Don't Care Words with an Application to the Automata-based Approach for Real Addition*

Jochen Eisinger[1] and Felix Klaedtke[2]

[1] Albert-Ludwigs-Universität Freiburg, Germany
[2] ETH Zurich, Switzerland

Abstract. Automata are a useful tool in infinite state model checking, since they can represent infinite sets of integers and reals. However, analogous to BDDs to represent finite sets, an obstacle of an automata-based set representation is the sizes of the automata. In this paper, we generalize the notion of "don't cares" for BDDs to word languages as a means to reduce the automata sizes. A general result in this paper shows that the minimal weak deterministic Büchi automaton (WDBA) with respect to a given don't care set with certain restrictions is uniquely determined and can be efficiently constructed. We apply don't cares to mechanize a more effective decision procedure for the first-order logic over the mixed linear arithmetic over the integers and the reals based on WDBAs.

1 Introduction

As Büchi observed almost 50 years ago [7, 8], automata can be used to decide arithmetical theories, like Presburger arithmetic. Roughly speaking, a Presburger arithmetic formula defines a regular language, for which one can build the automaton recursively over the structure of the formula. So, automata are used to represent sets of integers that are definable in Presburger arithmetic. More recently, model checkers for systems with unbounded integers, like FAST [1] and ALV [17] have been developed that use such an automata-based set representation using automata over finite words. The use of automata in these model checkers can be compared to the use of BDDs in model checkers for finite state systems, like SMV [15]: automata describe sets of system states. Moreover, automata constructions can be used for computing or overapproximating the set of all reachable states.

Sets of reals can be represented by ω-automata. Boigelot, Jodogne, and Wolper [5] have shown recently that even weak deterministic Büchi automata (WDBAs) suffice to represent the first-order definable sets in $(\mathbb{R}, Z, +, <)$, where Z is the unary predicate stating whether a number is an integer. WDBAs can be handled algorithmically almost as efficiently as automata over finite words. For instance, in contrast to Büchi automata, they can be efficiently minimized [14] and they are easy to complement. This result paves the way for an effective automata-based decision procedure for the first-order logic over $(\mathbb{R}, Z, +, <)$.

* This work was supported by the German Research Council (DFG) and the Swiss National Science Foundation (SNF).

The automata library LASH [13] provides implementations of all the needed operations for implementing such a decision procedure. WDBAs have a wide range of applications, e.g., Boigelot and others used them for the symbolic verification of linear hybrid automata [3, 4].

However, analogous to BDDs, a limiting factor in the automata-based representation of potential infinite sets of integers or reals is the size of the automata. For BDDs, many algorithms and methods have been developed to reduce the BDD sizes, which lead to performance boosts of BDD-based model-checkers. One of these techniques is the use of *don't cares* [10]. Roughly speaking, don't cares are inputs of a combinational circuit for which the circuit output is not specified or irrelevant. The BDD representation of a circuit can be reduced by choosing appropriate output values for the don't care inputs.

In this paper, we generalize the notion of don't cares for BDDs to languages. In the most general sense, a don't care set is a language over some alphabet. The set chosen depends on the application domain. The intuition of a don't care word is that it is irrelevant whether this word belongs to a language or not. Adding or removing don't care words to languages can result in smaller automata. A trivial example is where the don't care set consists of all words. In this case we can either add or remove all words and obtain an automaton with a single state. However, usually a don't care set is a proper subset of all words and it is not obvious which of these words have to be added or removed to obtain smaller automata. Furthermore, the order in which we add and remove words might lead to different (minimal) automata accepting the same language *modulo* the don't care set. We prove that under certain restrictions on the don't care set, the minimal WDBA is uniquely determined and can be efficiently constructed.

To demonstrate the effectiveness of don't cares for automata, we apply it to the approach for representing and manipulating sets of integers and reals by WDBAs. First, we define a straightforward don't care set when encoding reals by ω-words. Second, we present an automata construction for handling the existential quantification, which becomes more complicated when using don't cares. Third, we show by experiments that introducing don't care sets can reduce the automata sizes significantly in computing and representing sets of integers and reals.

Related to our work is [2] on widening sets of integers that are represented by automata. The widening approach differs from the concept of don't care words since they overapproximate the represented set. In order to obtain always an overapproximation of a set, the widening construction only adds words to the language. In contrast, we allow words to be removed, and adding or removing don't care words still yields an exact automata-based representation of the set. Moreover, the widening method is complementary to don't care words and hence, they can be combined in infinite state model checkers that use an automata-based representation for the reachable states of a system.

We proceed as follows. Preliminaries are given in §2. In §3, we introduce don't care words and present our general results about don't care sets. In §4, we present an automata construction for projecting sets of reals that are represented by WDBAs modulo a set of don't cares. In §5, we report on experimental results.

Finally, in §6, we draw conclusions. Proof details can be found in the appendix. In particular, Appendix §A contains the proofs of the claims in §3.3 for minimizing WDBAs with respect to a don't care set. Appendix §B contains the proof details of the correctness of the automata construction in §4 for handling the existential quantification.

2 Preliminaries

We assume that the reader is familiar with the basics of automata theory and first-order logic. The purpose of this section is to recall some background in these areas, and fix the notation and terminology used in the remainder of the text.

2.1 Languages and Deterministic Automata

Let Σ be an alphabet. We denote the set of all finite words over Σ by Σ^* and Σ^+ denotes the set $\Sigma^* \setminus \{\varepsilon\}$, where ε is the empty word. Σ^ω is the set of all ω-words over Σ. The *concatenation* of words is written as juxtaposition. We write $|w|$ for the *length* of $w \in \Sigma^*$. We often write a word $w \in \Sigma^*$ of length $\ell \geq 0$ as $w(0)\dots w(\ell-1)$ and an ω-word $\alpha \in \Sigma^\omega$ as $\alpha(0)\alpha(1)\alpha(2)\dots$, where $w(i)$ and $\alpha(i)$ denote the ith letter of w and α, respectively.

A *deterministic finite automaton* (DFA) $\mathcal{A}$ is a tuple $(Q, \Sigma, \delta, q_\mathrm{I}, F)$, where Q is a finite set of states, Σ is an alphabet, $\delta : Q \times \Sigma \to Q$ is the transition function, $q_\mathrm{I} \in Q$ is the initial state, and $F \subseteq Q$ is the set of accepting states. A state not in F is a *rejecting* state. The *size* of $\mathcal{A}$ is the cardinality of Q. We write $\mathcal{A}_q$ for the DFA that is identical to $\mathcal{A}$ except that $q \in Q$ is the initial state, i.e., $\mathcal{A}_q := (Q, \Sigma, \delta, q, F)$. We extend δ to the function $\hat{\delta} : Q \times \Sigma^* \to Q$ defined as $\hat{\delta}(q, \varepsilon) := q$ and $\hat{\delta}(q, bu) := \hat{\delta}(\delta(q, b), u)$, where $q \in Q$, $b \in \Sigma$, and $u \in \Sigma^*$. The DFA $\mathcal{A}$ defines the language $L_*(\mathcal{A}) := \{w \in \Sigma^* : \hat{\delta}(q_\mathrm{I}, w) \in F\}$.

The state $q \in Q$ is *reachable* from $p \in Q$ if there is a word $w \in \Sigma^*$ such that $\hat{\delta}(p, w) = q$. In the remainder of the text, we assume that every state in an automaton is reachable from its initial state. A *strongly connected component* (SCC) of $\mathcal{A}$ is a set $S \subseteq Q$ such that every $p \in S$ is reachable from every $q \in S$ and S is maximal. For $q \in Q$, $\mathrm{SCC}(q)$ denotes the SCC $S \subseteq Q$ with $q \in S$. We call an SCC S *accepting* if $S \subseteq F$, and *rejecting* if $S \cap F = \emptyset$.

We can view a DFA as a *deterministic Büchi automaton* (DBA). A *run* of the DBA $\mathcal{A}$ on the ω-word $\alpha \in \Sigma^\omega$ is an ω-word $\vartheta \in Q^\omega$ such that $\vartheta(0) = q_\mathrm{I}$ and $\vartheta(i+1) = \delta(\vartheta(i), \alpha(i))$, for all $i \in \mathbb{N}$. The run ϑ is *accepting* if $\mathrm{Inf}(\vartheta) \cap F \neq \emptyset$, where $\mathrm{Inf}(\vartheta)$ is the set of states that occur infinitely often in ϑ. The DBA $\mathcal{A}$ defines the ω-language $L_\omega(\mathcal{A}) := \{\alpha \in \Sigma^\omega : \text{the run of } \mathcal{A} \text{ on } \alpha \text{ is accepting}\}$. The DBA $\mathcal{A}$ is *weak* if every SCC of $\mathcal{A}$ is either accepting or rejecting. We use the initialism WDBA for "weak deterministic Büchi automaton." Similarly, we can view a DFA as a *deterministic co-Büchi automaton* (co-DBA). Runs of co-DBAs are defined as for DBAs. A run ϑ of a co-DBA $\mathcal{C}$ is *accepting* if $\mathrm{Inf}(\vartheta) \cap F = \emptyset$, where F is the set of "accepting" states of $\mathcal{C}$. We define $\overline{L}_\omega(\mathcal{C}) := \{\alpha \in \Sigma^\omega :$ the run of $\mathcal{C}$ on α is accepting (in the co-Büchi sense)$\}$.

2.2 Representing Sets of Reals with Automata

Let $\mathfrak{R}$ be the structure $(\mathbb{R}, Z, +, <)$, where $+$ and $<$ are as expected and Z is the unary predicate such that $Z(x)$ is true iff x is an integer. For a formula $\varphi(x_1,\dots,x_r)$ and $a_1,\dots,a_r \in \mathbb{R}$, we write $\mathfrak{R} \models \varphi[a_1,\dots,a_r]$ if φ is true in $\mathfrak{R}$ when the variable x_i is interpreted as a_i.

Boigelot, Jodogne, and Wolper have shown in [5] that for every first-order definable set $X \subseteq \mathbb{R}^r$ in $\mathfrak{R}$, there is a WDBA $\mathcal{A}$ that describes X. Moreover, they have shown that $\mathcal{A}$ can be effectively constructed from a formula $\varphi(x_1,\dots,x_r)$ that defines X, i.e., $X = \{\overline{a} \in \mathbb{R}^r : \mathfrak{R} \models \varphi[\overline{a}]\}$. We recall the precise correspondence between subsets of $\mathbb{R}^r$ and ω-languages from [5]. In the remainder of the text, let $\varrho > 1$ and $\Sigma := \{0,\dots,\varrho-1\}$ be fixed. ϱ is called the *base*.

Definition 1. *Let $r \geq 1$.*

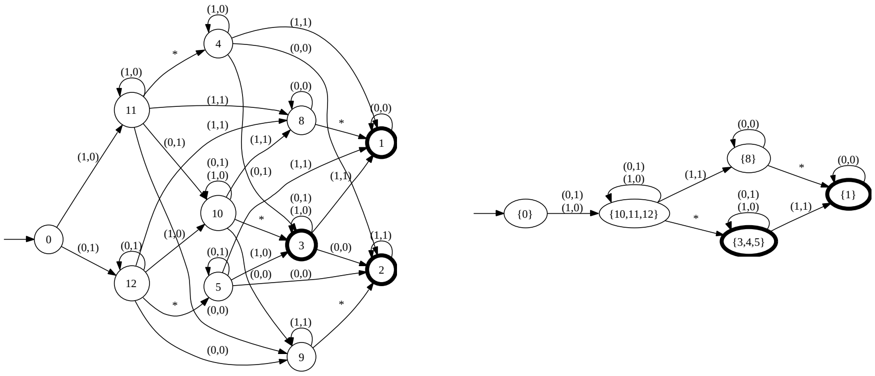
1. *V_r denotes the set of all ω-words over the alphabet $\Sigma^r \cup \{\star\}$ of the form $v \star \gamma$, where $v \in (\Sigma^r)^+$ with $v(0) \in \{0, \varrho-1\}^r$ and $\gamma \in (\Sigma^r)^\omega$.*
2. *An ω-word $v \star \gamma \in \mathsf{V}_r$ represents the vector of reals with r components*
$$\langle\!\langle v \star \gamma \rangle\!\rangle := -\varrho^{|v|-1} \cdot v(0) + \sum_{0<i<|v|} \varrho^{|v|-i-1} \cdot v(i) + \sum_{i \geq 0} \varrho^{-i-1} \cdot \gamma(i)\,,$$
where vector addition and scalar multiplication are componentwise.[3]
3. *The ω-language of formula $\varphi(x_1,\dots,x_r)$ is $L(\varphi) := \{\alpha \in \mathsf{V}_r : \mathfrak{R} \models \varphi[\langle\!\langle \alpha \rangle\!\rangle]\}$.*

Note that every vector in $\mathbb{R}^r$ can be represented by an ω-word in V_r. However, the representation is not unique. First, the first letter can be repeated arbitrarily often without changing the represented vector. Second, a vector that contains in a component a rational whose denominator has only prime factors that are also factors of the base ϱ, has distinct representations, e.g., in base $\varrho = 2$, $\langle\!\langle 0 \star 10^\omega \rangle\!\rangle = \langle\!\langle 0 \star 01^\omega \rangle\!\rangle = \frac{1}{2}$, where b^ω is the infinite repetition of the letter b.

Additional notation. Let $r \geq 1$ and $s,t \in \{1,\dots,r\}$ with $s \leq t$. We denote the tth coordinate of $b \in \Sigma^r$ by $b_{\restriction t}$ and $b_{\restriction s,t} := (b_{\restriction s}, b_{\restriction s+1},\dots,b_{\restriction t})$. We write $\alpha_{\restriction t}$ for the tth *track* of $\alpha \in (\Sigma^r \cup \{\star\})^\omega$, i.e., $\alpha_{\restriction t}$ is the ω-word $\gamma \in (\Sigma \cup \{\star\})^\omega$ defined as $\gamma(i) := \star$ if $\alpha(i) = \star$, and $\gamma(i) := \alpha(i)_{\restriction t}$ otherwise, for $i \in \mathbb{N}$. Analogously, $\alpha_{\restriction s,t}$ denotes the ω-word consisting of the tracks $s, s+1,\dots,t$ of α. For $m,n \geq 1$ and ω-words $\alpha \in (\Sigma^m \cup \{\star\})^\omega$ and $\beta \in (\Sigma^n \cup \{\star\})^\omega$, we write (α,β) for the ω-word $\gamma \in (\Sigma^{m+n} \cup \{\star\})^\omega$ with $\gamma_{\restriction 1,m} = \alpha$ and $\gamma_{\restriction m+1,m+n} = \beta$. Here, we make the assumption that $\alpha(i) = \star$ iff $\beta(i) = \star$, for all $i \in \mathbb{N}$. We use the same notation for finite words, which is defined analogously.

3 Don't Cares for Optimizing the Real Representation

In this section, we define our optimized representation of the reals as ω-words, which leads us to the general concept of don't care words for ω-languages. We first give a motivating example.



(a) straightforward encoding (b) optimized encoding

Fig. 1. Minimal WDBAs for the formula $x \neq 0 \wedge x+y=0$. For the sake of readability, we have omitted the rejecting sink states and their incoming transitions.

Example 2. Consider the formula $\varphi(x, y) := x \neq 0 \wedge x+y = 0$. The minimal WDBA accepting $L(\varphi)$ in base $\varrho = 2$ is shown in Figure 1(a). This WDBA is rather complex as it must either accept or reject all ω-words that represent the same pair of reals. For instance, the ω-words $\alpha := (1, 0) \star (1, 0)^\omega$ and $\beta := (0, 1) \star (0, 1)^\omega$ represent the pair $(0, 0)$ of reals, which does not satisfy φ and thus, the WDBA must reject them. In the optimized encoding we exploit that already the ω-word $\gamma := (0, 0) \star (0, 0)^\omega$ takes care of the fact that the pair of reals $(0, 0)$ is not in the represented set. That means, we can add α and β to the ω-language.

More general, an ω-word that has a suffix in which at least one of its tracks is of the form 1^ω is treated as a *don't care*, i.e., we can freely chose whether the automaton should accept or reject this ω-word. Observe that for every don't care representing the pair (x, y) of reals, there is an ω-word that also represents (x, y) and is not a don't care.

Consider again the ω-words α and β, which are don't cares. When reading these ω-words, we eventually loop in the states 4 and 5, respectively. Note that all runs that eventually stay in one of these states are don't cares. Making the states 4 and 5 accepting clearly alters the ω-language of the WDBA. However, we only add ω-words that are don't cares, like α and β. If the states 4 and 5 are accepting we can merge them with state 3. Analogously, we can make state 2 rejecting. Then, we can merge the states 2 and 9 with the rejecting sink state. We could also make the states 11 or 12 accepting. However, this would not be beneficial since it will prevent us from merging the states 10, 11, and 12. The resulting minimized automaton is depicted in Figure 1(b).

In the context of encoding reals by ω-words we use the following don't cares.

Definition 3. *Let $r \geq 1$. An ω-word $\alpha \in (\Sigma^r \cup \{\star\})^\omega$ is a* don't care word *if there are $t \in \{1, \ldots, r\}$ and $k \in \mathbb{N}$ such that $\alpha(i) \in \Sigma^r$ and $\alpha(i){\restriction_t} = \varrho - 1$, for all $i \geq k$. DC_r denotes the set of all don't care words in $(\Sigma^r \cup \{\star\})^\omega$.*

[3] Note that we do not distinguish between vectors and tuples.

Instead of constructing WDBAs that accept ω-languages $L(\varphi)$ for formulas φ, we are interested in constructing WDBAs that accept ω-languages that coincide on all the ω-words in $L(\varphi)$ that are not don't care words. Note that removing or adding *all* don't care words to $L(\varphi)$ does not necessarily result in a smaller automaton. Also note that by removing or adding all don't care words we can obtain ω-languages that are not recognizable by WDBAs.

In the following, we take a more general view.

Definition 4. *A* don't care set D *is an ω-language over some alphabet Γ. We write $L \equiv_D L'$ if $L \setminus D = L' \setminus D$, where $L, L' \subseteq \Gamma^\omega$ are ω-languages.*

In the remainder of this section, we present general results about ω-languages with respect to don't care sets. We focus on ω-languages that can be described by Büchi automata, in particular by WDBAs. In §3.1 and §3.2, we establish some straightforward facts. Namely, in §3.1, we observe that standard automata constructions carry over to handle the Boolean operations when using don't care sets, and in §3.2, we show how to solve the emptiness problem for Büchi automata with respect to an ω-regular don't care set $D \subseteq \Gamma^\omega$. In §3.3, we describe how to minimize WDBAs with respect to a don't care set $D \subseteq \Gamma^\omega$, where we assume that D fulfils the two properties: (1) $D \neq \Gamma^\omega$ and (2) $\alpha \in D \Leftrightarrow u\alpha \in D$, for all $u \in \Gamma^*$ and $\alpha \in \Gamma^\omega$. In particular, we show that the minimal WDBA is uniquely determined (up to isomorphism) and we give an efficient algorithm for constructing it under the assumption that D is ω-regular.

3.1 Boolean Operations

The automata construction for Boolean operations, like union and complementation of ω-languages, need not to be changed when using a don't care set $D \subseteq \Gamma^\omega$.

For WDBAs, we can use the standard product construction for the intersection and union. Let $\mathcal{A} = (Q, \Gamma, \delta, q_{\mathrm{I}}, F)$ and $\mathcal{B} = (Q', \Gamma, \delta', q'_{\mathrm{I}}, F')$ be WDBAs. For the intersection, we define $\mathcal{D} := (Q \times Q', \Gamma, \eta, (q_{\mathrm{I}}, q'_{\mathrm{I}}), F \times F')$, where $\eta((q, q'), b) := (\delta(q, b), \delta'(q', b))$, for $q \in Q$, $q' \in Q'$, and $b \in \Gamma$. The construction for the union is similar. Complementing WDBAs is done by flipping accepting and rejecting states of a WDBA. We define $\mathcal{C} := (Q, \Gamma, \delta, q_{\mathrm{I}}, Q \setminus F)$.

Proposition 5. *(a) For the WDBA $\mathcal{D}$, it holds that $L_\omega(\mathcal{D}) \equiv_D L_\omega(\mathcal{A}) \cap L_\omega(\mathcal{B})$. (b) For the WDBA $\mathcal{C}$, it holds that $L_\omega(\mathcal{C}) \equiv_D \Gamma^\omega \setminus L_\omega(\mathcal{A})$.*

3.2 Emptiness Check

The emptiness problem for Büchi automata modulo a don't care set D is to check whether a Büchi automaton $\mathcal{A}$ accepts an ω-word that is not in D. If D is ω-regular, then we can solve this problem by constructing the Büchi automaton accepting $L_\omega(\mathcal{A}) \setminus D$ and check whether the resulting Büchi automaton accepts an ω-word. The complexity is in $\mathrm{O}(n)$, where n is the number of states of $\mathcal{A}$. Note that D is fixed and hence, the size of the Büchi automaton for D is a constant.

3.3 Minimizing WDBAs with Don't Cares

Löding showed in [14] that the minimal WDBA can be constructed by using a standard DFA minimization algorithm, like that of Hopcroft [11]. However, before applying a DFA minimization algorithm, the WDBA has to be put into a normal form by determining a suitable set of accepting states. This preprocessing step can be done in linear time. We extend Löding's algorithm to WDBAs such that it takes a don't care set D over the alphabet Γ into account, where we require that (1) $D \neq \Gamma^\omega$ and (2) $\alpha \in D \Leftrightarrow u\alpha \in D$, for all $u \in \Gamma^*$ and $\alpha \in \Gamma^\omega$.

A WDBA $\mathcal{A}$ is *D-minimal* if there is no smaller WDBA $\mathcal{B}$ such that $L_\omega(\mathcal{A}) \equiv_D L_\omega(\mathcal{B})$. To minimize WDBAs with respect to the don't care set D, we need the following definitions.

Definition 6. *Let $\mathcal{A} = (Q, \Gamma, \delta, q_\mathrm{I}, F)$ be a WDBA.*

1. *A state $q \in Q$ is D-*recurrent *if $L_\omega(\mathcal{A}') \setminus D \neq \emptyset$, where $\mathcal{A}'$ is the WDBA $(Q, \Gamma, \delta, q, \mathrm{SCC}(q))$. A state is D-*transient *if it is not D-recurrent. An SCC is D-*recurrent *if it contains a D-recurrent state, otherwise, it is D-*transient.
2. *A mapping $c : Q \to \mathbb{N}$ is a D-*coloring *for $\mathcal{A}$ if the two conditions hold:*
   - *$c(q)$ is even $\Leftrightarrow q \in F$, for every D-recurrent state $q \in Q$, and*
   - *$c(p) \leq c(q)$, for all $p, q \in Q$ and $b \in \Gamma$ with $\delta(p, b) = q$.*

   *The D-coloring c is k-*maximal*, where $k \in \mathbb{N}$, if $c(q) \leq k$ and $c'(q) \leq c(q)$, for every $q \in Q$ and every D-coloring $c' : Q \to \mathbb{N}$ for $\mathcal{A}$.*
3. *$\mathcal{A}$ is in D-*normal form *if for some even $k \in \mathbb{N}$, there is a k-maximal D-coloring $c : Q \to \mathbb{N}$ such that $F = F_c$, where $F_c := \{q \in Q : c(q) \text{ is even}\}$.*[4]

Note that for the ω-words not in D, it is irrelevant whether a D-transient SCC is accepting or rejecting. Moreover, note that an SCC without loops is D-transient.

The algorithm in Figure 2 computes the D-normal form of a given WDBA $\mathcal{A} = (Q, \Gamma, \delta, q_\mathrm{I}, F)$. In line 11 of the algorithm, we must check whether an SCC S is D-transient. This can be done by checking whether $L_\omega(\mathcal{C}) \subseteq D$ holds, where $\mathcal{C}$ is the WDBA $(Q, \Sigma, \delta, q, S)$ and q is an arbitrarily chosen state in S. Note that $L_\omega(\mathcal{C}) \subseteq D$ iff $L_\omega(\mathcal{C}) \cap (\Gamma^\omega \setminus D) = \emptyset$. Under the assumption that D is ω-regular, it is easy to see that $L_\omega(\mathcal{C}) \cap (\Gamma^\omega \setminus D) = \emptyset$ can be checked in time $\mathrm{O}(|S|)$, since D is fixed and we can construct a Büchi automaton for the ω-language $\Gamma^\omega \setminus D$ in a preprocessing step. In summary, the checks performed in line 11 take time $\mathrm{O}(\sum_{S \text{ SCC of } \mathcal{A}} |S|) = \mathrm{O}(|Q|)$.

Lemma 7. *For a given WDBA $\mathcal{A} = (Q, \Gamma, \delta, q_\mathrm{I}, F)$, there is a set $F' \subseteq Q$ such that the WDBA $\mathcal{A}' := (Q, \Gamma, \delta, q_\mathrm{I}, F')$ is in D-normal form and $L_\omega(\mathcal{A}) \equiv_D L_\omega(\mathcal{A}')$. The set F' can be constructed in time $\mathrm{O}(|Q|)$ if D is ω-regular.*

Our minimization algorithm for WDBAs with the don't care set D is as follows: First, we put the given WDBA into D-normal form. Second, we apply to the WDBA in D-normal form the classical DFA minimization algorithm [11]. The overall complexity is in $\mathrm{O}(n \log n)$, where n is the size of $\mathcal{A}$. This algorithm returns the unique minimal WDBA for the don't care set D.

---

[4] Alternatively, we could require that k has to be odd. But we must fix some parity in order to obtain a canonical form for D-minimal WDBAs in D-normal form.

```
Compute the SCC graph $G$ of $\mathcal{A}$.
Compute a topological ordering $v_1, \dots, v_m$ on the vertices of $G$. To simplify
  notation, we identify a vertex $v_i$ with its corresponding SCC, i.e., a set of states.
Let $k \geq m$ be an even number.
for $i = m$ downto 1 do    /* Compute a $k$-maximal $D$-coloring $c : Q \to \mathbb{N}$ */
  if $v_i$ has no successors and $v_i$ is accepting then
    Define $c(q) := k$, for all $q \in v_i$.
  else if $v_i$ has no successors and $v_i$ is rejecting then
    Define $c(q) := k - 1$, for all $q \in v_i$.
  else
    Let $\ell := \min\{c(q) : v_j$ is a successor of $v_i$ and $q \in v_j\}$.
    if $v_i$ is $D$-transient then
      Define $c(q) := \ell$, for all $q \in v_i$.
    else if ($\ell$ is even and $v_i$ is accepting) or ($\ell$ is odd and $v_i$ is rejecting) then
      Define $c(q) := \ell$, for all $q \in v_i$.
    else
      Define $c(q) := \ell - 1$, for all $q \in v_i$.
    end if
  end if
end for
Return the WDBA $\mathcal{A}' := (Q, \Gamma, \delta, q_I, F_c)$.
```

Fig. 2. Algorithm for computing the D-normal form of a WDBA $\mathcal{A} = (Q, \Gamma, \delta, q_{\mathrm{I}}, F)$.

Theorem 8. *For a given WDBA $\mathcal{A} = (Q, \Gamma, \delta, q_{\mathrm{I}}, F)$, there is a D-minimal WDBA $\mathcal{A}'$ with $L_\omega(\mathcal{A}) \equiv_D L_\omega(\mathcal{A}')$. $\mathcal{A}'$ can be constructed in time $\mathrm{O}(|Q| \log |Q|)$ if D is ω-regular. Furthermore, every D-minimal WDBA $\mathcal{B}$ in D-normal form with $L_\omega(\mathcal{A}) \equiv_D L_\omega(\mathcal{B})$ is isomorphic to $\mathcal{A}'$.*

Remark 9. Similar to Definition 3, we can define for $r \geq 1$ the set I_r that consists of the ω-words over $\Sigma^r \cup \{\star\}$ that are not periodic in at least one track. Note that such a periodic track, if it is also in V_1, corresponds to an irrational number. Obviously, I_r has the properties (1) and (2). The decision procedure for the first-order logic over $\mathfrak{R}$ using WDBAs given in [5] can be understood as an automata-based decision procedure for the first-order logic over $(\mathbb{Q}, Z, +, <)$ using WDBAs with the don't care sets I_r. Note that the ω-languages definable in the first-order logic over $(\mathbb{Q}, Z, +, <)$ are in general not ω-regular using the encoding in Definition 1.2. From this point of view, we see that WDBAs modulo don't care sets can describe non-ω-regular languages and in this case, they even have a canonical minimal form (Theorem 8). So, it is uniquely determined which of the ω-words in I_r have to be added and removed in order to obtain the minimal WDBA. Analogously, WDBAs with the don't care sets DC_r can describe ω-regular languages that are not in the Borel class $F_\sigma \cap G_\delta$, which exactly captures the expressive power of WDBAs [16]. Furthermore, by Theorem 8, the ω-words in DC_r that have to be added to or removed from the ω-language are uniquely determined in order to obtain the minimal WDBA for the ω-language modulo the don't care set DC_r.

4 Quantification for the Reals

In this subsection, we present an automata construction for WDBAs that handles the existential quantification in the first-order logic over $\mathfrak{R}$ when using the don't care sets DC_r.

Roughly speaking, for the straightforward encoding, the existential quantification is done by eliminating the track of the quantified variable in the transitions of the WDBA.[5] Intuitively, this nondeterministic automaton guesses the digits of the quantified variable. As explained in [5], we can determinize this automaton by using the breakpoint construction for weak co-Büchi automata (see [12]). The construction for handling the existential quantification that we present in this subsection for the optimized encoding is also based on the breakpoint construction. However, the construction is more subtle because of the following problem: Assume that $\mathcal{A}$ is a WDBA for the formula $\varphi(x_1, \dots, x_r)$, i.e., $L_\omega(\mathcal{A}) \equiv_{\mathsf{DC}_r} L(\varphi)$. Eliminating the track of the variable x_r results in a nondeterministic Büchi automaton that might accept ω-words $\alpha \notin \mathsf{DC}_{r-1}$ for which there is only an ω-word $\gamma \in \mathsf{DC}_1$ such that $(\alpha, \gamma) \in L_\omega(\mathcal{A})$. A WDBA for $\exists x_r \varphi$ must not accept such ω-words α. A concrete instance of this problem is given in the following example.

Example 10. Consider again the formula $\varphi(x, y) := x \neq 0 \wedge x + y = 0$ and the WDBA in Figure 1(b) from Example 2. Eliminating the x-track, i.e., the first track, yields a nondeterministic Büchi automaton that accepts the ω-word $0 \star 0^\omega$, since we can infinitely loop in state $q := \{3, 4, 5\}$ by reading the letter 0. However, $\mathfrak{R} \not\models \exists x \varphi[\langle\!\langle 0 \star 0^\omega \rangle\!\rangle]$. Here, the problem is that the only ω-word γ such that $(\gamma, 0 \star 0^\omega)$ is accepted by the WDBA in Figure 1(b) is the don't care word $1 \star 1^\omega$. On the one hand, for the ω-word $0 \star 0^\omega$ the state q has to be rejecting. On the other hand, for the ω-word $0 \star (10)^\omega$ the state q has to be accepting.

Before we present our construction, we remark that removing all don't care words from the ω-language of the given WDBA before applying the construction in [5] for handling the existential quantification does not work. The reason is that the resulting DBA is not necessarily *weak* and hence, we cannot longer apply the breakpoint construction after eliminating the track of the quantified variable.

Assume that $\mathcal{A} = (Q, \Sigma^r \cup \{\star\}, \delta, q_{\mathrm{I}}, F)$ is a WDBA for the formula φ with r free variables, i.e., $L_\omega(\mathcal{A}) \equiv_{\mathsf{DC}_r} L(\varphi)$. We divide the construction of the WDBA for $\exists x_i \varphi$ into two steps. First, we construct from $\mathcal{A}$ a co-DBA $\mathcal{B}$ that accepts an ω-language for $\exists x_i \varphi$, i.e., $\overline{L}_\omega(\mathcal{B}) \equiv_{\mathsf{DC}_{r-1}} L(\exists x_i \varphi)$. Second, we show that $\mathcal{B}$ can be easily turned into a WDBA. To simplify notation, we assume without loss of generality that $i = r$ and $L_\omega(\mathcal{A}) \subseteq \mathsf{V}_r$.

To define $\mathcal{B}$'s transition function, we need the following definitions. For $u \in \Sigma^+$ with $u(0) \in \{0, \varrho - 1\}$, we define

$$\overline{u} := \begin{cases} 0^n & \text{if } u = (\varrho-1)^n \text{ with } n > 0, \\ 010^n & \text{if } u = 0(\varrho-1)^n \text{ with } n \geq 0, \\ v(c+1)0^n & \text{if } u = vc(\varrho-1)^n \text{ with } v \in \Sigma^+,\ c \in \Sigma \setminus \{\varrho - 1\}, \text{ and } n \geq 0. \end{cases}$$

[5] Some additional work is needed for the sign bit, see, e.g., [5, 6] for details.

Lemma 11. *It holds that* $\langle\!\langle u(\varrho-1)^n \star (\varrho-1)^\omega \rangle\!\rangle = \langle\!\langle \overline{u}0^n \star 0^\omega \rangle\!\rangle$, *for all* $n \geq 0$ *and* $u \in \Sigma^+$ *with* $u(0) \in \{0, \varrho-1\}$.

We define the relation $M \subseteq Q \times Q$ by pMq iff $p \in F$ and for every $\alpha \in (\Sigma^{r-1})^\omega \setminus \mathsf{DC}_{r-1}$, it holds that $(\alpha, (\varrho-1)^\omega) \in L_\omega(\mathcal{A}') \Rightarrow (\alpha, 0^\omega) \in L_\omega(\mathcal{A}_q)$, where $\mathcal{A}'$ is the WDBA $(Q, \Sigma^r \cup \{\star\}, \delta, p, \mathrm{SCC}(p))$.

Lemma 12. *Let* $p, q \in Q$, $b \in \Sigma^{r-1}$, *and* $c \in \Sigma \setminus \{\varrho-1\}$.

(a) If $\delta(p, (b, \varrho-1)) \in \mathrm{SCC}(p)$ *and* pMq *then* $\delta(p, (b, \varrho-1)) M \delta(q, (b, 0))$.
(b) If $\delta(p, (b, c)) \in \mathrm{SCC}(p)$ *and* $p \in F$ *then* $\delta(p, (b, c)) M \delta(p, (b, c+1))$.

Intuitively, the construction works as follows. As in the breakpoint construction, $\mathcal{B}$ has states of the form (R, S). Roughly speaking, in the first component we collect $\mathcal{A}$'s states that are reached by guessing the digits of the variable x_r. The second component checks whether we eventually stay in an accepting SCC of $\mathcal{A}$. In contrast to the breakpoint construction, R and S are not only subsets of Q but sets of pairs of states of $\mathcal{A}$. The reason for using pairs of states is the following. Assume that we reach the pair (R, S) from $\mathcal{B}$'s initial state by reading a finite prefix of an ω-word $\gamma \in \mathsf{V}_{r-1} \setminus \mathsf{DC}_{r-1}$. For $(p, q) \in R$, we have that p is reached by guessing a finite prefix of the digits of a real number for the quantified variable x_r. However, the guessed digits u could be a finite prefix of a don't care word $\alpha \in \mathsf{DC}_1 \cap \mathsf{V}_1$. Suppose that we visit p infinitely often when reading (γ, α). If p is accepting, $\mathcal{A}$ accepts (γ, α). However, since (γ, α) is a don't care word, $\langle\!\langle \alpha \rangle\!\rangle$ is not necessarily a real number such that $\mathfrak{R} \models \varphi[\langle\!\langle \gamma \rangle\!\rangle, \langle\!\langle \alpha \rangle\!\rangle]$. In order to detect such a case, we use the state q and the relation M. The state q is the state that is reached when guessing the corresponding digits for u of the ω-word $\beta \in \mathsf{V}_1 \setminus \mathsf{DC}_1$ such that $\langle\!\langle \alpha \rangle\!\rangle = \langle\!\langle \beta \rangle\!\rangle$. If pMq holds, then we know that $\mathfrak{R} \models \varphi[\langle\!\langle \gamma \rangle\!\rangle, \langle\!\langle \alpha \rangle\!\rangle]$, since $\langle\!\langle \alpha \rangle\!\rangle = \langle\!\langle \beta \rangle\!\rangle$ and $\mathcal{A}$ also accepts β. Hence, p is rightly an accepting state for the prefix of γ we have read so far. In the case where pMq does not hold, we have to treat p as a rejecting state.

Formally, $\mathcal{B}$ is the co-DBA $(\{q_{\mathrm{I}}'\} \cup (K \times K), \Sigma^{r-1} \cup \{\star\}, \eta, q_{\mathrm{I}}', K \times \{\emptyset\})$, where $K := \mathcal{P}(Q \times Q)$, q_{I}' is a fresh state and η is defined as follows. For the initial state, we define $\eta(q_{\mathrm{I}}', b) := (\emptyset, \emptyset)$, for $b \notin \{0, \varrho-1\}^{r-1}$, and for $b \in \{0, \varrho-1\}^{r-1}$, we define $\eta(q_{\mathrm{I}}', b) := (I(b), \emptyset)$, where

$$I(b) := \big\{ \big(\hat{\delta}(q_{\mathrm{I}}, (b^{|u|}, u)), \hat{\delta}(q_{\mathrm{I}}, (b^{|\overline{u}|}, \overline{u}))\big) \,:\, u \in \Sigma^+ \text{ with } u(0) \in \{0, \varrho-1\} \big\}\,.$$

For a state $(R, S) \in K \times K$ and $b \in \Sigma^{r-1}$, we define

$$\eta\big((R, S), b\big) := \begin{cases} (R', R' \cap M) & \text{if } S = \emptyset, \\ (R', S' \cap M) & \text{if } S \neq \emptyset, \end{cases}$$

where

$$\begin{aligned} R' := &\big\{ \big(\delta(p, (b, \varrho-1)), \delta(q, (b, 0))\big) \,:\, (p, q) \in R \big\} \cup \\ &\big\{ \big(\delta(p, (b, c)), \delta(p, (b, c+1))\big) \,:\, (p, q) \in R \text{ and } c \in \Sigma \setminus \{\varrho-1\} \big\}, \text{ and} \end{aligned}$$

and

$$S' := \{(\delta(p,(b,\varrho-1)),\delta(q,(b,0))) \,:\, (p,q)\in S\} \cup \{(\delta(p,(b,c)),\delta(p,(b,c+1))) \,:\, (p,q)\in S \text{ and } c\in\Sigma\setminus\{\varrho-1\}\}\,.$$

Finally, we define $\eta((R,S),\star) := (R', R'\cap M)$, where $R' := \{(\delta(p,\star),\delta(q,\star)) : (p,q)\in R\}$.

Lemma 13. *It holds that* $\overline{L}_\omega(\mathcal{B}) \equiv_{\mathsf{DC}_{r-1}} L(\exists x_r\varphi)$.

An SCC of $\mathcal{B}$ might contain accepting and rejecting states. The next lemma shows that if an SCC of $\mathcal{B}$ contains accepting and rejecting states then we can make all states in this SCC accepting. Given this, it is easy to turn the co-DBA $\mathcal{B}$ into a WDBA $\mathcal{A}'$ for $L(\exists x_r\varphi)$, i.e. $L_\omega(\mathcal{A}') \equiv_{\mathsf{DC}_{r-1}} L(\exists x_r\varphi)$.

Lemma 14. *Let* $\psi(y_1,\dots,y_s)$ *be a formula and let* $\mathcal{C} = (P,\Sigma^s\cup\{\star\},\mu,p_{\mathrm{I}},E)$ *be a co-DBA with* $\overline{L}_\omega(\mathcal{C}) \equiv_{\mathsf{DC}_s} L(\psi)$. *If* $S\subseteq P$ *is an SCC with* $S\cap E\neq\emptyset$ *then* $\overline{L}_\omega(\mathcal{C}') \equiv_{\mathsf{DC}_s} L(\psi)$, *where* $\mathcal{C}'$ *is the co-DBA* $(P,\Sigma^s\cup\{\star\},\mu,p_{\mathrm{I}},E\cup S)$.

The above given construction yields a WDBA that has $1+2^{2\cdot|Q|^2}$ states. However, some of the states are not reachable from the initial state q'_{I}, e.g., the states $(R,S)\in K\times K$ with $S\not\subseteq R$ are never reachable from q'_{I}. Next, we briefly discuss the auxiliary computations involved in the construction.

For the transitions from the initial state q'_{I}, we need to compute the sets $I(b)$, for every $b\in\Sigma^{r-1}$. Computing $I(b)$ separately for each $b\in\Sigma^{r-1}$, yields an algorithm that is exponential in r and is not practical. The algorithm described in [6] for determining the initial transitions of DFAs for quantifying Presburger arithmetic formulas, can be adopted to our construction and it works well in practice, although it has exponential worst case complexity in r.

For computing the relation M, we define the WDBAs $\mathcal{G} := (Q,\Sigma^{r-1},\delta_1,q_{\mathrm{I}},F)$ and $\mathcal{H} := (Q,\Sigma^{r-1},\delta_2,q_{\mathrm{I}},F)$, where $\delta_1(q,b) := \delta(p,(b,\varrho-1))$ and $\delta_2(q,b) := \delta(p,(b,0))$, for $q\in Q$ and $b\in\Sigma^{r-1}$. For states $p,q\in Q$, we have that pMq iff (1) $p\in F$ and (2) $L_\omega(\mathcal{G}')\cap L_\omega(\mathcal{H}_q)$ contains an ω-word not in DC_{r-1}, where $\mathcal{G}'$ is the WDBA $(Q,\Sigma^{r-1},\delta_1,p,\mathrm{SCC}(p))$. Since the SCC of p consists of at most $|F|$ states, condition (2) can be checked in time $\mathrm{O}(|Q|\cdot|F|)$, see §3.1 and §3.2. An upper bound for computing M is $\mathrm{O}(|Q|^2\cdot|F|^2)$, since the first component in M has to be a state in F. Note that with Lemma 12 we can reduce the number of pairs $(p,q)\in F\times Q$ for which we have to carry out the check in condition (2).

5 Experimental Results

In this section, we report on experimental results obtained from our prototype implementation of an automata-based decision procedure for the first-order logic over $\mathfrak{R}$.[6] We have carried out tests on two different classes of problems: (1) randomly generated formulas and (2) the iterative computation of the reachable

[6] Our prototype and further details on the evaluation are publicly available online at `http://www.informatik.uni-freiburg.de/~eisinger/research/rva.html`.

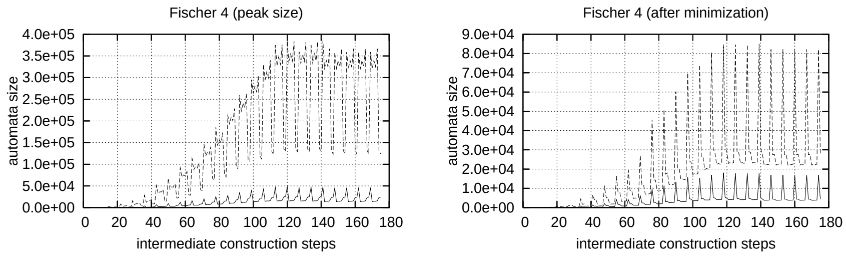


Fig. 3. Automata sizes encountered during the computation for Fischer's protocol with 4 processes. The solid (dashed) lines correspond to the optimized (straightforward) encoding. The intermediate construction steps correspond to the flows and jumps of the processes in Fischer's protocol. We obtain similar results for the other protocols.

states of infinite state systems. In the later case, we mainly focus on the sizes of the automata, as our prototype is not intended to compete with optimized tools for solving the reachability problem.

Random Formulas. We have applied our prototype to randomly generated formulas. For a test set of 100 formulas with 4 variables with about 10 disjunctions and conjunctions each, the savings in terms of automata sizes encountered during the construction are observable (on average 8.4%), although moderate. Our new construction for the quantification generates larger automata (on average 40.1%), however, after normalization and minimization the resulting automata with don't care sets are smaller (on average 7.7%). Our prototype requires up to one order of magnitude more runtime for the quantification when using don't care words. When restricting the 4 variables to the integer domain, the savings due to the don't care set become more substantial (on average 48.5%), as every integer has encodings that are in the don't care set. In comparison to an implementation based on LASH [13] without don't cares, our prototype is faster. The marginal difference in performance on small quantifier free formulas grows rapidly when the formulas contain quantifiers or have more variables.

Reachability Analysis. Infinite state systems, like systems with unbounded integers or linear hybrid automata can be analyzed symbolically in the first-order logic over $\mathfrak{R}$. We have analyzed the Bakery protocol, Fischer's protocol, and the railroad crossing example [9]. Using don't care words, the automata constructed during the iterative computation of the reachable states become smaller by an order of magnitude (see Figures 3 and 4). This saving can be explained by the following two observations. First, the formulas that describe the transitions of a system contain many variables (the formulas for Fischer's protocol with 5 processes have 34 variables). Note that the don't care sets contain more words if the formula contains many free variables. Second, the construction of the reachable state set requires a large number of automata constructions. Although the saving in a single automata construction might be small, the overall saving grows with the number of automata constructions.

		with don't cares			without don't cares		
	iterations	**peak**	**final**	**runtime**	**peak**	**final**	**runtime**
Fischer 2	9	212	53	35.8s	2,236	182	39.5s
Fischer 3	15	44,638	405	149.6s	90,422	2,045	174.9s
Fischer 4	21	48,207	4,377	3,954.7s	385,548	27,548	4,704.6s
Fischer 5	27	129,715	55,885	25,685.8s	1,582,115	430,727	49,872.3s
Railroad	8	151,634	7,735	1,227.4s	363,742	9,411	767.8s
Bakery 2	30	100	-	39.8s	535	-	44.5s
Bakery 3	30	297	-	74.4s	1,858	-	85.7s
Bakery 4	30	866	-	145.3s	8,166	-	211.0s

Fig. 4. Iterations required to reach the fixpoint of the reachable state set for several infinite state systems, construction times, and peak and final automata sizes. Note that the fixpoint for Bakery cannot be reached using our naive fixpoint computation.

6 Conclusions

We generalized the concept of *don't cares* for BDDs to automata and demonstrated that don't cares are effective in reducing the automata sizes. On the one hand, we were able to prove rather general results about don't cares sets, like the minimization of WDBAs. On the other hand, we presented an automata construction for the quantification in the first-order logic $\mathfrak{R}$, which depends on the used don't care set. We demonstrated the potential of don't cares by a prototype.

Future work includes the improvement of the mechanization of the given automata construction for handling the existential quantification in the first-order logic over $\mathfrak{R}$, which is currently the bottleneck in our prototype. Another direction we want to pursue is to exploit don't cares further. For example, for carrying out the quantification of x in the formula $(\exists x \varphi(x, \overline{y})) \vee \psi(\overline{y})$, we can use the language of the automaton for ψ as a don't care set for making the automaton for $\varphi(x, \overline{y})$ smaller before we apply the construction for the existential quantification. Moreover, we believe that don't care words have a high potential for making automata-based model checking for infinite states systems more effective.

References

1. S. Bardin, A. Finkel, J. Leroux, and L. Petrucci, *FAST: Fast acceleration of symbolic transition systems*, in Proc. of the 15th International Conference on Computer Aided Verification (CAV'03), vol. 2725 of Lecture Notes in Computer Science, 2003, pp. 118–121.
2. C. Bartzis and T. Bultan, *Widening arithmetic automata*, in Proc. of the 16th International Conference on Computer Aided Verification (CAV'04), vol. 3114 of Lecture Notes in Computer Science, 2004, pp. 321–333.
3. B. Boigelot, L. Bronne, and S. Rassart, *An improved reachability analysis method for strongly linear hybrid systems (extended abstract)*, in Proc. of the 9th International Conference on Computer Aided Verification (CAV'97), vol. 1254 of Lecture Notes in Computer Science, 1997, pp. 167–178.
4. B. Boigelot, F. Herbreteau, and S. Jodogne, *Hybrid acceleration using real vector automata*, in Proc. of the 15th International Conference on Computer-Aided

Verification (CAV'03), vol. 2725 of Lecture Notes in Computer Science, 2003, pp. 193–205.

5. B. Boigelot, S. Jodogne, and P. Wolper, *An effective decision procedure for linear arithmetic over the integers and reals*, ACM Trans. Comput. Log., 6 (2005), pp. 614–633.
6. B. Boigelot and L. Latour, *Counting the solutions of Presburger equations without enumerating them*, Theoretical Comput. Sci., 313 (2004), pp. 17–29.
7. J. Büchi, *Weak second-order arithmetic and finite automata*, Zeitschrift der mathematischen Logik und Grundlagen der Mathematik, 6 (1960), pp. 66–92.
8. ———, *On a decision method in restricted second order arithmetic*, in Logic, Methodology and Philosophy of Science (Proc. 1960 Internat. Congr.), Stanford University Press, 1962, pp. 1–11.
9. T. Henzinger, *The theory of hybrid automata*, in Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS '96), New Brunswick, New Jersey, 1996, pp. 278–292.
10. Y. Hong, P. A. Beerel, J. R. Burch, and K. L. McMillan, *Safe BDD minimization using don't cares*, in Proc. of the 34th Conference on Design Automation (DAC'97), ACM Press, 1997, pp. 208–213.
11. J. E. Hopcroft, *An $n \log n$ algorithm for minimizing the states in a finite automaton*, in Theory of Machines and Computations (Proc. of an International Symposium), Z. Kohavi and A. Paz, eds., Technion (Israel Institute of Technology), Haifa, Israel, 1971, Academic Press, pp. 189–196.
12. O. Kupferman and M. Vardi, *Weak alternating automata are not that weak*, ACM Trans. Comput. Log., 2 (2001), pp. 408–429.
13. LASH, *The Liège Automata-based Symbolic Handler.* `http://www.montefiore.ulg.ac.be/~boigelot/research/lash/`.
14. C. Löding, *Efficient minimization of deterministic weak ω-automata*, Information Processing Letters, 79 (2001), pp. 105–109.
15. K. L. McMillan, *Symbolic Model Checking*, Kluwer Academic Publishers, 1993.
16. L. Staiger and K. Wagner, *Automatentheoretische und automatenfreie Charakterisierungen topologischer Klassen regulärer Folgenmengen*, Elektronische Informationsverarbeitung und Kybernetik, 10 (1974), pp. 379–392.
17. T. Yavuz-Kahveci, C. Bartzis, and T. Bultan, *Action language verifier, extended*, in Proc. of the 17th International Conference on Computer Aided Verification (CAV'05), vol. 3576 of Lecture Notes in Computer Science, 2005, pp. 413–417.

A Proof Details for Minimizing with Don't Cares

We want to point out that most of the proofs in this section follow the lines of the proofs in Löding's article [14] on minimizing WDBAs. However, there are some subtleties that require to adjust and to generalize the argumentation for proving our results in §3.3 on minimizing WDBAs with respect to a don't care set D.

Recall that we require that the don't care set D fulflls the two properties: (1) $D \neq \Gamma^\omega$ and (2) $\alpha \in D \Leftrightarrow u\alpha \in D$, for all $u \in \Gamma^*$ and $\alpha \in \Gamma^\omega$.

Lemma 15. *Let $\mathcal{A} = (Q, \Gamma, \delta, q_\mathrm{I}, F)$ be a WDBA. For every state $p \in Q$, there is a state $q \in Q$ such that q is D-recurrent and q is reachable from p.*

Proof. For the sake of contradiction, assume that all states reachable from p are D-transient. That means, that all runs of $\mathcal{A}_p$ on any ω-word $\alpha \in \Gamma^\omega$ visit only D-transient states. It follows that all states reachable from q_I are D-transient because of the requirement (2). Therefore, all ω-words $\alpha \in \Gamma^\omega$ are in D. This contradicts the requirement (1). $\square$

We first prove Lemma 7, which states that a WDBA can be put into D-normal form in linear time if we additionally require that D is ω-regular.

Proof (Lemma 7). The proof follows the lines of the proof of Theorem 8 in Löding's article [14] and we do not repeat it here. However, the argumentation from Löding's proof about the correctness and complexity of the algorithm in Figure 2 for computing the set F' has to be extended by the following points. Let v_i be a vertex in the SCC graph and let $S_i \subseteq Q$ its corresponding SCC.

1. If v_i has no successors then S_i is D-recurrent. This follows from Lemma 15 and the definition of a D-recurrent SCC.
2. In line 11 of the algorithm, we have to check whether S_i is D-transient. This can be done by checking whether $L_\omega(\mathcal{C}) \subseteq D$ holds, where $\mathcal{C}$ is the WDBA $(Q, \Sigma, \delta, q, S_i)$, where q is an arbitrarily chosen state in S_i. Note that $L_\omega(\mathcal{C}) \subseteq D$ iff $L_\omega(\mathcal{C}) \cap (\Gamma^\omega \setminus D) = \emptyset$. It is easy to see that $L_\omega(\mathcal{C}) \cap (\Gamma^\omega \setminus D) = \emptyset$ can be checked in time $\mathrm{O}(|S_i|)$, since D is fixed and we can construct a Büchi automaton for the ω-language $\Gamma^\omega \setminus D$ in a preprocessing step. In summary, the checks performed in line 11 take time $\mathrm{O}(|S_1| + \cdots + |S_m|) = \mathrm{O}(|Q|)$. $\square$

For proving Theorem 8, we need some technical lemmas. In the following, let $\mathcal{A} = (Q, \Gamma, \delta, q_\mathrm{I}, F)$ be a WDBA and let $c : Q \to \mathbb{N}$ be a k-maximal D-coloring for $\mathcal{A}$. For an ω-word $\vartheta \in Q^\omega$, we define $c(\vartheta) := \max\{c(\vartheta(i)) : i \in \mathbb{N}\}$.

Lemma 16. *Let $\vartheta \in Q^\omega$ be the run on an ω-word in $\Gamma^\omega \setminus D$. It holds that ϑ is accepting iff $c(\vartheta)$ is even.*

Proof. Note that the states in an SCC have the same color. Since the run ϑ eventually stays in a D-recurrent SCC of $\mathcal{A}$, the color of the states in this SCC is even. Together with $c(\vartheta(i)) \leq c(\vartheta(j))$, for all $i \leq j$ we see that the claim is true. $\square$

Lemma 17. *For every state $p \in Q$ with $c(p) \leq k-2$ there is a state $q \in Q$ such that q is reachable from p and $c(q) = c(p) + 1$.*

Proof. We prove the claim by contradiction: assume that there is a $p \in Q$ not satisfying the property. We define a new D-coloring $c' : Q \to \mathbb{N}$ for $\mathcal{A}$ by

$$c'(q) := \begin{cases} c(q) + 2 & \text{if } q \text{ is reachable from } p \text{ and } c(p) = c(q), \\ c(q) & \text{otherwise.} \end{cases}$$

Because of c' the D-coloring c is not k-maximal. ☐

Lemma 18. *For every state $q \in Q$ there is an ω-word $\alpha \in \Gamma^\omega \setminus D$ such that $c(\vartheta) = c(q)$, where $\vartheta \in Q^\omega$ is the run of $\mathcal{A}$ on α.*

Proof. The claim is obviously true when q is D-recurrent. If the claim does not hold when q is D-transient then there is no D-recurrent state that is reachable from q and that has the same color as q. Note that because of Lemma 15 there is a D-recurrent state that is reachable from q and thus, $c(q) < k$. We define a new D-coloring $c' : Q \to \mathbb{N}$ for $\mathcal{A}$ by

$$c'(p) := \begin{cases} c(p) + 1 & \text{if } p \text{ is reachable from } q \text{ and } c(q) = c(p), \\ c(p) & \text{otherwise.} \end{cases}$$

Because of c' the D-coloring c is not k-maximal. ☐

Lemma 19. *For all states $p, q \in Q$, it holds that if $L_\omega(\mathcal{A}_p) \equiv_D L_\omega(\mathcal{A}_q)$ then $c(p) = c(q)$.*

Proof. For the sake of contradiction, assume that there are states $p, q \in Q$ such that $L_\omega(\mathcal{A}_p) \equiv_D L_\omega(\mathcal{A}_q)$ and $c(p) \neq c(q)$. Moreover, we assume that $c(p) + c(q)$ is maximal and $c(p) < c(q)$.

First, we show that $c(q) = c(p) + 1$. If $c(p) > k - 2$ then $c(q) = k$. Hence, $c(p) = k - 1$. Assume that $c(p) \leq k - 2$. By Lemma 17, there is word $u \in \Gamma^*$ such that $c(\hat{\delta}(p, u)) = c(p) + 1$. We have that $L_\omega(\mathcal{A}_{\hat{\delta}(p,u)}) \equiv_D L_\omega(\mathcal{A}_{\hat{\delta}(q,u)})$ since $L_\omega(\mathcal{A}_p) \equiv_D L_\omega(\mathcal{A}_q)$. Moreover, since $c(p) + c(q)$ is maximal, we conclude that $c(\hat{\delta}(p, u)) = c(\hat{\delta}(q, u))$. Also in this case, we have that $c(q) = c(\hat{\delta}(p, u)) = c(p)+1$, since $c(\hat{\delta}(q, u)) \geq c(q)$ and $c(q) \geq c(\hat{\delta}(p, u))$.

Now, let $\alpha \in \Gamma^\omega \setminus D$ be an ω-word such that $c(\vartheta) = c(p)$, where $\vartheta \in Q^\omega$ is the run of $\mathcal{A}_p$ on α. Note that such an ω-word α with such a run ϑ always exists because of Lemma 18. Let $\vartheta' \in Q^\omega$ be the run of $\mathcal{A}_q$ on α. If $c(\vartheta) > c(q)$ then let $v \in \Gamma^*$ be a prefix of α such that $c(\hat{\delta}(q, v)) > c(q)$. We have that $L_\omega(\mathcal{A}_{\hat{\delta}(p,v)}) \equiv_D L_\omega(\mathcal{A}_{\hat{\delta}(q,v)})$ and $c(\hat{\delta}(p, v)) + c(\hat{\delta}(q, v)) > c(p) + c(q)$. This is a contradiction to the choice of p and q. Note that $c(\hat{\delta}(p, v)) = c(p)$ since v is a prefix of α. Therefore, we have that $c(\vartheta) = c(p)$ and $c(\vartheta') = c(q)$. Since $c(q) = c(p)+1$, it follows from Lemma 16 that $\alpha \in L_\omega(\mathcal{A}_p) \Leftrightarrow \alpha \notin L_\omega(\mathcal{A}_q)$. This contradicts the assumption that $L_\omega(\mathcal{A}_p) \equiv_D L_\omega(\mathcal{A}_q)$. ☐

Lemma 20. *Under the assumption that $F = F_c$, it holds that if $L_*(\mathcal{A}_p) \neq L_*(\mathcal{A}_q)$ then $L_\omega(\mathcal{A}_p) \not\equiv_D L_\omega(\mathcal{A}_q)$, for every $p, q \in Q$.*

Proof. Let $p, q \in Q$ with $L_*(\mathcal{A}_p) \neq L_*(\mathcal{A}_q)$, i.e., there is a word $u \in \Gamma^*$ such that $\hat{\delta}(p, u) \in F \Leftrightarrow \hat{\delta}(q, u) \notin F$. Since $F = F_c$, we have that $c(\hat{\delta}(p, u)) \neq c(\hat{\delta}(q, u))$. From Lemma 19 it follows that $L_\omega(\mathcal{A}_{\hat{\delta}(p,u)}) \not\equiv_D L_\omega(\mathcal{A}_{\hat{\delta}(q,u)})$, i.e., there is an ω-word $\alpha \in \Gamma^\omega \setminus D$ such that $\alpha \in L_\omega(\mathcal{A}_{\hat{\delta}(p,u)}) \Leftrightarrow \alpha \notin L_\omega(\mathcal{A}_{\hat{\delta}(q,u)})$. We conclude that $L_\omega(\mathcal{A}_p) \not\equiv_D L_\omega(\mathcal{A}_q)$, since $u\alpha \notin D$ and $u\alpha \in L_\omega(\mathcal{A}_p) \Leftrightarrow u\alpha \notin L_\omega(\mathcal{A}_q)$. □

Definition 21. *For $L \subseteq \Gamma^\omega$, we define the relation $\approx_D^L \subseteq \Gamma^* \times \Gamma^*$ by $u \approx_D^L v$ iff $u\alpha \in L \Leftrightarrow v\alpha \in L$, for all $\alpha \in \Gamma^\omega \setminus D$.*

Lemma 22. *For $L \subseteq \Gamma^\omega$, $\approx_D^L$ is an equivalence relation and a right congruence.*

Proof. Obviously, $\approx_D^L$ is reflexive and symmetric. For showing that $\approx_D^L$ is transitive, assume that $u \approx_D^L v$ and $v \approx_D^L w$, where $u, v, w \in \Gamma^*$. For any $\alpha \in \Gamma^\omega \setminus D$, we have that $u\alpha \in L \Leftrightarrow v\alpha \in L \Leftrightarrow w\alpha \in L$ and thus, $u \approx_D^L w$.

We now show that $\approx_D^L$ is a right congruence. For $\alpha \in \Gamma^\omega \setminus D$, we have that $\alpha \notin D$ and by assumption $w\alpha \notin D$. It follows that $uw\alpha \in L \Leftrightarrow vw\alpha \in L$, and thus, $uw \approx_D^L vw$. □

We denote the equivalence class of $u \in \Gamma^*$ with respect to $\approx_D^L$ by $[u]_D^L$.

Lemma 23. *Let $L \subseteq \Gamma^\omega$ and let $\mathcal{A} = (Q, \Gamma, \delta, q_\mathrm{I}, F)$ be a WDBA with $L \equiv_D L_\omega(\mathcal{A})$. For each state $q \in Q$ that is reachable from q_I there is a word $u_q \in \Gamma^*$ such that $v \in [u_q]_D^L$, for all words $v \in \Gamma^*$ with $\hat{\delta}(q_\mathrm{I}, v) = q$.*

Proof. Let $q \in Q$ be a state and $u_q \in \Gamma^*$ be a word such that $\hat{\delta}(q_\mathrm{I}, u_q) = q$. It suffices to show that $u_q \approx_D^L v$, for every word $v \in \Gamma^*$ with $\hat{\delta}(q_\mathrm{I}, v) = q$. Let v be such a word, For $\alpha \in \Gamma^\omega$, it holds that $\mathcal{A}$ accepts $u_q\alpha \in L_\omega(\mathcal{A}) \Leftrightarrow v\alpha \in L_\omega(\mathcal{A})$. Note the runs on $u_q\alpha$ and $v\alpha$ only differ on a finite prefix; the runs are identical on α. Moreover, note that $\alpha \in \Gamma^\omega \setminus D \Leftrightarrow u_q\alpha, v\alpha \in \Gamma^\omega \setminus D$. □

Corollary 24. *Let $\mathcal{A} = (Q, \Gamma, \delta, q_\mathrm{I}, F)$ be a WDBA, where every state is reachable from q_I. $\mathcal{A}$ is D-minimal if $L_\omega(\mathcal{A}_p) \not\equiv_D L_\omega(\mathcal{A}_q)$, for all states $p, q \in Q$ with $p \neq q$.*

Proof. Let $L \subseteq \Gamma^\omega$ be an ω-language with $L \equiv_D L_\omega(\mathcal{A})$. Assume that $\mathcal{B}$ is a WDBA that has less than $|Q|$ states and $L_\omega(\mathcal{B}) \equiv_D L$. Without loss of generality, we assume that every state of $\mathcal{B}$ is reachable from its initial state. By Lemma 23 and the pigeon hole principle, there is a state s of $\mathcal{B}$ such that $u_p, u_q \in [u_s]_D^L$, for some state $p, q \in Q$ with $p \neq q$. Note that $u_s, u_p, u_q \in \Gamma^*$ denote the words from Lemma 23. It follows that $[u_p]_D^L = [u_q]_D^L$. This contradicts the assumption that $L_\omega(\mathcal{A}_p) \not\equiv_D L_\omega(\mathcal{A}_q)$. □

Lemma 25. *Let $\mathcal{A}$ and $\mathcal{B}$ be WDBAs in D-normal form. If $L_\omega(\mathcal{A}) \equiv_D L_\omega(\mathcal{B})$ then $L_*(\mathcal{A}) = L_*(\mathcal{B})$.*

Proof. Assume that $\mathcal{A} = (Q, \Gamma, \delta, q_\mathrm{I}, F_c)$ and $\mathcal{B} = (Q', \Gamma, \delta', q'_\mathrm{I}, F_{c'})$, where $c : Q \to \mathbb{N}$ is a ℓ-maximal D-coloring for $\mathcal{A}$, $c' : Q' \to \mathbb{N}$ is a ℓ'-maximal D-coloring for $\mathcal{B}$, and ℓ, ℓ' are even. Without loss of generality, we can assume that c and c' are k-maximal for some sufficiently large k by adding appropriate constants to the c and c' values. In the remainder of the proof, we make use of the following notation: For a word $w \in \Gamma^*$, we write q_w for $\hat{\delta}(q_\mathrm{I}, w)$ and q'_w for $\hat{\delta}'(q'_\mathrm{I}, w)$.

For the sake of contradiction, we assume that there is a $w \in \Gamma^*$ such that $c(q_w) \neq c'(q'_w)$. We only consider the case $c(q_w) < c'(q'_w)$. The other case is symmetric. We define a function $d : Q \to \mathbb{N}$ by $d(q_v) := \max\{c(q_v), c'(q'_v)\}$, for $v \in \Gamma^*$. Note that d is well-defined because if $q'_u = q'_v$ then $c'(q'_u) = c'(q'_v)$, for all $u, v \in \Gamma^*$. This fact follows from $u \approx_D^{L_\omega(\mathcal{A})} v$ and Lemma 19. In the following, we show that d is a D-coloring for $\mathcal{A}$. This contradicts the k-maximality of the D-coloring c and hence, $c(q_w) = c'(q'_w)$. Since w was chosen arbitrarily, we conclude that for all $u \in \Gamma^*$, $q_u \in F_c \Leftrightarrow q'_u \in F_{c'}$, i.e., $L_*(\mathcal{A}) = L_*(\mathcal{B})$.

By the definition of d, we have that $d(q) \leq d(\delta(q, b))$, for all $q \in Q$ and $b \in \Gamma$. It remains to show that $d(q)$ is even $\Leftrightarrow q \in F_c$, for every D-recurrent state $q \in Q$. So, let $q \in Q$ be a D-recurrent state and let $u \in \Gamma^*$ such that $q = q_u$. If $d(q) = c(q)$ there is nothing to prove. Assume that $d(q) \neq c(q)$, i.e., $d(q) = c'(q'_u)$. Since q is D-recurrent, there is an ω-word $\alpha \in L_\omega(\mathcal{A}') \setminus D$, where $\mathcal{A}'$ is the WDBA $(Q, \Gamma, \delta, q, \mathrm{SCC}(q))$. Without loss of generality, we assume that the run of $\mathcal{A}'$ on α visits infinitely often the state q. Since the run of $\mathcal{A}$ on $u\alpha$ infinitely often visits q, infinitely many prefixes of $u\alpha$ are $\approx_D^{L_\omega(\mathcal{A})}$-equivalent to u. Thus, the run ϑ' of $\mathcal{A}'$ on $u\alpha$ visits infinitely often a state $q' \in Q'$ such that $L_\omega(\mathcal{A}'_{q'}) \equiv_D L_\omega(\mathcal{A}'_{q'_u})$. From Lemma 19 we know that $c'(q') = c'(q'_u)$. It follows that the maximal color seen on the run ϑ' is $d(q)$, i.e., $c'(\vartheta') = c'(q'_u) = d(q)$. Since $\mathcal{A}'$ has to accept $u\alpha$ iff $\mathcal{A}$ accepts $u\alpha$, we conclude that $c'(q'_u)$ is even iff $c(q_u)$ is even. □

Now, we have all the needed ingredients for proving the result about the minimization of WDBAs with the don't care set D.

Proof (Theorem 8). By Lemma 7, we can put $\mathcal{A}$ into D-normal form in time $\mathrm{O}(|Q|)$. In the following, assume that $\mathcal{A}$ is in D-normal form and $c : Q \to \mathbb{N}$ is a k-maximal D-coloring for $\mathcal{A}$, where k is even and $F = F_c$. We apply Hopcroft's DFA minimization algorithm to $\mathcal{A}$. The running time is in $\mathrm{O}(|Q| \log |Q|)$. Let $\mathcal{A}' = (Q', \Gamma, \delta', q'_\mathrm{I}, F')$ be the result of Hopcroft's algorithm.

Note that there is a homomorphism h from $\mathcal{A}$ to $\mathcal{A}'$, i.e., $h : Q \to Q'$ is a function with $h(q_\mathrm{I}) = q'_\mathrm{I}$, $q \in F \Leftrightarrow h(q) \in F'$ and if $\delta(p, b) = q$ then $\delta'(h(p), b) = h(q)$, for all $q \in Q$ and $b \in \Gamma$.

1. The DBA $\mathcal{A}'$ is weak, since h preserves accepting and rejecting states. A cycle in $\mathcal{A}'$ that contains accepting and rejecting states would yield a cycle in $\mathcal{A}$ that contains accepting and rejecting states.
2. The mapping $c' : Q' \to \mathbb{N}$ defined by $c'(q) := \max\{c(p) : h(p) = q \text{ for } p \in Q\}$ is a k-maximal D-coloring for $\mathcal{A}'$. Moreover, it holds that $F' = F_{c'}$.

Since the DFA $\mathcal{A}'$ is minimal, we have that $L_*(\mathcal{A}'_p) \neq L_*(\mathcal{A}'_q)$, for all states $p, q \in Q'$ with $p \neq q$. By Lemma 20, we have that $L_\omega(\mathcal{A}'_p) \not\equiv_D L_\omega(\mathcal{A}'_q)$, for all $p, q \in Q'$ with $p \neq q$. By Corollary 24, we have that the WDBA $\mathcal{A}'$ is D-minimal.

Assume that $\mathcal{B}$ is a D-minimal WDBA in D-normal form with $L_\omega(\mathcal{A}') \equiv_D L_\omega(\mathcal{B})$. From Lemma 25, it follows that $L_*(\mathcal{A}') = L_*(\mathcal{B})$. The WDBAs $\mathcal{A}'$ and $\mathcal{B}$ are isomorphic, since $\mathcal{A}'$ and $\mathcal{B}$ are minimal DFAs. □

B Proof Details for the Automata Construction for the Existential Quantification

Lemma 11 is straightforward to prove. We omit it.

B.1 Proof Details of Lemma 12

The proof of Lemma 12 about the relation M is as follows. Recall the claim, which comprises two parts:

(a) If $\delta(p,(b,\varrho-1)) \in \mathrm{SCC}(p)$ and pMq then $\delta(p,(b,\varrho-1))M\delta(q,(b,0))$, and
(b) If $\delta(p,(b,c)) \in \mathrm{SCC}(p)$ and $p \in F$ then $\delta(p,(b,c))M\delta(p,(b,c+1))$,

where $p, q \in Q$ are states of the WDBA $\mathcal{A} = (Q, \Sigma^r \cup \{\star\}, \delta, q_\mathrm{I}, F)$, $b \in \Sigma^{r-1}$, $c \in \Sigma \setminus \{\varrho - 1\}$.

Proof (Lemma 12). Let $\mathcal{A}'$ be the WDBA $(Q, \Sigma^r \cup \{\star\}, \delta, p, \mathrm{SCC}(p))$.
(a): Let $p' := \delta(p,(b,\varrho-1))$ and $q' := \delta(q,(b,0))$. Assume that $\mathcal{A}'_{p'}$ accepts the ω-word $(\alpha,(\varrho-1)^\omega)$. Then, $\mathcal{A}'$ accepts the ω-word $(b\alpha,(\varrho-1)^\omega)$. From the assumption pMq, it follows that $\mathcal{A}_q$ accepts the ω-word $(b\alpha, 0^\omega)$. It follows that $\mathcal{A}_{q'}$ accepts the ω-word $(\alpha, 0^\omega)$. We conclude that $p'Mq'$.
(b): Let $p' := \delta(p,(b,c))$ and $q' := \delta(p,(b,c+1))$. For the sake of contradiction, assume that there is an ω-word $\alpha \notin (\Sigma^{r-1})^\omega \setminus \mathsf{DC}_{r-1}$ such that $(\alpha,(\varrho-1)^\omega) \in L_\omega(\mathcal{A}'_{p'})$ and $(\alpha, 0^\omega) \notin L_\omega(\mathcal{A}_{q'})$.

Let $u \in (\Sigma^r \cup \{\star\})^*$ be a word such that $\hat{\delta}(q_\mathrm{I}, u) = p$. Moreover, let $v \in (\Sigma^r)^*$ be a word such that $((b,c)v)^\omega \notin \mathsf{DC}_r$ and $\hat{\delta}(p', v) = p$. Note that such a word v exists, since $\alpha \notin \mathsf{DC}_{r-1}$ and we do not leave the SCC of p when reading $(\alpha,(\varrho-1)^\omega)$ from p'. Let $\mathcal{D} = (P, \Sigma^r \cup \{\star\}, \mu, p_\mathrm{I}, E)$ be a WDBA that accepts $L(\varphi)$.

We define a sequence of words $w_0, w_1, \dots$ such that w_i is a proper prefix of w_{i+1} and $\hat{\delta}(q_\mathrm{I}, w_i) = p$, for all $i \in \mathbb{N}$. Let $w_0 := u$ and for $i > 0$, we define $w_i := w_{i-1}ww'$, where the words $w, w' \in (\Sigma^r)^*$ are defined as follows depending on whether $\hat{\mu}(p_\mathrm{I}, w_{i-1})$ is an accepting state or not.

- Case 1: $\hat{\mu}(p_\mathrm{I}, w_{i-1}) \in E$. The word w is a finite prefix of the ω-word $(b\alpha, c(\varrho-1)^\omega)$ such that $\hat{\mu}(p_\mathrm{I}, w_{i-1}w) \notin E$. Such a word w exists, since $\langle\!\langle w_{i-1}(b\alpha,(c+1)0^\omega)\rangle\!\rangle = \langle\!\langle w_{i-1}(b\alpha, c(\varrho-1)^\omega)\rangle\!\rangle$ and the don't care word $w_{i-1}(b\alpha,(c+1)0^\omega)$ is rejected by $\mathcal{A}$, and hence, $\mathcal{D}$ has to reject $w_{i-1}(b\alpha, c(\varrho-1)^\omega)$. The word w' is chosen in such way that $\hat{\delta}(q_\mathrm{I}, w_{i-1}ww') = p$. The existence of w' is guaranteed, since we stay in the SCC of p when reading w from p.

– Case 2: $\hat{\mu}(p_\mathrm{I}, w_{i-1}) \notin E$. The word w is a finite prefix of the ω-word $((b,c)v)^\omega$ such that $\hat{\mu}(p_\mathrm{I}, w_{i-1}w) \in E$. Such a word w exists, since $w_{i-1}((b,c)v)^\omega$ is a don't care word and accepted by $\mathcal{A}$. Hence, $\mathcal{D}$ has to accept it. The word w' is chosen as in Case 1, i.e., such that $\hat{\delta}(q_\mathrm{I}, w_{i-1}ww') = p$. This is also always possible, since we do not leave the SCC of p when reading w from p.

Note that in both cases the length of w is greater than or equal to 1. Let γ be the ω-word that is the closure of the construction of these w_is. We observe that the run of $\mathcal{D}$ on γ infinitely often alternates between accepting and rejecting states. This contradicts the weakness of $\mathcal{D}$. □

B.2 Proof Details of Lemma 13

Proof (Lemma 13).
($\subseteq$): Assume that $\beta \in \overline{L}_\omega(\mathcal{B}) \setminus \mathsf{DC}_{r-1}$. It is straightforward to see that $\beta \in \mathsf{V}_{r-1}$, i.e., β is of the form $\beta_I \star \beta_F$, where $\beta_I \in (\Sigma^{r-1})^+$ with $\beta(0) \in \{0, \varrho - 1\}^{r-1}$ and $\beta_F \in (\Sigma^{r-1})^\omega \setminus \mathsf{DC}_{r-1}$.

Let $\vartheta \in (K \times K)^\omega$ be the run of $\mathcal{B}$ on β, where $\vartheta(i) = (R_i, S_i)$, for $i \in \mathbb{N}$. According to the co-Büchi acceptance condition there is some $k \in \mathbb{N}$ such that $S_j \neq \emptyset$, for all $j \geq k$. We assume without loss of generality that $k > |\beta_I|$. For showing that $\beta \in L(\exists x_r \varphi)$, it suffices to show that there is an ω-word $\alpha \in L_\omega(\mathcal{A}) \setminus \mathsf{DC}_r$ such that $\langle\!\langle \alpha_{\restriction 1,r-1} \rangle\!\rangle = \langle\!\langle \beta \rangle\!\rangle$. In order to show the existence of such an ω-word α, we need the following fact, which we prove later: there are $\gamma \in (\Sigma^r)^\omega$ and $\xi \in Q^\omega$ with the properties:

(i) $\gamma(i)_{\restriction 1,r-1} = \beta(k+i)$, for all $i \in \mathbb{N}$,
(ii) ξ is a run of $\mathcal{A}_{\xi(0)}$ on γ and
(iii) $\xi(i) \in S'_{k+i}$, for all $i \in \mathbb{N}$, where $S'_{k+i} := \{p : (p,q) \in S_{k+i}\}$.

Note that the run ξ is accepting since $\xi(i) \in S'_{k+i}$, for all $i \in \mathbb{N}$.

Since $\xi(0) \in S'_k$, there is a word $v \in (\Sigma^r \cup \{\star\})^+$ and $(p,q) \in I(\beta(0))$ such that $\hat{\delta}(p, v) = \xi(0)$ and $v_{\restriction 1,r-1}$ is a prefix of β. Moreover, there is a word $u \in \Sigma^+$ such that $\hat{\delta}(q_\mathrm{I}, (\beta(0)^{|u|}, u)) = p$ and $\hat{\delta}(q_\mathrm{I}, (\beta(0)^{|\overline{u}|}, \overline{u})) = q$. Let $\alpha' := (\beta(0)^{|u|}, u)v\gamma$.

If $\alpha' \notin \mathsf{DC}_r$ then α' is the ω-word α we are looking for. So, assume that $\alpha' \in \mathsf{DC}_r$. For some $k' \in \mathbb{N}$, we have that $\gamma(i)_{\restriction r} = \varrho - 1$ and $\xi(i) \in \mathrm{SCC}(\xi(k'))$, for all $i \geq k'$. Without loss of generality we can assume that $k' = 0$, since k can be chosen arbitrarily large. From $\xi(0) \in S'_k$ it follows that there is a state $s \in Q$, such that $(\xi(0), s) \in S_k$ and $\xi(0) M s$. Since the WDBA $\mathcal{A}' = (Q, \Sigma^r \cup \{\star\}, \delta, \xi(0), \mathrm{SCC}(\xi(0)))$ accepts the ω-word

$$\binom{\gamma(0)_{\restriction 1,r-1}}{\varrho-1}\binom{\gamma(1)_{\restriction 1,r-1}}{\varrho-1}\dots,$$

if follows from the definition of M that $\mathcal{A}_s$ accepts the ω-word

$$\gamma' := \binom{\gamma(0)_{\restriction 1,r-1}}{0}\binom{\gamma(1)_{\restriction 1,r-1}}{0}\dots.$$

Let $\xi' \in Q^\omega$ be the accepting run of $\mathcal{A}_s$ on γ'. We make a case split on v.

Case I: there is no $j \in \mathbb{N}$ such that $v(j) \in \Sigma^{r-1} \times \{0, \dots, \varrho - 2\}$. Let $v' \in (\Sigma^r \cup \{\star\})^+$ be the word of length $|v|$ such that

$$v'(i) := \begin{cases} \star & \text{if } v(i) = \star, \\ \big(v(i)_{\restriction 1,r-1}, 0)\big) & \text{if } v(i) \in \Sigma^r, \end{cases}$$

where $0 \le i < |v|$. From Lemma 11 it follows that $\langle\!\langle \overline{u} v'_{\restriction r} \gamma'_{\restriction r} \rangle\!\rangle = \langle\!\langle \alpha'_{\restriction r} \rangle\!\rangle$. We have that $\hat{\delta}(q, v') = s$. From this and from the run ξ' it is straightforward to construct an accepting run of $\mathcal{A}$ on the ω-word $\alpha := (\beta(0)^{|\overline{u}|}, \overline{u}) v' \gamma'$.

Case II: There is a $j \in \mathbb{N}$ such that $v(j) \in \Sigma^{r-1} \times \{0, \dots, \varrho - 2\}$. Let j be maximal, i.e., $j := \max\{i : v(i) \in \Sigma^{r-1} \times \{0, \dots, \varrho - 2\}\}$. Let $v' \in (\Sigma^+ \cup \{\star\})^+$ of length $|v|$ such that

$$v'(i) := \begin{cases} \star & \text{if } v(i) = \star, \\ v(i) & \text{if } i < j \text{ and } v(i) \in \Sigma^r, \\ \big(v(i)_{\restriction 1,r-1}, v(i)_{\restriction r} + 1\big) & \text{if } i = j \text{ and } v(i) \in \Sigma^r, \\ \big(v(i)_{\restriction 1,r-1}, 0\big) & \text{if } i > j \text{ and } v(i) \in \Sigma^r, \end{cases}$$

where $1 \le i < |v|$. Let $\alpha := (\beta(0)^{|u|}, u) v' \gamma'$. It is easy to verify that $\langle\!\langle \alpha_{\restriction r} \rangle\!\rangle = \langle\!\langle \alpha'_{\restriction r} \rangle\!\rangle$. Analogously as in the above case, we can construct an accepting run on α and hence, α is the ω-word we are looking for.

It remains to prove the existence of the ω-word $\gamma \in (\Sigma^r)^\omega$ and the run $\xi \in Q^\omega$ with the properties (i)–(iii). We define the graph G with the set of vertices $V := \bigcup_{i \ge 0} (S_{k+i} \times \{i\})$. There is an edge from $((p,q), i)$ to $((p', q'), j)$ in G iff $j = i + 1$ and at least one of the two conditions holds:

1. $p' = \delta(p, (\beta(k+i), \varrho - 1))$ and $q' = \delta(q, (\beta(k+i), 0))$;
2. $p' = \delta(p, (\beta(k+i), c))$ and $q' = \delta(p', (\beta(k+i), c+1))$, for some $c \in \Sigma \setminus \{\varrho - 1\}$.

Note that graph G is finitely branching and V is infinite. Moreover, note that every vertex in G is reachable from some vertex $((p,q), 0)$, where $(p,q) \in S_k$. $R(v)$ denotes the set of vertices that are reachable from vertex $v \in V$. The finite boundary B is the set of vertices $v \in V$ for which $R(v)$ is finite.

It is easy to see by contradiction that there is a vertex $v \in V \setminus B$, where v is of the form $((p,q), 0)$. Assume that there is no such vertex v in $V \setminus B$, i.e., $R((p,q), 0)$ is finite, for every vertex $((p,q), 0) \in V$. We obtain a contradiction since G is an infinite, finitely branching graph, where all states are reachable from some vertex of the form $((p,q), 0)$. Note that there are only finitely many such vertices $((p,q), 0)$, since S_k is finite.

We claim that for every $v \in V \setminus B$, there is an infinite path π in G starting from v. The existence of π follows from König's Lemma, since $R(v) \setminus B$ is infinite and G is finitely branching. From an infinite path starting in a vertex $((p,q), 0)$ it is straightforward to define γ and ξ.

($\supseteq$): Assume that $\beta \in L(\exists x_r \varphi) \setminus \mathsf{DC}_{r-1}$. There is an ω-word $\alpha \in L_\omega(\mathcal{A}) \setminus \mathsf{DC}_r$ with $\beta = \alpha_{\restriction 1,r-1}$. Let $\vartheta \in Q^\omega$ be the run of $\mathcal{A}$ on α and let ϑ' bet the run of

$\mathcal{B}$ on β. Assume that $\vartheta'(i) = (R_i, S_i)$, for $i \geq 0$. We have to show that ϑ' is accepting, i.e., there is a $k \in \mathbb{N}$ such that $S_j \neq \emptyset$, for all $j \geq k$.

Since ϑ is accepting, there is an SCC $S \subseteq F$ and an integer $k \geq 1$ such that $\vartheta(i) \in S$, for all $j \geq k$. Without loss of generality, we assume that $\alpha(k)_{\restriction r} \neq \varrho - 1$ and $\alpha(i) \neq \star$, for all $i \geq k$. Let $J := \{j \in \mathbb{N} : j \geq k \text{ and } \alpha(j)_{\restriction r} \neq \varrho - 1\}$ and let $\zeta \in Q^\omega$ be the ω-word defined as

$$\zeta(i+1) := \begin{cases} \delta\big(\vartheta(i), (\alpha(i)_{\restriction 1,r-1}, \alpha(i)_{\restriction r} + 1)\big) & \text{if } i \in J, \\ \delta\big(\zeta(i), (\alpha(i)_{\restriction 1,r-1}, 0)\big) & \text{if } i \notin J, \end{cases}$$

for $i \geq k$ and $\zeta(i) := q_{\mathrm{I}}$, for $i \leq k$. By the definition of $\mathcal{B}$'s transition function η, we have that $(\vartheta(j), \zeta(j)) \in R_j$, for all $j > k$. It suffices to show that $\vartheta(j+1)M\zeta(j+1)$, for all $j \geq k$.

Case I: $j \in J$. Since $\vartheta(j) \in S$, $\delta(\vartheta(j), \alpha(j)) \in S$, and $\alpha(j)_{\restriction r} \neq \varrho - 1$, we conclude from Lemma 12(b) that $\vartheta(j+1)M\zeta(j+1)$.

Case II: $j \notin J$. Let $j' := \max\{i \in J : i < j\}$. Note that $\{i \in J : i < j\}$ is nonempty because $k \in J$. From the previous case we know that $\vartheta(j'+1)M\zeta(j'+1)$. Using Lemma 12(a) it is straightforward to show that $\vartheta(i+1)M\zeta(i+1)$, for all $i \in \{j'+1, \dots, j\}$. Note that $\vartheta(i+1) = \delta(\vartheta(i), \alpha(i)) \in S$ and $\alpha(i)_{\restriction r} = \varrho - 1$. □

B.3 Proof Details of Lemma 14

Our proof of Lemma 14 uses results from the article [5] by Boigelot, Jodogne, and Wolper on the "dense oscillating sequence property" for ω-languages. We refer the reader to this article for further details on this topic.

Proof (Lemma 14). There is nothing to prove if there are no ω-words $\alpha, \beta \in \mathsf{V}_s \setminus \mathsf{DC}_s$ such that (1) the run on α visits a state in $S \cap E$ infinitely often and (2) the run on β visits only states in $S \setminus E$ infinitely often. Assume that such $\alpha, \beta \in \mathsf{V}_s \setminus \mathsf{DC}_s$ with (1) and (2) exist. Note that $\alpha \notin \overline{L}_\omega(\mathcal{C})$ and $\beta \in \overline{L}_\omega(\mathcal{C})$.

From the existence of the ω-word α, we conclude that there is a state $p \in S \cap E$ and words $u, w \in (\Sigma^s \cup \{\star\})^+$ such that $\hat{\mu}(p, u) = p$, $\hat{\mu}(p_{\mathrm{I}}, w) = p$, and $wu^\omega \in \mathsf{V}_s \setminus \mathsf{DC}_s$. From the existence of the ω-word β, we conclude that there is a state $q \in S \setminus F$ and word $v \in (\Sigma^s)^+$ such that $\hat{\mu}(q, v) = q$, $v^\omega \notin \mathsf{DC}_s$. and no accepting state is visited when reading v from q. Moreover, since p and q are in the same SCC, there are words $w_1, w_2 \in (\Sigma^s)^+$ such that $\hat{\mu}(p, w_1) = q$ and $\hat{\mu}(q, w_2) = p$. For a sequence $k_2, k_3, \ldots \in \mathbb{N}$, we define the words $x_1 := w$ and

$$x_i := \begin{cases} x_{i-1} v^{k_i} w_2 & \text{if } i \text{ is odd}, \\ x_{i-1} u^{k_i} w_1 & \text{if } i \text{ is even}, \end{cases}$$

for $i > 1$. Furthermore, we define the ω-words

$$\gamma_i := \begin{cases} x_i u^\omega & \text{if } i \text{ is odd}, \\ x_i v^\omega & \text{if } i \text{ is even}, \end{cases}$$

for $i \geq 1$. Note that $\gamma_i \in \mathsf{V}_s \setminus \mathsf{DC}_s$, for all $i \geq 1$. and $\gamma_i \in \overline{L}_\omega(\mathcal{C})$ iff i is even, for every $i \geq 1$. It follows that $\gamma_i \in L(\psi)$ iff i is even, for $i \geq 1$.

Similar to Lemma 5.4 in [5], it follows from the γ_is that $L(\psi)$ has the "dense oscillating sequence property." We conclude by Lemma 5.5 in [5] that $L(\psi)$ is not in the Borel class $F_\sigma \cap G_\delta$ with respect to the ω-word distance. We obtain a contradiction, since from the Theorems 3.1 and 6.1 in [5], it follows that $L(\psi)$ is in the Borel class $F_\sigma \cap G_\delta$ with respect to the ω-word distance. $\square$