2002). Contexts and the concept of mild context-sensitivity. *Linguistics and Philosophy*, 26(6):703–725.

[Lange and Zeugmann, 1996] Lange, S. and Zeugmann, T. (1996). Incremental learning from positive data. *Journal of Computer and System Sciences*, 53:88–103.

[Makinen, 1992] Makinen, E. (1992). On the structural grammatical inference problem for some classes of context-free grammars. *Information Processing Letters*, 42:193–199.

[Manaster-Ramer, 1999] Manaster-Ramer, A. (1999). Some uses and abuses of mathematics in linguistics. In Martn-Vide, C., editor, *Issues in Mathematical Linguistics*, pages 73–130. John Benjamins, Amsterdam.

[Marcotte, 2004] Marcotte, J. (2004). Language acquisition, linguistic evidence, the baby, and the bathwater. *Journal of Linguistics*, pages 1–61.

[Marcus, 1993] Marcus, G. (1993). Negative evidence in language acquisition. *Cognition*, 46:53–95.

[Marcus, 1969] Marcus, S. (1969). Contextual grammars. *Revue Roumaine des Mathématiques Pures et Appliquées*, 14:1525–1534.

[Marcus, 1997] Marcus, S. (1997). Contextual grammars and natural languages. In Rozenberg, G. and Salomaa, A., editors, *Handbook of Formal Languages*, volume 2, pages 215–235. Springer-Verlag, Berlin.

[Marcus et al., 1998] Marcus, S., Martín-Vide, C., and Păun, G. (1998). Contextual grammars as generative models of natural languages. *Computational Linguistics*, 24(2):245–274.

[Martín-Vide, 1996] Martín-Vide, C. (1996). Natural language understanding: a new challenge for grammar systems. *Acta Cybernetica*, 12:461–472.

[Martín-Vide et al., 2004] Martín-Vide, C., Mitrana, V., and Păun, G., editors (2004). *Formal Languages and Applications*. Springer-Verlag, Berlin.

[Miclet, 1986] Miclet, L. (1986). Grammatical inference. In Bunke, H. and Sanfeliu, A., editors, *Syntactic and Structural Pattern Recognition - Theory and Applications*, pages 237–290. Word Scientific, Singapore.

[Moerk, 1983] Moerk, E. (1983). *The mother of Eve – As a first language teacher*. Ablex, Norwood, NJ.

[Mohri et al., 1998] Mohri, M., Pereira, F., and Riley, M. (1998). A rational design for a weighted finite-state transducer library. volume 1436 of *Lecture Notes in Computer Science*. Springer-Verlag.

[Morgan et al., 1995] Morgan, J., Bonamo, K., and Travis, L. (1995). Negative evidence on negative evidence. *Developmental Psychilogy*, 31:180–197.

[Nelson et al., 1995] Nelson, K., Denninger, M., Bonvillian, J., Kaplan, B., and Baker, N. (1995). Maternal adjustments and non-adjustments as related to children's linguistic advances and language acquistion theories. In *The development of oral and written languages: Readings in developmental and applied lingistics*, pages 135–182. Ablex, Norwood, NJ.

[Oncina et al., 1993] Oncina, J., García, P., and Vidal, E. (1993). Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:448–458.

[Parekh and Honavar, 2000] Parekh, R. and Honavar, V. (2000). Grammar inference, automata induction and language acquisition. In Dale, M. and Somers, editors, *Handbook of Natural Language Processing*, pages 727–774. Marcel Dekker, New York, NY.

[Pearl, 2004] Pearl, L. (2004). How learning works - a view from english participle morphology. In *Cognitive Neuroscience of Language Lunch Talks*. University of Maryland at College Park.

[Piaget, 1953] Piaget, J. (1953). *The Origin of Intelligence in the Child*. Routledge-Kegan Paul, London.

[Piaget, 1962] Piaget, J. (1962). *Play, Dreams and Imitation in Childhood*. Routledge-Kegan Paul, London.

[Pinker, 1979] Pinker, S. (1979). Formal models of language learning. *Cognition*, 7:217–283.

[Pinker, 1989] Pinker, S. (1989). *Learnability and cognition: The acquisition of the argument structure*. MIT Press, Cambridge, MA.

[Pinker, 1995] Pinker, S. (1995). Language acquistion. In *An Invitation to Cognitive Science*, pages 135–182. MIT Press.

[Pitt, 1989] Pitt, L. (1989). Inductive inference, dfas, and computational complexity. In *Proceedings of the AII-89 Workshop on Analogical and Inductive Inference*, volume 397 of *Lecture Notes in Computer Science, Springer*, pages 18–44. Springer-Verlag.

[Păun, 1997] Păun, G. (1997). *Marcus Contextual Grammars*. Kluwer, Dordrecht.

[Păun and Nguyen, 1980] Păun, G. and Nguyen, X. M. (1980). On the inner contextual grammars. *Revue Roumaine des Mathématiques Pures et Appliquées*, 25:641–651.

[Pullum and Gazdar, 1982] Pullum, G. K. and Gazdar, G. (1982). Natural languages and context-free languages. *Linguistics and Philosophy*, 4:471–504.

[Rivest and E.Schapire, 1993] Rivest, R. L. and E.Schapire, R. (1993). Inference of finite automata using homing sequences. *Information and Computation*, 103(2):299–347.

[Roach, 1987] Roach, K. (1987). Formal porperties of head grammars. In Manaster-Ramer, A., editor, *Mathematics of Language*, pages 293–348. John Benjamins, Amsterdam.

[Ron et al., 1994] Ron, D., Singer, Y., and Tishby, N. (1994). Learning probabilistic automata with variable memory length. In *Proceedings on the Seventh Annual ACM Conference on Computational Learning Theory*, pages 35–46. ACM Press.

[Rozenberg and Salomaa, 1997] Rozenberg, G. and Salomaa, A., editors (1997). *Handbook of Formal Languages*, volume 1–3. Springer-Verlag, Berlin.

[Sakakibara, 1990] Sakakibara, Y. (1990). Learning context-free grammars from structural data in polynomial time. *Theoretical Computer Science*, 76:223–242.

[Sakakibara, 1992] Sakakibara, Y. (1992). Efficient learning of context-free grammars from positive structural examples. *Information Processing Letters*, 97:23–60.

[Sakakibara, 1997] Sakakibara, Y. (1997). Recent advances of grammatical inference. *Theoretical Computer Science*, 185:15–45.

[Saxton, 1997] Saxton, M. (1997). The contrast theory of negative input. *Journal of Child Language*, 24:139–161.

[Saxton et al., 1998] Saxton, M., Kulcsar, B., Marshall, G., and Rupra, M. (1998). Long term effects of corrective input and experimental approach. *Journal of Child Language*, 25:701–721.

[Sempere and García, 1994] Sempere, J. and García, P. (1994). A characterization of even linear languages and its application to the learning problem. In *Proceedings of the Second International Colloquium on Grammatical Inference (ICGI-94): Grammatical Inference and Applications*, volume 862 of *Lecture Notes in Computer Science*, pages 38–44. Springer-Verlag.

[Shieber, 1987] Shieber, S. (1987). Evidence against the context-freeness of natural languages. In Savitch, W., Bach, E., Marsh, W., and Safran-Naveh, G., editors, *The Formal Complexity of Natural Language*, pages 320–334. D. Reidel, Dordrecht.

[Shinohara, 1990] Shinohara, T. (1990). Inductive inference from positive data is powerful. In Fulk, M., editor, *Proceedings of the Third Annual Workshop on Computational Learning Theory (COLT 1990)*. Morgan Kaufmann, Rochester, NY.

[Shinohara, 1994] Shinohara, T. (1994). Rich classes inferable from positive data: Length-bounded elementary formal systems. *Information and Computation*, 108:175–186.

[Skinner, 1957] Skinner, B. (1957). *Verbal Behavior*. Printice Hall, New York, NY.

[Smullyan, 1961] Smullyan, R. (1961). Theory of formal systems. In *Annals of Mathematical Studies*. Princeton Univ. Press.

[Steedman, 1985] Steedman, M. (1985). Dependency and coordination in the grammar of dutch and english. *Language*, 61:523–568.

[Takada, 1988] Takada, Y. (1988). Grammatical inference for even linear languages based on control sets. *Information Processing Letters*, 28(4):193–199.

[Turing, 1936] Turing, A. (1936). On computable real numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(2):230–265.

[Valiant, 1984] Valiant, L. (1984). A theory of the learnable. *Communication of the ACM*, 27:1134–1142.

[Victor, 2005] Victor, M. (2005). Marcus external contextual grammars: From one to many dimensions. *Fundamenta Informaticae*, 54:307–316.

[Wexler and Culicover, 1980] Wexler, K. and Culicover, P. (1980). *Formal Principles of Languages Acquisition*. MIT Press, Cambridge, MA.

[Wintner, 2002] Wintner, S. (2002). Formal language theory for natural language processing. In *Effective tools and methodologies for teaching natural*

language processing and computational linguistics, an ACL'02 Workshop, pages 1–46. Philadelphia, PA.

[Yokomori, 1991] Yokomori, T. (1991). Polynomial-time learning of very simple grammars from positive data. In *COLT'91: Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 213–227. Morgan Kaufmann, Santa Cruz, CA.

[Yokomori, 1994] Yokomori, T. (1994). Learning non-deterministic finite automata from queries and counterexamples. In Furukawa, K., Michie, D., and Muggleton, S., editors, *Machine Intelligence*, volume 13, pages 169–189. Clarendon Press, Oxford.

[Yokomori, 1995] Yokomori, T. (1995). On polynomial-time learnability of strictly deterministic automata. *Machine Learning*, 19:153–179.

[Yokomori, 1996] Yokomori, T. (1996). Learning two-tape automata from queries and counterexamples. *Mathematical Systems Theory*, pages 259–270.

[Yokomori, 2004] Yokomori, T. (2004). Grammatical inference and learning. In Martín-Vide, C., Mitrana, V., and Păun, G., editors, *Formal Languages and Applications*, pages 507–528. Springer-Verlag, Berlin.

[Yokomori and Kobayashi, 1998] Yokomori, T. and Kobayashi, S. (1998). Learning local languages and their applications to DNA sequence analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1067–1079.

Index

UNIVERSITAT ROVIRA I VIRGILI

ON THE LEARNABILITY OF MILDLY CONTEXT-SENSITIVE LANGUAGES USING POSITIVE DATA AND CORRECTION QUERIES

LEONOR BECERRA - BONACHE

PhD Dissertation

Tarragona, 2006