Second-Order Finite Automata

Alexsander Andrade de Melo[1] · Mateus de Oliveira Oliveira[2]

Abstract

Traditionally, finite automata theory has been used as a framework for the representation of possibly infinite sets of strings. In this work, we introduce the notion of second-order finite automata, a formalism that combines finite automata with ordered decision diagrams, with the aim of representing possibly infinite *sets of sets* of strings. Our main result states that second-order finite automata can be canonized with respect to the second-order languages they represent. Using this canonization result, we show that sets of sets of strings represented by second-order finite automata are closed under the usual Boolean operations, such as union, intersection, difference and even under a suitable notion of complementation. Additionally, emptiness of intersection and inclusion are decidable. We provide two algorithmic applications for second-order automata. First, we show that several width/size minimization problems for deterministic and nondeterministic ODDs are solvable in fixed-parameter tractable time when parameterized by the width of the input ODD. In particular, our results imply FPT algorithms for corresponding width/size minimization problems for ordered binary decision diagrams (OBDDs) with a fixed variable ordering. Previously, only algorithms that take exponential time in the size of the input OBDD were known for width minimization, even for OBDDs of constant width. Second, we show that for each k and w one can count the number of distinct functions computable by ODDs of width at most w and length k in time $h(|\Sigma|, w) \cdot k^{O(1)}$, for a suitable $h : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$. This improves exponentially on the time necessary to explicitly enumerate all such functions, which is exponential in both the width parameter w and in the length k of the ODDs.

✉ Mateus de Oliveira Oliveira
mateus.oliveira@uib.no

Alexsander Andrade de Melo
aamelo@cos.ufrj.br

[1] Federal University of Rio de Janeiro, Rio de Janeiro, Brazil

[2] University of Bergen, Bergen, Norway

1 Introduction

In its most traditional setting, automata theory has been used as a framework for
the representation and manipulation of (possibly infinite) sets of strings. This frame-
work has been generalized in many ways to allow the representation of sets of more
elaborate combinatorial objects, such as trees [10], partial orders [36], graphs [6], pic-
tures [19], etc. Such notions of automata have encountered innumerous applications
in fields such as formal verification [5, 20], finite model theory [16], concurrency
theory [34], parameterized complexity [11, 12], etc. Still, these generalized notions
of automata share in common the fact that they are designed to represent (possibly
infinite) sets of isolated objects.

In this work, we combine traditional finite automata with ordered decision dia-
grams (ODDs) of bounded width to introduce a formalism that can be used to
represent and manipulate *sets of sets* of strings, or alternatively speaking, classes
of languages. We call this combined formalism *second-order finite automata*. We
will show that the width of an ODD is a useful parameter when studying classes
of languages from a complexity-theoretic point of view. Additionally, we will use
second-order finite automata to show that several computational problems involv-
ing ordered decision diagrams are fixed-parameter tractable when parameterized
by width.

Given a finite alphabet Σ and a number $w \in \mathbb{N}_+$, a (Σ, w)-ODD is a sequence
$D = B_1 B_2 \ldots B_k$ of (Σ, w)-layers. Each such a layer B_i has a set of left-states (a
subset of $\{1, \ldots, w\}$), a set of right-states (also a subset of $\{1, \ldots, w\}$), and a set of
transitions, labeled with letters in Σ, connecting left states to right states. We require
that for each $i \in \{1, \ldots, k-1\}$, the set of right-states of the layer B_i is equal to
the set of left states of the layer B_{i+1}. The language of an ODD D is the set of
strings labelling paths from its set of initial states (a subset of the left states of B_1)
to its final states (a subset of the right states of B_k). Since the number of distinct
(Σ, w)-layers is finite, the set $\mathcal{B}(\Sigma, w)$ of all (Σ, w)-layers can itself be regarded as
an alphabet. A finite automaton $\mathcal{F}$ over the alphabet $\mathcal{B}(\Sigma, w)$ is said to be a second-
order finite automaton if each string $D = B_1 \ldots B_k$ in the language $\mathcal{L}(\mathcal{F})$ accepted
by $\mathcal{F}$ is a valid ODD. In this case, the second language of $\mathcal{F}$ is defined as the class
$\mathcal{L}_2(\mathcal{F}) = \{\mathcal{L}(D) \ : \ D \in \mathcal{L}(\mathcal{F})\}$ of languages accepted by ODDs in $\mathcal{L}(\mathcal{F})$. We say
that a class of languages $\mathcal{X}$ is *regular-decisional* if there is some second-order finite
automaton $\mathcal{F}$ such that $\mathcal{L}_2(\mathcal{F}) = \mathcal{X}$.

Canonical Forms for Second Order Finite Automata Our main result (Theorem 10)
states that second-order finite automata can be effectively canonized with respect
to their second languages. More specifically, there is an algorithm that maps each
second-order finite automaton $\mathcal{F}$ to a second-order finite automaton $\mathcal{C}_2(\mathcal{F})$, called
the second canonical form of $\mathcal{F}$, in such a way that the following three properties
are satisfied. First, $\mathcal{C}_2(\mathcal{F})$ and $\mathcal{F}$ have the same second language. That is to say,

$\mathcal{L}_2(\mathcal{C}_2(\mathcal{F})) = \mathcal{L}_2(\mathcal{F})$. Second, any two second-order finite automata $\mathcal{F}$ and $\mathcal{F}'$ with identical second languages are mapped to the same canonical form. More formally, $\mathcal{L}_2(\mathcal{F}) = \mathcal{L}_2(\mathcal{F}') \Rightarrow \mathcal{C}_2(\mathcal{F}) = \mathcal{C}_2(\mathcal{F}')$. Third, $\mathcal{L}(\mathcal{C}_2(\mathcal{F})) = \{\mathcal{C}(D) \;:\; D \in \mathcal{L}(\mathcal{F})\}$. Here, $\mathcal{C}(D)$ is the unique deterministic, complete, normalized[1] ODD with minimum number of states with the same language as D. Intuitively, the language of $\mathcal{C}_2(\mathcal{F})$ consists precisely of the set of canonical forms of ODDs in the language of $\mathcal{F}$. For this reason, we say that Theorem 10 is a *canonical form of canonical forms theorem*. From a complexity-theoretic point of view, $\mathcal{C}_2(\mathcal{F})$ can be constructed in time $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{O(|\Sigma| \cdot w \cdot 2^w)}}$, where $\mathsf{nSt}(\mathcal{F})$ is the number of states of $\mathcal{F}$. Additionally this construction can be sped up to time $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{O(|\Sigma| \cdot w \cdot \log w)}}$ if all ODDs in $\mathcal{L}(\mathcal{F})$ are deterministic and complete (Observation 11).

We note that canonizing a second-order finite automaton $\mathcal{F}$ with respect to its second language $\mathcal{L}_2(\mathcal{F})$ is *not* equivalent to canonizing $\mathcal{F}$ with respect to its language $\mathcal{L}(\mathcal{F})$. For instance, let D and D' be distinct ODDs such that $\mathcal{L}(D) = \mathcal{L}(D')$. Let $\mathcal{F}$ and $\mathcal{F}'$ be second-order finite automata with $\mathcal{L}(\mathcal{F}) = \{D\}$ and $\mathcal{L}(\mathcal{F}') = \{D'\}$. Then the languages of $\mathcal{F}$ and $\mathcal{F}'$ are distinct ($\mathcal{L}(\mathcal{F}) \neq \mathcal{L}(\mathcal{F}')$) even though their second languages are equal ($\mathcal{L}_2(\mathcal{F}) = \{\mathcal{L}(D)\} = \{\mathcal{L}(D')\} = \mathcal{L}_2(\mathcal{F}')$).

At a high level, what our canonization algorithm does is to eliminate ambiguity in the language of a given second-order finite automaton. More specifically, any two ODDs D and D' with $\mathcal{L}(D) = \mathcal{L}(D')$ in the language of a second-order finite automaton $\mathcal{F}$ correspond to a single ODD $\mathcal{C}(D) = \mathcal{C}(D')$ in the language of $\mathcal{C}_2(\mathcal{F})$. This implies almost immediately that the collection of regular-decisional classes of languages is closed under union, intersection, set difference, and even under a suitable notion of complementation. Furthermore, emptiness of intersection and inclusion for the second languages of second-order finite automata are decidable (Theorem 13). It is interesting to note that non-emptiness of intersection for the second languages of second-order finite automata can be tested in fixed-parameter tractable time, where the parameter is the maximum width of an ODD accepted by one of the input automata (Observation 14). Finally, closure under several operations that are specific to classes of languages, such as *pointwise union*, *pointwise intersection* and *pointwise negation*, among others can also be obtained as a direct corollary (Corollary 16) of a technical lemma from [15].

Main Technical Tool Let $\mathcal{B}(\Sigma, w)^{\circledast}$ be the set of all (Σ, w)-ODDs and $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ be the set of all deterministic, complete (Σ, w)-ODDs. The main technical tool of this work (Theorem 9) states that the transduction $\mathfrak{can}[\Sigma, w] = \{(D, \mathcal{C}(D)) \;:\; D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular. In other words, there is an NFA with $2^{O(|\Sigma| \cdot w \cdot 2^w)}$ states accepting the language $\{D \otimes \mathcal{C}(D) \;:\; D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$. Additionally, the transduction $\widehat{\mathfrak{can}}[\Sigma, w] = \{(D, \mathcal{C}(D)) \;:\; D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}\}$, whose domain is restricted to deterministic, complete ODDs, is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular.

Most results of our work follow as a consequence of Theorem 9. If we do not take complexity theoretic issues into account, then some of our *decidability* results also

---

[1] By *normalized* we mean that the states of the ODD are numbered according to their lexicographical order. In this way $\mathcal{C}(D)$ is *syntactically* unique and not only unique up to isomorphism.

follow by employing other notions of canonizing relations (see Section 7 for further discussion on this topic). Nevertheless, the transductions $\mathfrak{can}[\Sigma, w]$ and $\widehat{\mathfrak{can}}[\Sigma, w]$ enjoy special properties that make them attractive from a complexity theoretic point of view. In particular, as we will see next, these transductions have applications in the fixed-parameter tractability theory of computational problems related to ordered decision diagrams ODDs. It is worth noting that ODDs comprise the well studied notion of ordered binary decision diagrams (OBDDs) with fixed variable ordering as a special case. And indeed, the width parameter has relevance in several contexts, such as learning theory [17], the theory of pseudo-random generators [18], the theory of symbolic algorithms [15], and structural graph theory [13]. Additionally, Theorem 9 implies that the set $\{\mathcal{C}(D) \quad : \quad D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$ of all minimized, deterministic, complete ODDs accepting the language of some ODD in $\mathcal{B}(\Sigma, w)^{\circledast}$ is regular (Corollary 12), and therefore, can be accepted by some deterministic finite automaton $\mathcal{F}$. This result may be of independent interest since the fact that the canonical form $\mathcal{C}(D)$ has minimum number of states among all deterministic, complete ODDs with the same language as D is a relevant complexity theoretic information about the language $\mathcal{L}(D)$. One interesting consequence of this result is that there is a bijection b from the set of accepting paths of $\mathcal{F}$ and the class of languages accepted by ODDs in $\mathcal{B}(\Sigma, w)^{\circledast}$. Additionally, the ODD corresponding to each such a path $\mathfrak{p}$ has minimum number of states among all deterministic, complete ODDs accepting the language $b(\mathfrak{p})$.

Algorithmic Applications Although ODDs of constant width constitute a simple computational model, they can already be used to represent many interesting functions. It is worth noting that for each width $w \geq 3$, the class of functions that can be represented by ODDs of constant width is at least as difficult to learn in the PAC-learning model as the problem of learning DNFs [17]. Additionally, the study of ODDs of constant width is still very active in the theory of pseudo-random generators [18]. Our main results can be used to show that several width/size minimization problems for nondeterministic and deterministic ODDs can be solved in fixed parameter tractable time when parameterized by width. For instance, we show that given an ODD D of length k and width w over an alphabet Σ, one can compute in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot k$ an ODD D' of minimum width such that $\mathcal{L}(D') = \mathcal{L}(D)$. A more efficient algorithm, running in time $2^{O(|\Sigma| \cdot w \cdot \log w)} \cdot k$ can be obtained if the input ODD is deterministic (Theorem 20). Our algorithm is in fact more general and can be used to minimize other complexity measures, such as number of states and number of transitions among all ODDs belonging to the language of a given second-order finite automaton $\mathcal{F}$ (Theorem 19).

Our algorithm for width minimization of ODDs parameterized by width naturally can be used to minimize the width of ordered binary decision diagrams (OBDDs), since OBDDs *with a fixed variable ordering* correspond to ODDs over a binary alphabet. Width minimization problems for OBDDs have been considered before in the literature [3, 4], but previously known algorithms are exponential on the size of the OBDD even for OBDDs of constant width, and even in the case of when one is not allowed to vary the order of the input variables. Our FPT result shows that width

minimization for OBDDs of constant width with a fixed variable ordering can be achieved in polynomial time.

As a second application of our main results, we show that the problem of counting the number of distinct functions computable by ODDs of a given width w and a given length k can be solved in time $2^{2^{O(|\Sigma| \cdot w \cdot 2^w)}} \cdot k^{O(1)}$. This running time can be improved to $2^{2^{O(|\Sigma| \cdot w \cdot \log w)}} \cdot k^{O(1)}$ if we are interested in counting the number of functions computable by deterministic, complete ODDs of width w and length k (Corollary 24). We note that this restricted case is relevant because ordered binary decision diagrams (OBDDs) defined in the literature are usually deterministic and complete. Our results imply that counting the number of functions computable by OBDDs of width w with a fixed variable ordering can be solved in time polynomial in the number of variables. This improves exponentially on the approach of explicit enumeration without repetitions, which takes time exponential in k. This result is obtained as a consequence of a more general theorem analyzing the complexity of the problem counting functions represented by ODDs of a given length in the language of a given second-order finite automaton $\mathcal{F}$ (Theorem 22).

The reminder of this paper is organized as follows. Next, in Section 2, we define some basic concepts and state well-known results concerning finite automata and ordered decision diagrams. Subsequently, in Section 3, we formally define the notion of second-order finite automata and state our main results (Theorem 9 and Theorem 10). In Section 4, we state several closure properties for second-order finite automata. In Section 5, we discuss several algorithmic applications of our main results. In Section 6 we prove Theorem 9. Finally, in Section 7 we draw some concluding remarks and establish connections with related work.

2 Preliminaries

2.1 Basics

We denote by $\mathbb{N} \doteq \{0, 1, \ldots\}$ the set of natural numbers (including zero), and by $\mathbb{N}_+ \doteq \mathbb{N} \setminus \{0\}$ the set of positive natural numbers. For each $c \in \mathbb{N}_+$, we let $[c] \doteq \{1, 2, \ldots, c\}$ and $[\![c]\!] \doteq \{0, 1, \ldots, c-1\}$. For each finite set X, we let $\mathcal{P}(X) \doteq \{X' : X' \subseteq X\}$ denote the *power set* of X. For each two sets X and Y, each function $f : X \to Y$ and each subset $X' \subseteq X$, we let $f|_{X'}$ denote the *restriction* of f to X', i.e. the function $f|_{X'} : X' \to Y$ such that $f|_{X'}(x) = f(x)$ for each $x \in X'$.

2.1.1 Alphabets and Strings

An *alphabet* is any finite, non-empty set Σ. A *string* over an alphabet Σ is any finite sequence of symbols from Σ. The *empty string*, denoted by λ, is the unique string of length zero. We denote by Σ^* the set of all strings over Σ, including the empty string λ, and by $\Sigma^+ \doteq \Sigma^* \setminus \{\lambda\}$ the set of all non-empty strings over Σ. A *language* over Σ is any subset L of Σ^*. In particular, for each $k \in \mathbb{N}$, we let Σ^k be the language of all strings of length k over Σ. We say that an alphabet Σ is ordered if it is endowed with

a total order $<_\Sigma: \Sigma \times \Sigma$. Such an order $<_\Sigma$ is extended naturally to a lexicographical order $\prec_\Sigma \subseteq \Sigma^* \times \Sigma^*$ on the set Σ^*. Unless stated otherwise, we assume that each alphabet considered in this paper is endowed with a fixed total order.

2.1.2 Finite Automata

A *finite automaton* (FA) over an alphabet Σ is a tuple $\mathcal{F} = (\Sigma, Q, I, F, T)$, where Q is a finite set of *states*, $I \subseteq Q$ is a set of *initial states*, $F \subseteq Q$ is a set of *final states* and $T \subseteq Q \times \Sigma \times Q$ is a set of *transitions*. The *size* of $\mathcal{F}$ is defined as $|\mathcal{F}| \doteq |Q| + |T| \cdot \log |\Sigma|$. We denote the number of states of $\mathcal{F}$ by $\mathsf{nSt}(\mathcal{F}) \doteq |Q|$, and the number of transitions of $\mathcal{F}$ by $\mathsf{nTr}(\mathcal{F}) \doteq |T|$.

Let $s \in \Sigma^*$, and $q, q' \in Q$. We say that s reaches q' from q if either $s = \lambda$ and $q = q'$, or if $s = \sigma_1 \ldots \sigma_k$ for some $k \in \mathbb{N}_+$ and there is a sequence

$$\langle (q_0, \sigma_1, q_1), (q_1, \sigma_2, q_2), \ldots, (q_{k-1}, \sigma_k, q_k) \rangle,$$

of transitions such that $q_0 = q$, $q_k = q'$ and $(q_i, \sigma_{i+1}, q_{i+1}) \in T$ for each $i \in [k-1]$. We say that $\mathcal{F}$ *accepts* s if there exist states $q \in I$ and $q' \in F$ such that s reaches q' from q. The *language* of $\mathcal{F}$ is defined as the set

$$\mathcal{L}(\mathcal{F}) \doteq \{s \in \Sigma^* : s \text{ is accepted by } \mathcal{F}\}$$

of all finite strings over Σ accepted by $\mathcal{F}$. For $\alpha \in \mathbb{N}$, we say that a language $L \subseteq \Sigma^*$ is *α-regular* if there exists a finite automaton with at most α states such that $\mathcal{L}(\mathcal{F}) = L$.

We say that $\mathcal{F}$ is *deterministic* if $\mathcal{F}$ contains exactly one initial state, i.e. $|I| = 1$, and for each $q \in Q$ and each $\sigma \in \Sigma$, there exists at most one state $q' \in Q$ such that (q, σ, q') is a transition in T. We say that $\mathcal{F}$ is *complete* if it has at least one initial state, and for each $q \in Q$ and each $\sigma \in \Sigma$, there exists at least one state $q' \in Q$ such that (q, σ, q') is a transition in T. We say that $\mathcal{F}$ is *reachable* if for each state $q \in Q$, there is a sequence of transitions from some initial state of $\mathcal{F}$ to q. If $\mathcal{F}$ is a reachable finite automaton, then for each state $q \in Q$, we let $\mathsf{lex}(q)$ denote the lexicographically first string that reaches q from some initial state, according to the order $\prec_\Sigma$. We say that $\mathcal{F}$ is *normalized* if $Q = [\![n]\!]$ for some $n \in \mathbb{N}_+$, and $q < q'$ if and only if $\mathsf{lex}(q) \prec_\Sigma \mathsf{lex}(q')$ for each $q, q' \in Q$.

In what follows, we may write $Q(\mathcal{F})$, $T(\mathcal{F})$, $I(\mathcal{F})$ and $F(\mathcal{F})$ to refer to the sets Q, T, I and F, respectively.

The following theorem, stating the existence of canonical forms for finite automata, is one of the most fundamental results in automata theory.

Theorem 1 *For each finite automaton $\mathcal{F}$, there exists a unique finite automaton $\mathcal{C}(\mathcal{F})$ with minimum number of states such that $\mathcal{C}(\mathcal{F})$ is deterministic, complete, normalized, and satisfies $\mathcal{L}(\mathcal{C}(\mathcal{F})) = \mathcal{L}(\mathcal{F})$.*

We note that given a (possibly non-deterministic) finite automaton $\mathcal{F}$, the canonical form $\mathcal{C}(\mathcal{F})$ of $\mathcal{F}$ can be obtained by the following process. First, one applies Rabin's power-set construction to $\mathcal{F}$ in order to obtain a *deterministic, complete* finite automaton $\mathcal{F}'$ that accepts the same language as $\mathcal{F}$. Subsequently, by using

Hopcroft's algorithm [25] for instance, one minimizes $\mathcal{F}'$ in order to obtain a deterministic finite automaton $\mathcal{F}''$ that accepts the same language as $\mathcal{F}$ and has the minimum number of states. At this point, the finite automaton $\mathcal{F}''$ is unique *up to renaming of states*. Thus, as a last step, one obtains the canonical form $\mathcal{C}(\mathcal{F})$ by renaming the states of $\mathcal{F}''$ in such a way that the normalization property is satisfied. Note that the automaton $\mathcal{C}(\mathcal{F})$ is finally syntactically unique. In particular, for each two finite automata $\mathcal{F}$ and $\mathcal{F}'$, $\mathcal{L}(\mathcal{F}) = \mathcal{L}(\mathcal{F}')$ if and only if $\mathcal{C}(\mathcal{F}) = \mathcal{C}(\mathcal{F}')$.

2.2 Ordered Decision Diagrams

2.2.1 Layers

Let Σ be an alphabet and $w \in \mathbb{N}_+$. A (Σ, w)-*layer* is a tuple $B \doteq (\ell, r, T, I, F, \iota, \phi)$, where $\ell \subseteq [\![w]\!]$ is a set of *left states*, $r \subseteq [\![w]\!]$ is a set of *right states*, $T \subseteq \ell \times \Sigma \times r$ is a set of *transitions*, $I \subseteq \ell$ is a set of *initial states*, $F \subseteq r$ is a set of *final states* and $\iota, \phi \in \{0, 1\}$ are Boolean flags satisfying the two following conditions:

1. if $\iota = 0$, then $I = \emptyset$;
2. if $\phi = 0$, then $F = \emptyset$.

In what follows, we may write $\ell(B), r(B), T(B), I(B), F(B), \iota(B)$ and $\phi(B)$ to refer to the sets ℓ, r, T, I and F and to the Boolean flags ι and ϕ, respectively.

We let $\mathcal{B}(\Sigma, w)$ denote the set of all (Σ, w)-layers. Note that $\mathcal{B}(\Sigma, w)$ is non-empty and has at most $2^{\mathcal{O}(|\Sigma| \cdot w^2)}$ elements. Therefore, $\mathcal{B}(\Sigma, w)$ may be regarded as an alphabet.

2.2.2 Ordered Decision Diagrams

Let Σ be an alphabet and $w, k \in \mathbb{N}_+$. A (Σ, w)-*ordered decision diagram* (or simply, (Σ, w)-*ODD*) of *length* k is a string $D \doteq B_1 \cdots B_k \in \mathcal{B}(\Sigma, w)^k$ of length k over the alphabet $\mathcal{B}(\Sigma, w)$ satisfying the following conditions:

1. for each $i \in [k-1]$, $\ell(B_{i+1}) = r(B_i)$;
2. $\iota(B_1) = 1$ and, for each $i \in \{2, \ldots, k\}$, $\iota(B_i) = 0$;
3. $\phi(B_k) = 1$ and, for each $i \in [k-1]$, $\phi(B_i) = 0$.

Intuitively, Condition 1 expresses that for each $i \in [k-1]$, the set of right states of B_i can be identified with the set of left states of B_{i+1}. Condition 2 guarantees that only the first layer of an ODD is allowed to have initial states. Analogously, Condition 3 guarantees that only the last layer of an ODD is allowed to have final states.

Let $D = B_1 \cdots B_k$ be a (Σ, w)-ODD of length k, for some $k \in \mathbb{N}_+$. We let $\mathsf{len}(D) \doteq k$ denote the length of D, $\mathsf{nSt}(D) \doteq |\ell(B_1)| + \sum_{i \in [k]} |r(B_i)|$ denote the *number of states* of D, $\mathsf{nTr}(D) \doteq |T(B_1)| + \sum_{i \in [k]} |T(B_i)|$ denote the *number of transitions* of D,

$$\mathsf{w}(D) \doteq \max\{|\ell(B_1)|, \ldots, |\ell(B_k)|, |r(B_k)|\}$$

denote the *width* of D. We remark that $\mathsf{w}(D) \leq w$.

For each subset $\mathcal{S} \subseteq \mathcal{B}(\Sigma, w)$ and each positive integer $k \in \mathbb{N}_+$, we denote by $\mathcal{S}^{\circ k}$ the set of all (Σ, w)-ODDs of length k whose layers belong to the set $\mathcal{S}$. Additionally, for each subset $\mathcal{S} \subseteq \mathcal{B}(\Sigma, w)$, we denote by $\mathcal{S}^{\circledast} \doteq \bigcup_{k \in \mathbb{N}_+} \mathcal{S}^{\circ k}$ the set of all (Σ, w)-ODDs whose layers belong to the set $\mathcal{S}$. In particular, we denote by $\mathcal{B}(\Sigma, w)^{\circ k}$ the set of all (Σ, w)-ODDs of length k, and we denote by $\mathcal{B}(\Sigma, w)^{\circledast}$ the set of all (Σ, w)-ODDs.

2.2.3 Length Typed Subsets of Σ^k

Let Σ be an alphabet and $k \in \mathbb{N}_+$. In this work, it is convenient to assume that subsets of Σ^k are typed with their length. This can be achieved by viewing each subset $L \subseteq \Sigma^k$ as a pair of the form (k, L). We let $\mathcal{P}_k(\Sigma) = \{(k, L) : L \subseteq \Sigma^k\}$ be the set of all length typed subsets of Σ^k. Given length typed sets (k, L_1) and (k, L_2), we define $(k, L_1) \cup (k, L_2) \doteq (k, L_1 \cup L_2)$, $(k, L_1) \cap (k, L_2) \doteq (k, L_1 \cap L_2)$, $(k, L_1) \setminus (k, L_2) \doteq (k, L_1 \setminus L_2)$, $(k, L_1) \otimes (k, L_2) \doteq (k, L_1 \otimes L_2)$, and for maps $g : \Sigma \to \Sigma'$ and $h : \Sigma' \to \Sigma$, we let $g(k, L) \doteq (k, g(L))$ and $h^{-1}(k, L) \doteq (k, h^{-1}(L))$.

2.2.4 Language Accepted by an ODD

Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, $D = B_1 \cdots B_k$ be an ODD in $\mathcal{B}(\Sigma, w)^{\circ k}$ and $s = \sigma_1 \cdots \sigma_k$ be a string in Σ^k. A *valid sequence* for s in D is a sequence of transitions

$$\langle (\mathfrak{p}_1, \sigma_1, \mathfrak{q}_1), \ldots, (\mathfrak{p}_k, \sigma_k, \mathfrak{q}_k) \rangle$$

such that $\mathfrak{p}_{i+1} = \mathfrak{q}_i$ for each $i \in [k-1]$, and $(\mathfrak{p}_i, \sigma_i, \mathfrak{q}_i) \in T(B_i)$ for each $i \in [k]$. Such a valid sequence is called *accepting* for s if, additionally, $\mathfrak{p}_1$ is an initial state in $I(B_1)$ and $\mathfrak{q}_k$ is a final state in $F(B_k)$. We say that D accepts s if there exists an accepting sequence for s in D. The *language* of D is defined as the (length-typed) set

$$\mathcal{L}(D) \doteq (k, \{s \in \Sigma^k : s \text{ is accepted by } D\})$$

of all strings accepted by D. Note that every string accepted by D has length k.

In Fig. 1, we depict an ODD $D \in \mathcal{B}(\{0, 1\}, 2)^{\circ 5}$ whose language is the length-typed set $\mathcal{L}(D) = (5, \{s = \sigma_1 \cdots \sigma_5 \in \{0, 1\}^5 : \sigma_1 + \cdots + \sigma_5 \equiv 0 \pmod{2}\})$ of all binary strings of length 5 with an even number of occurrences of the symbol '1'. For instance,

$$\langle (0, 0, 0), (0, 1, 1), (1, 0, 0), (0, 1, 0), (0, 0, 0) \rangle$$

is an accepting sequence in D for the string 01010, which has two occurrences of the symbol '1'.

2.2.5 Deterministic and Complete ODDs

Let Σ be an alphabet and $w \in \mathbb{N}_+$. A (Σ, w)-layer B is called *deterministic* if the following conditions are satisfied:

1. if $\iota(B) = 1$, then $I(B) = \ell(B)$ and $|\ell(B)| = 1$;
2. for each $\mathfrak{p} \in \ell(B)$ and each $\sigma \in \Sigma$, there exists at most one right state $\mathfrak{q} \in r(B)$ such that $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B)$.
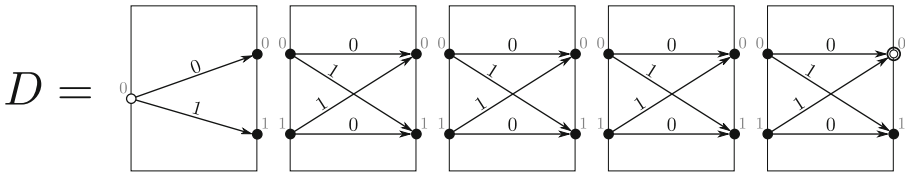
Fig. 1 Example of ODD $D \in \mathcal{B}(\{0, 1\}, 2)^{\circ 5}$ whose language consists of all binary strings of length 5 with an even number of occurrences of the symbol '1'

A (Σ, w)-layer B is called *complete* if the following conditions are satisfied:

1. if $\iota(B) = 1$, then $I(B) \neq \emptyset$;
2. for each $\mathfrak{p} \in \ell(B)$ and each $\sigma \in \Sigma$, there exists at least one right state $\mathfrak{q} \in r(B)$ such that $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B)$.

We let $\widehat{\mathcal{B}}(\Sigma, w)$ be the subset of $\mathcal{B}(\Sigma, w)$ comprising all deterministic, complete (Σ, w)-layers.

Observation 2 Let Σ be an alphabet, and $w \in \mathbb{N}_+$.

1. The alphabet $\widehat{\mathcal{B}}(\Sigma, w)$ has $2^{O(|\Sigma|w \log w)}$ layers.
2. The alphabet $\mathcal{B}(\Sigma, w)$ has $2^{O(|\Sigma|w^2)}$ layers.

Proof 1. Let Σ be an alphabet, and $x, y \in \{0, 1, \ldots, w\}$. We note that there are at most

$$d(\Sigma, x, y) = \binom{w}{x}\binom{w}{y}(x + 1)(1 + 2^y)y^{|\Sigma|x} = w^{O(|\Sigma|w)} = 2^{O(|\Sigma|w \log w)}$$

deterministic complete layers with x left states, y right states and transitions labeled by symbols in Σ. Indeed, there are $\binom{w}{x}$ ways of choosing x left states, out of the set $\{1, \ldots, w\}$, $\binom{w}{y}$ ways of choosing y right states out of the set $\{1, \ldots, w\}$, $(w + 1)$ ways of choosing the initial set of states $I(B)$ together with the initial flag $\iota(B)$ (because $I(B) = \emptyset$ if $\iota(B) = 0$ and $|I(B)| = 1$ if $\iota(B) = 1$, due to determinism), $1 + 2^y$ ways of choosing the subset of final states $F(B)$ together with the final flag $\phi(B)$ (because $F(B) = 0$ if $\phi(B) = 0$ and $F(B)$ is an arbitrary subset of $r(B)$ if $\phi(B) = 1$), and $y^{|\Sigma|x}$ ways of choosing the transition relation $T(B)$ (because there are x left states, and for each such state q and each symbol $a \in \Sigma$ there are y ways of choosing the unique transition with label a leaving q). Therefore, we have that $|\widehat{\mathcal{B}}(\Sigma, w)| \leq \sum_{x,y=0}^{w} d(\Sigma, x, y) = (w + 1)^2 \cdot 2^{O(|\Sigma|w \log w)} = 2^{O(|\Sigma|w \log w)}$.

2. By a similar analysis we can conclude that for each alphabet Σ, and each $x, y \in \{0, 1, \ldots, w\}$ there are at most at most

$$n(\Sigma, x, y) = \binom{w}{x}\binom{w}{y}(1 + 2^x)(1 + 2^y)2^{|\Sigma|xy} = 2^{O(|\Sigma|w^2)}$$

(possibly nondeterministic) layers with x left state, y right states, and transitions labeled with symbols from Σ. The essential differences are that in the

nondeterministic case, there are $(1 + 2^x)$ ways of choosing the set of initial states together with the initial flag (because $I(B) = \emptyset$ if $\iota(B) = 0$, and $I(B)$ may be an arbitrary subset of $\ell(B)$ if $\iota(B) = 1$), and that there are $2^{|\Sigma|xy}$ ways of choosing the transition relation $T(B)$ (because there are x left states, and for each such a state q and each symbol $a \in \Sigma$ there are 2^y ways of choosing the set of transitions with label a leaving q). Therefore, we have that $|\mathcal{B}(\Sigma, w)| \leq \sum_{x,y=0}^{w} n(\Sigma, x, y) = (w+1)^2 \cdot 2^{O(|\Sigma|w^2)} = 2^{O(|\Sigma|w^2)}.$　　　□

Let $k \in \mathbb{N}_+$ and $D = B_1 \cdots B_k \in \mathcal{B}(\Sigma, w)^{\circ k}$. We say that D is *deterministic* (*complete*, resp.) if for each $i \in [k]$, B_i is a deterministic (complete, resp.) layer. We remark that if D is deterministic, then there exists at most one valid sequence in D for each string in Σ^k. On the other hand, if D is complete, then there exists at least one valid sequence in D for each string in Σ^k.

For each $k \in \mathbb{N}_+$, we denote by $\widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$ the subset of $\mathcal{B}(\Sigma, w)^{\circ k}$ comprising all deterministic, complete (Σ, w)-ODDs of length k. We denote by $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ the subset of $\mathcal{B}(\Sigma, w)^{\circledast}$ comprising all deterministic, complete (Σ, w)-ODDs.

2.2.6 Isomorphism of ODDs

Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, and let $D = B_1 \cdots B_k$ and $D' = B'_1 \cdots B'_k$ be two ODDs in $\mathcal{B}(\Sigma, w)^{\circ k}$. An *isomorphism from D to D'* is a sequence $\overline{\pi} \doteq \langle \pi_0, \ldots, \pi_k \rangle$ of functions that satisfy the following conditions:

1. $\pi_0 \colon \ell(B_0) \to \ell(B'_0)$ is a bijection from $\ell(B_0)$ to $\ell(B'_0)$;
2. $\pi_0|_{I(B_0)}$ is a bijection from $I(B_0)$ to $I(B'_0)$;
3. for each $i \in [k]$, $\pi_i \colon r(B_i) \to r(B'_i)$ is a bijection from $r(B_i)$ to $r(B'_i)$;
4. $\pi_k|_{F(B_k)}$ is a bijection from $F(B_k)$ to $F(B'_k)$;
5. for each $i \in [k]$, each left state $\mathfrak{p} \in \ell(B_i)$, each symbol $\sigma \in \Sigma$ and each right state $\mathfrak{q} \in r(B_i)$, $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B_i)$ if and only if $(\pi_{i-1}(\mathfrak{p}), \sigma, \pi_i(\mathfrak{q})) \in T(B'_i)$.

We remark that if $\overline{\pi} = \langle \pi_0, \ldots, \pi_k \rangle$ is an isomorphism from D to D', then the sequence $\overline{\pi}^{-1} \doteq \langle \pi_0^{-1}, \ldots, \pi_k^{-1} \rangle$ is an isomorphism from D' to D, where π_i^{-1} denotes the inverse function of π_i for each $i \in [\![k+1]\!]$. We say that D and D' are *isomorphic* if there exists an isomorphism $\overline{\pi}$ between D and D'. The following proposition is immediate.

Proposition 3 *Let Σ be an alphabet, $w \in \mathbb{N}_+$, and let D and D' be two (Σ, w)-ODDs. If D and D' are isomorphic, then $\mathcal{L}(D) = \mathcal{L}(D')$.*

2.2.7 Normalized ODDs.

Let Σ be an alphabet, $w \in \mathbb{N}_+$, and let B be a (Σ, w)-layer. We say that B is *reachable* if for each right state $\mathfrak{q} \in r(B)$, there exist a symbol $\sigma \in \Sigma$ and a left state $\mathfrak{p} \in \ell(B)$ such that $(\mathfrak{p}\,\sigma, \mathfrak{q})$ is a transition in $T(B)$. If B is reachable, then we let $\chi_B \colon r(B) \to \ell(B) \times \Sigma$ be the function such that for each right state $\mathfrak{q} \in r(B)$,

$$\chi_B(\mathfrak{q}) \doteq \min\{(\mathfrak{p}, \sigma) \colon (\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B)\},$$

where the minimum is taken lexicographically, i.e., for each two left states $\mathfrak{p}, \mathfrak{p}' \in \ell(B)$ and each two symbols $\sigma, \tau \in \Sigma$, we have that $(\mathfrak{p}, \sigma) < (\mathfrak{p}', \tau)$ if and only if $\mathfrak{p} < \mathfrak{p}'$, or $\mathfrak{p} = \mathfrak{p}'$ and $\sigma <_\Sigma \tau$. (Recall we are assuming that the alphabet Σ is endowed with a fixed total order $<_\Sigma \subseteq \Sigma \times \Sigma$.) We say that B is *well-ordered* if it is a reachable, deterministic layer such that for each two right states $\mathfrak{q}, \mathfrak{q}' \in r(B)$, we have that $\mathfrak{q} < \mathfrak{q}'$ if and only if $\chi_B(\mathfrak{q}) < \chi_B(\mathfrak{q}')$. We say that B is *contiguous* if $\ell(B) = [\![w_1]\!]$ and $r(B) = [\![w_2]\!]$ for some $w_1, w_2 \in [w]$. Then, we say that B is *normalized* if it is both well-ordered and contiguous.

Let $k \in \mathbb{N}_+$ and $D = B_1 \cdots B_k$ be an ODD in $\mathcal{B}(\Sigma, w)^{\circ k}$. We say that D is *reachable/well-ordered/contiguous/normalized* if for each $i \in [k]$, the layer B_i is reachable/well-ordered/contiguous/normalized. Note that D is normalized if and only if it is both well-ordered and contiguous.

2.2.8 Minimized ODDs

Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, and let $D = B_1 \cdots B_k$ be a deterministic, complete ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$. We say that D is *minimized* if for each $w' \in \mathbb{N}_+$ and each $D' = B'_1 \cdots B'_k \in \widehat{\mathcal{B}}(\Sigma, w')^{\circ k}$, with $\mathcal{L}(D) = \mathcal{L}(D')$, we have that $\mathsf{nSt}(D) \leq \mathsf{nSt}(D')$. In other words, D is minimized if no deterministic, complete ODD with the same language as D has less states than D. The following theorem is the analog of Theorem 1 in the realm of the theory of ordered decision diagrams.

Theorem 4 *Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, and let D be an ODD in $\mathcal{B}(\Sigma, w)^{\circ k}$. There exists a unique minimized ODD $\mathcal{C}(D) \in \widehat{\mathcal{B}}(\Sigma, 2^w)^{\circ k}$ such that $\mathcal{C}(D)$ is deterministic, complete, normalized and satisfies $\mathcal{L}(\mathcal{C}(D)) = \mathcal{L}(D)$. Additionally, if $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$ then $\mathcal{C}(D) \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$.*

We call the ODD $\mathcal{C}(D)$ of Theorem 4 the *canonical form* of D. We note that $\mathcal{C}(D)$ is unique not only up to isomorphism, but also unique up to equality. In particular, this implies that for each alphabet Σ, each $w, w', k \in \mathbb{N}$, and each two ODDs $D \in \mathcal{B}(\Sigma, w)^{\circ k}$ and $D' \in \mathcal{B}(\Sigma, w')^{\circ k}$ with $\mathcal{L}(D) = \mathcal{L}(D')$, we have that $\mathcal{C}(D) = \mathcal{C}(D')$. The construction of $\mathcal{C}(D)$ follows a similar process to the construction of canonical forms of OBDDs with a fixed variable, or equivalently, read-once oblivious branching programs [37].

2.3 Regular Transductions

Let Σ_1 and Σ_2 be two alphabets. In this work, a (Σ_1, Σ_2)-*transduction* is a binary relation $\mathfrak{t} \subseteq \Sigma_1^+ \times \Sigma_2^+$ where $|s| = |u|$ for each $(s, u) \in \mathfrak{t}$. We let

$$\mathsf{Im}(\mathfrak{t}) \doteq \{u \in \Sigma_2^+ : \exists\, s \in \Sigma_1^+, (s, u) \in \mathfrak{t}\}$$

be the *image* of $\mathfrak{t}$, and we let

$$\mathsf{Dom}(\mathfrak{t}) \doteq \{s \in \Sigma_1^+ : \exists\, u \in \Sigma_2^+, (s, u) \in \mathfrak{t}\}$$

be the *domain* of $\mathfrak{t}$. We say that a (Σ_1, Σ_2)-transduction $\mathfrak{t}$ is *functional* if, for each string $s \in \Sigma_1^+$, there exists at most one string $u \in \Sigma_2^+$ such that $(s, u) \in \mathfrak{t}$.

Let Σ_1, Σ_2 and Σ_3 be three (not-necessarily distinct) alphabets. If $\mathfrak{t}$ is a (Σ_1, Σ_2)-transduction and $\mathfrak{t}'$ is a (Σ_2, Σ_3)-transduction, then the *composition* of $\mathfrak{t}$ with $\mathfrak{t}'$ is defined as the (Σ_1, Σ_3)-transduction

$$\mathfrak{t} \circ \mathfrak{t}' \doteq \{(s, v) \in \Sigma_1^+ \times \Sigma_3^+ : \exists u \in \Sigma_2^+, (s, u) \in \mathfrak{t} \text{ and } (u, v) \in \mathfrak{t}'\}.$$

For each language $L \subseteq \Sigma_1^+$, we let

$$\mathfrak{d}(L) \doteq \{(s, s) : s \in L\}$$

be the (Σ_1, Σ_1)-*transduction derived from* L. Then, for each language $L \subseteq \Sigma_1^+$ and each (Σ_1, Σ_2)-transduction $\mathfrak{t}$, we let

$$\mathfrak{t}(L) \doteq \mathsf{Im}(\mathfrak{d}(L) \circ \mathfrak{t}) = \{u \in \Sigma_2^+ : \exists s \in L, (s, u) \in \mathfrak{t}\}$$

be the image of L under $\mathfrak{t}$.

2.3.1 Tensor Product

Let $\Sigma_1, \ldots, \Sigma_{\mathfrak{a}}$ be $\mathfrak{a}$ alphabets and $k \in \mathbb{N}_+$. For each $i \in [\mathfrak{a}]$, let $s_i = \sigma_{i,1} \cdots \sigma_{i,k}$ be a string of length k over the alphabet Σ_i. The *tensor product* of $s_1, \ldots, s_{\mathfrak{a}}$ is defined as the string

$$s_1 \otimes \cdots \otimes s_{\mathfrak{a}} \doteq (\sigma_{1,1}, \ldots, \sigma_{\mathfrak{a},1}) \cdots (\sigma_{k,1}, \ldots, \sigma_{k,\mathfrak{a}})$$

of length k over the alphabet $\Sigma_1 \times \cdots \times \Sigma_{\mathfrak{a}}$. For each $i \in [\mathfrak{a}]$, let $L_i \subseteq \Sigma_i^+$ be a language over Σ_i. The *tensor product* of $L_1, \ldots, L_{\mathfrak{a}}$ is defined as the language

$$L_1 \otimes \cdots \otimes L_{\mathfrak{a}} \doteq \{s_1 \otimes \cdots \otimes s_{\mathfrak{a}} : |s_1| = \cdots = |s_{\mathfrak{a}}|, \ s_i \in L_i \text{ for each } i \in [\mathfrak{a}]\}.$$

2.3.2 Regular transductions

For $\alpha \in \mathbb{N}_+$, we say that a (Σ_1, Σ_2)-transduction $\mathfrak{t}$ is α-*regular* if the language

$$\mathcal{L}(\mathfrak{t}) \doteq \{s \otimes u : (s, u) \in \mathfrak{t}\} \subseteq (\Sigma_1 \times \Sigma_2)^+$$

is α-regular. The following proposition states some straightforward quantitative properties of regular transductions.

Proposition 5 *Let Σ_1, Σ_2 and Σ_3 be three alphabets, $\mathfrak{t}$ be an α-regular (Σ_1, Σ_2)-transduction, $\mathfrak{t}'$ be a β-regular (Σ_2, Σ_3)-transduction, and let $L \subseteq \Sigma_1^+$ be a γ-regular language, for some $\alpha, \beta, \gamma \in \mathbb{N}_+$. The following statements hold.*

1. *The languages $\mathsf{Im}(\mathfrak{t})$ and $\mathsf{Dom}(\mathfrak{t})$ are α-regular.*
2. *The composition $\mathfrak{t} \circ \mathfrak{t}'$ is $(\alpha \cdot \beta)$-regular.*
3. *The transduction $\mathfrak{d}(L)$ is γ-regular.*
4. *The language $\mathfrak{t}(L)$ is $(\gamma \cdot \alpha)$-regular.*

Proof Let $\mathcal{F}_{\mathfrak{t}}$ be a finite automaton with α states and language $\mathcal{L}(\mathcal{F}_{\mathfrak{t}}) = \mathcal{L}(\mathfrak{t})$, $\mathcal{F}_{\mathfrak{t}'}$ be a finite automaton with β states and language $\mathcal{L}(\mathcal{F}_{\mathfrak{t}'}) = \mathcal{L}(\mathfrak{t}')$, and let $\mathcal{F}_L$ be a finite

automaton with γ states and language $\mathcal{L}(\mathcal{F}_L) = L$. Note that, such automata $\mathcal{F}_t$, $\mathcal{F}_{t'}$ and $\mathcal{F}_L$ exist, since by hypothesis t is α-regular, t' is β-regular and L is γ-regular, respectively.

1. We let $\mathcal{F}_{\mathsf{Im}(t)}$ and $\mathcal{F}_{\mathsf{Dom}(t)}$ be the finite automata over the alphabets Σ_2 and Σ_1, respectively, defined exactly as $\mathcal{F}_t$ except for their transition sets, which is defined as follows:

$$T(\mathcal{F}_{\mathsf{Im}(t)}) = \{(q, \tau, q') \colon \exists \sigma \in \Sigma_1, (q, (\sigma, \tau), q') \in T(\mathcal{F}_t)\}, \text{ and}$$

$$T(\mathcal{F}_{\mathsf{Dom}(t)}) = \{(q, \sigma, q') \colon \exists \tau \in \Sigma_2, (q, (\sigma, \tau), q') \in T(\mathcal{F}_t)\}.$$

Clearly, $\mathcal{F}_{\mathsf{Im}(t)}$ and $\mathcal{F}_{\mathsf{Dom}(t)}$ have at most α states each. Moreover, $\mathcal{F}_{\mathsf{Im}(t)}$ accepts a string $u \in \Sigma_2^+$ if and only if there exists a string $s \in \Sigma_1^+$ such that $s \otimes u \in \mathcal{L}(t)$. Analogously, one can verify that $\mathcal{F}_{\mathsf{Dom}(t)}$ accepts a string $s \in \Sigma_1^+$ if and only if there exists a string $u \in \Sigma_2^+$ such that $s \otimes u \in \mathcal{L}(t)$. Therefore, the language of $\mathcal{F}_{\mathsf{Im}(t)}$ is $\mathcal{L}(\mathcal{F}_{\mathsf{Im}(t)}) = \mathsf{Im}(t)$, and the language of $\mathcal{F}_{\mathsf{Dom}(t)}$ is $\mathcal{L}(\mathcal{F}_{\mathsf{Dom}(t)}) = \mathsf{Dom}(t)$.

2. We let $\mathcal{F}_{t \circ t'}$ be the finite automaton over the alphabet $\Sigma_1 \times \Sigma_3$, with state set $Q(\mathcal{F}_{t \circ t'}) = Q(\mathcal{F}_t) \times Q(\mathcal{F}_{t'})$, initial state set $I(\mathcal{F}_{t \circ t'}) = I(\mathcal{F}_t) \times I(\mathcal{F}_{t'})$, final state set $F(\mathcal{F}_{t \circ t'}) = F(\mathcal{F}_t) \times F(\mathcal{F}_{t'})$ and transition set

$$T(\mathcal{F}_{t \circ t'}) = \{((p, p'), (\sigma, \tau'), (q, q')) \colon \exists \tau \in \Sigma_2,$$
$$(p, (\sigma, \tau), q) \in T(\mathcal{F}_t), (p', (\tau, \tau'), q') \in T(\mathcal{F}_{t'})\}.$$

We remark $\mathcal{F}_{t \circ t'}$ is a finite automaton with at most $(\alpha \cdot \beta)$ states. Moreover, $\mathcal{F}_{t \circ t'}$ accepts a string $s \otimes v \in (\Sigma_1 \times \Sigma_3)^+$ if and only if there exists $u \in \Sigma_2^+$ such that $s \otimes u \in \mathcal{L}(t)$ and $u \otimes v \in \mathcal{L}(t')$. Therefore, the language of $\mathcal{F}_{t \circ t'}$ is $\mathcal{L}(\mathcal{F}_{t \circ t'}) = \mathcal{L}(t \circ t')$.

3. We let $\mathcal{F}_{\mathfrak{d}(L)}$ be the finite automata over the alphabet $\Sigma_1 \times \Sigma_1$ defined exactly as $\mathcal{F}_L$ except for its transition set, which is defined as follows:

$$T(\mathfrak{d}(L)) = \{(q, (\sigma, \sigma), q') \colon (q, \sigma, q') \in T(\mathcal{F}_L)\}.$$

Clearly, $\mathcal{F}_{\mathfrak{d}(L)}$ has at most γ states. Moreover, $\mathcal{F}_{\mathfrak{d}(L)}$ accepts a string $s \otimes u \in (\Sigma_1 \times \Sigma_1)^+$ if and only if $u = s$ and $s \in \mathcal{L}(t)$. Therefore, the language of $\mathcal{F}_{\mathfrak{d}(L)}$ is $\mathcal{L}(\mathcal{F}_{\mathfrak{d}(L)}) = \mathcal{L}(\mathfrak{d}(L))$.

4. We let $\mathcal{F}_{t(L)}$ be the finite automaton over the alphabet $\Sigma_1 \times \Sigma_3$ such that $\mathcal{F}_{t(L)} = \mathcal{F}_{\mathsf{Im}(\mathfrak{d}(L) \circ t)}$. Based on (2)–(4), $\mathcal{F}_{t(L)}$ is a finite automaton with at most $(\gamma \cdot \alpha)$ states and with language $\mathcal{L}(\mathcal{F}_{t(L)}) = t(L) = \mathsf{Im}(\mathfrak{d}(L) \circ t)$. □

3 Second-Order Finite Automata

In this section, we formally define the main object of study of this work, namely, the notion of second-order finite automata.

Definition 6 (Second-Order Finite Automata) Let Σ be an alphabet and $w \in \mathbb{N}_+$. A finite automaton $\mathcal{F}$ over the alphabet $\mathcal{B}(\Sigma, w)$ is called a (Σ, w)-*second-order finite automaton* (SOFA) if $\mathcal{L}(\mathcal{F}) \subseteq \mathcal{B}(\Sigma, w)^{\circledast}$.

In other words, a (Σ, w)-second-order finite automaton $\mathcal{F}$ is a finite automaton over the alphabet $\mathcal{B}(\Sigma, w)$ such that each string $D = B_1 \cdots B_k$ in $\mathcal{L}(\mathcal{F})$ is a (Σ, w)-ODD, for some $k \in \mathbb{N}_+$.

From now on, for every (Σ, w)-second-order finite automaton $\mathcal{F}$, we may refer to $\mathcal{L}(\mathcal{F})$ as the *first language* of $\mathcal{F}$. Since each string $D \in \mathcal{L}(\mathcal{F})$ is a (Σ, w)-ODD, we can also associate with $\mathcal{F}$ a *second language*, denoted by $\mathcal{L}_2(\mathcal{F})$, which consists of the set of languages accepted by ODDs in $\mathcal{L}(\mathcal{F})$. More precisely, the *second language* of a (Σ, w)-second-order finite automaton $\mathcal{F}$ is defined as the set

$$\mathcal{L}_2(\mathcal{F}) \doteq \{\mathcal{L}(D) \colon D \in \mathcal{L}(\mathcal{F})\}.$$

Note that $\mathcal{L}_2(\mathcal{F})$ is a possibly infinite subset of $\bigcup_{k \in \mathbb{N}_+} \mathcal{P}_k(\Sigma)$. We say that a subset $\mathcal{X} \subseteq \bigcup_{k \in \mathbb{N}_+} \mathcal{P}_k(\Sigma)$ is *regular-decisional* if there is a second-order finite automaton $\mathcal{F}$ such that $\mathcal{X} = \mathcal{L}_2(\mathcal{F})$.

Lemma 7 *Let Σ be an alphabet and $w \in \mathbb{N}_+$. For each $\mathcal{S} \subseteq \mathcal{B}(\Sigma, w)$, there exists a (Σ, w)-second-order finite automaton $\mathcal{F}_\mathcal{S}$ with $(|\mathcal{S}| + 1)$ states such that $\mathcal{L}(\mathcal{F}_\mathcal{S}) = \mathcal{S}^\circledast$.*

Proof Let $\mathcal{F}_\mathcal{S}$ be the (Σ, w)-second-order finite automaton over the alphabet $\mathcal{S}$, with state set $Q(\mathcal{F}_\mathcal{S}) = \{q\} \cup \{q_B \colon B \in \mathcal{S}\}$, initial state set $I(\mathcal{F}_\mathcal{S}) = \{q\}$, final state set $F(\mathcal{F}_\mathcal{S}) = \{q_B \in Q(\mathcal{F}_\mathcal{S}) \colon \phi(B) = 1\}$ and transition set $T(\mathcal{F}_\mathcal{S}) = \{(q, B, q_B) \colon B \in \mathcal{S}, \iota(B) = 1\} \cup \{(q_B, B, q_{B'}) \colon B, B' \in \mathcal{S}, \ell(B') = r(B), \phi(B) = 0, \iota(B') = 0\}$. Since each transition is labeled with some element from $\mathcal{S}$, it should be clear that $\mathcal{L}(\mathcal{F}_\mathcal{S}) \subseteq \mathcal{S}^\circledast$. Now, let $k \in \mathbb{N}$ and $D = B_1 B_2 \ldots B_k$ be an ODD in $\mathcal{S}^\circledast$. Then it should be clear that the sequence of transitions $(q, B_1, q_{B_1})(q_{B_1}, B_2, q_{B_2}) \ldots (q_{B_{k-1}}, B_k, q_{B_k})$ is an accepting sequence in $\mathcal{F}_\mathcal{S}$. This implies that $\mathcal{L}(\mathcal{F}_\mathcal{S}) \supseteq \mathcal{S}^\circledast$. $\quad\square$

The following Corollary is an immediate consequence of Lemma 7 and Observation 2.

Corollary 8 *Let Σ be an alphabet, and $w \in \mathbb{N}_+$.*

1. *The (Σ, w)-SOFA $\mathcal{F}_{\mathcal{B}(\Sigma, w)}$ has $2^{O(|\Sigma| \cdot w^2)}$ states and $\mathcal{L}(\mathcal{F}_{\mathcal{B}(\Sigma, w)}) = \mathcal{B}(\Sigma, w)^\circledast$.*
2. *The (Σ, w)-SOFA $\mathcal{F}_{\widehat{\mathcal{B}}(\Sigma, w)}$ has $2^{O(|\Sigma| \cdot w \log w)}$ states and $\mathcal{L}(\mathcal{F}_{\widehat{\mathcal{B}}(\Sigma, w)}) = \widehat{\mathcal{B}}(\Sigma, w)^\circledast$.*

Example 1 The Even Language

In Fig. 2, we depict a $(\{0, 1\}, 2)$-second-order finite automaton $\mathcal{F}$ whose second language consists of all (length-typed) sets

$$\mathsf{Even}_k = (k, \{s = \sigma_1 \cdots \sigma_k \in \{0, 1\}^k \colon \sigma_1 + \cdots + \sigma_k \equiv 0 \pmod 2)\})$$

of all binary strings of length k with an even number of occurrences of the symbol '1', for each $k \in \mathbb{N}_+$. Note that, for each $k \in \mathbb{N}_+$, $\mathcal{F}$ accepts a unique $(\{0, 1\}, 2)$-ODD of length k, whose language is Even_k. In particular, the language Even_5 is represented by the ODD depicted in Fig. 1, which is accept by $\mathcal{F}$ upon following the sequence of states q_0, q_1, q_1, q_1, q_2.
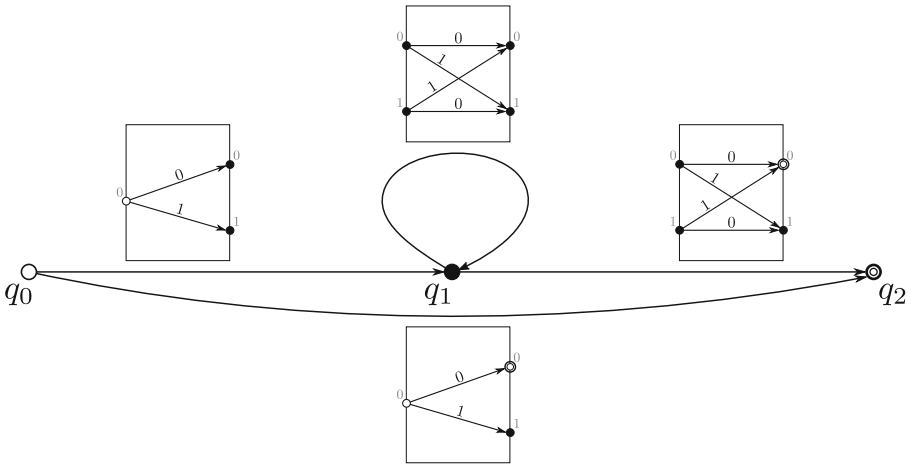
Fig. 2 A $(\{0, 1\}, 2)$-second-order finite automaton $\mathcal{F}$ with second language $\mathcal{L}_2(\mathcal{F}) = \{\mathsf{Even}_k : k \in \mathbb{N}_+\}$

Example 2 The Hypercube Language.

The *hypercube of dimension k* can be defined as the graph H_k with vertex set $V(H) = \{0, 1\}^k$ and edge set

$$E(H_k) = \{(s, s') \colon s, s' \in \{0, 1\}^k, \exists! \, j \in [k] \; s_j \neq s'_j\}.$$

Intuitively, vertices of the hypercube H_k are strings in $\{0, 1\}^k$ and edges are pairs of strings from $\{0, 1\}^k$ that differ in exactly one position. From a formal language standpoint, the edge set of the graph H_k can be encoded by the language

$$\hat{H}_k = (k, \{(s_1, s'_1) \cdots (s_k, s'_k) \colon (s, s') \in E(H_k)\}),$$

Note that, $\hat{H}_k$ is a language over the alphabet $\{0, 1\}^{\times 2}$.

In Fig. 3, we depict a $(\{0, 1\}^{\times 2}, 2)$-second-order finite automaton $\mathcal{H}$ whose second language is $\mathcal{L}_2(\mathcal{H}) = \{\hat{H}_k : k \in \mathbb{N}_+\}$. Similarly to the second-order finite automata illustrated in the previous example, for each $k \in \mathbb{N}_+$, $\mathcal{H}$ accepts a unique $(\{0, 1\}^{\times 2}, 2)$-ODD D_k of length k, whose language is $\hat{H}_k$. In particular, the language $\hat{H}_5$ is represented by the ODD D_5 depicted in Fig. 4, which is accept by $\mathcal{H}$ upon following the sequence of states q_0, q_1, q_1, q_1, q_2.

3.1 Main Results

The main result of this work (Theorem 10) states that second order finite automata can be canonized with respect to their *second* languages. In other words, there is an algorithm that sends each SOFA $\mathcal{F}$ to a SOFA $\mathcal{C}_2(\mathcal{F})$ with $\mathcal{L}_2(\mathcal{F}) = \mathcal{L}_2(\mathcal{C}_2(\mathcal{F}))$ in such a way that $\mathcal{C}_2(\mathcal{F}) = \mathcal{C}_2(\mathcal{F}')$ for any SOFA $\mathcal{F}'$ with the same second language as $\mathcal{F}$. Indeed, $\mathcal{C}_2(\mathcal{F})$ satisfies the following interesting property: $\mathcal{L}(\mathcal{C}_2(\mathcal{F})) = \{\mathcal{C}(D) \;:$
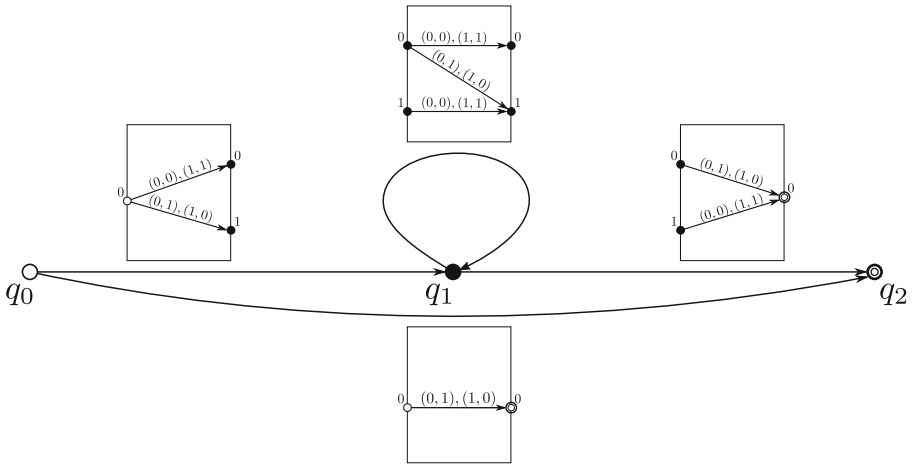
Fig. 3 A $(\{0,1\}^2, 2)$-second-order finite automaton $\mathcal{H}$ with second language $\mathcal{L}_2(\mathcal{H}) = \{\hat{H}_k : k \in \mathbb{N}_+\}$

$D \in \mathcal{L}(\mathcal{F})\}$. Here, for each ODD D, $\mathcal{C}(D)$ denotes the unique deterministic, complete, normalized and minimized ODD with the same language as D, as specified in Theorem 4. In other words, the first language of $\mathcal{C}_2(\mathcal{F})$ is precisely the set of canonical forms of ODDs in the first language of $\mathcal{F}$.

We note that even though $\mathcal{F}$ and $\mathcal{C}_2(\mathcal{F})$ have the same *second* language, i.e. $\mathcal{L}_2(\mathcal{C}_2(\mathcal{F})) = \mathcal{L}_2(\mathcal{F})$, the *first* languages of $\mathcal{F}$ and $\mathcal{C}_2(\mathcal{F})$ may differ. In other words, it may be the case that $\mathcal{L}(\mathcal{C}_2(\mathcal{F})) \neq \mathcal{L}(\mathcal{F})$. As a simple example for this observation, let D be an ODD in $\mathcal{B}(\Sigma, w)^{\circledast}$ for some alphabet Σ and $w \in \mathbb{N}_+$. Let $\mathcal{F}_D$ be the second order finite automaton such that $\mathcal{L}(\mathcal{F}_D) = \{D\}$. Then the language $\mathcal{L}(\mathcal{C}_2(\mathcal{F}_D)) = \{\mathcal{C}(D)\}$ is distinct from $\mathcal{L}(\mathcal{F}_D)$ whenever $\mathcal{C}(D) \neq D$. Therefore, canonization of a finite automaton $\mathcal{F}$ with respect to its second language $\mathcal{L}_2(\mathcal{F})$ cannot be achieved by simply canonizing $\mathcal{F}$ with respect to its first language $\mathcal{L}(\mathcal{F})$ according to Theorem 1.

The proof of our main result is a direct consequence of the following theorem, stating that the traditional minimization and canonization algorithm for ODDs can be simulated in terms of functional regular transductions.
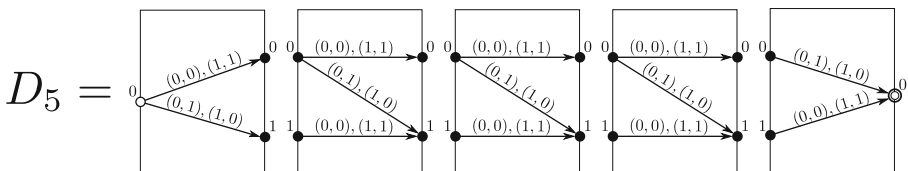

Fig. 4 The $(\{0,1\}^{\times 2}, 2)$-ODD D_5, with language $\mathcal{L}(D_5) = (5, \hat{H}_5)$, accepted by $\mathcal{H}$ upon following the sequence of states q_0, q_1, q_1, q_1, q_2

Theorem 9 (Canonization as Transduction Theorem) *Let Σ be an alphabet and let $w \in \mathbb{N}_+$.*

1. *The functional transduction $\mathfrak{can}[\Sigma, w] = \{(D, \mathcal{C}(D)) \colon D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular.*
2. *The functional transduction $\widehat{\mathfrak{can}}[\Sigma, w] = \{(D, \mathcal{C}(D)) \colon D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}\}$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular.*

Intuitively, the transduction $\mathfrak{can}[\Sigma, w]$ is obtained as a composition of regular transductions that simulate the application of the usual steps in the canonization of a single ODD: determinization, elimination of unreachable states, merging of equivalent states and normalization. The transduction $\widehat{\mathfrak{can}}[\Sigma, w]$ is obtained by a similar process, except that one may skip the application of the determinization transduction, yielding in this way, a more efficient construction. Due to its technical nature, the proof of Theorem 9 will be postponed to Section 6. Next, we show how Theorem 9 can be used to provide a canonization procedure for second order finite automata. Later, in Section 5, we will provide some algorithmic applications of this theorem in the realm of the theory of ODDs of bounded width.

Theorem 10 (Canonical Form of Canonical Forms Theorem) *Let Σ be an alphabet (endowed with a total order $<_\Sigma \subset \Sigma \times \Sigma$), $w \in \mathbb{N}_+$, and let $\mathcal{F}$ be a (Σ, w)-SOFA. One can construct in time $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{O(|\Sigma| \cdot w \cdot 2^w)}}$ a deterministic, complete, normalized $(\Sigma, 2^w)$-SOFA $\mathcal{C}_2(\mathcal{F})$ satisfying the following properties.*

1. $\mathcal{L}(\mathcal{C}_2(\mathcal{F})) = \{\mathcal{C}(D) \colon D \in \mathcal{L}(\mathcal{F})\}$;
2. $\mathcal{L}_2(\mathcal{C}_2(\mathcal{F})) = \mathcal{L}_2(\mathcal{F})$;
3. *For each $w' \in \mathbb{N}_+$ and each (Σ, w')-SOFA $\mathcal{F}'$, if $\mathcal{L}_2(\mathcal{F}') = \mathcal{L}_2(\mathcal{F})$, then $\mathcal{C}_2(\mathcal{F}') = \mathcal{C}_2(\mathcal{F})$.*

Proof Let $\mathcal{F}$ be a (Σ, w)-SOFA and $\mathfrak{can}[\Sigma, w]$ be the $(\mathcal{B}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-transduction specified in Theorem 9. Then, the image of $\mathcal{L}(\mathcal{F})$ under the transduction $\mathfrak{can}[\Sigma, w]$ is the language $\mathfrak{can}[\Sigma, w](\mathcal{L}(\mathcal{F})) = \{\mathcal{C}(D) \colon D \in \mathcal{L}(\mathcal{F})\}$. Here, for each ODD $D \in \mathcal{B}(\Sigma, w)^{\circledast}$, $\mathcal{C}(D) \in \widehat{\mathcal{B}}(\Sigma, 2^w)^{\circledast}$ denotes the unique ODD with minimum number of states such that $\mathcal{C}(D)$ is deterministic, complete, normalized and satisfies $\mathcal{L}(\mathcal{C}(D)) = \mathcal{L}(D)$, as specified in Theorem 4. Since $\mathfrak{can}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular, it follows from Proposition 5.(4), that one can construct a $(\Sigma, 2^w)$-SOFA $\mathcal{F}^\dagger$ with $\mathsf{nSt}(\mathcal{F}) \cdot 2^{O(|\Sigma| \cdot w \cdot 2^w)}$ states such that $\mathcal{L}(\mathcal{F}^\dagger) = \mathfrak{can}[\Sigma, w](\mathcal{L}(\mathcal{F}))$. Now, let $\mathcal{C}(\mathcal{F}^\dagger)$ be the unique finite automaton with minimum number of states such that $\mathcal{C}(\mathcal{F}^\dagger)$ is deterministic, complete, normalized and satisfies $\mathcal{L}(\mathcal{C}(\mathcal{F}^\dagger)) = \mathcal{L}(\mathcal{F}^\dagger)$, as specified in Theorem 1. Then $\mathcal{C}(\mathcal{F}^\dagger)$ can be constructed in time $2^{\mathsf{nSt} \cdot 2^{O(|\Sigma| \cdot w \cdot 2^w)}}$ by applying the standard power-set construction to $\mathcal{F}^\dagger$, followed by a DFA minimization algorithm, such as Hopcroft's algorithm. Now, by defining $\mathcal{C}_2(\mathcal{F})$ as $\mathcal{C}(\mathcal{F}^\dagger)$, we have that $\mathcal{L}(\mathcal{C}_2(\mathcal{F})) = \{\mathcal{C}(D) \colon D \in \mathcal{L}(\mathcal{F})\}$, and therefore, Condition 1 is satisfied. This immediately implies that $\mathcal{L}_2(\mathcal{C}_2(\mathcal{F})) = \mathcal{L}_2(\mathcal{F})$, since each ODD $D \in \mathcal{L}(\mathcal{F})$ has the same language as its canonical form $\mathcal{C}(D)$ in $\mathcal{L}(\mathcal{C}_2(\mathcal{F}))$. Therefore, Condition 2 is also satisfied. Finally, $\mathcal{C}_2(\mathcal{F}) = \mathcal{C}_2(\mathcal{F}')$ for any (Σ, w')-SOFA $\mathcal{F}'$ satisfying

$\mathcal{L}_2(\mathcal{F}') = \mathcal{L}_2(\mathcal{F})$, since for any two ODDs $D \in \mathcal{L}(\mathcal{F})$ and $D' \in \mathcal{L}(\mathcal{F}')$, $\mathcal{L}(D) = \mathcal{L}(D')$ if and only if $\mathcal{C}(D) = \mathcal{C}(D')$. Therefore, Condition 3 is also satisfied. $\qquad\square$

Let $\mathcal{F}$ be a (Σ, w)-SOFA. We call the $(\Sigma, 2^w)$-SOFA $\mathcal{C}_2(\mathcal{F})$ specified in Theorem 10 the *second canonical form* of $\mathcal{F}$. We note that if all ODDs in the language $\mathcal{F}$ are deterministic and complete, then $\mathcal{C}_2(\mathcal{F})$ is actually a (Σ, w)-SOFA, and a faster canonization algorithm can be obtained, since in this case, the transduction $\mathfrak{can}[\Sigma, w]$ used in the proof of Theorem 10 can be replaced by the transduction $\widehat{\mathfrak{can}}[\Sigma, w]$, which is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular.

Observation 11 If $\mathcal{F}$ is a (Σ, w)-SOFA such that $\mathcal{L}(\mathcal{F}) \subseteq \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$, then $\mathcal{C}_2(\mathcal{F})$ is also a (Σ, w)-SOFA and can be constructed in time $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{O(|\Sigma| \cdot w \log w)}}$.

An immediate consequence of Theorem 9 and of Proposition 5.(1) is that for each alphabet Σ, and each $w \in \mathbb{N}_+$, the set of canonical forms of ODDs in $\mathcal{B}(\Sigma, w)^{\circledast}$ is a regular set. The same holds for the set of canonical forms of ODDs in $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$.

Corollary 12 *Let Σ be an alphabet and $w \in \mathbb{N}_+$.*

1. *The language $\mathsf{Im}(\mathfrak{can}[\Sigma, w]) = \{\mathcal{C}(D) : D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular.*
2. *The language $\mathsf{Im}(\widehat{\mathfrak{can}}[\Sigma, w]) = \{\mathcal{C}(D) : D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}\}$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular.*

4 Closure Properties

4.1 Basic Closure Properties

Theorem 10 implies that regular-decisional subsets of $\bigcup_{k \in \mathbb{N}_+} \mathcal{P}_k(\Sigma)$ are closed under Boolean operations such as union, intersection and even a suitable notion of bounded width complementation. These closure properties are formally stated in Theorem 13 below. Let Σ be an alphabet and $w \in \mathbb{N}_+$. We denote by

$$\mathsf{Det}(\Sigma, w) \doteq \{\mathcal{L}(D) : D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}\}$$

the set of all sets of strings accepted by some deterministic, complete (Σ, w)-ODD. Moreover, given a subset $\mathcal{S} \subseteq \bigcup_{k \in \mathbb{N}_+} \mathcal{P}_k(\Sigma)$, we denote by $\overline{\mathcal{S}}^w \doteq \mathsf{Det}(\Sigma, w) \backslash \mathcal{S}$ the *width-w complement* of $\mathcal{S}$.

Theorem 13 *Let Σ be an alphabet, $w \in \mathbb{N}_+$, and let $\mathcal{F}, \mathcal{F}_1$ and $\mathcal{F}_2$ be (Σ, w)-second-order finite automata. The following statements hold.*

1. *There is a $(\Sigma, 2^w)$-second-order finite automaton $\mathsf{intersec}_2(\mathcal{F}_1, \mathcal{F}_2)$ such that*

$$\mathcal{L}_2(\mathsf{intersec}_2(\mathcal{F}_1 \mathcal{F}_2)) = \mathcal{L}_2(\mathcal{F}_1) \cap \mathcal{L}_2(\mathcal{F}_2).$$

2. *There is a* $(\Sigma, 2^w)$*-second-order finite automaton* $\mathsf{union}_2(\mathcal{F}_1, \mathcal{F}_2)$ *such that*
$$\mathcal{L}_2(\mathsf{union}_2(\mathcal{F}_1\mathcal{F}_2)) = \mathcal{L}_2(\mathcal{F}_1) \cup \mathcal{L}_2(\mathcal{F}_2).$$

3. *There is a* $(\Sigma, 2^w)$*-second-order finite automaton* $\mathsf{diff}_2(\mathcal{F}_1, \mathcal{F}_2)$ *such that*
$$\mathcal{L}_2(\mathsf{diff}_2(\mathcal{F}_1\mathcal{F}_2)) = \mathcal{L}_2(\mathcal{F}_1) \backslash \mathcal{L}_2(\mathcal{F}_2).$$

4. *There is a* (Σ, w)*-second-order finite automaton* $\mathcal{F}(\Sigma, w)$ *such that*
$$\mathcal{L}_2(\mathcal{F}(\Sigma, w)) = \mathsf{Det}(\Sigma, w).$$

5. *For each* $w' \in \mathbb{N}_+$, *there is a* $(\Sigma, 2^{\max\{w, w'\}})$*-second-order finite automaton* $\mathsf{compl}_2(\mathcal{F}, w')$ *such that*
$$\mathcal{L}_2(\mathsf{compl}_2(\mathcal{F}, w')) = \overline{\mathcal{L}_2(\mathcal{F})}^{w'}.$$

6. *It is decidable whether* $\mathcal{L}_2(\mathcal{F}_1) \cap \mathcal{L}_2(\mathcal{F}_2) = \emptyset$.

7. *It is decidable whether* $\mathcal{L}_2(\mathcal{F}_1) \subseteq \mathcal{L}_2(\mathcal{F}_2)$.

Proof Let $\mathcal{F}_1' = \mathcal{C}_2(\mathcal{F}_1)$ and $\mathcal{F}_2' = \mathcal{C}_2(\mathcal{F}_2)$ be the second canonical forms specified in Theorem 10 of the automata $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. It is well-known that regular languages are closed under intersection, union and complementation [26]. Consequently, there exist finite automata $\mathsf{intersec}_1(\mathcal{F}_1', \mathcal{F}_2')$, $\mathsf{union}_1(\mathcal{F}_1', \mathcal{F}_2')$ and $\mathsf{compl}_1(\mathcal{F}_2')$ over the alphabet $\widehat{\mathcal{B}}(\Sigma, 2^w)$, such that
$$\mathcal{L}(\mathsf{intersec}_1(\mathcal{F}_1', \mathcal{F}_2')) = \mathcal{L}(\mathcal{F}_1') \cap \mathcal{L}(\mathcal{F}_2'),$$
$\mathcal{L}(\mathsf{union}_1(\mathcal{F}_1', \mathcal{F}_2')) = \mathcal{L}(\mathcal{F}_1') \cup \mathcal{L}(\mathcal{F}_2')$ and $\mathcal{L}(\mathsf{compl}_1(\mathcal{F}_2')) = \widehat{\mathcal{B}}(\Sigma, 2^w)^* \setminus \mathcal{L}(\mathcal{F}_2')$.
Clearly,
$$\mathcal{L}(\mathsf{union}_1(\mathcal{F}_1', \mathcal{F}_2')) = \{\mathcal{C}(D) \colon D \in \mathcal{L}(\mathcal{F}_1)\} \cup \{\mathcal{C}(D') \colon D' \in \mathcal{L}(\mathcal{F}_2)\}.$$

Thus, $\mathsf{union}_2(\mathcal{F}_1, \mathcal{F}_2) = \mathsf{union}_1(\mathcal{F}_1', \mathcal{F}_2')$ is a $(\Sigma, 2^w)$-second-order finite automaton with second language
$$\mathcal{L}_2(\mathsf{union}_2(\mathcal{F}_1, \mathcal{F}_2)) = \mathcal{L}_2(\mathcal{F}_1) \cup \mathcal{L}_2(\mathcal{F}_2).$$

Moreover, owing to the fact that any two ODDs with the same language have the same canonical form, one can verify that
$$\mathcal{L}(\mathsf{intersec}_1(\mathcal{F}_1', \mathcal{F}_2')) = \{\mathcal{C}(D) \colon D \in \mathcal{L}(\mathcal{F}_1), \exists D' \in \mathcal{L}(\mathcal{F}_2), \mathcal{L}(D) = \mathcal{L}(D')\}.$$

Thus, $\mathsf{intersec}_2(\mathcal{F}_1, \mathcal{F}_2) = \mathsf{intersec}_1(\mathcal{F}_1', \mathcal{F}_2')$ is a $(\Sigma, 2^w)$-second-order finite automata with second language
$$\mathcal{L}_2(\mathsf{intersec}_2(\mathcal{F}_1, \mathcal{F}_2)) = \mathcal{L}_2(\mathcal{F}_1) \cap \mathcal{L}_2(\mathcal{F}_2).$$

Furthermore, we have that $\mathsf{diff}_2(\mathcal{F}_1, \mathcal{F}_2) = \mathsf{intersec}_2(\mathcal{F}_1, \mathsf{compl}_1(\mathcal{F}_2'))$ is a $(\Sigma, 2^w)$-second-order finite automata with first language
$$\mathcal{L}(\mathsf{diff}_2(\mathcal{F}_1, \mathcal{F}_2)) = \mathcal{L}(\mathcal{F}_1') \cap \mathcal{L}(\mathsf{compl}_1(\mathcal{F}_2')) = \mathcal{L}(\mathcal{F}_1') \cap \left(\widehat{\mathcal{B}}(\Sigma, 2^w)^* \setminus \mathcal{L}(\mathcal{F}_2')\right)$$
$$= \mathcal{L}(\mathcal{F}_1') \setminus \mathcal{L}(\mathcal{F}_2').$$

Thus, since ODDs with the same language have the same canonical form, the second language of $\mathsf{diff}_2(\mathcal{F}_1, \mathcal{F}_2)$ is
$$\mathcal{L}(\mathsf{diff}_2(\mathcal{F}_1, \mathcal{F}_2)) = \mathcal{L}_2(\mathcal{F}_1) \setminus \mathcal{L}_2(\mathcal{F}_2).$$

Based on Lemma 7, we let $\mathcal{F}(\Sigma, w) = \mathcal{F}_{\mathcal{S}}$ be the (Σ, w)-second-order finite automaton over the alphabet $\mathcal{S}$, where $\mathcal{S} = \widehat{\mathcal{B}}(\Sigma, w)$. One can readily verify that $\mathcal{L}_2(\mathcal{F}(\Sigma, w)) = \mathsf{Det}(\Sigma, w)$.

Now, let $\mathcal{F}' = \mathcal{C}_2(\mathcal{F})$ be the second canonical form specified in Theorem 10 of the automaton $\mathcal{F}$. For each $w' \in \mathbb{N}_+$, we let $\mathsf{compl}_2(\mathcal{F}, w') = \mathsf{diff}_2(\mathcal{F}(\Sigma, w'), \mathcal{F}')$. It is straightforward that $\mathsf{compl}_2(\mathcal{F}, w')$ is a $(\Sigma, 2^{\max\{w, w'\}})$-second-order finite automaton with second language

$$\mathcal{L}_2(\mathsf{compl}_2(\mathcal{F}, w')) = \overline{\mathcal{L}_2(\mathcal{F})}^{w'}.$$

Finally, we note that deciding whether $\mathcal{L}_2(\mathcal{F}_1) \cap \mathcal{L}_2(\mathcal{F}_2) = \emptyset$ is equivalent to deciding whether $\mathcal{L}(\mathcal{F}_1') \cap \mathcal{L}(\mathcal{F}_2') = \emptyset$. Similarly, we have that deciding whether $\mathcal{L}_2(\mathcal{F}_1) \subseteq \mathcal{L}_2(\mathcal{F}_2)$ is equivalent to deciding whether $\mathcal{L}(\mathcal{F}_1') \subseteq \mathcal{L}(\mathcal{F}_2')$, which in turn is equivalent to deciding whether

$$\mathcal{L}(\mathcal{F}_1') \cap (\widehat{\mathcal{B}}(\Sigma, 2^w)^* \setminus \mathcal{L}(\mathcal{F}_2')) = \mathcal{L}(\mathcal{F}_1') \cap \mathsf{compl}_1(\mathcal{F}_2') = \emptyset.$$

Therefore, since disjointness of regular languages is a decidable problem [26], we obtain that the problems of verifying whether $\mathcal{L}_2(\mathcal{F}_1) \cap \mathcal{L}_2(\mathcal{F}_2) = \emptyset$ and verifying whether $\mathcal{L}_2(\mathcal{F}_1) \subseteq \mathcal{L}_2(\mathcal{F}_2)$ are both decidable. □

We note that all binary operations described in Theorem 13 are also defined when $\mathcal{F}_1$ is a (Σ, w_1)-second-order finite automaton and $\mathcal{F}_2$ is a (Σ, w_2)-second-order finite automaton, for distinct positive integers w_1 and w_2. Indeed, it suffices to view both finite automata as $(\Sigma, \max\{w_1, w_2\})$-second-order finite automata. We also note that the SOFAs $\mathsf{intersec}_2(\mathcal{F}_1, \mathcal{F}_2)$, $\mathsf{union}_2(\mathcal{F}_1, \mathcal{F}_2)$ and $\mathsf{diff}_2(\mathcal{F}_1, \mathcal{F}_2)$ are actually (Σ, w)-SOFAs if all ODDs in the languages $\mathcal{L}(\mathcal{F}_1)$ and $\mathcal{L}(\mathcal{F}_2)$ are deterministic and complete, since in this case one can use the more efficient construction given in Observation 11. Finally, it is worth remarking that non-emptiness of intersection of the second languages of SOFAs is not only decidable, but can be achieved in fixed-parameter tractable time (Observation 14).

Observation 14 Let Σ be an alphabet, and $w \in \mathbb{N}_+$ and $\mathcal{F}_1$ and $\mathcal{F}_2$ be (Σ, w)-SOFAs.

1. One can determine whether $\mathcal{L}_2(\mathcal{F}_1) \cap \mathcal{L}_2(\mathcal{F}_2) \neq \emptyset$ in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}_1) \cdot \mathsf{nSt}(\mathcal{F}_2)$.
2. If all ODDs in $\mathcal{L}(\mathcal{F}_1)$ and $\mathcal{L}(\mathcal{F}_2)$ are deterministic and complete, then one one can can determine whether $\mathcal{L}_2(\mathcal{F}_1) \cap \mathcal{L}_2(\mathcal{F}_2) \neq \emptyset$ in time $2^{O(|\Sigma| \cdot w \cdot \log w)} \cdot \mathsf{nSt}(\mathcal{F}_1) \cdot \mathsf{nSt}(\mathcal{F}_2)$.

Proof Since $\mathfrak{can}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular, for each $i \in \{1, 2\}$, one can construct from $\mathcal{F}_i$ a finite automaton $\mathcal{F}_i'$ with $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}_i)$ states such that $\mathcal{L}(\mathcal{F}_i') = \mathfrak{can}[\Sigma, w](\mathcal{L}(\mathcal{F}_i)) = \{\mathcal{C}(D) : D \in \mathcal{L}(\mathcal{F}_i)\}$. Therefore, testing whether $\mathcal{L}_2(\mathcal{F}_1) \cap \mathcal{L}_2(\mathcal{F}_2) \neq \emptyset$ is equivalent to testing whether $\mathcal{L}(\mathcal{F}_1') \cap \mathcal{L}(\mathcal{F}_2') \neq \emptyset$, which can be done in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}_1) \cdot \mathsf{nSt}(\mathcal{F}_2)$. If the languages of the automata $\mathcal{F}_1$ and $\mathcal{F}_2$ only contain deterministic, complete ODDs, then one can apply a similar argument using the transduction $\widehat{\mathfrak{can}}[\Sigma, w]$ instead of $\mathfrak{can}[\Sigma, w]$ to infer that

non-emptiness of intersection for the languages $\mathcal{L}_2(\mathcal{F}_1)$ and $\mathcal{L}_2(\mathcal{F}_2)$ can be tested in time $2^{O(|\Sigma| \cdot w \cdot \log w)} \cdot \mathsf{nSt}(\mathcal{F}_1) \cdot \mathsf{nSt}(\mathcal{F}_2)$. □

4.2 Closure Properties Specific for Language Classes

In this subsection, we show that regular-decisional classes of languages are also closed under operations that are specific to language classes. Let Σ_1 and Σ_2 be alphabets, and $g : \Sigma_1 \to \Sigma_2$ be a map from Σ_1 to Σ_2. Given languages $L \subseteq \Sigma_1^+$ and $L' \subseteq \Sigma_2^+$, we let

$$g(L) = \{u \; : \; \exists w \in L, \; |u| = |w|, \; u_i = g(w_i) \text{ for each } i \in [|u|]\}$$

and

$$g^{-1}(L') = \{u \; : \; \exists w \in L', \; |u| = |w|, \; u_i \in g^{-1}(w_i) \text{ for each } i \in [|u|]\}.$$

The following lemma from [15] states that several operations that are effective for regular languages may be realized on ODDs using maps that act layerwisely. Below, for ODDs $D = B_1 B_2 \ldots B_k$ and $D' = B_1' B_2' \ldots B_k'$, we let $D \otimes D' = (B_1, B_1')(B_2, B_2') \ldots (B_k, B_k')$.

Lemma 15 (Simulation Lemma (see Lemma 2 of [15])) *Let Σ_1 and Σ_2 be alphabets, $w_1, w_2 \in \mathbb{N}_+$, and $g : \Sigma_1 \to \Sigma_2$ be a map from Σ_1 to Σ_2. There exist maps*

1. $f_\cup : \mathcal{B}(\Sigma_1, w_1) \times \mathcal{B}(\Sigma_2, w_2) \to \mathcal{B}(\Sigma_1 \cup \Sigma_2, w_1 + w_2)$,
2. $f_\cap : \mathcal{B}(\Sigma_1, w_1) \times \mathcal{B}(\Sigma_2, w_2) \to \mathcal{B}(\Sigma_1 \cup \Sigma_2, w_1 \cdot w_2)$,
3. $f_\otimes : \mathcal{B}(\Sigma_1, w_1) \times \mathcal{B}(\Sigma_2, w_2) \to \mathcal{B}(\Sigma_1 \times \Sigma_2, w_1 \cdot w_2)$,
4. $f_g : \mathcal{B}(\Sigma_1, w_1) \to \mathcal{B}(\Sigma_2, w_1)$,
5. $f_{g^{-1}} : \mathcal{B}(\Sigma_2, w_2) \to \mathcal{B}(\Sigma_1, w_2)$,
6. $f_\neg : \widehat{\mathcal{B}}(\Sigma_1, w_1) \to \widehat{\mathcal{B}}(\Sigma_1, w_1)$,

such that for each (Σ_1, w_1)-ODD $D = B_1 B_2 \ldots B_k$, each (Σ_2, w_2)-ODD $D' = B_1' B_2' \ldots B_k'$, and each deterministic, complete (Σ_1, w_1)-ODD $D'' = B_1'' B_2'' \ldots B_k''$, the following hold.

1. $f_\cup(D \otimes D') \doteq f_\cup(B_1, B_1') f_\cup(B_2, B_2') \ldots f_\cup(B_k, B_k')$ *is a* $(\Sigma_1 \cup \Sigma_2, w_1 + w_2)$-*ODD such that*

$$\mathcal{L}(f_\cup(D \otimes D')) = \mathcal{L}(D) \cup \mathcal{L}(D').$$

2. $f_\cap(D \otimes D') \doteq f_\cap(B_1, B_1') f_\cap(B_2, B_2') \ldots f_\cap(B_k, B_k')$ *is a* $(\Sigma_1 \cup \Sigma_2, w_1 \cdot w_2)$-*ODD such that*

$$\mathcal{L}(f_\cap(D \otimes D')) = \mathcal{L}(D) \cap \mathcal{L}(D').$$

3. $f_\otimes(D \otimes D') \doteq f_\otimes(B_1, B_1') f_\otimes(B_2, B_2') \ldots f_\otimes(B_k, B_k')$ *is a* $(\Sigma_1 \times \Sigma_2, w_1 \cdot w_2)$-*ODD such that*

$$\mathcal{L}(f_\otimes(D \otimes D')) = \mathcal{L}(D) \otimes \mathcal{L}(D').$$

4. $f_g(D) \doteq f_g(B_1) f_g(B_2) \ldots f_g(B_k)$ *is a* (Σ_2, w_1)-*ODD such that*

$$\mathcal{L}(f_g(D)) = g(\mathcal{L}(D)).$$

5. $f_{g^{-1}}(D') \doteq f_{g^{-1}}(B'_1)f_{g^{-1}}(B'_2)\dots f_{g^{-1}}(B'_k)$ *is a* (Σ_1, w_2)*-ODD such that*

$$\mathcal{L}(f_{g^{-1}}(D')) = g^{-1}(\mathcal{L}(D)).$$

6. $f_\neg(D) \doteq f_\neg(B''_1)f_\neg(B''_2)\dots f_\neg(B''_k)$ *is a deterministic, complete* (Σ_1, w)*-ODD such that*

$$\mathcal{L}(f_\neg(D)) = \Sigma^k \backslash \mathcal{L}(D).$$

Lemma 15 immediately implies implies that the collection of regular-decisional classes of languages is effectively closed under several *pointwise* operations, as stated in the next corollary.

Corollary 16 *Let* Σ_1 *and* Σ_2 *be alphabets,* $w_1, w_2 \in \mathbb{N}_+$, $g : \Sigma_1 \to \Sigma_2$ *be a map from* Σ_1 *to* Σ_2, $\mathcal{F}$ *be a* (Σ_1, w_1)*-SOFA, and* $\mathcal{F}'$ *be a* (Σ_2, w_2)*-SOFA.*

1. **Pointwise union.** *There is a SOFA* $\mathcal{F}\dot\cup\mathcal{F}'$ *such that*

$$\mathcal{L}_2(\mathcal{F}\dot\cup\mathcal{F}') = \{\mathcal{L}(D) \cup \mathcal{L}(D') \ : \ D \in \mathcal{L}(\mathcal{F}), \ D' \in \mathcal{L}(\mathcal{F}'), \ \mathsf{len}(D) = \mathsf{len}(D')\}.$$

2. **Pointwise intersection.** *There is a SOFA* $\mathcal{F}\dot\cap\mathcal{F}'$,

$$\mathcal{L}_2(\mathcal{F}\dot\cap\mathcal{F}') = \{\mathcal{L}(D) \cap \mathcal{L}(D') \ : \ D \in \mathcal{L}(\mathcal{F}), \ D' \in \mathcal{L}(\mathcal{F}'), \mathsf{len}(D) = \mathsf{len}(D')\}.$$

3. **Pointwise tensor product.** *There is a SOFA* $\mathcal{F}\dot\otimes\mathcal{F}'$,

$$\mathcal{L}_2(\mathcal{F}\dot\otimes\mathcal{F}') = \{\mathcal{L}(D) \otimes \mathcal{L}(D') \ : \ D \in \mathcal{L}(\mathcal{F}), \ D' \in \mathcal{L}(\mathcal{F}'), \mathsf{len}(D) = \mathsf{len}(D')\}.$$

4. **Pointwise map.** *There is a SOFA* $\dot{g}(\mathcal{F})$ *such that*

$$\mathcal{L}_2(\dot{g}(\mathcal{F})) = \{g(\mathcal{L}(D)) \ : \ D \in \mathcal{L}(\mathcal{F})\}.$$

5. **Pointwise inverse map:** *There is a SOFA* $\dot{g}^{-1}(\mathcal{F}')$ *such that*

$$\mathcal{L}_2(\dot{g}^{-1}(\mathcal{F}')) = \{g^{-1}(\mathcal{L}(D)) \ : \ D \in \mathcal{L}(\mathcal{F}')\}.$$

6. **Pointwise negation:** *There is a SOFA* $\dot{\neg}\mathcal{F}$ *such that*

$$\mathcal{L}_2(\dot{\neg}\mathcal{F}) = \{(k, \Sigma^k)\backslash\mathcal{L}(D) \ : \ D \in \mathcal{L}(\mathcal{F}), \mathsf{len}(D) = k\}.$$

Proof The proof follows directly from the fact that regular languages are closed under maps, together with Lemma 15. The SOFAs $\dot{g}(\mathcal{F})$, $\dot{g}^{-1}(\mathcal{F})$, and $\dot{\neg}\mathcal{F}$ are obtained from $\mathcal{F}$ by replacing each transition (q, B, q') with the transitions $(q, f_g(B), q')$, $(q, f_{g^{-1}}(B), q')$, and $(q, f_\neg(B), q')$ respectively. For the binary operations, we first compute a finite automaton $\mathcal{F} \otimes \mathcal{F}'$ over the alphabet $\mathcal{B}(\Sigma_1, w_1) \times \mathcal{B}(\Sigma_2, w_2)$ that accepts a string $D \otimes D' = (B_1, B'_1)(B_2, B'_2)\dots(B_k, B'_k)$ if and only if $D = B_1 B_2 \dots B_k$ is accepted by $\mathcal{F}$ and $D' = B'_1 B'_2 \dots B'_k$ is accepted by $\mathcal{F}'$. Subsequently we define $\mathcal{F}\dot\cup\mathcal{F}'$, $\mathcal{F}\dot\cap\mathcal{F}'$ and $\mathcal{F}\dot\otimes\mathcal{F}'$ by replacing each transition $(q, (B, B'), q')$ of $\mathcal{F} \otimes \mathcal{F}'$ with the transitions $(q, f_\cup(B, B'), q')$, $(q, f_\cap(B, B'), q')$, and $(q, f_\otimes(B, B'), q')$ respectively.

We exemplify how Lemma 15 can be used to complete the proof with the first item. The others follow an analogous argument. From the construction of $\mathcal{F}\dot\cup\mathcal{F}'$, we have that $D \in \mathcal{L}(\mathcal{F})$ and $D' \in \mathcal{L}(\mathcal{F}')$ are such that $\mathsf{len}(D) = \mathsf{len}(D')$ if and only if $f_\cup(D \otimes D')$ belongs to $\mathcal{L}(\mathcal{F}\dot\cup\mathcal{F}')$. Since $\mathcal{L}(f_\cup(D \otimes D')) = \mathcal{L}(D) \cup \mathcal{L}(D')$, we

have that $\mathcal{L}_2(\mathcal{F} \dot{\cup} \mathcal{F}') = \{\mathcal{L}(D) \cup \mathcal{L}(D') \ : \ D \in \mathcal{L}(\mathcal{F}), \ D' \in \mathcal{L}(\mathcal{F}'), \ \mathsf{len}(D) = \mathsf{len}(D')\}$. □

5 Algorithmic Applications

In this section, we show that Theorems 9 and Theorem 10 can be used to provide novel algorithmic applications in the realm of the theory of ODDs of bounded width, and therefore also in the realm of the theory of ordered binary decision diagrams (OBDDs) of bounded width. In Section 5.1 we will show that several minimization problems for deterministic and nondeterministic ODDs can be solved in fixed parameter tractable time when parameterized by width. Subsequently, in Section 5.2 we will show that the problem of counting the number of distinct functions computable by some ODD of length k and width w can be solved in time $h(|\Sigma|, w) \cdot k^{O(1)}$ for a suitable $h : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$.

5.1 Width and Size Minimization of Nondeterministic ODDs

Models of computation comprised by ODDs of constant width have been studied in a variety of fields, such as symbolic computation, machine learning and property testing [3, 21, 33, 35]. In this section, we show that width minimization for ODDs is fixed-parameter tractable in the width parameter. Additionally, the space of ODDs where the minimization will take place may be selected as the language $\mathcal{L}(\mathcal{F})$ of a given second-order finite automaton $\mathcal{F}$. Furthermore, if such a minimum width ODD D' with $\mathcal{L}(D) = \mathcal{L}(D')$ exists in $\mathcal{L}(\mathcal{F})$, then one can furthermore impose that D' has minimum number of states or minimum number of transitions.

As important special cases, if we set $\mathcal{F}$ to be the finite automaton accepting the language $\mathcal{B}(\Sigma, w)^{\circledast}$ the minimization occurs in the space of all (possibly nondeterministic) ODDs of width at most w, while by setting $\mathcal{F}$ to be the finite automaton accepting the language $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$, the minimization takes place over the space of deterministic, complete ODDs of width at most w.

Lemma 17 *Let Σ be an alphabet, $w \in \mathbb{N}_+$, D be a (Σ, w)-ODD, and $\mathcal{F}$ be a (Σ, w)-SOFA. One can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k$ an acyclic (Σ, w)-SOFA $X(\mathcal{F}D)$ with $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k$ states such that $\mathcal{L}(X(\mathcal{F}D)) = \{D' \in \mathcal{L}(\mathcal{F}) \ : \ \mathcal{L}(D) = \mathcal{L}(D')\}$.*

Proof Let $\mathcal{F}$ be a (Σ, w)-SOFA, and $D \in \mathcal{B}(\Sigma, w)^{\circ k}$. Consider the $(\widehat{\mathcal{B}}(\Sigma, 2^w), \mathcal{B}(\Sigma, w))$-transduction $\mathsf{t}_D = \{(\mathcal{C}(D), D)\}$. Note that t_D is a singleton, and therefore, it is $(k+1)$-regular, since the language $L(\mathsf{t}_D) = \{\mathcal{C}(D) \otimes D\}$ is accepted by a finite automaton $\widetilde{\mathcal{F}}$ with $(k+1)$ states $\{q_0, \ldots, q_k\}$. Here, q_0 is the unique initial state and q_k is the unique final state. Indeed, let $\mathcal{C}(D) = B_1' B_2' \ldots B_k'$. Note that this canonical form can be constructed in time $2^{O(w)} \cdot |\Sigma| \cdot k$ by applying the standard minimization algorithm for a single ODD (Theorem 4). Then, for each $i \in \{0, \ldots, k-1\}$, the automaton has a unique transition leaving q_i, namely, the transition

$(q_i, (B_i', B_i), q_{i+1})$. It should be clear that $\mathcal{C}(D) \otimes D = (B_1', B_1)(B_2', B_2) \ldots (B_k', B_k)$. is the only string accepted by $\hat{\mathcal{F}}$.

Now consider the transduction $\mathfrak{can}[\Sigma, w]$. Since this transduction is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular (Theorem 9), it follows from Proposition 5.(2) that the transduction

$$\mathfrak{can}[\Sigma, w] \circ \mathfrak{t}_D = \{(D', D) \;:\; D' \in \mathcal{B}(\Sigma, w)^{\circledast}, \mathcal{C}(D') = \mathcal{C}(D)\}$$
$$= \{(D', D) \;:\; D' \in \mathcal{B}(\Sigma, w)^{\circledast}, \mathcal{L}(D') = \mathcal{L}(D)\}$$

is $(2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot k)$-regular, and therefore, the language

$$\mathsf{Dom}(\mathfrak{can}[\Sigma, w] \circ \mathfrak{t}_D) = \{D' \in \mathcal{B}(\Sigma, w)^{\circledast} \;:\; \mathcal{L}(D') = \mathcal{L}(D)\}$$

is $(2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot k)$-regular. Additionally, an automaton $\mathcal{F}'$ accepting $\mathsf{Dom}(\mathfrak{can}[\Sigma, w] \circ \mathfrak{t}_D)$ can be constructed in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot k$. This implies that one can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k$ a finite automaton $X(\mathcal{F}D)$ with $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k$ states accepting the language $\mathsf{Dom}(\mathfrak{can}[\Sigma, w] \circ \mathfrak{t}_D) \cap \mathcal{L}(\mathcal{F}D) = \{D' \in \mathcal{L}(\mathcal{F}) \;:\; \mathcal{L}(D) = \mathcal{L}(D')\}$. Since all ODDs accepted by $X(\mathcal{F}D)$ have length k, we may assume that this automaton is acyclic, where the set of states is partitioned into a sequence of levels, and transitions can only exist from a given level to the next level in the sequence. □

Let $\oplus : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be a commutative, associative binary operation (typically max or $+$), and $\omega : \mathcal{B}(\Sigma, w) \to \mathbb{N}$ be a weighting function. Then the weight of an ODD $D \in \mathcal{B}(\Sigma, w)^{\circ k}$ is defined as $\omega_{\oplus}(D) = \bigoplus_{i=1}^{k} \omega(B_i)$.

Proposition 18 *Let $\omega : \mathcal{B}(\Sigma, w) \to \mathbb{N}$ be a fixed weighting function on the alphabet $\mathcal{B}(\Sigma, w)$ and $\oplus : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be one of the operations $+$ or $\max$. Let $\mathcal{J}$ be a (Σ, w)-SOFA and $a \in \mathbb{N}$. Then, in time $O(|\mathcal{J}| \cdot \log |\mathcal{J}| \cdot \log a)$, one can determine whether there is an ODD of weight at most a in $\mathcal{L}(\mathcal{J})$, and if yes, construct an ODD $D \in \mathcal{L}(\mathcal{J})$ of minimum weight. The time complexity can be improved to $O(|\mathcal{J}| \cdot \log a)$ if $\mathcal{J}$ is acyclic.*

Proof Let $G = (V, E, \gamma)$ be the weighted directed graph with vertex set $V = Q(\mathcal{J})$, edge set $E = \{(q, q') \;:\; \exists B \in \mathcal{B}(\Sigma, w), (q, B, q') \in T(\mathcal{J})\}$, and weighting function γ such that for each edge $(q, q') \in E$, $\gamma(q, q') = \{\min\{\omega(B)\} \;:\; (q, B, q') \in T(\mathcal{J}), \omega(B) \le a\}$. Then there is an ODD $D \in \mathcal{L}(\mathcal{J})$ of weight at most a if and only if the weight of a shortest (weighted) path from some initial state q to some final state q' of $\mathcal{J}$ is at most a. Let $q_0 q_1 \ldots q_k$ be such a shortest path. Then any ODD $D = B_1 B_2 \ldots B_k$ where $(q_{i-1}, B, q_i) \in T(\mathcal{J})$ for all $i \in [k]$ is a valid solution. Using standard algorithms, the time for computing a shortest path is upper bounded by $O((|E| + |V| \log |V|) \log a) = O(|\mathcal{F}| \cdot \log |\mathcal{F}| \cdot \log a)$. If by $\mathcal{J}$ is acyclic, then this running time can be improved to $O((|V| + |E|) \log a) = O(|\mathcal{F}| \cdot \log a)$. □

By combining Lemma 17 with Proposition 18 we obtain the following theorem.

Theorem 19 *Let D be an ODD in $\mathcal{B}(\Sigma, w)^{\circ k}$ and let $\mathcal{F}$ be a (Σ, w)-SOFA. One can determine in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k$ whether there is an ODD $D' \in \mathcal{L}(\mathcal{F})$ such that $\mathcal{L}(D') = \mathcal{L}(D)$. Suppose such an ODD exists.*

1. *One can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k$ an ODD $D' \in \mathcal{L}(\mathcal{F})$ of minimum width such that $\mathcal{L}(D) = \mathcal{L}(D')$.*
2. *One can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k \cdot \log k$ an ODD $D' \in \mathcal{L}(\mathcal{F})$ with minimum number of states such that $\mathcal{L}(D) = \mathcal{L}(D')$.*
3. *One can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k \cdot \log k$ an ODD $D' \in \mathcal{L}(\mathcal{F})$ with minimum number of transitions such that $\mathcal{L}(D) = \mathcal{L}(D')$.*

Proof 1. In Proposition 18, set $\mathcal{J} = X(\mathcal{F}D)$, $a = w$, $\oplus(x, y) = \max\{x, y\}$, and for each $B \in \mathcal{B}(\Sigma, w)$ set $\omega(B) = \mathsf{w}(B)$. Then for each $D' \in \mathcal{B}(\Sigma, w)^{\circ k}$, $\omega_\oplus(D) = \mathsf{w}(D)$. Therefore, by Proposition 18, one can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k$ an ODD $D' \in \mathcal{L}(\mathcal{F})$ of minimum width such that $\mathcal{L}(D') = \mathcal{L}(D)$.

2. In Proposition 18, set $\mathcal{J} = X(\mathcal{F}D)$, $a = w \cdot (k + 1)$, $\oplus(x, y) = x + y$, and for each $B \in \mathcal{B}(\Sigma, w)$ set $\omega(B) = |\ell(B)| + \phi(B) \cdot |r(B)|$. Then for each $D' \in \mathcal{B}(\Sigma, w)^{\circ k}$, $\omega_\oplus(D) = \mathsf{nSt}(D)$. Therefore, by Proposition 18, one can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k \cdot \log k$ an ODD $D' \in \mathcal{L}(\mathcal{F})$ with minimum number of states such that $\mathcal{L}(D') = \mathcal{L}(D)$.

3. In Proposition 18, set $\mathcal{J} = X(\mathcal{F}D)$, $a = |\Sigma| \cdot w^2 \cdot k$, $\oplus(x, y) = x + y$, and for each $B \in \mathcal{B}(\Sigma, w)$ set $\omega(B) = |T(B)|$. Then for each $D' \in \mathcal{B}(\Sigma, w)^{\circ k}$, $\omega_\oplus(D) = \mathsf{nTr}(D)$. Therefore, by Proposition 18, one can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot \mathsf{nSt}(\mathcal{F}) \cdot k \cdot \log k$ an ODD $D' \in \mathcal{L}(\mathcal{F})$ with minimum number of transitions such that $\mathcal{L}(D') = \mathcal{L}(D)$. $\square$

Let $\mathcal{S} \subseteq \mathcal{B}(\Sigma, w)$. Then by plugging $\mathcal{F}_\mathcal{S}$ in Theorem 19, the following theorem, which can be used to address several minimization problems for ODDs over the space of ODDs in $\mathcal{S}^\circledast$.

Theorem 20 *Let D be an ODD in $\mathcal{B}(\Sigma, w)^{\circ k}$ and let $\mathcal{S} \subseteq \mathcal{B}(\Sigma, w)$. One can determine in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot k$ whether there is an ODD $D' \in \mathcal{S}^{\circ k}$ such that $\mathcal{L}(D') = \mathcal{L}(D)$. Suppose such an ODD exists.*

1. *One can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot k$ an ODD $D' \in \mathcal{S}^\circledast$ of minimum width such that $\mathcal{L}(D) = \mathcal{L}(D')$.*
2. *One can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot k \cdot \log k$ an ODD $D' \in \mathcal{S}^\circledast$ with minimum number of states such that $\mathcal{L}(D) = \mathcal{L}(D')$.*
3. *One can construct in time $2^{O(|\Sigma| \cdot w \cdot 2^w)} \cdot k \cdot \log k$ an ODD $D' \in \mathcal{S}^\circledast$ with minimum number of transitions such that $\mathcal{L}(D) = \mathcal{L}(D')$.*

5.2 Counting Functions Computable by ODDs of a Given Width.

Let Σ be an alphabet and $w, k \in \mathbb{N}_+$. Each ODD $D \in \mathcal{B}(\Sigma, w)^{\circ k}$ can be regarded as a representation of a function $f_D : \Sigma^k \to \{0, 1\}$. More precisely, for each $s \in \Sigma^k$, $f_D(s) = 1$ if and only if $s \in \mathcal{L}(D)$. We say that f_D is the function computed by D.

In this subsection, we analyze the problem of counting the number of functions of type $\Sigma^k \to \{0, 1\}$ that can be computed by some ODD of width w over the alphabet Σ. We note that to solve this problem it is not enough to count the number

of ODDs in $\mathcal{B}(\Sigma, w)^{\circ k}$. The caveat is that several ODDs in $\mathcal{B}(\Sigma, w)^{\circ k}$ may represent the same function. Fortunately, we can solve the issue of multiple representatives for a given function by resorting to our canonical form of canonical forms theorem (Theorem 10).

It is well known that the problem of counting the number of strings of length k accepted by a given deterministic finite automaton $\mathcal{A}$ can be solved in time polynomial in k and in the number of states of $\mathcal{A}$. Below we state a more precise upper bound.

Proposition 21 *Let $\mathcal{A}$ be a deterministic finite automaton over an alphabet Γ. Then, for each $k \in \mathbb{N}$, one can count in time $O(\mathsf{nSt}(\mathcal{A}) \cdot k^2 \cdot |\Gamma| \cdot \log |\Gamma|)$ the number of words of length k accepted by $\mathcal{A}$.*

Proof Let $\mathcal{A} = (\Gamma, Q, I, F, T)$. Since $\mathcal{A}$ is deterministic, $I = \{q_0\}$ for some state q_0. Additionally, there is a bijection from the set words of length k accepted by $\mathcal{A}$ to the set accepting sequences of transitions connecting the initial state q_0 to some final state in F.

We start by constructing a matrix $M : [\![k]\!] \times Q \to \mathbb{N}$ such that for each $i \in k$ and each $q \in Q$, the entry $M(i, q)$ is equal to the number of valid sequences of transitions of length $k - i$ from q to some final state in F. In particular, $M(0, q_0)$ is the number of valid sequences of transitions of length k from q_0 to some final state in F. The matrix M is constructed by induction on $k - i$. In the base case, $i = k$. In this case, we set $M(k, q) = 1$ if $q \in F$, and set $M(k, q) = 0$ otherwise. Now, let $i \in [\![k - 1]\!]$ and assume that the value $M(i + 1, q)$ has been determined for every $q \in Q$. Then, for each $q \in Q$, we let $M(i, q) = \sum_{(q,\sigma,q') \in T} M(i + 1, q')$. In other words, $M(i, q)$ is defined as the sum of all $M(i, q')$ for which (q, σ, q') is a transition in $\mathcal{A}$ for some $\sigma \in \Gamma$.

Since, there are at most $|\Gamma|^k$ words of length k, we have that each entry of M can be represented using $k \cdot \log |\Gamma|$ bits. Additionally, the computation of each entry involves the summation of $|\Gamma|$ entries, which in overall can be performed in time $O(k \cdot |\Gamma| \cdot \log \Gamma)$. Since the matrix has $(k + 1) \cdot \mathsf{nSt}(\mathcal{A})$ entries, the whole matrix can be constructed in time $O(\mathsf{nSt}(\mathcal{A}) \cdot k^2 \cdot |\Gamma| \cdot \log |\Gamma|)$. $\square$

Theorem 22 *Let $\mathcal{F}$ be a (Σ, w)-second order finite automaton. For each $k \in \mathbb{N}$, one can count in time $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{\mathcal{O}(|\Sigma| \cdot w \cdot 2^w)}} \cdot k^2$ the number of functions $f : \Sigma^k \to \{0, 1\}$ computable by some ODD of length k in $\mathcal{L}(\mathcal{F})$.*

Proof By Theorem 10, one can construct in time $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{\mathcal{O}(|\Sigma| \cdot w \cdot 2^w)}}$ a deterministic second-order finite automaton $\mathcal{C}_2(\mathcal{F})$ (with at most $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{\mathcal{O}(|\Sigma| \cdot w \cdot 2^w)}}$ states) such that $\mathcal{L}(\mathcal{C}_2(\mathcal{F})) = \{\mathcal{C}(D) : D \in \mathcal{L}(\mathcal{F})\}$. This implies that for each language $L \in \mathcal{L}_2(\mathcal{F})$, there is a unique ODD $D \in \mathcal{L}(\mathcal{C}_2(\mathcal{F}))$ such that $\mathcal{L}(D) = L$. Therefore, counting the number of functions of type $\Sigma^w \to \{0, 1\}$ computable by some ODD in $\mathcal{L}(\mathcal{F})$ amounts to counting the number of ODDs of length k accepted by $\mathcal{C}_2(\mathcal{F})$. By setting $\mathcal{A} = \mathcal{C}_2(\mathcal{F})$ and $\Gamma = \mathcal{B}(\Sigma, 2^w)$ in Proposition 21, and by using the facts that $|\mathcal{A}| =$

$2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{\mathcal{O}(|\Sigma| \cdot w \cdot 2^w)}}$ and $|\Gamma| = 2^{O(|\Sigma| \cdot 2^w \cdot w)}$, we have that this counting problem can be solved in time $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{\mathcal{O}(|\Sigma| \cdot w \cdot 2^w)}} \cdot k^2$. □

If all ODDs in the language of $\mathcal{F}$ are deterministic and complete then one can adapt the proof of Theorem 22 by using Observation 11 and by setting $\Gamma = \widehat{\mathcal{B}}(\Sigma, w)$ in order to obtain a more efficient counting algorithm.

Observation 23 Let $\mathcal{F}$ be a (Σ, w)-second order finite automaton such that $\mathcal{L}(\mathcal{F}) \subseteq \widehat{\mathcal{B}}(\Sigma, w)$. For each $k \in \mathbb{N}$, one can count in time $2^{\mathsf{nSt}(\mathcal{F}) \cdot 2^{\mathcal{O}(|\Sigma| \cdot w \cdot \log w)}} \cdot k^2$ the number of functions $f : \Sigma^k \to \{0, 1\}$ computable by some ODD of length k in $\mathcal{L}(\mathcal{F})$.

By combining Lemma 7 with Theorem 22 and Observation 23, we obtain the following corollary.

Corollary 24 Let Σ be an alphabet, $w \in \mathbb{N}_+$, $\mathcal{S} \subseteq \mathcal{B}(\Sigma, w)$, and $\widehat{\mathcal{S}} \subseteq \widehat{\mathcal{B}}(\Sigma, w)$.

1. *One can count in time $2^{2^{\mathcal{O}(|\Sigma| \cdot w \cdot 2^w)}} \cdot k^2$ the number of functions $f : \Sigma^k \to \{0, 1\}$ computable by some ODD in $\mathcal{S}^{\circledast}$.*
2. *One can count in time $2^{2^{\mathcal{O}(|\Sigma| \cdot w \cdot \log w)}} \cdot k^2$ the number of functions $f : \Sigma^k \to \{0, 1\}$ computable by some ODD in $\widehat{\mathcal{S}}^{\circledast}$.*

Proof By Lemma 7, one can construct SOFAs $\mathcal{F}_{\mathcal{S}}$ and $\mathcal{F}_{\widehat{\mathcal{S}}}$ with $(|\mathcal{S}| + 1)$ and $(|\widehat{\mathcal{S}}| + 1)$ states respectively such that $\mathcal{L}(\mathcal{F}_{\mathcal{S}}) = \mathcal{S}^{\circledast}$, and $\mathcal{L}(\mathcal{F}_{\widehat{\mathcal{S}}}) = \widehat{\mathcal{S}}^{\circledast}$. Since $|\mathcal{S}| = 2^{O(|\Sigma| w^2)}$, it follows from Theorem 22 that one can count the number of functions $f : \Sigma^k \to \{0, 1\}$ computable by ODDs in $\mathcal{S}^{\circledast}$ in time $2^{2^{\mathcal{O}(|\Sigma| \cdot w \cdot 2^w)}} \cdot k^2$. Analogously, since $|\widehat{\mathcal{S}}| = 2^{O(|\Sigma| w \log w)}$, it follows from Observation 23 that one can count the number of functions $f : \Sigma^k \to \{0, 1\}$ computable by ODDs in $\widehat{\mathcal{S}}^{\circledast}$ in time $2^{2^{\mathcal{O}(|\Sigma| \cdot w \cdot \log w)}}$. □

6 Proof of the Canonization as Transduction Theorem

In this section, we prove Theorem 9, which states that for each alphabet Σ, and each $w \in \mathbb{N}_+$ the following holds.

1. The functional transduction $\widehat{\mathfrak{can}}[\Sigma, w] = \{(D, \mathcal{C}(D)) : D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}\}$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular.
2. The functional transduction $\mathfrak{can}[\Sigma, w] = \{(D, \mathcal{C}(D)) : D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular.

Although the complete proof of Theorem 9 is quite technical, it is possible to give an intuitive overview of the main steps in the proof. More specifically, we will show that the transduction $\widehat{\mathfrak{can}}[\Sigma, w]$ can be cast a composition

$$\widehat{\mathfrak{can}}[\Sigma, w] \doteq \mathfrak{d}(\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}) \circ \mathfrak{rea}[\Sigma, 2^w] \circ \mathfrak{mer}[\Sigma, 2^w] \circ \mathfrak{nor}[\Sigma, 2^w], \quad (1)$$

of regular transductions satisfying the following properties.

1. $\mathfrak{d}(\widehat{\mathcal{B}}(\Sigma, w)^{\circledast})$ is a functional $2^{O(|\Sigma| \cdot w \log w)}$-regular $(\widehat{\mathcal{B}}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-trans-
   duction that sends each ODD $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ to itself. This transduction is used
   to limit the domain of $\widehat{\mathfrak{can}}[\Sigma, w]$ to deterministic, complete (Σ, w)-ODDs.

2. $\mathfrak{rea}[\Sigma, w]$ is a functional $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular $(\widehat{\mathcal{B}}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-transduc-
   tion that sends each ODD $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ to a reachable ODD $D' \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$
   with $\mathcal{L}(D) = \mathcal{L}(D')$. This transduction simulates the process of eliminating
   unreachable states from D.

3. $\mathfrak{mer}[\Sigma, w]$ is a functional $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular $(\widehat{\mathcal{B}}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-transduc-
   tion that sends each reachable, deterministic, complete ODD $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ to a
   minimized, deterministic, complete ODD $D' \in \mathcal{B}(\Sigma, w)^{\circledast}$ with $\mathcal{L}(D) = \mathcal{L}(D')$.
   This transduction simulates the process of merging equivalent states in a ODD.

4. $\mathfrak{nor}[\Sigma, w]$ is a functional $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular $(\widehat{\mathcal{B}}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-transduc
   tion that sends each deterministic, complete ODD $D \in \mathcal{B}(\Sigma, w)^{\circledast}$ to its nor-
   malized version $D' \in \mathcal{B}(\Sigma, w)^{\circledast}$. This transduction simulates the process
   of numbering the states of an ODD according to their lexicographical order.
   This guarantees that the ODD is unique not only up to isomorphism, but also
   syntactically unique.

Intuitively, the regular transductions above simulate the steps used in the stan-
dard ODD minimization algorithm. By using Proposition 5.(2), we have that the
transduction $\mathfrak{can}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular. The fact that each of the five trans-
ductions above is functional implies that $\mathfrak{can}[\Sigma, w]$ is also functional. Additionally,
it is straightforward to note that $\mathsf{Dom}(\widehat{\mathfrak{can}}[\Sigma, w]) = \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$. Finally, a pair of
ODDs (D, D') belongs to $\widehat{\mathfrak{can}}[\Sigma, w]$ if and only if D' is deterministic, complete,
minimized, normalized and $\mathcal{L}(D) = \mathcal{L}(D')$. In other words, if and only if D' is the
canonical form $\mathcal{C}(D)$ of Theorem 4.

Now, the transduction $\mathfrak{can}[\Sigma, w]$ can be obtained as the composition

$$\mathfrak{can}[\Sigma, w] \doteq \mathfrak{d}(\mathcal{B}(\Sigma, w)^{\circledast}) \circ \mathfrak{det}[\Sigma, w] \circ \widehat{\mathfrak{can}}[\Sigma, 2^w]. \tag{2}$$

Here, $\mathfrak{det}[\Sigma, w]$ is a functional 2-regular $(\mathcal{B}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, 2^w))$-transduction that
sends each ODD $D \in \mathcal{B}(\Sigma, w)^{\circledast}$ to a deterministic, complete ODD $D' \in \mathcal{B}(\Sigma, 2^w)^{\circledast}$
with $\mathcal{L}(D) = \mathcal{L}(D')$. This transduction simulates the application of the stan-
dard power set construction to the states of a ODD, and blows the width of the
original ODD at most exponentially. Since $\widehat{\mathfrak{can}}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular, we
have that $\widehat{\mathfrak{can}}[\Sigma, 2^w]$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular. This implies that $\mathfrak{can}[\Sigma, w]$ is also
$2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular.

Next, in Section 6.1, we will define two elementary types of regular transduc-
tions: the *multimap transductions* and the *compatibility transductions*. Subsequently
we will define $\mathfrak{det}[\Sigma, w]$, $\mathfrak{rea}[\Sigma, w]$, $\mathfrak{mer}[\Sigma, w]$ and $\mathfrak{nor}[\Sigma, w]$ using these elemen-
tary transductions. The determinization transduction $\mathfrak{det}[\Sigma, w]$ will be defined in
Section 6.2 and its properties analyzed in Lemma 27. The reachability transduction
$\mathfrak{rea}[\Sigma, w]$ will be defined in Section 6.3, and its properties analyzed in Lemma 30.
The merging transduction $\mathfrak{mer}[\Sigma, w]$ will be defined in Section 6.4, and its properties
analyzed in Lemma 36. The normalization transduction will be defined Section 6.5

and its properties analyzed in Lemma 39. Finally, in Section 6.6 we will combine Observation 26 with these four lemmas to conclude the proof of Theorem 9.

6.1 Basic Transductions

Let Σ be an alphabet and $R \subseteq \Sigma \times \Sigma$ be a binary relation over Σ. For each $k \in \mathbb{N}_+$ and each string $s = \sigma_1 \cdots \sigma_k \in \Sigma^k$, we say that s is R-*compatible* if $(\sigma_i, \sigma_{i+1}) \in R$ for each $i \in [k-1]$. We let

$$\mathfrak{cp}[R] \doteq \{(s, s) \in \Sigma^+ \times \Sigma^+ : s \text{ is } R\text{-compatible}\}$$

be the R-*compatibility transduction*, i.e. the (Σ, Σ)-transduction that sends each R-compatible string $s \in \Sigma^+$ to itself.

Let Σ_1 and Σ_2 be two alphabets and $R \subseteq \Sigma_1 \times \Sigma_2$ be a relation. We let

$$\mathfrak{mm}[R] \doteq \{(s, u): s = \sigma_1 \cdots \sigma_k \in \Sigma_1^k, \ u = \tau_1 \cdots \tau_k \in \Sigma_2^k, (\sigma_i, \tau_i)$$
$$\in R \text{ for each } i \in [k], \ k \in \mathbb{N}_+\}$$

be the R-*multimap transduction*. If $g \colon \Sigma_1 \to \Sigma_2$ is a map, then we write $\mathfrak{mm}[g]$ to denote the transduction $\mathfrak{mm}[R_g]$, where $R_g \doteq \{(\sigma, g(\sigma)): \sigma \in \Sigma_1\}$.

Proposition 25 *Let Σ, Σ_1 and Σ_2 be three alphabets, and let $R \subseteq \Sigma \times \Sigma$ and $R' \subseteq \Sigma_1 \times \Sigma_2$ be binary relations. The following statements hold.*

1. *The transduction $\mathfrak{mm}[R']$ is 2-regular.*
2. *The transduction $\mathfrak{cp}[R]$ is $(|\Sigma| + 2)$-regular.*

Proof 1. We let $\mathcal{F}_{\mathfrak{mm}[R']}$ be the finite automaton with state set $Q(\mathcal{F}_{\mathfrak{mm}[R']}) = \{q, q'\}$, initial state set $I(\mathcal{F}_{\mathfrak{mm}[R']}) = \{q\}$, final state set $F_{\mathfrak{mm}[R']} = \{q'\}$ and transition set $T_{\mathfrak{mm}[R']} = \{(q, (\sigma, \tau), q'): (\sigma, \tau) \in R'\} \cup \{(q', (\sigma, \tau), q'): (\sigma, \tau) \in R'\}$. Clearly, $\mathcal{F}$ has exactly two states, namely q and q'. Moreover, for each two strings $s \in \Sigma_1^+$ and $u \in \Sigma_2^+$, $\mathcal{F}_{\mathfrak{mm}[R']}$ accepts the string $s \otimes u \in (\Sigma_1 \times \Sigma_2)^+$ if and only if $|s| = |u|$ and $(\sigma_i, \tau_i) \in R'$ for each $i \in [k]$, where $s = \sigma_1 \cdots \sigma_k, u = \tau_1 \cdots \tau_k$ and $k = |s|$.

2. We let $\mathcal{F}_{\mathfrak{cp}[R]}$ be the finite automaton over the alphabet $\Sigma \times \Sigma$, with state set $Q(\mathcal{F}_{\mathfrak{cp}[R]}) = \{q, q'\} \cup \{q_\sigma : \sigma \in \Sigma\}$, initial state set $I(\mathcal{F}_{\mathfrak{cp}[R]}) = \{q\}$, final state set $F(\mathcal{F}_{\mathfrak{cp}[R]}) = \{q'\}$ and transition set $T(\mathcal{F}_{\mathfrak{cp}[R]}) = \{(q, (\sigma, \sigma), q_\sigma): \sigma \in \Sigma\} \cup \{(q_\sigma, (\tau, \tau), q_\tau): (\sigma, \tau) \in R\} \cup \{(q_\sigma, (\tau, \tau), q'): (\sigma, \tau) \in R\}$. Clearly, $\mathcal{F}_{\mathfrak{cp}[R]}$ has at most $|\Sigma| + 2$ states. Moreover, it is not hard to check that, for each $k \in \mathbb{N}_+$, $\mathcal{F}_{\mathfrak{cp}[R]}$ accepts a string $s = \sigma_1 \cdots \sigma_k \in \Sigma^k$ if and only if $(\sigma_i, \sigma_{i+1}) \in R$ for each $i \in [k-1]$. Therefore, the language of $\mathcal{F}_{\mathfrak{cp}[R]}$ is $\mathcal{L}(\mathcal{F}_{\mathfrak{cp}[R]}) = \mathcal{L}(\mathfrak{cp}[R])$. $\square$

The next observation is a direct consequence of Proposition 5.(3) and Corollary 8.

Observation 26 *Let Σ be an alphabet and $w \in \mathbb{N}_+$.*

1. $\mathfrak{d}(\mathcal{B}(\Sigma, w)^\circledast)$ *is* $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-*regular.*
2. $\mathfrak{d}(\widehat{\mathcal{B}}(\Sigma, w)^\circledast)$ *is* $2^{O(|\Sigma| \cdot w \cdot \log w)}$-*regular.*

6.2 Determinization Transduction

In this subsection, we define the determinization transduction $\mathfrak{det}[\Sigma, w]$, which intuitively simulates the application of the well known power-set construction to the layers of a (Σ, w)-ODD.

For each $w \in \mathbb{N}_+$, we let $\Omega \colon \mathcal{P}(\llbracket w \rrbracket)) \to \llbracket 2^w \rrbracket$ be the bijection that sends each subset $X \subseteq \llbracket w \rrbracket$ to the natural number $\Omega(X) \doteq \sum_{i \in X} 2^i$. In particular, we remark that $\Omega(\emptyset) = 0$ and $\Omega(\{i\}) = 2^i$ for each $i \in X$.

Let Σ be an alphabet, $w \in \mathbb{N}_+$, $B \in \mathcal{B}(\Sigma, w)$, $X \subseteq \ell(B)$ and $\Sigma' \subseteq \Sigma$. We let $\mathbf{N}(B, X, \Sigma')$ be the set of all right states of B that are reachable from some left state in X by reading some symbol in Σ'. More formally,

$$\mathbf{N}(B, X, \Sigma') \doteq \{\mathfrak{q} \in r(B) \colon \exists \mathfrak{p} \in X, \exists \sigma \in \Sigma', (\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B)\}.$$

For each alphabet Σ and each $w \in \mathbb{N}_+$, we let $\mathsf{pw}[\Sigma, w] \colon \mathcal{B}(\Sigma, w) \to \widehat{\mathcal{B}}(\Sigma, 2^w)$ be the map that sends each layer $B \in \mathcal{B}(\Sigma, w)$ to the deterministic, complete layer $\mathsf{pw}(B) \in \widehat{\mathcal{B}}(\Sigma, 2^w)$ defined as follows:

- $\ell(\mathsf{pw}(B)) \doteq \begin{cases} \{\Omega(I(B))\} & \text{if } \iota(B) = 1 \\ \{\Omega(X) \colon X \subseteq \ell(B)\} & \text{otherwise;} \end{cases}$
- $r(\mathsf{pw}(B)) \doteq \{\Omega(X) \colon X \subseteq r(B)\};$
- $T(\mathsf{pw}(B)) \doteq \begin{cases} \{(\Omega(I(B)), \sigma, \Omega(\mathbf{N}(B, I(B), \{\sigma\}))), \sigma \in \Sigma\} & \text{if } \iota(B) = 1 \\ \{(\Omega(X), \sigma, \Omega(\mathbf{N}(B, X, \{\sigma\})) \colon X \subseteq \ell(B), \sigma \in \Sigma\} & \text{otherwise;} \end{cases}$
- $I(\mathsf{pw}(B)) \doteq \begin{cases} \{\Omega(I(B))\} & \text{if } \iota(B) = 1 \\ \emptyset & \text{otherwise;} \end{cases}$
- $F(\mathsf{pw}(B)) \doteq \{\Omega(X) \colon X \subseteq r(B), X \cap F(B) \neq \emptyset\};$
- $\iota(\mathsf{pw}(B)) \doteq \iota(B);$
- $\phi(\mathsf{pw}(B)) \doteq \phi(B).$

Let Σ be an alphabet, $w \in \mathbb{N}_+$, and let $B \in \mathcal{B}(\Sigma, w)$. Since Ω is a bijection, there exists precisely one right state $\mathfrak{q} \in r(\mathsf{pw}(B))$, namely $\mathfrak{q} = \Omega(\mathbf{N}(B, X, \{\sigma\}))$, such that $(\Omega(X), \sigma, \mathfrak{q}) \in T(\mathsf{pw}(B))$ for each subset $X \subseteq \llbracket w \rrbracket$ with $\Omega(X) \in \ell(\mathsf{pw}(B))$ and each symbol $\sigma \in \Sigma$. Furthermore, note that $\iota(\mathsf{pw}(B)) = 1$ implies $\iota(B) = 1$. Thus, if $\iota(\mathsf{pw}(B)) = 1$, then $I(\mathsf{pw}(B)) = \ell(\mathsf{pw}(B)) = \{\Omega(I(B))\}$. As a result, $\mathsf{pw}(B)$ is indeed a deterministic, complete layer in $\widehat{\mathcal{B}}(\Sigma, 2^w)$.

Now, for each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, we define the $(\mathcal{B}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-transduction $\mathfrak{det}[\Sigma, w] \doteq \mathfrak{mm}[\mathsf{pw}[\Sigma, w]]$. The next lemma states that $\mathfrak{det}[\Sigma, w]$ sends each ODD $D \in \mathcal{B}(\Sigma, w)^{\circledast}$ to a deterministic, complete ODD $D' \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ that has the same language as D.

Lemma 27 (Determinization Transduction) *For each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, the following statements hold.*

1. $\mathfrak{det}[\Sigma, w]$ *is functional.*
2. $\mathsf{Dom}(\mathfrak{det}[\Sigma, w]) \supseteq \mathcal{B}(\Sigma, w)^{\circledast}.$
3. *For each pair* $(D, D') \in \mathfrak{det}[\Sigma, w]$, *if* $D \in \mathcal{B}(\Sigma, w)^{\circledast}$, *then* $D' \in \widehat{\mathcal{B}}(\Sigma, 2^w)^{\circledast}$ *and* $\mathcal{L}(D') = \mathcal{L}(D)$.

4. $\mathfrak{det}[\Sigma, w]$ *is 2-regular.*

Proof First, we note that $\mathsf{Dom}(\mathfrak{det}[\Sigma, w]) = \mathcal{B}(\Sigma, w)^+$. This follows from the fact that pw is a *map from the alphabet $\mathcal{B}(\Sigma, w)$ to the alphabet $\widehat{\mathcal{B}}(\Sigma, 2^w)$*. Thus, for each $k \in \mathbb{N}_+$ and each string $D = B_1 \cdots B_k \in \mathcal{B}(\Sigma, w)^k$, there exists exactly one string D' over $\widehat{\mathcal{B}}(\Sigma, 2^w)$ such that $(D, D') \in \mathfrak{det}[\Sigma, w]$, namely the string $D' = \mathsf{pw}(D) = \mathsf{pw}(B_1) \cdots \mathsf{pw}(B_k)$. Consequently, $\mathsf{Dom}(\mathfrak{det}[\Sigma, w]) \supseteq \mathcal{B}(\Sigma, w)^{\circledast}$. Moreover, by the uniqueness of the string $D' = \mathsf{pw}(D)$ with $(D, D') \in \mathfrak{det}[\Sigma, w]$ for each $D \in \mathcal{B}(\Sigma, w)^+$, we obtain that $\mathfrak{det}[\Sigma, w]$ is a functional transduction.

Now, let $D = B_1 \cdots B_k \in \mathcal{B}(\Sigma, w)^{\circ k}$ for some $k \in \mathbb{N}_+$. Since Ω is a bijection, for each $i \in [k-1]$, $\ell(\mathsf{pw}(B_{i+1})) = r(\mathsf{pw}(B_i))$ if and only if $\ell(B_{i+1}) = r(B_i)$. Furthermore, $\iota(\mathsf{pw}(B_i)) = \iota(B_i)$ and $\phi(\mathsf{pw}(B_i)) = \phi(B_i)$ for each $i \in [k]$. Thus, owing to fact that $D \in \mathcal{B}(\Sigma, w)^{\circ k}$, $\mathsf{pw}(D) = \mathsf{pw}(B_1) \cdots \mathsf{pw}(B_k) \in \mathcal{B}(\Sigma, 2^w)^{\circ k}$. More specifically, $\mathsf{pw}(D)$ is a deterministic, complete ODD in $\widehat{\mathcal{B}}(\Sigma, 2^w)^{\circ k}$. Indeed, this follows from the fact that $\mathsf{pw}(B_i)$ is a deterministic, complete (Σ, w)-layer for each $i \in [k]$. Thus, it just remains to prove that $\mathcal{L}(\mathsf{pw}(D)) = \mathcal{L}(D)$. Let $s = \sigma_1 \cdots \sigma_k$ be a string in Σ^k.

First, suppose that $s \in \mathcal{L}(D)$. Then, there exists an accepting sequence

$$\langle (\mathfrak{p}_1, \sigma_1, \mathfrak{q}_1), \ldots, (\mathfrak{p}_k, \sigma_k, \mathfrak{q}_k) \rangle$$

for s in D. Let $X_0 = I(B_1)$ and, for each $i \in [\![k]\!]$, let $X_{i+1} = \mathbf{N}(B_{i+1} X_i \{\sigma_{i+1}\})$. Note that $X_i \subseteq \ell(B_{i+1})$ for each $i \in [\![k]\!]$. Furthermore, for each $i \in [k]$, we have that $\mathfrak{q}_i \in X_i$, i.e. $\mathfrak{q}_i \in \mathbf{N}(B_i X_{i-1} \{\sigma_i\})$, otherwise $(\mathfrak{p}_i, \sigma_i, \mathfrak{q}_i) \notin T(B_i)$. Therefore,

$$\langle (\Omega(X_0), \sigma_1, \Omega(X_1)), \ldots, (\Omega(X_{k-1}), \sigma_k, \Omega(X_k)) \rangle$$

is an accepting sequence for s in $\mathsf{pw}(D)$, and we obtain that $s \in \mathcal{L}(\mathsf{pw}(D))$.

Conversely, suppose that $s \in \mathcal{L}(\mathsf{pw}(D))$. Then, there exists an accepting sequence

$$\langle (\Omega(X_0), \sigma_1, \Omega(X_1)), \ldots, (\Omega(X_{k-1}), \sigma_k, \Omega(X_k)) \rangle$$

for s in $\mathsf{pw}(D)$, where $X_0 = I(B_1)$ and $X_{i+1} = \mathbf{N}(B_{i+1}, X_i, \{\sigma_{i+1}\})$ for each $i \in [\![k]\!]$. Thus, let $\mathfrak{p}_k \in X_{k-1}$ and $\mathfrak{q}_k \in X_k$ such that $(\mathfrak{p}_k, \sigma_k, \mathfrak{q}_k) \in T(B_k)$. Moreover, for each $i \in [k-1]$, let $\mathfrak{p}_i \in X_{i-1}$ and $\mathfrak{q}_i \in X_i$ such that $\mathfrak{q}_i = \mathfrak{p}_{i+1}$ and $(\mathfrak{p}_i, \sigma_i, \mathfrak{q}_i) \in T(B_i)$. We note that for each $i \in [\![k]\!]$, there exist left states and right states $\mathfrak{p}_{i+1}$ and $\mathfrak{q}_{i+1}$ as described above, otherwise $(\Omega(X_i), \sigma_{i+1}, \Omega(X_{i+1}))$ would not be a transition in $T(\mathsf{pw}(B_{i+1}))$. Therefore,

$$\langle (\mathfrak{p}_1, \sigma_1, \mathfrak{q}_1), \ldots, (\mathfrak{p}_k, \sigma_k, \mathfrak{q}_k) \rangle$$

is an accepting sequence for s in D, and $s \in \mathcal{L}(D)$. Finally, the fact that $\mathfrak{det}[\Sigma, w]$ is 2-regular follows from the fact that $\mathfrak{det}[\Sigma, w] \doteq \mathfrak{mm}[\mathsf{pw}[\Sigma, w]]$ is an instantiation of a multimap transduction and that multimap transductions are 2-regular (Proposition 25.(1)). $\qquad \square$

6.3 Reachability Transduction

In this subsection, we define the *reachability transduction*, which intuitively simulates the process of eliminating unreachable states from the frontiers of each layer of

an ODD. It is worth noting that unlike the determinization transduction, that can be defined using a map that acts layerwisely, the reachability transduction will require the use of a compatibility transduction. The issue is that reachability of a given state q in a given B belonging to a given ODD D is a property that depends on which layers have been read before B. To circumvent this issue, the action of the reachability transduction on a ODD D can be described in three intuitive steps. First, we use a multimap transduction to expand each layer of the ODD into a set of *annotated* layers. Each annotation splits states of a layer into two classes: those that are deemed to be useful, and those that should be deleted. Subsequently, we use a compatibility transduction to ensure that only sequences of annotated layers with compatible annotations are considered to be legal. The crucial observation is that each ODD D has a unique annotated version where each two adjacent annotated layers are compatible with each other. Finally, we apply a mapping that sends each annotated layer to the layer obtained by deleting the states that have been marked for deletion. The resulting ODD is then the unique ODD obtained from D by eliminating unreachable states.

Let Σ be an alphabet, $w \in \mathbb{N}_+$ and $B \in \widehat{\mathcal{B}}(\Sigma, w)$. A *reachability annotation* for B is a pair (ϑ, η) of functions $\vartheta \colon \ell(B) \to \{0, 1\}$ and $\eta \colon r(B) \to \{0, 1\}$ that satisfies the following conditions:

1. if $\iota(B) = 1$, then, for each left state $\mathfrak{p} \in \ell(B)$, $\vartheta(\mathfrak{p}) = 1$ if and only if $\mathfrak{p} \in I(B)$;
2. for each right state $\mathfrak{q} \in r(B)$, $\eta(\mathfrak{q}) = 1$ if and only if there exists $\mathfrak{p} \in \ell(B)$ and $\sigma \in \Sigma$ such that $\vartheta(\mathfrak{p}) = 1$ and $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B)$.

Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, and let $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$. A *reachability annotation* for D is a sequence $\langle (\vartheta_1, \eta_1), \ldots, (\vartheta_k, \eta_k) \rangle$ that satisfies the following conditions:

1. for each $i \in [k]$, (ϑ_i, η_i) is a reachability annotation for B_i;
2. for each $i \in [k-1]$, $\eta_i = \vartheta_{i+1}$.

Proposition 28 *Let Σ be an alphabet and $w \in \mathbb{N}_+$. Every ODD $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ admits a unique reachability annotation.*

Proof First, we observe that for each layer $B \in \widehat{\mathcal{B}}(\Sigma, w)$ and each function $\vartheta \colon \ell(B) \to \{0, 1\}$, there exists exactly one function $\eta \colon r(B) \to \{0, 1\}$ such that (ϑ, η) is a reachability annotation for B.

Let $k \in \mathbb{N}_+$ and $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$, such that $\ell(B_{i+1}) = r(B_i)$ for each $i \in [k-1]$, and $\iota(B_1) = 1$ and $\iota(B_i) = 0$ for each $i \in \{2, \ldots, k\}$. Based on the previous observation, we prove by induction on k that the following statement holds: there exists a unique sequence $\langle (\vartheta_1, \eta_1), \ldots, (\vartheta_k, \eta_k) \rangle$ such that $\vartheta_i = \eta_{i+1}$ for each $i \in [k-1]$, and (ϑ_i, η_i) is a reachability annotation for B_i for each $i \in [k]$.

Base case. Consider $k = 1$. Since $\iota(B_k) = 1$, the function $\vartheta_k \colon \ell(B_k) \to \{0, 1\}$ is uniquely determined. Indeed, by definition, for each left state $\mathfrak{p} \in \ell(B_k)$, $\vartheta_k(\mathfrak{p}) = 1$ if $\mathfrak{p} \in I(B_k)$, and $\vartheta_k(\mathfrak{p}) = 0$ otherwise. Thus, there exists a unique sequence $\langle (\vartheta_k, \eta_k) \rangle$ such that (ϑ_k, η_k) is a reachability annotation for B_k.

Inductive step. Consider $k > 1$. Let $D' = B_1 \cdots B_{k-1}$ be the string obtained from $D = B_1 \cdots B_k$ by removing the layer B_k. It follows from the inductive hypothesis that there exists a unique sequence $\langle (\vartheta_1, \eta_1), \ldots, (\vartheta_{k-1}, \eta_{k-1}) \rangle$ such that $\vartheta_i = \eta_{i+1}$ for each $i \in [k-2]$, and (ϑ_i, η_i) is a reachability annotation for B_i for each $i \in [k-1]$. In particular, we note that the function η_{k-1} is uniquely determined. Furthermore, based on the previous observation, for each function $\vartheta_k \colon \ell(B_k) \to \{0, 1\}$, there exists a unique function $\eta_k \colon r(B_k) \to \{0, 1\}$ such that (ϑ_k, η_k) is a reachability annotation for B_k. Therefore, since ϑ_k must be equal to η_{k-1}, there exists a unique sequence $\langle (\vartheta_1, \eta_1), \ldots, (\vartheta_k, \eta_k) \rangle$ such that $\vartheta_i = \eta_{i+1}$ for each $i \in [k-1]$ and (ϑ_i, η_i) is a reachability annotation for B_i for each $i \in [k]$. $\qquad \square$

Let Σ be an alphabet and $w \in \mathbb{N}_+$. We denote by $\mathcal{R}(\Sigma, w)$ the set consisting of all triples (B, ϑ, η) such that B is a layer in $\widehat{\mathcal{B}}(\Sigma, w)$ and (ϑ, η) is a reachability annotation for B. Additionally, we denote by $\xi[\Sigma, w] \colon \mathcal{R}(\Sigma, w) \to \widehat{\mathcal{B}}(\Sigma, w)$ the map that sends each triple $(B, \vartheta, \eta) \in \mathcal{R}(\Sigma, w)$ to the layer $\xi[\Sigma, w](B, \vartheta, \eta) \in \widehat{\mathcal{B}}(\Sigma, w)$ obtained from B by removing the left states $\mathfrak{p} \in \ell(B)$ with $\vartheta(\mathfrak{p}) = 0$, the right states $\mathfrak{q} \in r(B)$ with $\eta(\mathfrak{q}) = 0$, and the transitions incident with such left and right states. More formally, for each triple $(B, \vartheta, \eta) \in \mathcal{R}(\Sigma, w)$, we let $\xi[\Sigma, w](B, \vartheta, \eta) = B'$, where B' is the layer belonging to $\widehat{\mathcal{B}}(\Sigma, w)$ defined as follows:

- $\ell(B') \doteq \ell(B) \setminus \{\mathfrak{p} \colon \vartheta(\mathfrak{p}) = 0\}$;
- $r(B') \doteq r(B) \setminus \{\mathfrak{q} \colon \eta(\mathfrak{q}) = 0\}$;
- $T(B') \doteq T(B) \setminus \{(\mathfrak{p}, \sigma, \mathfrak{q}) \colon \vartheta(\mathfrak{p}) = 0\}$;
- $\iota(B') \doteq \iota(B); \phi(B') \doteq \phi(B)$;
- $I(B') \doteq I(B); F(B') \doteq r(B') \cap F(B)$.

We let $\xi[\Sigma, w] \colon \widehat{\mathcal{B}}(\Sigma, w)^{\circledast} \to \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ be the map that for each $k \in \mathbb{N}_+$, sends each ODD $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$ to the ODD

$$\xi[\Sigma, w](D) \doteq \xi[\Sigma, w](B_1, \vartheta_1, \eta_1) \cdots \xi[\Sigma, w](B_k, \vartheta_k, \eta_k) \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k},$$

where $\langle (\vartheta_1, \eta_1), \ldots, (\vartheta_k, \eta_k) \rangle$ denotes the unique reachability annotation for D (see Proposition 28).

Proposition 29 *Let Σ be an alphabet, $w \in \mathbb{N}_+$ and $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$. Then, $\xi[\Sigma, w](D)$ is a reachable ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ such that $\mathcal{L}(\xi[\Sigma, w](D)) = \mathcal{L}(D)$.*

Proof Assume that $D = B_1 \cdots B_k$ and $\xi[\Sigma, w](D) = B'_1 \cdots B'_k$, for some $k \in \mathbb{N}_+$, where $B'_i = \xi[\Sigma, w](B_i, \vartheta_i, \eta_i)$ for each $i \in [k]$ and $\langle (\vartheta_1, \eta_1), \ldots, (\vartheta_k, \eta_k) \rangle$ is the unique reachability annotation of D. First, we prove that $\xi[\Sigma, w](D)$ is reachable. Note that for each $i \in [k]$ and each $\mathfrak{q} \in r(B_i)$,

$$\mathfrak{q} \in r(B'_i) \Leftrightarrow \exists \mathfrak{p} \in \ell(B_i), \text{ with } \vartheta_i(\mathfrak{p}) = 1, \text{ and } \exists \sigma \in \Sigma \text{ such that } (\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B_i)$$
$$\Leftrightarrow \exists \mathfrak{p} \in \ell(B'_i) \text{ and } \exists \sigma \in \Sigma \text{ such that } (\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B'_i).$$

This implies that for each $i \in [k]$, B'_i is a reachable layer since $r(B'_i) \subseteq r(B_i)$. Therefore, $\xi[\Sigma, w](D)$ is a reachable ODD. Now, we prove that $\mathcal{L}(\xi[\Sigma, w](D)) = \mathcal{L}(D)$. It is immediate from the definition of $\xi[\Sigma, w](D)$ that $\mathcal{L}(\xi[\Sigma, w](D)) \subseteq$

$\mathcal{L}(D)$. On the other hand, it is not hard to check that for each string $s \in \Sigma^k$, every accepting sequence for s in D is also an accepting sequence for s in $\xi[\Sigma w](D)$. Consequently, $\mathcal{L}(\xi[\Sigma, w](D)) \supseteq \mathcal{L}(D)$.

To prove that $\xi[\Sigma, w]$ preserves determinism, it is enough to note that $T(B'_i) \subseteq T(B_i)$ for each $i \in [k]$. As a result, since D is deterministic, so is $\xi[\Sigma, w](D)$. Finally, since D is complete, by definition, for each $i \in [k]$ and each $\mathfrak{p} \in \ell(B_i) \cap \ell(B'_i)$, there exists a symbol σ and a right state $\mathfrak{q} \in r(B_i)$ such that $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B_i)$. This implies that for each $i \in [k]$ and each $\mathfrak{p} \in \ell(B'_i)$, there exists a symbol σ and a right state $\mathfrak{q} \in r(B'_i)$ such that $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B'_i)$. Therefore, $\xi[\Sigma, w](D)$ is also complete. $\qquad\square$

For each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, we let $\mathrm{RR}[\Sigma, w] \subseteq \widehat{\mathcal{B}}(\Sigma, w) \times \mathcal{R}(\Sigma, w)$ and $\mathrm{RC}[\Sigma, w] \subseteq \mathcal{R}(\Sigma, w) \times \mathcal{R}(\Sigma, w)$ be the relations defined as follows.

$$\mathrm{RR}[\Sigma, w] \doteq \{(B, (B, \vartheta, \eta)) \colon (B, \vartheta, \eta) \in \mathcal{R}(\Sigma, w)\}.$$

$$\begin{aligned} \mathrm{RC}[\Sigma, w] \doteq \{((B, \vartheta, \eta), (B', \vartheta', \eta')) \colon (B, \vartheta, \eta), (B', \vartheta', \eta') \in \mathcal{R}(\Sigma, w), r(B) \\ = \ell(B'), \ \eta = \vartheta'\}. \end{aligned}$$

Now, for each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, we define $\mathfrak{rea}[\Sigma, w]$ as the $(\widehat{\mathcal{B}}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-transduction

$$\mathfrak{rea}[\Sigma, w] \doteq \mathfrak{mm}[\mathrm{RR}[\Sigma, w]] \circ \mathfrak{cp}[\mathrm{RC}[\Sigma, w]] \circ \mathfrak{mm}[\xi[\Sigma w]].$$

The next lemma states that $\mathfrak{rea}[\Sigma, w]$ is a transduction that sends each ODD $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ to a *reachable* ODD $D' \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ that has the same language as D, and that preserves the determinism and completeness properties.

Lemma 30 (Reachability Transduction) *For each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, the following statements hold.*

1. $\mathfrak{rea}[\Sigma, w]$ *is functional.*
2. $\mathsf{Dom}(\mathfrak{rea}[\Sigma, w]) \supseteq \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$.
3. *For each pair* $(D, D') \in \mathfrak{rea}[\Sigma, w]$, $\mathcal{L}(D') = \mathcal{L}(D)$ *and D' is reachable.*
4. $\mathfrak{rea}[\Sigma, w]$ *is* $2^{O(|\Sigma| \cdot w \cdot \log w)}$-*regular.*

Proof We note that $\mathfrak{rea}[\Sigma, w]$ consists of all pairs (D, D') of non-empty strings over the alphabet $\widehat{\mathcal{B}}(\Sigma, w)$ satisfying the conditions that $|D| = |D'|$ and that, if $D = B_1 \cdots B_k$ and $D' = B'_1 \cdots B'_k$ for some $k \in \mathbb{N}_+$, then there exists a reachability annotation (ϑ_i, η_i) for the layer B_i such that $B'_i = \xi[\Sigma, w](B_i, \vartheta_i, \eta_i)$ for each $i \in [k]$, and $r(B_j) = \ell(B_{j+1})$ and $\eta_j = \vartheta_{j+1}$ for each $j \in [k-1]$. Additionally, based on Proposition 28, each (Σ, w)-ODD admits a unique reachability annotation. As a result, we obtain that $\mathsf{Dom}(\mathfrak{rea}[\Sigma, w]) \supseteq \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$. Moreover, $D' = \xi[\Sigma, w](D)$; thus, by the uniqueness of $\xi[\Sigma, w](D)$, the transduction $\mathfrak{rea}[\Sigma, w]$ is functional. Finally, it follows from Proposition 29 that for each pair $(D, D') \in \mathfrak{rea}[\Sigma, w]$, $D' = \xi[\Sigma, w](D)$ is a reachable ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ that has the same language as D.

The fact that $\mathfrak{rea}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular follows from Proposition 5.(2) together with the fact that the multimap transductions $\mathfrak{mm}[RR[\Sigma, w]]$ and $\mathfrak{mm}[\xi[\Sigma, w]]$ are 2-regular (Proposition 25.(1)), and that the transduction $\mathfrak{cp}[RC[\Sigma, w]]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular (Proposition 25.(2)), given that $RC[\Sigma, w] \subseteq \mathcal{R}(\Sigma, w) \times \mathcal{R}(\Sigma, w)$ and that $|\mathcal{R}(\Sigma, w)| = 2^{O(|\Sigma| \cdot w \cdot \log w)}$. □

6.4 Merging Transduction

In this subsection, we define the *merging transduction*, which intuitively simulates the process of merging equivalent states in the frontiers of each layer of an ODD D. As in the case of the reachability transduction, the merging transduction will be defined as the composition of three elementary transductions. First, we use a multimap transduction to expand each layer of the ODD into a set of *annotated* layers. Each annotation partitions each frontier of the layer into cells containing states that are deemed to be equivalent. Subsequently, we use a compatibility transduction to ensure that only sequences of annotated layers with compatible annotations are considered to be legal. As in the case of the reachability transduction, it is possible to show that each ODD D has a unique annotated version where each two adjacent annotated layers are compatible with each other. Finally, we apply a mapping that sends each annotated layer to the layer obtained by merging all states in each cell of each partition to the smallest state in the cell. The result is a minimized ODD with same language as D.

Let Σ be an alphabet, $w \in \mathbb{N}_+$, $B \in \widehat{\mathcal{B}}(\Sigma, w)$ and ν be a partition of $r(B)$. Two (not necessarily distinct) left states $\mathfrak{p}, \mathfrak{p}' \in \ell(B)$ are said to be ν-*equivalent* if, for each symbol $\sigma \in \Sigma$, there exists a right state $\mathfrak{q} \in r(B)$ such that $(\mathfrak{p}, \sigma, \mathfrak{q})$ is a transition in $T(B)$ if and only if there exists a right state $\mathfrak{q}' \in r(B)$ such that $(\mathfrak{p}', \sigma, \mathfrak{q}')$ is a transition in $T(B)$, and $\mathfrak{q}$ and $\mathfrak{q}'$ belong to the same cell of ν. We remark that each left state $\mathfrak{p}$ is trivially ν-equivalent to itself.

A *merging annotation* for B is a pair (μ, ν), where μ is a partition of $\ell(B)$ and ν is a partition of $r(B)$, that satisfies the following two conditions:

1. if $\phi(B) = 1$, then $\nu = \{r(B) \setminus F(B), F(B)\}$ whenever $r(B) \setminus F(B) \neq \emptyset$ and $F(B) \neq \emptyset$, and $\nu = \{r(B)\}$ whenever $r(B) \setminus F(B) = \emptyset$ or $F(B) = \emptyset$;
2. for each two left states $\mathfrak{p}, \mathfrak{p}' \in \ell(B)$, $\mathfrak{p}$ and $\mathfrak{p}'$ belong to the same cell of μ if and only if $\mathfrak{p}$ and $\mathfrak{p}'$ are ν-equivalent.

Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, and let $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$. A *merging annotation* for D is a sequence $\langle (\mu_1, \nu_1) \cdots (\mu_k, \nu_k) \rangle$ that satisfies the following conditions:

1. for each $i \in [k]$, (μ_i, ν_i) is a merging annotation for B_i;
2. for each $i \in [k-1]$, $\nu_i = \mu_{i+1}$.

Proposition 31 *Let Σ be an alphabet and $w \in \mathbb{N}_+$. Every deterministic, complete (Σ, w)-ODD admits a unique merging annotation.*

Proof First, we claim that for each layer $B \in \widehat{\mathcal{B}}(\Sigma, w)$ and each partition ν of $r(B)$, there exists a unique partition μ of $\ell(B)$ such that (μ, ν) is a merging annotation for B. Indeed, any two left states $\mathfrak{p}, \mathfrak{p}' \in \ell(B)$ belong to the same cell of μ if and only if they are ν-equivalent. Thus, the partition μ is uniquely defined as the set of all maximal subsets $X \subseteq \ell(B)$ of pairwise ν-equivalent left states.

Let $k \in \mathbb{N}_+$ and $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$, be such that $\ell(B_{i+1}) = r(B_i)$ for each $i \in [k-1]$, $\phi(B_i) = 0$ for each $i \in [k-1]$ and $\phi(B_k) = 1$. Based on the previous claim, we prove by induction on j that the following statement holds for each $j \in \{0, \ldots, k-1\}$: there exists a unique sequence $\langle (\mu_{k-j}, \nu_{k-j}) \cdots (\mu_k, \nu_k) \rangle$ such that (μ_i, ν_i) is a merging annotation for B_i for each $i \in \{j, \ldots, k\}$, and $\nu_i = \mu_{i+1}$ for each $i \in \{k-j, \ldots, k-1\}$. In particular, this implies that the ODD D admits a unique merging annotation $\langle (\mu_1, \nu_1) \cdots (\mu_k, \nu_k) \rangle$.

Base case.   Consider $j = 0$. Then $k - j = k$. Since $\phi(B_k) = 1$, the partition ν_k is uniquely determined. Indeed, $\nu_k = \{F(B_k), r(B_k) \setminus F(B_k)\}$ if both $r(B_k) \setminus F(B_k) \neq \emptyset$ and $F(B_k) \neq \emptyset$, and $\nu_k = \{r(B_k)\}$ otherwise. Thus, there exists a unique sequence $\langle (\mu_k, \nu_k) \rangle$ such that (μ_k, ν_k) is a merging annotation for B_k.

Inductive step.   Consider $j \in \{1, \ldots, k-1\}$. We show that there is a unique sequence
$$\langle (\mu_{k-j}, \nu_{k-j}) \cdots (\mu_k, \nu_k) \rangle$$
such that (μ_i, ν_i) is a merging annotation for B_i for each $i \in \{k-j, \ldots, k\}$, and $\nu_i = \mu_{i+1}$ for each $i \in \{k-j, \ldots, k-1\}$. It follows from the inductive hypothesis that there exists a unique sequence $\langle (\mu_{k-(j-1)}, \nu_{k-(j-1)}) \cdots (\mu_k, \nu_k) \rangle$ such that (μ_i, ν_i) is a merging annotation for B_i for each $i \in \{k-(j-1), \ldots, k\}$, and $\nu_i = \mu_{i+1}$ for each $i \in \{k-(j-1), \ldots, k-1\}$. Now, let (μ_{k-j}, ν_{k-j}) be the merging annotation of B_{k-j} with the property that $\nu_{k-j} = \mu_{k-(j-1)}$. Such a merging annotation exists (since $r(B_{k-j}) = \ell(B_{k-(j-1)})$) and is unique since μ_{k-j} is uniquely determined by ν_{k-j}. This concludes the proof of the inductive step, and therefore of the proposition. □

Let Σ be an alphabet, $w, k \in \mathbb{N}_+$ and $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$. For each $i \in [k]$, we say that a string $s = \sigma_1 \ldots \sigma_k$ is *accepted by D from a left state* $\mathfrak{p} \in \ell(B_i)$ if there exists a sequence $\langle (\mathfrak{p}_i, \sigma_i, \mathfrak{q}_i), \ldots, (\mathfrak{p}_k, \sigma_k, \mathfrak{q}_k) \rangle$ of transitions such that $\mathfrak{p}_i = \mathfrak{p}$, $\mathfrak{q}_k \in F(B_k)$ and, for each $j \in \{i, \ldots, k\}$, $(\mathfrak{p}_j, \sigma_j, \mathfrak{q}_j) \in T(B_j)$. For each $i \in [k]$ and each left state $\mathfrak{p} \in \ell(B_i)$, we let
$$\mathcal{L}(D, i, \mathfrak{p}) \doteq \{s \in \Sigma^{k-i+1} : s \text{ is accepted by } D \text{ from } \mathfrak{p}\}.$$

Proposition 32 *Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, $D = B_1 \cdots B_k$ be a deterministic, complete ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$, and let $\langle (\mu_1, \nu_1) \cdots (\mu_k, \nu_k) \rangle$ be the unique merging annotation for D. For each $i \in [k]$ and each two left states $\mathfrak{p}, \mathfrak{p}' \in \ell(B_i)$, $\mathfrak{p}$ and $\mathfrak{p}'$ belong to the same cell of μ_i if and only if $\mathcal{L}(D, i, \mathfrak{p}) = \mathcal{L}(D, i, \mathfrak{p}')$.*

Proof The proof is by induction on $k - i$. *Base case.* Consider $k - i = 0$. Then $i = k$. By definition, two left states $\mathfrak{p}, \mathfrak{p}' \in \ell(B_k)$ belong to the same cell of μ_k if and only

if $\mathfrak{p}$ and $\mathfrak{p}'$ are ν_k-equivalent. In other words, $\mathfrak{p}$ and $\mathfrak{p}'$ belong to the same cell of μ_k if and only if, for each symbol $\sigma \in \Sigma$, there exists a final state $\mathfrak{q} \in F(B_k)$ such that $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B_k)$ if and only if there exists a final state $\mathfrak{q}' \in F(B_k)$ (possibly $\mathfrak{q}' = \mathfrak{q}$) such that $(\mathfrak{p}', \sigma, \mathfrak{q}') \in T(B_k)$. Consequently, $\mathfrak{p}$ and $\mathfrak{p}'$ belong to the same cell of μ_k if and only if $\mathcal{L}(D, i, \mathfrak{p}) = \mathcal{L}(D, i, \mathfrak{p}')$.

Inductive step. Consider $k - i > 0$. Since $r(B_i) = \ell(B_{i+1})$ and $\nu_i = \mu_{i+1}$, it follows from the inductive hypothesis that any two right states $\mathfrak{q}, \mathfrak{q}' \in r(B_i)$ belong to the same cell of ν_i if and only if $\mathcal{L}(D, i + 1, \mathfrak{q}) = \mathcal{L}(D, i + 1, \mathfrak{q}')$. Moreover, note that for each left state $\mathfrak{p} \in \ell(B_i)$,

$$\mathcal{L}(D, i, \mathfrak{p}) = \bigcup_{\mathfrak{q} \in r(B_i)} \{\sigma u \in \Sigma^{k-i+1} : (\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B_i), u \in \mathcal{L}(D, i + 1, \mathfrak{q})\}. \quad (3)$$

Let $\mathfrak{p}, \mathfrak{p}' \in \ell(B_i)$. We will prove that $\mathfrak{p}, \mathfrak{p}'$ belong to the same cell of μ_i if and only if $\mathcal{L}(D, i, \mathfrak{p}) = \mathcal{L}(D, i, \mathfrak{p}')$. The proof is split in two parts.

First, suppose that $\mathfrak{p}$ and $\mathfrak{p}'$ belong to the same cell of μ_i. Then $\mathfrak{p}$ and $\mathfrak{p}'$ are ν_i equivalent. In other words, for each symbol $\sigma \in \Sigma$, there exists $\mathfrak{q} \in r(B_i)$ such that $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B_i)$ if and only if there exists $\mathfrak{q}' \in r(B_i)$ such that $(\mathfrak{p}', \sigma, \mathfrak{q}') \in T(B_i)$ and $\mathfrak{q}_i$ and $\mathfrak{q}'_i$ belong to the same cell of ν_i. Using the induction hypothesis, we have that for each pair $\mathfrak{q}, \mathfrak{q}'$ belonging to the same cell of ν_i, $\mathcal{L}(D, i + 1, \mathfrak{q}) = \mathcal{L}(D, i + 1, \mathfrak{q}')$. Therefore, using (3), that $\mathcal{L}(D, i, \mathfrak{p}) = \mathcal{L}(D, i, \mathfrak{p}')$.

Now, in order to prove the converse, suppose for contradiction that $\mathfrak{p}$ and $\mathfrak{p}'$ do not belong to the same cell of μ_i and that $\mathcal{L}(D, i, \mathfrak{p}) = \mathcal{L}(D, i, \mathfrak{p}')$. Since B_i is a deterministic, complete layer, for each symbol $\sigma \in \Sigma$, there exists exactly one right state $\mathfrak{q} \in r(B_i)$ such that $(\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B_i)$. Similarly, for each symbol $\sigma \in \Sigma$, there exists exactly one right state $\mathfrak{q}' \in r(B_i)$ such that $(\mathfrak{p}', \sigma, \mathfrak{q}') \in T(B_i)$. Consequently, for some symbol $\sigma \in \Sigma$, the right states $\mathfrak{q}$ and $\mathfrak{q}'$ associated with σ, and $\mathfrak{p}$ and $\mathfrak{p}'$, respectively, belong to distinct cells of ν_i. Then, it follows from the induction hypothesis that $\mathcal{L}(D, i + 1, \mathfrak{q}) \neq \mathcal{L}(D, i + 1, \mathfrak{q}')$. Assume without loss of generality that $\mathcal{L}(D, i + 1, \mathfrak{q}) \setminus \mathcal{L}(D, i + 1, \mathfrak{q}') \neq \emptyset$, and let $u \in \mathcal{L}(D, i + 1, \mathfrak{q}) \setminus \mathcal{L}(D, i + 1, \mathfrak{q}')$. Based on (3), we have that $\sigma u \in \mathcal{L}(D, i, \mathfrak{p})$ but $\sigma u \notin \mathcal{L}(D, i, \mathfrak{p}')$. This implies that $\mathcal{L}(D, i, \mathfrak{p}) \neq \mathcal{L}(D, i, \mathfrak{p}')$, contradicting our initial supposition. $\qquad \square$

Let Σ be an alphabet and $w \in \mathbb{N}_+$. We denote by $\mathcal{M}(\Sigma, w)$ the set consisting of all triples (B, μ, ν) such that B is a deterministic, complete layer in $\widehat{\mathcal{B}}(\Sigma, w)$, and (μ, ν) is a merging annotation for B. Additionally, we denote by $\zeta[\Sigma, w] \colon \mathcal{M}(\Sigma, w) \to \widehat{\mathcal{B}}(\Sigma, w)$ the map that sends each triple $(B, \mu, \nu) \in \mathcal{M}(\Sigma, w)$ to the layer $\zeta[\Sigma, w](B, \mu, \nu) \in \widehat{\mathcal{B}}(\Sigma, w)$ obtained from B by identifying, for each $X \in \mu \cup \nu$, all states belonging to X with the smallest state that belongs to X. More formally, for each triple $(B, \mu, \nu) \in \mathcal{M}(\Sigma, w)$, we let $\zeta[\Sigma, w](B, \mu, \nu) = B'$, where B' is the deterministic, complete layer belonging to $\widehat{\mathcal{B}}(\Sigma, w)$ defined as follows:

- $\ell(B') \doteq \bigcup_{X \in \mu} \{\min X\}$; $r(B') \doteq \bigcup_{X' \in \nu} \{\min X'\}$;
- $T(B') \doteq \bigcup_{X \in \mu, X' \in \nu} \{(\min X, \sigma, \min X') \colon \exists \mathfrak{p} \in X, \exists \mathfrak{q} \in X', (\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B)\}$;
- $\iota(B') \doteq \iota(B)$; $\phi(B') \doteq \phi(B)$;
- $I(B') \doteq I(B)$; $F(B') \doteq r(B') \cap F(B)$.

Let $\zeta[\Sigma, w]\colon \widehat{\mathcal{B}}(\Sigma, w)^{\circledast} \to \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ be the map that for each $k \in \mathbb{N}_+$, sends each deterministic, complete ODD $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$ to the deterministic, complete ODD

$$\zeta[\Sigma, w](D) \doteq \zeta[\Sigma, w](B_1, \mu_1, \nu_1) \cdots \zeta[\Sigma, w](B_k, \mu_k, \nu_k) \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k},$$

where $\langle (\mu_1, \nu_1), \ldots, (\mu_k, \nu_k) \rangle$ denotes the unique merging annotation for D (see Proposition 31).

Let Σ be an alphabet, $w, k \in \mathbb{N}_+$ and $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$. We recall that since D is a deterministic, complete ODD, we have that for each string $s = \sigma_1 \cdots \sigma_k \in \Sigma^k$, there is a unique valid sequence $\langle (\mathfrak{p}_1, \sigma_1, \mathfrak{q}_1), \ldots, (\mathfrak{p}_k, \sigma_k, \mathfrak{q}_k) \rangle$ for s in D. Thus, for each string $s \in \Sigma^k$ and each $i \in [k]$, we let $\mathfrak{q}_{[D,s,i]} \doteq \mathfrak{q}_i$ denote the unique right state $\mathfrak{q}_i \in r(B_i)$ that belongs to the valid sequence for s in D. Moreover, we let

$$[D, s, i] \doteq \{s' \in \Sigma^k : \mathfrak{q}_{[D,s',i]} = \mathfrak{q}_{[D,s,i]}\}$$

denote the equivalence class of s with respect to D and i.

Proposition 33 *Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$, and let $\mathfrak{q}$ be a right state in $r(B_i)$ such that $\mathfrak{q} = \mathfrak{q}_{[D,s,i]}$ for some string $s \in \Sigma^k$ and some $i \in [k-1]$. For each string $s' = \sigma'_1 \cdots \sigma'_k \in [D, s, i]$ and each string $u \in \Sigma^{k-i}$, we have that $u \in \mathcal{L}(D, i, \mathfrak{q})$ if and only if $\sigma'_1 \cdots \sigma'_i u \in \mathcal{L}(D)$.*

Proof Let $s' = \sigma'_1 \cdots \sigma'_k \in [D, s, i]$ and $u = \tau_{i+1} \cdots \tau_k \in \Sigma^{k-i}$. Also, let

$$\langle (\mathfrak{p}_1, \sigma'_1, \mathfrak{q}_1), \ldots, (\mathfrak{p}_k, \sigma'_k, \mathfrak{q}_k) \rangle$$

be the unique valid sequence for s' in D. We note that $\mathfrak{p}_1 \in I(B_1)$ and $\mathfrak{p}_{i+1} = \mathfrak{q}$. Suppose that $u \in \mathcal{L}(D_i \mathfrak{p}_{i+1})$. By definition, there is a sequence

$$\langle (\mathfrak{p}'_{i+1}, \tau_{i+1}, \mathfrak{q}'_{i+1}), \ldots, (\mathfrak{p}'_k, \tau_k, \mathfrak{q}'_k) \rangle$$

of transitions such that $\mathfrak{p}'_{i+1} = \mathfrak{p}_{i+1}$, $\mathfrak{q}'_k \in F(B_k)$ and, for each $j \in \{i+1, \ldots, k\}$, $(\mathfrak{p}'_j, \sigma'_j, \mathfrak{q}'_j) \in T(B_j)$. Thus, $\langle (\mathfrak{p}_1, \sigma'_1, \mathfrak{q}_1), \ldots, (\mathfrak{p}_i, \sigma'_i, \mathfrak{q}_i), (\mathfrak{p}'_{i+1}, \tau_{i+1}, \mathfrak{q}'_{i+1}), \ldots, (\mathfrak{p}'_k, \tau_k, \mathfrak{q}'_k) \rangle$ is an accepting sequence for the string $\sigma'_1 \cdots \sigma'_i u$ in D, and therefore $\sigma'_1 \cdots \sigma'_i u \in \mathcal{L}(D)$.

Conversely, suppose that $\sigma'_1 \cdots \sigma'_i u \in \mathcal{L}(D)$. Then, there exists a unique accepting sequence

$$\langle (\mathfrak{p}'_1, \sigma'_1, \mathfrak{q}'_1), \ldots, (\mathfrak{p}'_i, \sigma'_i, \mathfrak{q}'_i), (\mathfrak{p}'_{i+1}, \tau_{i+1}, \mathfrak{q}'_{i+1}), \ldots, (\mathfrak{p}'_k, \tau_k, \mathfrak{q}'_k) \rangle$$

for $\sigma'_1 \cdots \sigma'_i u$ in D. By the uniqueness of this sequence, we have that $\mathfrak{p}'_1 = \mathfrak{p}_1$ and $\mathfrak{q}'_j = \mathfrak{q}_j$ for each $j \in [i]$. In particular, $\mathfrak{p}'_{i+1} = \mathfrak{q}'_i = \mathfrak{q}$. Therefore, $u \in \mathcal{L}(D, i, \mathfrak{q})$. □

Proposition 34 *Let Σ be an alphabet, $w, k \in \mathbb{N}_+$, and let D and D' be two deterministic, complete ODDs in $\widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$. If $\mathcal{L}(\zeta[\Sigma, w](D)) = \mathcal{L}(\zeta[\Sigma, w](D'))$, then $[\zeta[\Sigma, w](D), s, i] = [\zeta[\Sigma, w](D'), s, i]$ for each $s \in \Sigma^k$ and each $i \in [k]$.*

Proof For the sake of contradiction, suppose that $\mathcal{L}(\zeta[\Sigma, w](D)) = \mathcal{L}(\zeta[\Sigma, w](D'))$ but, for some string $s = \sigma_1 \cdots \sigma_k \in \Sigma^k$ and some $i \in [k]$, $[\zeta[\Sigma, w](D), s, i] \neq [\zeta[\Sigma, w](D'), s, i]$.

Assume without loss of generality that $[\zeta[\Sigma, w](D), s, i] \setminus [\zeta[\Sigma, w](D'), s, i] \neq \emptyset$. Then, let $s' = \sigma'_1 \cdots \sigma'_k \in [\zeta[\Sigma, w](D), s, i] \setminus [\zeta[\Sigma, w](D'), s, i]$. Consider $\mathfrak{p}'_{i+1} = \mathfrak{q}_{[\zeta[\Sigma, w](D'), s, i]}$ and $\mathfrak{p}''_{i+1} = \mathfrak{q}_{[\zeta[\Sigma, w](D'), s', i]}$. We note that $i < k$, otherwise $\mathcal{L}(\zeta[\Sigma, w](D))$ would be different from $\mathcal{L}(\zeta[\Sigma, w](D'))$. Moreover, since $\mathfrak{p}'_{i+1} \neq \mathfrak{p}''_{i+1}$, we obtain by Proposition 32 that

$$\mathcal{L}(\zeta[\Sigma, w](D'), i+1, \mathfrak{p}'_{i+1}) \neq \mathcal{L}(\zeta[\Sigma, w](D'), i+1, \mathfrak{p}''_{i+1}).$$

Assume without loss of generality $\mathcal{L}(\zeta[\Sigma, w](D'), i+1, \mathfrak{p}'_{i+1}) \setminus \mathcal{L}(\zeta[\Sigma, w](D'), i+1, \mathfrak{p}''_{i+1}) \neq \emptyset$. Let $u \in \mathcal{L}(\zeta[\Sigma, w](D'), i+1, \mathfrak{p}'_{i+1}) \setminus \mathcal{L}(\zeta[\Sigma, w](D'), i+1, \mathfrak{p}''_{i+1})$. Since $\zeta[\Sigma, w](D')$ is deterministic, there exists a unique valid sequence for the string $\sigma'_1 \cdots \sigma'_i u$ in $\zeta[\Sigma, w](D')$, and by definition this sequence must contain the left state $\mathfrak{p}''_{i+1}$. Consequently, it follows from Proposition 33 and from the fact that u is not accepted by $\zeta[\Sigma, w](D')$ from $\mathfrak{p}''_{i+1}$ that

$$\sigma'_1 \cdots \sigma'_i u \notin \mathcal{L}(\zeta[\Sigma, w](D')). \tag{4}$$

On the other hand, u is accepted by $\zeta[\Sigma, w](D')$ from $\mathfrak{p}'_{i+1}$. As a result, we obtain by Proposition 33 that $\sigma_1 \cdots \sigma_i u \in \mathcal{L}(\zeta[\Sigma, w](D'))$. In addition, we have that $\sigma_1 \cdots \sigma_i u \in \mathcal{L}(\zeta[\Sigma, w](D))$ since $\mathcal{L}(\zeta[\Sigma, w](D)) = \mathcal{L}(\zeta[\Sigma, w](D'))$. This further implies that $u \in \mathcal{L}(\zeta[\Sigma, w](D), i+1, \mathfrak{p}_{i+1})$, where $\mathfrak{p}_{i+1}$ denotes $\mathfrak{q}_{[\zeta[\Sigma, w](D), s, i]}$. However, since $s' \in [\zeta[\Sigma, w](D), s, i]$, it follows from Proposition 33 that

$$\sigma'_1 \cdots \sigma'_i u \in \mathcal{L}(\zeta[\Sigma, w](D)), \tag{5}$$

which, along with (4), implies that $\mathcal{L}(\zeta[\Sigma, w](D)) \neq \mathcal{L}(\zeta[\Sigma, w](D'))$. □

Proposition 35 *Let Σ be an alphabet, $w \in \mathbb{N}_+$ and $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$. If D is reachable, then $\zeta[\Sigma, w](D)$ is a minimized ODD such that $\mathcal{L}(\zeta[\Sigma, w](D)) = \mathcal{L}(D)$.*

Proof Assume that $D = B_1 \cdots B_k$, for some $k \in \mathbb{N}_+$, and let $\langle (\mu_1, \nu_1) \cdots (\mu_k, \nu_k) \rangle$ be the unique merging annotation for D. First, we prove that $\mathcal{L}(\zeta[\Sigma, w](D)) = \mathcal{L}(D)$. Let $s = \sigma_1 \cdots \sigma_k \in \Sigma^k$. Suppose that $s \in \mathcal{L}(D)$. Then, there exists an accepting sequence $\langle (\mathfrak{p}_1, \sigma_1, \mathfrak{q}_1), \ldots, (\mathfrak{p}_k, \sigma_k, \mathfrak{q}_k) \rangle$ for s in D. For each $i \in [k]$, let X_i be the unique cell of ν_i that contains $\mathfrak{q}_i$. Then, we have that

$$\langle (\mathfrak{p}_1, \sigma_1, \min X_1), (\min X_1, \sigma_k, \min X_2), \ldots, (\min X_{k-1}, \sigma_k, \min X_k) \rangle$$

is an accepting sequence for s in $\zeta[\Sigma, w](D)$. As a result, we obtain that $\mathcal{L}(\zeta[\Sigma, w](D)) \subseteq \mathcal{L}(D)$. Now, suppose that $s \in \mathcal{L}(\zeta[\Sigma, w](D))$. Then, there exists an accepting sequence $\langle (\mathfrak{p}'_1, \sigma_1, \mathfrak{q}'_1), \ldots, (\mathfrak{p}'_k, \sigma_k, \mathfrak{q}'_k) \rangle$ for s in $\zeta[\Sigma, w](D)$. We note that for each $i \in [k]$, there exists a right state $\mathfrak{q}_i \in r(B_i)$ such that $\mathfrak{q}_i$ and $\mathfrak{q}'_i$ belong to a same cell of ν_i and $(\mathfrak{p}_i, \sigma_i, \mathfrak{q}_i) \in T(B_i)$, where $\mathfrak{p}_1 = \mathfrak{p}'_1$ and $\mathfrak{p}_j = \mathfrak{q}_{j-1}$ for each $j \in \{2, \ldots, k\}$. Thus, there exists an accepting sequence $\langle (\mathfrak{p}_1, \sigma_1, \mathfrak{q}_1), \ldots, (\mathfrak{p}_k, \sigma_k, \mathfrak{q}_k) \rangle$ for s in D. Therefore, $\mathcal{L}(\zeta[\Sigma, w](D)) \supseteq \mathcal{L}(D)$.

Now, we prove that $\zeta[\Sigma, w](D)$ is minimized if D is reachable. Thus, assume that D is reachable. This implies that $\zeta[\Sigma, w](D)$ is also reachable and thus, for each

$i \in [k]$ and each $\mathfrak{q} \in r(\zeta[\Sigma, w](B_i, \mu_i, \nu_i))$, $\mathfrak{q} = \mathfrak{q}_{[\zeta[\Sigma, w](D), s, i]}$ for some $s \in \Sigma^k$. Then, for each $i \in [k]$, let $w_i = |r(\zeta[\Sigma, w](B_i, \mu_i, \nu_i))|$ and let $s_1^i, \ldots, s_{w_i}^i$ be strings such that $[\zeta[\Sigma, w](D), s_j^i, i] \neq [\zeta[\Sigma, w](D), s_{j'}^i, i]$ for each $j \in [w_i]$ and each $j' \in [w_i]$ with $j \neq j'$. Also, let $D' = B'_1 \cdots B'_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast k}$ be a minimized ODD such that $\mathcal{L}(D') = \mathcal{L}(D)$. We note that D' is reachable. Thus, for each $i \in [k]$ and each $\mathfrak{q}' \in r(B'_i)$, $\mathfrak{q}' = \mathfrak{q}_{[D', s', i]}$ for some $s' \in \Sigma^k$. Moreover, we have that $\zeta[\Sigma w](D') = D'$, otherwise D' would not be minimized. Then, for each $i \in [k]$, we let $\pi_i \colon r(\zeta[\Sigma, w](B_i, \mu_i, \nu_i)) \to r(B'_i)$ be the mapping such that for each $j \in [w_i]$, $\pi_i(\mathfrak{q}_{[\zeta[\Sigma, w](D), s_j^i, i]}) = \mathfrak{q}_{[D', s_j^i, i]}$. It follows from Proposition 34 that π_i is a bijection. Consequently, we obtain that $\langle \pi_0, \ldots, \pi_k \rangle$ is a isomorphism between $\zeta[\Sigma, w](D)$ and D', where $\pi_0 \colon \ell(\zeta[\Sigma, w](B_1, \mu_1, \nu_1)) \to \ell(B'_1)$ is the trivial bijection that sends the unique left state in $\ell(\zeta[\Sigma, w](B_1, \mu_1, \nu_1))$ to the unique left state in $\ell(B'_1)$. Therefore, $\zeta[\Sigma, w](D)$ is minimized. $\qquad \square$

For each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, we let $\text{MR}[\Sigma, w] \subseteq \widehat{\mathcal{B}}(\Sigma, w) \times \mathcal{M}(\Sigma, w)$ and $\text{MC}[\Sigma, w] \subseteq \mathcal{M}(\Sigma, w) \times \mathcal{M}(\Sigma, w)$ be the following relations.

$$\text{MR}[\Sigma, w] \doteq \{(B, (B, \mu, \nu)) \colon (B, \mu, \nu) \in \mathcal{M}(\Sigma, w)\} \text{ and}$$

$$\begin{aligned} \text{MC}[\Sigma, w] \doteq \{((B, \mu, \nu), (B', \mu', \nu')) \colon (B, \mu, \nu), (B', \mu', \nu') \in \mathcal{M}(\Sigma, w), r(B) \\ = \ell(B'), \ \nu = \mu'\}. \end{aligned}$$

For each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, we define the $(\widehat{\mathcal{B}}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-transduction $\mathfrak{mer}[\Sigma, w]$ as

$$\mathfrak{mer}[\Sigma, w] \doteq \mathfrak{mm}[\text{MR}[\Sigma, w]] \circ \mathfrak{cp}[\text{MC}[\Sigma, w]] \circ \mathfrak{mm}[\zeta[\Sigma, w]].$$

The next lemma states that $\mathfrak{mer}[\Sigma, w]$ is a transduction that sends each deterministic, complete ODD $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ to a *minimized* deterministic, complete ODD $D' \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ that has the same language as D.

Lemma 36 (Merging Transduction) *For each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, the following statements hold.*

1. $\mathfrak{mer}[\Sigma, w]$ *is functional.*
2. $\text{Dom}(\mathfrak{mer}[\Sigma, w]) \supseteq \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$.
3. *For each pair* $(D, D') \in \mathfrak{mer}[\Sigma, w]$, *if D is a reachable ODD, then* $D' \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$, $\mathcal{L}(D') = \mathcal{L}(D)$ *and D' is minimized.*
4. $\mathfrak{mer}[\Sigma, w]$ *is* $2^{O(|\Sigma| \cdot w \log w)}$-*regular.*

Proof We note that $\mathfrak{mer}[\Sigma, w]$ consists of all pairs (D, D') of non-empty strings over the alphabet $\widehat{\mathcal{B}}(\Sigma, w)$ satisfying the conditions that $|D| = |D'|$ and that, if $D = B_1 \cdots B_k$ and $D' = B'_1 \cdots B'_k$ for some $k \in \mathbb{N}_+$, then there exists a merging annotation (μ_i, ν_i) for the layer B_i such that $B'_i = \zeta[\Sigma, w](B_i, \mu_i, \nu_i)$ for each $i \in [k]$, and $r(B_j) = \ell(B_{j+1})$ and $\nu_j = \mu_{j+1}$ for each $j \in [k-1]$. Additionally, based on Proposition 31, each (Σ, w)-ODD admits a unique merging annotation.

As a result, we obtain that $\mathsf{Dom}(\mathfrak{mer}[\Sigma, w]) \supseteq \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$. Moreover, if $(D, D') \in \mathfrak{mer}[\Sigma, w]$, then $D' = \zeta[\Sigma, w](D)$; thus, by the uniqueness of $\zeta[\Sigma, w](D)$, the transduction $\mathfrak{mer}[\Sigma, w]$ is functional. Finally, it follows from Proposition 35 that for each pair $(D, D') \in \mathfrak{mer}[\Sigma, w]$ such that D is a reachable ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$, we have that $D' = \zeta[\Sigma, w](D)$ is a minimized ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ that has the same language as D.

The fact that $\mathfrak{mer}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular follows from Proposition 5.(2) together with the fact that the multimap transductions $\mathfrak{mm}[\mathrm{MR}[\Sigma, w]]$ and $\mathfrak{mm}[\zeta[\Sigma, w]]$ are 2-regular (Proposition 25.(1)), and that the transduction $\mathfrak{cp}[\mathrm{MC}[\Sigma, w]]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular (Proposition 25.(2)), given that $\mathrm{MC}[\Sigma, w] \subseteq \mathcal{M}(\Sigma, w) \times \mathcal{M}(\Sigma, w)$, and that $|\mathcal{M}(\Sigma, w)| = 2^{O(|\Sigma| \cdot w \cdot \log w)}$. $\qquad \square$

6.5 Normalization Transduction.

In this subsection, we define the *normalization transduction*, which intuitively simulates the process of numbering the states in each frontier of each layer of an ODD D according to their lexicographical order. This transduction can be defined as the composition of three elementary transductions. First, we use a multimap transduction to expand each layer of the ODD into a set of *annotated* layers. Each annotation relabels the left and right frontier vertices of the layer in such a way that the layer itself is normalized. Subsequently, we use a compatibility transduction that defines two consecutive annotated layers to be compatible if and only if the relabeling of the right-frontier of the first is equal to the relabeling of the left frontier of the second. It is possible to show that each reachable ODD D gives rise to a unique sequence of annotated layers where each two consecutive layers are compatible. Finally, we apply a mapping that sends each annotated layer to the layer obtained sending the numbers in the frontiers to their relabeled versions. The resulting ODD is isomorphic to the original one, and therefore besides preserving the language, it also preserves reachability and minimality.

Let Σ be an alphabet, $w \in \mathbb{N}_+$, and let $B \in \widehat{\mathcal{B}}(\Sigma, w)$. For each two bijections $\pi \colon \ell(B) \to [\![|\ell(B)|]\!]$ and $\pi' \colon r(B) \to [\![|r(B)|]\!]$, we denote by $\langle \pi B \pi' \rangle$ the (Σ, w)-layer obtained from B by applying the bijection π to the left frontier of B and by applying the bijection π' to the right frontier of B. More formally, $\langle \pi B \pi' \rangle = B'$ is the (Σ, w)-layer defined as follows:

- $\ell(B') \doteq \{\pi(\mathfrak{p}) \colon \mathfrak{p} \in \ell(B)\};\ r(B') \doteq \{\pi'(\mathfrak{q}) \colon \mathfrak{q} \in r(B)\};$
- $I(B') \doteq \{\pi(\mathfrak{p}) \colon \mathfrak{p} \in I(B)\};\ F(B') \doteq \{\pi'(\mathfrak{q}) \colon \mathfrak{q} \in F(B)\};$
- $T(B') \doteq \{(\pi(\mathfrak{p}), \sigma, \pi'(\mathfrak{q})) \colon (\mathfrak{p}, \sigma, \mathfrak{q}) \in T(B)\};$
- $\iota(B') \doteq \iota(B);\ \phi(B') \doteq \phi(B).$

We note that since $B \in \widehat{\mathcal{B}}(\Sigma, w)$, $\langle \pi B \pi' \rangle$ also belongs to $\widehat{\mathcal{B}}(\Sigma, w)$.

Let Σ be an alphabet, $w \in \mathbb{N}_+$. A *normalizing isomorphism* for a reachable layer $B \in \widehat{\mathcal{B}}(\Sigma, w)$ is a pair (π, π') of bijections $\pi \colon \ell(B) \to [\![|\ell(B)|]\!]$ and $\pi' \colon r(B) \to [\![|r(B)|]\!]$ such that the layer $\langle \pi B \pi' \rangle$ is normalized. Let $k \in \mathbb{N}_+$ and $D = B_1 \cdots B_k$ be a reachable ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$. A *normalizing isomorphism* for D is a sequence $\overline{\pi} =$

$\langle \pi_0, \pi_1, \ldots, \pi_k \rangle$ such that for each $i \in [k]$, (π_{i-1}, π_i) is a normalizing isomorphism for B_i.

Proposition 37 *Let Σ be an alphabet and w. Every reachable ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ admits a unique normalizing isomorphism.*

Proof First, we claim that for each reachable, layer $B \in \widehat{\mathcal{B}}(\Sigma, w)$ and each bijection $\pi : \ell(B) \to [\![|\ell(B)|]\!]$, there exists a unique bijection $\pi' : r(B) \to [\![|r(B)|]\!]$ such that $\langle \pi B \pi' \rangle$ is normalized. Indeed, consider $B' = \langle \pi B \pi'' \rangle$, where $\pi'' : r(B) \to [\![|r(B)|]\!]$ denotes the identity function. Then, let $\pi' : r(B) \to [\![|r(B)|]\!]$ be a bijection such that for each two right states $\mathfrak{q}, \mathfrak{q}' \in r(B)$, we have that $\pi'(\mathfrak{q}) \leq \pi'(\mathfrak{q}')$ if and only if $\chi_{B'}(\mathfrak{q}) \leq \chi_{B'}(\mathfrak{q}')$. One can readily verify that $\langle \pi B \pi' \rangle$ is normalized. Furthermore, since B' is deterministic, $\chi_{B'}$ is an injection from $r(B')$ to $\ell(B') \times \Sigma$, i.e., for each two distinct right states $\mathfrak{q}, \mathfrak{q}' \in r(B')$, we have that either $\chi_{B'}(\mathfrak{q}) < \chi_{B'}(\mathfrak{q}')$ or $\chi_{B'}(\mathfrak{q}') < \chi_{B'}(\mathfrak{q})$. In other words, $\chi_{B'}$ describes a total order on $r(B')$. Therefore, π' is the unique bijection from $r(B)$ to $[\![|r(B)|]\!]$ such that $\langle \pi B \pi' \rangle$ is normalized.

Let $k \in \mathbb{N}_+$ and $D = B_1 \cdots B_k \in \widehat{\mathcal{B}}(\Sigma, w)^{\circ k}$ be an ODD such that B_i is a reachable layer for each $i \in [k]$, $\ell(B_{i+1}) = r(B_i)$ for each $i \in [k-1]$, $\iota(B_1) = 1$ and $\iota(B_i) = 0$ for each $i \in \{2, \ldots, k\}$. Based on the previous claim, we prove by induction on k that the following statement holds: there exists a unique sequence $\langle \pi_0, \pi_1, \ldots, \pi_k \rangle$ such that (1) $\pi_0 : \ell(B_1) \to [\![|\ell(B_1)|]\!]$ is a bijection, (2) $\pi_i : r(B_i) \to [\![|r(B_i)|]\!]$ is a bijection for each $i \in [k]$, and (3) $\langle \pi_{i-1} B_i \pi_i \rangle$ is a normalized layer for each $i \in [k]$.

Base case. Consider $k = 1$. Since B_1 is deterministic, $|\ell(B_1)| = 1$. Thus, the bijection $\pi_0 : \ell(B_1) \to [\![|\ell(B_1)|]\!]$ is trivially uniquely determined. As a result, there exists a unique sequence $\langle \pi_0, \pi_1 \rangle$ satisfying the required conditions (1)–(3).

Inductive step. Consider $k > 1$. Let $D' = B_1 \cdots B_{k-1}$ be the string obtained from $D = B_1 \cdots B_k$ by removing the layer B_k. It follows from the inductive hypothesis that there exists a unique sequence $\langle \pi_0, \pi_1, \ldots, \pi_{k-1} \rangle$ such that $\pi_0 : \ell(B_1) \to [\![|\ell(B_1)|]\!]$ is a bijection, $\pi_i : r(B_i) \to [\![|r(B_i)|]\!]$ is a bijection for each $i \in [k-1]$, and $\langle \pi_{i-1} B_i \pi_i \rangle$ is a normalized layer for each $i \in [k-1]$. In particular, we note that the bijection π_{k-1} is uniquely determined. Furthermore, based on the previous claim, there exists a unique bijection $\pi_k : r(B_k) \to [\![|r(B_k)|]\!]$ such that $\langle \pi_{k-1} B_k \pi_k \rangle$ is normalized. Therefore, there exists a unique sequence $\langle \pi_0, \pi_1, \ldots, \pi_k \rangle$ satisfying the required conditions (1)–(3). $\qquad\square$

Proposition 38 *Let Σ be an alphabet, $w, k \in \mathbb{N}_+$ and $D \in \mathcal{B}(\Sigma, w)^{\circ k}$. If D is a reachable, deterministic ODD and $\overline{\pi} = \langle \pi_0, \pi_1, \ldots, \pi_k \rangle$ is the unique normalizing isomorphism for D, then $D' = \langle \pi_0 B_1 \pi_1 \rangle \cdots \langle \pi_{k-1} B_k \pi_k \rangle$ is a normalized ODD such that $\mathcal{L}(D') = \mathcal{L}(D)$.*

Proof It immediately follows from the definition of normalizing isomorphism that D' is normalized. Finally, we note that $\overline{\pi}$ is an isomorphism from D to D'. Therefore, by Proposition 3, $\mathcal{L}(D') = \mathcal{L}(D)$. $\qquad\square$

For each finite set X, we denote by $\mathbb{S}_X \doteq \{\pi \colon X \to [\![|X|]\!] \colon \pi$ is a bijection$\}$ the set of all bijections from X to $[\![|X|]\!]$. For each alphabet Σ and each $w \in \mathbb{N}_+$ we define the following set.

$$\mathcal{S}[\Sigma, w] = \{(\pi, B, \pi') \colon B \in \widehat{\mathcal{B}}(\Sigma, w), \pi \in \mathbb{S}_{\ell(B)}, \pi' \in \mathbb{S}_{r(B)}, \langle \pi B \pi' \rangle \text{ is normalized}\}.$$

We let $\eta[\Sigma, w] \colon \mathcal{S}[\Sigma, w] \to \widehat{\mathcal{B}}(\Sigma, w)$ be the map that sends each triple $(\pi, B, \pi') \in \mathcal{S}[\Sigma, w]$ to the layer $\langle \pi B \pi' \rangle$. Moreover, we let $\mathrm{NR}[\Sigma, w] \subseteq \widehat{\mathcal{B}}(\Sigma, w) \times \mathcal{S}[\Sigma, w]$ and $\mathrm{NC}[\Sigma, w] \subseteq \mathcal{S}[\Sigma, w] \times \mathcal{S}[\Sigma, w]$ be the following relations.

$$\mathrm{NR}[\Sigma, w] \doteq \{(B, (\pi, B, \pi')) \colon (\pi, B, \pi') \in \mathcal{S}[\Sigma, w]\}.$$

$$\mathrm{NC}[\Sigma, w] \doteq \{((\pi, B, \pi'), (\pi', B', \pi'')) \in \mathcal{S}[\Sigma, w] \times \mathcal{S}[\Sigma, w] \colon r(B) = \ell(B'), \}.$$

Finally, for each alphabet Σ, and each positive integer $w \in \mathbb{N}_+$, we let $\mathfrak{nor}[\Sigma, w]$ be the $(\widehat{\mathcal{B}}(\Sigma, w), \widehat{\mathcal{B}}(\Sigma, w))$-transduction

$$\mathfrak{nor}[\Sigma, w] \doteq \mathfrak{mm}[\mathrm{NR}[\Sigma, w]] \circ \mathfrak{cp}[\mathrm{NC}[\Sigma, w]] \circ \mathfrak{mm}[\eta[\Sigma, w]].$$

The next lemma states that $\mathfrak{nor}$ is a transduction that sends each reachable, deterministic, complete ODD $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ to as *normalized*, deterministic, complete ODD $D' \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ that has the same language as D.

Lemma 39 (Normalization Transduction) *For each alphabet Σ and each positive integer $w \in \mathbb{N}_+$, the following statements hold.*

1. $\mathfrak{nor}[\Sigma, w]$ *is functional.*
2. $\mathsf{Dom}(\mathfrak{nor}[\Sigma, w]) \supseteq \{D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast} \colon D$ *is reachable*$\}$.
3. *For each pair $(D, D') \in \mathfrak{nor}[\Sigma, w]$, if D is reachable then $D' \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$, $\mathcal{L}(D') = \mathcal{L}(D)$ and D' is normalized.*

Proof We note that $\mathfrak{nor}[\Sigma, w]$ consists of all pairs (D, D') of non-empty strings over the alphabet $\widehat{\mathcal{B}}(\Sigma, w)$ satisfying the conditions that $|D| = |D'|$ and that, if $D = B_1 \cdots B_k$ and $D' = B'_1 \cdots B'_k$ for some $k \in \mathbb{N}_+$, then there exists a sequence of permutations $\overline{\pi} = \langle \pi_0, \pi_1, \dots, \pi_k \rangle$ such that for each $i \in [k]$, $(\pi_{i-1}, B_i, \pi_i) \in \mathcal{S}[\Sigma, w]$ and $B'_i = \langle \pi_{i-1} B_i \pi_i \rangle$. Additionally, based on Proposition 37, each reachable, deterministic (Σ, w)-ODD admits a unique normalizing isomorphism. As a result, we obtain that $\mathsf{Dom}(\mathfrak{nor}[\Sigma, w]) \supseteq \{D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast} \colon D$ is reachable$\}$. Moreover, if $(D, D') \in \mathfrak{d}(\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}) \circ \mathfrak{nor}[\Sigma, w]$, then $D' = \langle \pi_0 B_1 \pi_1 \rangle \cdots \langle \pi_{k-1} B_k \pi_k \rangle$, where $\overline{\pi} = \langle \pi_0, \pi_1, \dots, \pi_k \rangle$ denotes the unique normalizing isomorphism of D; thus, by the uniqueness of $\overline{\pi}$, the transduction $\mathfrak{d}(\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}) \circ \mathfrak{nor}[\Sigma, w]$ is functional. Finally, it follows from Proposition 38 that for each pair $(D, D') \in \mathfrak{nor}[\Sigma, w]$ such that D is a reachable ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$, we have that D' is a normalized ODD in $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ that has the same language as D.

The fact that $\mathfrak{nor}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular follows from Proposition 5.(2) together with the fact that the multimap transductions $\mathfrak{mm}[\mathrm{NR}[\Sigma, w]]$ and $\mathfrak{mm}[\eta[\Sigma, w]]$ are 2-regular (Proposition 25.(1)), and that the transduction $\mathfrak{cp}[\mathrm{NC}[\Sigma, w]]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular (Proposition 25.(2)), given that $\mathrm{NC}[\Sigma, w] \subseteq \mathcal{S}[\Sigma, w] \times \mathcal{S}[\Sigma, w]$, and that $|\mathcal{S}[\Sigma, w]| = 2^{O(|\Sigma| \cdot w \cdot \log w)}$. $\qquad \square$

6.6 Putting All Steps Together

In this subsection we combine Observation 26 with Lemma 27, Lemma 30, Lemma 36 and Lemma 39 to prove our Canonization as Transduction Theorem (Theorem 9). Consider the transduction

$$\widehat{\mathfrak{can}}[\Sigma, w] \doteq \mathfrak{d}(\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}) \circ \mathfrak{rea}[\Sigma, w] \circ \mathfrak{mer}[\Sigma, w] \circ \mathfrak{nor}[\Sigma, w].$$

Since each of the four transductions in the composition is at most $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular, we have that $\widehat{\mathfrak{can}}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular. Since each these four transductions is functional, the transduction $\widehat{\mathfrak{can}}[\Sigma, w]$ is functional. Since $\mathsf{Dom}(\mathfrak{d}(\widehat{\mathcal{B}}(\Sigma, w)^{\circledast})) = \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ and the image of each of the three first transductions is contained in the domain of the next transduction (from left to right), we have that $\mathsf{Dom}(\widehat{\mathfrak{can}}[\Sigma, w]) = \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$. Now, let (D, D') be a pair of ODDs in $\widehat{\mathfrak{can}}[\Sigma, w]$. Then there exist ODDs D_1 and D_2 such that $(D, D_1) \in \mathfrak{rea}[\Sigma, w]$, $(D_1, D_2) \in \mathfrak{mer}[\Sigma, w]$, and $(D_2, D') \in \mathfrak{nor}[\Sigma, w]$. Since each of these transductions is language preserving, we have that $\mathcal{L}(D) = \mathcal{L}(D_1) = \mathcal{L}(D_2) = \mathcal{L}(D')$. Since $D \in \widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$, we have that D is by definition deterministic and complete. By Lemma 30, D_1 is deterministic, complete and reachable. By Lemma 36, D_2 is deterministic, complete and minimized. Finally, by Lemma 39, D' is deterministic, complete, minimized and normalized. Since for each ODD D, there is a unique deterministic, complete, minimized and normalized ODD $\mathcal{C}(D)$ with the same language as D, we have that $D' = \mathcal{C}(D)$. This shows that $\widehat{\mathfrak{can}}[\Sigma, w] = \{(D, \mathcal{C}(D)) \ : \ D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$.

Now, consider the transduction

$$\mathfrak{can}[\Sigma, w] \doteq \mathfrak{d}(\mathcal{B}(\Sigma, w)^{\circledast}) \circ \mathfrak{det}[\Sigma, w] \circ \widehat{\mathfrak{can}}[\Sigma, 2^w].$$

Since $\widehat{\mathfrak{can}}[\Sigma, w]$ is $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular, we have that $\widehat{\mathfrak{can}}[\Sigma, 2^w]$ is $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular. This implies that $\mathfrak{can}[\Sigma, w]$ is also $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular. Since $\mathcal{B}(\Sigma, w)^{\circledast} = \mathsf{Dom}(\mathfrak{det}[\Sigma, w])$ and $\mathsf{Im}(\mathfrak{det}[\Sigma, w])$ is included in $\mathsf{Dom}(\widehat{\mathfrak{can}}[\Sigma, 2^w])$, we have that $\mathsf{Dom}(\mathfrak{can}[\Sigma, w]) = \mathcal{B}(\Sigma, w)^{\circledast}$. Now, let (D, D') be a pair of ODDs in $\mathfrak{can}[\Sigma, w]$. Then there is an ODD $D_1 \in \widehat{\mathcal{B}}(\Sigma, 2^w)^{\circledast}$ such that $(D, D_1) \in \mathfrak{det}[\Sigma, w]$ and $(D_1, D') \in \widehat{\mathfrak{can}}[\Sigma, 2^w]$. By Lemma 27, we have that D_1 is complete, deterministic and $\mathcal{L}(D) = \mathcal{L}(D_1)$. Additionally, $D' = \mathcal{C}(D_1)$. Since $\mathcal{L}(D) = \mathcal{L}(D_1) = \mathcal{L}(\mathcal{C}(D_1)) = \mathcal{L}(D')$, we have that $D' = \mathcal{C}(D_1) = \mathcal{C}(D)$. This shows that $\mathfrak{can}[\Sigma, w] = \{(D, \mathcal{C}(D)) \ : \ D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$. $\square$

7 Conclusion

In this work, we have introduced the notion of second-order finite automata, a formalism that combines traditional finite automata with ODDs of bounded width in order to represent possibly infinite *classes of languages*. Our main result (Theorem 10) is a *canonical form of canonical forms theorem*. It states for each second-order finite automaton $\mathcal{F}$, one can construct a canonical form $\mathcal{C}_2(\mathcal{F})$ whose language $\mathcal{L}(\mathcal{C}_2(\mathcal{F})) = \{\mathcal{C}(D) \ : \ D \in \mathcal{L}(\mathcal{F})\}$ is precisely the set of canonical forms of ODDs in $\mathcal{L}(\mathcal{F})$. Here, the canonical form $\mathcal{C}(D)$ of an ODD D is the usual deterministic,

complete, normalized ODD with *minimum number of states* having the same language as D. In this sense, the ODDs in $\mathcal{L}(\mathcal{C}_2(\mathcal{F}))$ carry useful complexity theoretic information about the languages they represent in the class $\mathcal{L}_2(\mathcal{F}) = \mathcal{L}_2(\mathcal{C}_2(\mathcal{F}))$.

Our canonization result immediately implies that the collection of regular-decisional classes of languages is closed under union, intersection, set difference, and a suitable notion of bounded-width complementation. This result also implies that inclusion and non-emptiness of intersection for regular-decisional classes of languages are decidable. Furthermore, non-emptiness of intersection for the second languages of second-order finite automata $\mathcal{F}_1$ and $\mathcal{F}_2$ can be solved in fixed parameter tractable time when the parameter is the maximum width of an ODD accepted by $\mathcal{F}_1$ or $\mathcal{F}_2$.

We also provided two algorithmic applications of second-order automata to the theory of ODDs. First, we have shown that several width/size minimization problems for ODDs can be solved in fixed-parameter tractable time when parameterized by the width of the input ODD. This implies corresponding FPT algorithms for width/size minimization of ordered binary decision diagrams (OBDDs) with a fixed ordering. Previous to our work, only exponential algorithms were known. Finally, we have shown that second-order finite automata can be used to count the exact number of distinct functions computable by (Σ, w)-ODDs of a given width w and a given length k in time $2^{2^{O(|\Sigma| \cdot w \cdot 2^w)}} \cdot k^{O(1)}$, and in time $2^{2^{O(|\Sigma| \cdot w \cdot \log w)}} \cdot k^{O(1)}$ if only deterministic, complete ODDs are considered. It is worth noting that the naive process of enumerating functions while eliminating repetitions takes time (and space) exponential in both w and in k.

Regular Canonizing Relations Most results in this work are obtained as a consequence of Theorem 9, which states that the relation $\mathfrak{can}[\Sigma, w] = \{(D, \mathcal{C}(D)) : D \in \mathcal{B}(\Sigma, w)^{\circledast}\}$ is a regular relation. It is worth noting that aside from complexity theoretic considerations, Theorem 10 and Theorem 13 have identical proofs if we replace $\mathfrak{can}[\Sigma, w]$ with any *regular* canonizing relation $R(\Sigma, w)$ for $\mathcal{B}(\Sigma, w)^{\circledast}$ in the sense we will define below. Nevertheless, when taking complexity considerations into account, and also when considering our applications in Section 5, the fact that the transductions $\mathfrak{can}[\Sigma, w]$ and $\widehat{\mathfrak{can}}[\Sigma, w]$ are $2^{O(|\Sigma| \cdot w \cdot 2^w)}$-regular and $2^{O(|\Sigma| \cdot w \cdot \log w)}$-regular respectively play an important role. Additionally, some of our results use explicitly the fact the canonical form $\mathcal{C}(D)$ has minimum number of states among all deterministic, complete ODDs with the same language as D.

Say that a relation $R(\Sigma, w) \subseteq \mathcal{B}(\Sigma, w)^{\circledast} \times \mathcal{B}(\Sigma, w)^{\circledast}$ is canonizing for $\mathcal{B}(\Sigma, w)^{\circledast}$ if the following three conditions are verified.

1. $R(\Sigma, w)$ is functional and the domain of $R(\Sigma, w)$ is equal to $\mathcal{B}(\Sigma, w)^{\circledast}$.
2. For each $(D, D') \in R(\Sigma, w)$, $\mathcal{L}(D) = \mathcal{L}(D')$.
3. $(D, D') \in R(\Sigma, w)$ implies that $(D'', D') \in R(\Sigma, w)$ for each $D'' \in \mathcal{B}(\Sigma, w)^{\circledast}$ with $\mathcal{L}(D) = \mathcal{L}(D'')$.

The notion of a relation $\widehat{R}(\Sigma, w)$ that is canonizing for $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ can be defined analogously. An interesting question is whether there are canonizing relations with significantly better complexity than the ones of $\mathfrak{can}[\Sigma, w]$ and $\widehat{\mathfrak{can}}[\Sigma, w]$. More

specifically, is there some canonizing relation for $\mathcal{B}(\Sigma, w)^{\circledast}$ that is α-regular for $\alpha = 2^{f(|\Sigma|) \cdot 2^{o(w)}}$ where f is a function depending only on the size of the alphabet? Similarly, is there some canonizing relation for $\widehat{\mathcal{B}}(\Sigma, w)^{\circledast}$ that is α-regular for some $\alpha = 2^{o(f(|\Sigma|) \cdot w \cdot \log w)}$? In view of Observation 14, a canonizing relation of complexity $\alpha = 2^{O(f(|\Sigma|) \cdot w)}$ would imply that emptiness of intersection regular-decisional classes of languages can be realized in polynomial time even when w is logarithmic in the size of the input second-order finite automata representing these classes of languages.

Connections with the Theory of Automatic Structures Finite automata operating with ODDs and tuples of ODDs were first considered in [15] as a formalism to provide a uniform representation of classes of finite relational structures of bounded ODD-width. The technical results from [13] rely on two observations. First, that the relation $\mathcal{R}_{\in}(\Sigma, w) = \{(D, s) \ : \ D \in \mathcal{B}(\Sigma, w)^{\circledast}, \ s \in \mathcal{L}(D)\}$ is regular (Proposition 6.3 of [13]). Second, that the relation $\mathcal{R}_{\subseteq}(\Sigma, w) = \{(D, D') \ : \ D, D' \in \mathcal{B}(\Sigma, w)^{\circledast}, \mathcal{L}(D) \subseteq \mathcal{L}(D')\}$ is regular (Proposition 6.6 of [13]). Similar observations have been used in [32] to study second-order finite automata using the framework of the theory of automatic structures [1, 2, 31]. In particular, some of our decidability and closure results have been rederived in [32] using this framework, and some new applications of second-order finite automata to partial-order theory have been obtained.

As pointed out in [32], the fact that automatic structures are closed under first-order interpretations [8, 23, 31] has also applications in our framework. For instance, one can infer that the relation $\mathcal{R}_{\subseteq}(\Sigma, w)$ is automatic from the fact that $\mathcal{R}_{\in}(\Sigma, w)$ is automatic, together with the fact that for each two ODDs D, D', the inclusion $\mathcal{L}(D) \subseteq \mathcal{L}(D')$ is defined by the first-order formula $\forall s, (D, s) \in \mathcal{R}_{\in}(\Sigma, w) \Rightarrow (D', s) \in \mathcal{R}_{\in}(\Sigma, w)$. Similarly, the relation $\mathcal{R}_{=}(\Sigma, w) = \{(D, D') : D, D' \in \mathcal{B}(\Sigma, w)^{\circledast}, \mathcal{L}(D) = \mathcal{L}(D')\}$ can be inferred to be automatic from the fact the equality $\mathcal{L}(D) = \mathcal{L}(D')$ is defined by the first-order formula $\forall s, (D, s) \in \mathcal{R}_{\in}(\Sigma, w) \Leftrightarrow (D', s) \in \mathcal{R}_{\in}(\Sigma, w)$. Since relations definable in this way can be accepted by finite automata of size $f(|\Sigma|, w)$, for some function f, the interplay between automatic structures and first-order logics may be useful in the development of algorithms that decide properties of ODDs in $\mathcal{B}(\Sigma, w)^{\circ k}$ in time fixed-parameter linear time $f(|\Sigma|, w) \cdot k$, although the functions $f(|\Sigma|, w)$ obtained by a direct application of this approach may be very large. We refer to [22] for an updated survey on the theory of automatic structures.

Jain, Luo and Stephan have introduced the notion of automatic indexed classes of languages as a tool to address some problems in computational learning theory [29]. An indexed class of languages $\{L_\alpha \ : \ \alpha \in I\}$ is said to be *automatic* if the relation $E = \{(\alpha, x) \ : \ x \in L_\alpha, \ \alpha \in I\}$ is automatic. The fact that $\mathcal{R}_{\in}(\Sigma, w)$ is regular immediately implies that any regular-decisional class of languages corresponds to an automatic class of languages. Indeed, given a second-order finite automaton $\mathcal{F}$, the second language of $\mathcal{F}$, $\mathcal{L}_2(\mathcal{F}) = \{\mathcal{L}(D) \ : \ D \in \mathcal{L}(\mathcal{F})\}$, is an automatic class of languages where each $D \in \mathcal{L}(\mathcal{F})$ is regarded as an index, and $\mathcal{L}(D)$ is regarded as the language indexed by D. Henning Fernau conjectured that if $\{L_\alpha \ : \ \alpha \in I\}$ is

an automatic class of languages where each index $\alpha \in I$ is a finite string and all strings in L_α have the same length, then this class is regular-decisional (i.e. is equal to the second language of some second-order finite automaton). This conjecture has recently been confirmed by Kuske in [32]. Similar connections can be established with the framework of uniform classes of automatic structures [39], which are defined with basis on the notion of automatic structures with advice. In this context, an ODD D may be regarded as an advice string, while the language $\mathcal{L}(D)$ may be regarded as the set of strings associated with the advice D. This point of view is particularly relevant when ODDs are used to represent relations, as done for instance in [13].

In view of the connections discussed above, our framework provides a suitable parameterization for problems arising in the realm of the theory of automatic classes of languages [29] and in the realm of the theory of uniformly automatic classes of structures [39]. The intuition is that the size of the representation for the whole class of languages/structures (i.e, the size of the second-order finite automaton given at the input) is completely dissociated from the complexity of the languages/structures being represented in the class (i.e. the ODD-width w necessary to represent languages/structures in the class). Since the concepts in [29, 39] have applications in the fields of learning theory [7, 24, 27, 28] and algebra [9, 30, 38, 39], an interesting line of research would be the investigation of potential applications of our fixed-parameter tractable algorithms to problems in these fields.

References

1. Baranyi, V., Grädel, E., Rubin, S.: Automata-based presentations of infinite structures. Finite and Algorithmic Model Theory **379**, 1 (2011)
2. Blumensath, A., Grädel, E.: Automatic structures. In: Proc. of the 15th Annual IEEE Symposium on Logic in Computer Science (LICS 2000), pp. 51–62. IEEE Computer Society (2000)
3. Bollig, B.: On the width of ordered binary decision diagrams. In: Zhang, Z., Wu, L., Xu, W., Du, D. (eds.) Proc. of the 8th International Conference on Combinatorial Optimization and Applications (COCOA 2014) volume 8881 of Lecture Notes in Computer Science, pp. 444–458. Springer (2014)

4. Bollig, B.: On the minimization of (complete) ordered binary decision diagrams. Theory Comput. Syst. **59**(3), 532–559 (2016)
5. Bouajjani, A., Habermehl, P., Rogalewicz, A., Vojnar, T.: Abstract regular tree model checking. Electron. Notes Theor. Comput. Sci. **149**(1), 37–48 (2006)
6. Bozapalidis, S., Kalampakas, A.: Graph automata. Theor. Comput. Sci. **393**(1–3), 147–165 (2008)
7. Case, J., Jain, S., Ong, Y.S., Semukhin, P., Stephan, F.: Automatic learners with feedback queries. J. Comput. Syst. Sci. **80**(4), 806–820 (2014)
8. Case, J., Jain, S., Stephan, F., Stephan, F.: Automatic functions, linear time and learning. Log. Methods Comput. Sci. (2013)
9. Colcombet, T., Löding, C.: Transforming structures by set interpretations. Log. Methods Comput. Sci. **3**(2), 4 (2007)
10. Courcelle, B.: On recognizable sets and tree automata. In: Algebraic Techniques, pp. 93–126. Elsevier (1989)
11. Courcelle, B.: The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. Inf. Comput. **85**(1), 12–75 (1990)
12. Courcelle, B., Durand, I.: Verifying monadic second order graph properties with tree automata. In: Rhodes, C. (ed.) Proc. of the 3rd European Lisp Symposium (ELS 2010), pp. 7–21. ELSAA (2010)
13. De Melo, A.A., De Oliveira Oliveira, M.: On the width of regular classes of finite structures. In: Fontaine, P. (ed.) Proc. of the 27th International Conference on Automated Deduction (CADE 2019), volume 11716 of Lecture Notes in Computer Science, pp. 18–34. Springer (2019)
14. De Melo, A.A., De Oliveira Oliveira, M.: Second-order finite automata. In: Fernau, H. (ed.) Proc. of the 15th International Computer Science Symposium in Russia (CSR 2020), volume 12159 of Lecture Notes in Computer Science, pp. 46–63 (2020)
15. De Melo, A.A., De Oliveira Oliveira, M.: Symbolic solutions for symbolic constraint satisfaction problems. In: Calvanese, D., Erdem, E., Thielscher, M. (eds.) Proc. of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020), pp. 49–58 (2020)
16. Ebbinghaus, H.-D., Flum, J.: Finite automata and logic: A microcosm of finite model theory. In: Finite Model Theory, pp. 107–118. Springer (1995)
17. Ergün, F., Kumar, R., Rubinfeld, R.: On learning bounded-width branching programs. In: Maass, W. (ed.) Proc. of the Eigth Annual Conference on Computational Learning Theory (COLT 1995), pp. 361–368. ACM (1995)
18. Forbes, M.A., Kelley, Z.: Pseudorandom generators for read-once branching programs, in any order. In: Thorup, M. (ed.) Proc. of the 59th IEEE Annual Symposium on Foundations of Computer Science (FOCS 2018), pp. 946–955. IEEE Computer Society (2018)
19. Giammarresi, D., Restivo, A.: Recognizable picture languages. Int. J. Pattern Recognit. Artif. Intell. **6**(2&3), 241–256 (1992)
20. Godefroid, P.: Using partial orders to improve automatic verification methods. In: Clarke, E.M., Kurshan, R.P. (eds.) 2nd International Workshop on Computer Aided Verification (CAV 1990), volume 531 of Lecture Notes in Computer Science, pp. 176–185. Springer (1990)
21. Goldreich, O.: On testing computability by small width obdds. In: Serna, M.J., Shaltiel, R., Jansen, K., Rolim, J.D.P. (eds.) Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, Proc. of the 13th International Workshop, APPROX 2010, and 14th International Workshop, RANDOM 2010, volume 6302 of Lecture Notes in Computer Science, pp. 574–587. Springer (2010)
22. Grädel, E.: Automatic structures: twenty years later. In: Inproceedings of the 35Th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '20, pp. 21–34. New York, NY,USA, Association for Computing Machinery (2020)
23. Hodgsonl, B.R.: Décidabilité par automate fini. Annales des sciences mathématiques du Québec **7**(1), 39–57 (1983)
24. Hölzl, R., Jain, S., Stephan, F.: Learning pattern languages over groups. Theor. Comput. Sci. **742**, 66–81 (2018)
25. Hopcroft, J.: An n log n algorithm for minimizing states in a finite automaton. In: Theory of machines and computations, pp. 189–196. Elsevier (1971)
26. Hopcroft, J., Motwani, R., Ullman, J.: Introduction to Automata Theory Languages, and Computation. Pearson/Addison Wesley (2007)
27. Howar, F., Steffen, B.: Active automata learning in practice - an annotated bibliography of the years 2011 to 2016. In: Bennaceur, A., Hähnle, R., Meinke, K. (eds.) Machine Learning for Dynamic

Software Analysis: Potentials and Limits - International Dagstuhl Seminar 16172, volume 11026 of Lecture Notes in Computer Science, pp. 123–148. Springer (2018)

28. Jain, S., Kinber, E.B.: Automatic learning from positive data and negative counterexamples. In: Stoltz, G., Vayatis, N., Zeugmann, T. (eds.) Algorithmic Learning Theory - 23rd International Conference, ALT 2012, Lyon, France, October 29-31 Proceedings, volume 7568 of Lecture Notes in Computer Science, pp. 66–80. Springer (2012)

29. Jain, S., Luo, Q., Stephan, F.: Learnability of automatic classes. J. Comput. Syst. Sci. **78**(6), 1910–1927 (2012)

30. Kartzow, A., Schlicht, P.: Structures without scattered-automatic presentation. In: Bonizzoni, P., Brattka, V., Löwe, B. (eds.) Proc. of the 9th Conference on Computability in Europe (CiE 2013), volume 7921 of Lecture Notes in Computer Science, pp. 273–283. Springer (2013)

31. Khoussainov, B., Nerode, A.: Automatic presentations of structures. In: International Workshop on Logical and Computational Complexity (LCC 1994), volume 960 of Lecture Notes in Computer Science, pp. 367–392. Springer (1995)

32. Kuske, D.: Second-order finite automata: Expressive power and simple proofs using automatic structures. In: Moreira, N., Reis, R. (eds.) Proc. of the 25th International Conference on Developments in Language Theory (DLT 2021), volume 12811 of Lecture Notes in Computer Science, pp. 242–254. Springer (2021)

33. Newman, I.: Testing membership in languages that have small width branching programs. SIAM J. Comput. **31**(5), 1557–1570 (2002)

34. Priese, L.: Automata and concurrency. Theor. Comput. Sci. **25**(3), 221–265 (1983)

35. Ron, D., Tsur, G.: Testing computability by width two obdds. In: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, pp. 686–699. Springer (2009)

36. Thomas, W.: Automata theory on trees and partial orders. In: Bidoit, M., Dauchet, M. (eds.) Proc. of the 7th International Joint Conference on Theory and Practice of Software Development (TAPSOFT 1997), volume 1214 of Lecture Notes in Computer Science, pp. 20–38. Springer (1997)

37. Wegener, I.: Branching programs and binary decision diagrams. SIAM (2000)

38. Zaid, F.A.: Algorithmic solutions via model theoretic interpretations. Ph.D. Dissertation RWTH Aachen University (2016)

39. Zaid, F.A., Grädel, E., Reinhardt, F.: Advice automatic structures and uniformly automatic classes. In: Goranko, V., Dam, M. (eds.) Proc. of the 26th EACSL Annual Conference on Computer Science Logic (CSL 2017), volume 82 of LIPIcs, pp. 35:1–35:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2017)