Fundamenta Informaticae 189(1) : 69–91 (2022)
Available at IOS Press through:
https://doi.org/10.3233/FI-222152

arXiv:2211.08787v5 [cs.FL] 13 Jun 2023

On Minimization and Learning of Deterministic ω-Automata in the Presence of Don't Care Words

Christof Löding[*†], **Max Philip Stachon**
Department of Computer Science
RWTH Aachen University, Germany
loeding@automata.rwth-aachen.de
max.stachon@rwth-aachen.de

Abstract. We study minimization problems for deterministic ω-automata in the presence of don't care words. We prove that the number of priorities in deterministic parity automata can be efficiently minimized under an arbitrary set of don't care words. We derive that from a more general result from which one also obtains an efficient minimization algorithm for deterministic parity automata with informative right-congruence (without don't care words). We then analyze languages of don't care words with a trivial right-congruence. For such sets of don't care words it is known that weak deterministic Büchi automata (WDBA) have a unique minimal automaton that can be efficiently computed from a given WDBA (Eisinger, Klaedtke 2006). We give a congruence-based characterization of the corresponding minimal WDBA, and show that the don't care minimization results for WDBA do not extend to deterministic ω-automata with informative right-congruence: for this class there is no unique minimal automaton for a given don't care set with trivial right congruence, and the minimization problem is NP-hard. Finally, we extend an active learning algorithm for WDBA (Maler, Pnueli 1995) to the setting with an additional set of don't care words with trivial right-congruence.

Keywords: deterministic ω-automata, don't care words, learning, minimization

[*]Supported by DFG grant LO 1174/7-1
[†]Address for correspondence: Department of Computer Science, RWTH Aachen University, Germany.

Received November 2022; accepted April 2023.

1. Introduction

In this paper we consider minimization and learning problems with don't care sets for deterministic automata on infinite words, called ω-automata in the following. For finite words, it is well known that there is a unique minimal deterministic finite automaton (DFA) for each regular language L, which can be computed efficiently by merging language equivalent states in a given DFA for L (see, e.g., [1]). The minimal DFA can be characterized by the Myhill-Nerode congruence on finite words, in which two words are equivalent if they have the same extensions into the language L. The states of the minimal DFA correspond to the equivalence classes of this congruence. This characterization is also the basis for active learning algorithms that identify a DFA for an unknown language L by equivalence and membership queries [2].

In some applications, some input words might not play a role because they do not occur and thus it does not matter whether the automaton accepts them. Such a set of input words is called a don't care set in this paper. The minimization problem with a given don't care set is then to compute a minimal automaton that behaves the same as the given one on all words that are not in the don't care set. Minimal automata in this setting need not to be unique, and the problem (its decision variant) becomes NP-hard [3].

For deterministic ω-automata, the minimization problem (without don't cares) is harder than for DFAs. There is a variety of acceptance conditions for ω-automata, the most basic one being the Büchi condition. A deterministic Büchi automaton (DBA) has a set of accepting states, like a DFA, and a run (an infinite state sequence) is accepting if an accepting state is visited infinitely often. It turns out that the minimization problem for DBA is NP-hard [4], and minimal DBA for an ω-language are, in general, not unique. Since a deterministic parity automaton (DPA) for a DBA-recognizable language can be turned into an equivalent DBA without changing the transition structure (as shown in [5] for Rabin conditions, which include parity conditions), minimization for DPA is also NP-hard. In a parity condition, each state is assigned a priority (a natural number), and a run is accepting if the maximal priority that occurs infinitely often is even. DPAs are interesting for algorithmic use because they capture the full class of regular ω-languages (see [6, 7] for background on ω-automata), and their acceptance condition is specified in a very compact way. For this reason we mainly study DPA and subclasses thereof in this paper (note that a Büchi condition can be expressed with priorities 2 for accepting and 1 for non-accepting).

The only class of deterministic ω-automata for which an efficient minimization algorithm is known is the one of weak deterministic Büchi automata (WDBA) [8], in which every strongly connected set consists only of accepting states, or only of non-accepting states. Minimal WDBA can also be characterized by the Myhill-Nerode congruence (its natural extension to infinite words) [9, 10, 11], and (a variant of) the active learning algorithm for DFAs from [2] has been adapted to WDBA [12]. As for minimization, this is the only class of ω-automata for which a polynomial time active learning algorithm is known. Other active learning algorithms for regular ω-languages as in [13] learn different representations that can only be translated to deterministic ω-automata with an exponential cost.

In [14] it is shown that there is an interesting class of don't care sets for which the minimization problem for WDBAs can be solved efficiently: If the don't care set is a regular ω-language with a trivial Myhill-Nerode congruence consisting only of one class (in the following, we refer to the Myhill-

Nerode congruence simply as the canonical right-congruence of the language). For finite words the only languages with this property are the trivial languages (all words or the empty language). For ω-words, however, all languages in which membership only depends on the infinite suffix of the word have a trivial right-congruence, for example, the set of all infinite words with infinitely many b. In [14] such don't care sets are used for reducing the size of WDBAs for sets of real numbers by making certain representations of real numbers don't cares.

This raises one of the questions that we study in this paper, which we answer positively: Can the active learning algorithm from [12] be extended to this setting of a don't care set with trivial right-congruence?

We also investigate whether the minimization result for WDBAs with don't care sets can be extended to larger classes of ω-automata. Since the correspondence between the states of the minimal automaton and the classes of the right-congruence plays an important role in all these results, we consider the class of deterministic ω-automata with informative right-congruence (abbreviated IRC) [15]. These are automata that use only one state per equivalence class, so their transition system is isomorphic to the transition system induced by the canonical right-congruence of the accepted language. Minimal WDBA have informative right-congruence, and we study here the question whether efficient minimization of DPA with IRC is possible under a don't care set with trivial right-congruence.

Finally, we also consider the problem of optimizing the acceptance condition of a DPA under a don't care set. It is known that for a given DPA, one can efficiently compute an equivalent parity condition on the same transition structure that uses the minimal number of different priorities [16]. In [17] such a priority minimization has turned out to be a useful step before applying heuristics for reducing the state space of DPA resulting from a determinization procedure. The number of priorities might be further reducible if we are given a set of don't cares.

Our contributions on these questions can be summarized as follows:

- We show that polynomial time priority minimization on the transition structure of a given DPA is possible for a don't care set that is given by a DPA (in this case, it does not need to have trivial right-congruence).

- We show that the minimization result for WDBA under don't care sets with trivial right-congruence does not extend to DPA with IRC: Given a DPA with IRC and a don't care set with trivial right-congruence, there is, in general, not a unique minimal DPA under this don't care set, and (the decision variant of) this don't care minimization problem is NP-hard.

- The active learning algorithm for WDBAs can be adapted to the setting with a don't care set with trivial right-congruence.

The remainder of this paper is structured as follows. In Section 2 we introduce basic notation and terminology. In Section 3 we consider the problem of optimizing parity conditions under a given set of don't care words. The state minimization problem for don't care sets with trivial right-congruence is discussed in Section 4, and in Section 5 we present the active learning algorithm for WDBAs under a don't care set with trivial right-congruence. In Section 6 we conclude.

2. Preliminaries

An alphabet Σ is a finite set whose elements are called letters or symbols. By Σ^* we denote the set of all finite words over Σ, by ϵ the empty word, $\Sigma^+ := \Sigma^* \backslash \{\epsilon\}$, and Σ^ω denotes the set of all ω-words over Σ. For $\alpha \in \Sigma^\omega$ we write $\alpha(i)$ for the letter of α at position i. An ω-word $\alpha \in \Sigma^\omega$ is ultimately periodic if $\alpha = uv^\omega := uvvv\ldots$ for some $u, v \in \Sigma^*$ with $v \neq \epsilon$. An ω-language is a subset of Σ^ω, simply referred to as language in the following. The complement of a language is denoted by $\overline{L} = \Sigma^\omega \setminus L$.

Deterministic ω-automata consist of a deterministic transition graph (or deterministic transition system), and an acceptance condition. The deterministic transition graph is a tuple $\mathcal{T} = (\Sigma, Q, \delta, q_0)$, where Σ is an alphabet, Q is a finite set of states, $\delta : Q \times \Sigma \to Q$ is a transition function, and q_0 is the initial state. We define the extended transition function as $\delta^* : \Sigma^* \to Q$ with $\delta^*(q, \epsilon) := q$ and $\delta^*(q, \sigma w) = \delta^*(\delta(q, \sigma), w)$ for $q \in Q$, $\sigma \in \Sigma$, and $w \in \Sigma^*$. We sometimes write $\delta^*(w)$ for $\delta^*(q_0, w)$. Further, we assume that all states are reachable from the initial state.

We say that two states $p, q \in Q$ are strongly connected if we can reach p from q and vice versa, that is, there are u, v with $\delta^*(p, u) = q$ and $\delta^*(q, v) = p$. A strongly connected component (SCC) is a subset of states that are strongly connected, and a maximal strongly connected component (MSCC) is an SCC that is not a proper subset of any other SCC. We define $\text{MSCC}(q) := \{p \in Q \mid p \text{ is in the same MSCC as } q\}$.

For $\alpha \in \Sigma^\omega$, the run of α in $\mathcal{T}$ is an infinite sequence of states $\rho \in Q^\omega$ with $\rho(0) = q_0$ and $\rho(i+1) = \delta(\rho(i), \alpha(i))$ for all $i \in \mathbb{N}$. The set of states that occur infinitely often in this run is called the infinity set, and is denoted by $Inf(\rho) := \{q \in Q : \rho(i) = q \text{ for infinitely many } i \in \mathbb{N}\}$. Instead of referring to ρ directly, we often write $Inf_{\mathcal{T}}(\alpha)$ for the set of states occurring infinitely often in the run of α in $\mathcal{T}$ (we also use this notation for automata instead of transition systems).

Below we define different standard types of ω-automata. A deterministic Büchi automaton (DBA) $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$ has an acceptance condition in the form of a set $F \subseteq Q$ of accepting (or final) states. An ω-word is accepted by the DBA if and only if it produces a run that visits an accepting state infinitely often. More formally, we say that $\mathcal{A}$ recognizes the ω-language $L(\mathcal{A}) = \{\alpha \in \Sigma^\omega \mid Inf_{\mathcal{A}}(\alpha) \cap F \neq \emptyset\}$. The class of DBA-recognizable ω-languages is denoted as $\mathbb{DB}$.

A deterministic co-Büchi automaton (DCA) $\mathcal{B} = (\Sigma, Q, \delta, q_0, F)$ also has a set $F \subseteq Q$ of accepting states and recognizes the ω-language $L(\mathcal{B}) = \{\alpha \in \Sigma^\omega \mid Inf_{\mathcal{A}}(\alpha) \subseteq F\}$. In other words, a DCA accepts an ω-word if and only if it produces a run that exclusively visits accepting states infinitely often. We denote the class of DCA-recognizable ω-language as $\mathbb{DC}$. Note that for a DBA $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$, the DCA $\mathcal{B} = (\Sigma, Q, \delta, q_0, Q \backslash F)$ recognizes $\overline{L(\mathcal{A})}$ (while the classes $\mathbb{DB}$ and $\mathbb{DC}$ are not closed under complement, see [6]).

A weak deterministic Büchi automaton (WDBA) is a DBA with the additional property that each MSCC is either contained in F or disjoint from F. It follows that any WDBA can be interpreted as a DCA recognizing the same ω-language. Furthermore, the intersection $\mathbb{DB} \cap \mathbb{DC}$ defines precisely the class of WDBA-recognizable ω-languages [9, 10, 11].

A deterministic parity automaton (DPA) $\mathcal{A} = (\Sigma, Q, \delta, q_0, c)$ has an acceptance condition in the form of a priority function $c : Q \to \mathbb{N}$. An ω-word is accepted by the DPA if and only if it produces a run such that the highest priority of any state that is visited infinitely often is even. More formally, we

say that $\mathcal{A}$ recognizes the ω-language $L(\mathcal{A}) = \{\alpha \in \Sigma^\omega \mid \max\{c(q) \mid q \in \mathit{Inf}_{\mathcal{A}}(\alpha)\}$ is even.$\}$. For a set $P \subseteq Q$ of states we let $c(P) = \{c(p) \mid p \in P\}$. For a deterministic Muller automaton (DMA) $\mathcal{A} = (\Sigma, Q, \delta, q_0, \mathcal{F})$ the acceptance condition is given as the set of accepting infinity sets $\mathcal{F} \subseteq 2^Q$ with $L(\mathcal{A}) = \{\alpha \in \Sigma^\omega \mid \mathit{Inf}_{\mathcal{A}}(\alpha) \in \mathcal{F}\}$. We refer to the class of DPA-recognizable ω-languages as $\mathbb{DP}$ and the class of DMA-recognizable ω-languages as $\mathbb{DM}$.

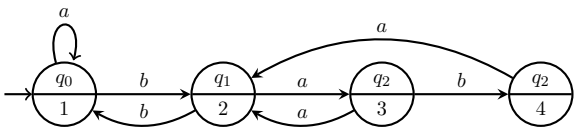
Note that Muller conditions are the most general ones since all the other conditions that are defined here can be expressed as Muller condition. For algorithmic questions, however, Muller conditions are not well suited because a Muller condition needs to explicitly list all accepting infinity sets. For this reason, we mainly consider WDBA, DBA, DCA, and DPA in this article. It is well known that DPA are as expressive as DMA, and that DBA and DCA are less expressive, that is, $\mathbb{DB} \cup \mathbb{DC} \subsetneq \mathbb{DP} = \mathbb{DM}$ (see, e.g., [7]). An ω-language is regular if it can be accepted by a DPA or DMA.

A deterministic ω-automaton is called minimal if there is no deterministic ω-automaton of the same type with less states that accepts the same language. Besides the number of states, there is also the question of the number of priorities used in a DPA. We thus call the priority function of a DPA optimal if there is no priority function on the same transition graph that uses less priorities and accepts the same language. Note that one could also use the number of transitions as criterion for minimality, but for (complete) deterministic automata there is one transition for every combination of state and input letter, so these two measures for minimality are equivalent in our setting.

A right-congruence (on finite words) is an equivalence relation $\sim \subseteq \Sigma^* \times \Sigma^*$ with the property that $u \sim v$ implies $u\sigma \sim v\sigma$ for all $u, v \in \Sigma^*$ and $\sigma \in \Sigma$. We write $[u]_\sim$ for the equivalence class of u in $\sim$. If $\sim$ is clear from the context, we simply write $[u]$. The index of $\sim$, which is the number of equivalence classes of $\sim$, is denoted by $\mathit{Ind}(\sim)$. A right-congruence $\sim$ of finite index induces a finite deterministic transition system $\mathcal{T}_\sim = (\Sigma, Q_\sim, \delta_\sim, [\epsilon]_\sim)$ with $Q_\sim = \{[u]_\sim \mid u \in \Sigma^*\}$ and $\delta_\sim([u], \sigma) = [u\sigma]_\sim$. Vice versa, each deterministic transition system induces a right congruence in which two words are equivalent if they lead to the same state.

For $L \subseteq \Sigma^\omega$, the canonical right-congruence $\sim_L$ of L is defined by $u \sim_L v$ if and only if $(u\alpha \in L \Leftrightarrow v\alpha \in L)$ for all $\alpha \in \Sigma^\omega$. A deterministic ω-automaton that accepts a language L has informative right-congruence (IRC) if its transition graph is isomorphic to $\mathcal{T}_{\sim_L}$. It is not hard to see that, in general, regular ω-languages cannot be accepted by automata with informative right-congruence (see [15] for a detailed analysis of automata with IRC). By $\mathbb{IB}$, $\mathbb{IC}$, $\mathbb{IP}$, and $\mathbb{IM}$ we denote the classes of ω-languages that can be accepted by a deterministic ω-automaton with IRC and the corresponding accepting condition (Büchi, co-Büchi, parity, Muller, respectively). Note that in any DMA (and hence also DBA, DCA, DPA) for L, two words $u, v \in \Sigma^*$ leading to the same state must be $\sim_L$-equivalent. Thus, there is a surjective homomorphism from the transition graph $\mathcal{T}$ of the DMA into $\mathcal{T}_{\sim_L}$. The notion of homomorphism for two transition graphs $\mathcal{T}_1, \mathcal{T}_2$ with $\mathcal{T}_i = (\Sigma, Q_i, \delta_i, q_{0,i})$ is defined as usual. It is a function $h : Q_1 \to Q_2$ with $h(q_{0,1}) = q_{0,2}$ such that $h(\delta_1(q, a)) = \delta_2(h(q), a)$ for all $q \in Q_1$ and $a \in \Sigma$. If there is a surjective homomorphism from $\mathcal{T}_1$ to $\mathcal{T}_2$, we also say that $\mathcal{T}_2$ is a quotient of $\mathcal{T}_1$.

Don't care words are redundant ω-words that we need not consider, that is to say we do not care whether these words are accepted or rejected by an ω-automaton. Let $L, L' \subseteq \Sigma^\omega$ be two ω-languages and $D \subseteq \Sigma^\omega$ a set of don't care words. We say that L and L' are D-equivalent, denoted as $L \equiv_D L'$, if and only if $L \backslash D = L' \backslash D$. We say that a deterministic ω-automaton $\mathcal{A}$ is D-minimal if and only if



With $\mathcal{A} = (\mathcal{T}, c)$ on the left, $D = \Sigma^* a^\omega$, and $\mathcal{T}' = \mathcal{T}$, we obtain

$$\mathcal{F}'_0 = \{\{q_0, q_1, q_2, q_3\}, \{q_1, q_2, q_3\}, \{q_0, q_1\}\}$$
$$\mathcal{F}'_1 = \{\{q_0, q_1, q_2\}\}$$

Figure 1. Illustration of DON'T CARE PRIORITY OPTIMIZATION. An optimal parity condition consistent with $(\mathcal{F}'_0, \mathcal{F}'_1)$ can redefine the priority of q_0 to 2 and thus use one priority less.

there is no deterministic ω-automaton $\mathcal{B}$ of the same type with $L(\mathcal{A}) \equiv_D L(\mathcal{B})$ with less states than $\mathcal{A}$.

3. Priority optimization

In this section we consider the problem of optimizing a parity condition under a given don't care set, referred to as DON'T CARE PRIORITY OPTIMIZATION: Given DPAs $\mathcal{A}, \mathcal{D}$ with $\mathcal{A} = (\mathcal{T}, c)$ and $D := L(\mathcal{D})$, find a parity condition c' on $\mathcal{T}$ with the smallest number of priorities such that $L(\mathcal{T}, c') \equiv_D L(\mathcal{T}, c)$. Figure 1 shows an example of a DPA $\mathcal{A}$ in which one priority less can be used if the don't care set is $\Sigma^* a^\omega$.

We show that this problem can be solved in polynomial time. In fact, we consider a more general problem, in which the transition system on which we compute the parity condition is a quotient of the given DPA $\mathcal{A}$. Formally, we define the problem of generalized parity optimization under don't cares, QUOTIENT DON'T CARE PRIORITY OPTIMIZATION, as follows: Given DPAs $\mathcal{A}, \mathcal{D}$ with $\mathcal{A} = (\mathcal{T}, c)$ and $D := L(\mathcal{D})$, a deterministic transition system $\mathcal{T}'$, and a surjective homomorphism $h : \mathcal{T} \to \mathcal{T}'$, decide if there is a parity condition c' on $\mathcal{T}'$ such that $L(\mathcal{T}', c') \equiv_D L(\mathcal{T}, c)$, and construct such a c' with the smallest possible number of priorities if one exists.

Theorem 3.1. The problem QUOTIENT DON'T CARE PRIORITY OPTIMIZATION can be solved in polynomial time.

The proof of this theorem follows from Lemma 3.2, Lemma 3.4, and Theorem 3.3 below. We start with some definitions.

Let $\mathcal{T} = (\Sigma, Q, \delta, q_0)$ and $\mathcal{T}' = (\Sigma, Q', \delta', q'_0)$, and for $P \subseteq Q$ define $h(P) = \{h(q) \mid q \in P\}$. From the DPA $\mathcal{A}$, the don't care set D, and the homomorphism h, we obtain two families $(\mathcal{F}'_0, \mathcal{F}'_1)$ of subsets of Q', those that need to be accepting, and those that need to be rejecting in order to accept a language that is D-equivalent to $L(\mathcal{A})$. Formally, we define

$$\mathcal{F}'_0 = \{R' \subseteq Q' \mid \exists \alpha \in L(\mathcal{A}) \setminus D \ : \ h(Inf_{\mathcal{A}}(\alpha)) = R'\}$$
$$\mathcal{F}'_1 = \{R' \subseteq Q' \mid \exists \alpha \in \overline{L(\mathcal{A})} \setminus D \ : \ h(Inf_{\mathcal{A}}(\alpha)) = R'\}$$

We say that a parity condition $c' : Q' \to \mathbb{N}$ is consistent with $(\mathcal{F}'_0, \mathcal{F}'_1)$ if $\max(c'(R'))$ is even for all $R' \in \mathcal{F}'_0$, and odd for all $R' \in \mathcal{F}'_1$. See Figure 1 for an example.

Lemma 3.2. With the notations introduced above, $L(\mathcal{T}', c') \equiv_D L(\mathcal{T}, c)$ iff c' is consistent with $(\mathcal{F}'_0, \mathcal{F}'_1)$.

Proof:
The proof is immediate from the definitions: Let $\alpha \in \Sigma^\omega \setminus D$. Since h is a homomorphism, $Inf_{\mathcal{T}'}(\alpha) = h(Inf_{\mathcal{A}}(\alpha))$. Thus, $Inf_{\mathcal{T}'}(\alpha) \in \mathcal{F}'_0$ if $\alpha \in L(\mathcal{A})$, and $Inf_{\mathcal{T}'}(\alpha) \in \mathcal{F}'_1$ if $\alpha \in \overline{L(\mathcal{A})}$. So $L(\mathcal{T}', c') \equiv_D L(\mathcal{T}, c)$ iff $(\mathcal{T}', c')$ accepts precisely those words in $\Sigma^\omega \setminus D$ with infinity set in $\mathcal{F}'_0$, which is the definition of consistency with $(\mathcal{F}'_0, \mathcal{F}'_1)$. □

So for solving the problem QUOTIENT DON'T CARE PRIORITY OPTIMIZATION, we need to find an optimal parity condition c' that is consistent with $(\mathcal{F}'_0, \mathcal{F}'_1)$. There is an algorithm that solves this problem in polynomial time, which is presented in [18]. Unfortunately, the size of $(\mathcal{F}'_0, \mathcal{F}'_1)$ can be exponential in general, so we cannot construct it explicitly in a polynomial time algorithm. However, the algorithm from [18] only needs to access $(\mathcal{F}'_0, \mathcal{F}'_1)$ via queries of the following type: Given a set $P' \subseteq Q'$ and $t \in \{0, 1\}$, return the set

$$U_t(P') = \bigcup \{R' \subseteq P' \mid R' \in \mathcal{F}'_t\},$$

that is, the union of all sets $R' \in \mathcal{F}'_t$ with $R' \subseteq P'$. Let us call such a function that returns $U_t(P')$ a *Subset-Parity-Oracle* for $(\mathcal{F}'_0, \mathcal{F}'_1)$. We can thus reformulate the result from [18] as follows.

Theorem 3.3. ([18, Theorem 4])
There is a polynomial time algorithm that, given a *Subset-Parity-Oracle* for $(\mathcal{F}'_0, \mathcal{F}'_1)$ that is computable in polynomial time, constructs an optimal parity condition c' on $\mathcal{T}'$ that is consistent with $(\mathcal{F}'_0, \mathcal{F}'_1)$ if one exists.

Thus, for a polynomial time algorithm solving QUOTIENT DON'T CARE PRIORITY OPTIMIZATION, it suffices to show that one can implement a *Subset-Parity-Oracle* for $(\mathcal{F}'_0, \mathcal{F}'_1)$ in polynomial time, which is stated in the following lemma.

Lemma 3.4. There is a polynomial time algorithm that, given the inputs $\mathcal{A} = (\mathcal{T}, c), \mathcal{D}, \mathcal{T}', h$ for the problem QUOTIENT DON'T CARE PRIORITY OPTIMIZATION, a set P' of states of $\mathcal{T}'$, and $t \in \{0, 1\}$, returns the set $U_t(P')$.

Proof:
Let $\mathcal{A} = (\mathcal{T}, c)$ with $\mathcal{T} = (\Sigma, Q, \delta, q_0)$, $\mathcal{D} = (\Sigma, Q_{\mathcal{D}}, \delta_{\mathcal{D}}, q_0^{\mathcal{D}}, d)$, and $\mathcal{T}' = (\Sigma, Q', \delta', q'_0)$. We consider the case $t = 0$, the case $t = 1$ being symmetric. Let $R' \subseteq P'$ with $R' \in \mathcal{F}'_0$. By definition of $\mathcal{F}'_0$, there is $\alpha \in L(\mathcal{A}) \setminus D$ with $h(Inf_{\mathcal{A}}(\alpha)) = R'$. For the condition $\alpha \in L(\mathcal{A}) \setminus D$, we consider the product of $\mathcal{A}$ and $\mathcal{D}$, which is obtained by the standard product construction, restricted to reachable states, and with the two priority functions c, d extended to pairs of states. Formally we define $\hat{\mathcal{A}} = \mathcal{A} \times \mathcal{D} = (\Sigma, \hat{Q}, \hat{\delta}, \hat{q_0}, \hat{c}, \hat{d})$ with

- $\hat{q_0} = (q_0, q_0^{\mathcal{D}})$,
- $\hat{Q} = Q \times Q_{\mathcal{D}}$ restricted to the states reachable from $\hat{q_0}$ by application of $\hat{\delta}$,

- $\hat{\delta}((q_1, q_2), a) = (\delta(q_1, a), \delta_{\mathcal{D}}(q_2, a))$
- $\hat{c}(q_1, q_2) = c(q_1)$ and $\hat{d}(q_1, q_2) = d(q_2)$

Then $\alpha \in L(\mathcal{A}) \setminus D$ is equivalent to $\max(\hat{c}(Inf_{\hat{\mathcal{A}}}(\alpha)))$ being even and $\max(\hat{d}(Inf_{\hat{\mathcal{A}}}(\alpha)))$ being odd. This means that the set $Inf_{\hat{\mathcal{A}}}(\alpha)$ is a strongly connected set in $\hat{\mathcal{A}}$ on which the maximal c-priority is even, and the maximal d-priority is odd.

We can view h also as a homomorphism from the product transition system $\hat{\mathcal{A}}$ to $\mathcal{T}'$ by setting $h(q_1, q_2) = h(q_1)$ for all $(q_1, q_2) \in \hat{Q}$. The condition $h(Inf_{\mathcal{A}}(\alpha)) = R'$ then directly translates to $\hat{\mathcal{A}}$ as $h(Inf_{\hat{\mathcal{A}}}(\alpha)) = R'$.

In summary, the sets $R' \in \mathcal{F}_0$ with $R' \subseteq P'$ are precisely those sets $h(S)$ where S is a strongly connected subset of states in $\hat{\mathcal{A}}$ with $h(S) \subseteq P'$, $\max(\hat{c}(S))$ even, and $\max(\hat{d}(S))$ odd.

Hence, for computing $U_0(P')$, we can proceed as follows.

- $U_0 := \emptyset$
- For all even i in the range of c, and all odd j in the range of d:
  - Let $\hat{\mathcal{A}}_{P',i,j}$ be $\hat{\mathcal{A}}$ restricted to the states $(q_1, q_2) \in \hat{Q}$ with $h(q_1) \in P'$, $c(q_1) \leq i$, and $d(q_2) \leq j$.
  - For each MSCC S of $\hat{\mathcal{A}}_{P',i,j}$ with $\max(\hat{c}(S)) = i$ and $\max(\hat{d}(S)) = j$, let $U_0 := U_0 \cup h(S)$.
- Return U_0

This procedure can be implemented to run in polynomial time because there are only polynomially many pairs i, j to consider, the product $\hat{\mathcal{A}}$ and its restrictions $\hat{\mathcal{A}}_{P',i,j}$ as well as the MSCCs can all be computed in polynomial time. □

4. State space minimization

In this section we consider minimization problems for deterministic ω-automata with informative right-congruence under a given don't care set. As explained in the introduction, we only consider don't care sets D with trivial right-congruence. We start with the observation that for such don't care sets, one can define a canonical right-congruence as a generalization of the standard Myhill-Nerode congruence.

In the following, if not stated otherwise, let $L \subseteq \Sigma^\omega$ be a regular ω-language and $D \subseteq \Sigma^\omega$ a regular ω-language of don't care words with trivial right-congruence.

Definition 4.1. **(D-congruence)**
Given $w, w' \in \Sigma^*$, we say that w and w' are D-equivalent for L, denoted as $w \sim_{L,D} w'$, if and only if for all $\alpha \in \Sigma^\omega \setminus D$ it holds that $w\alpha \in L \iff w'\alpha \in L$. Note that $\sim_{L,\emptyset}$ is the same as $\sim_L$.

Clearly, $w \sim_L w'$ implies $w \sim_{L,D} w'$ for any don't care set D. The opposite is obviously not true because for $D = \Sigma^\omega$, all finite words are D-equivalent for every L. If D has trivial right-congruence, it follows that $\alpha \in D \iff w\alpha \in D$ for all $\alpha \in \Sigma^\omega$ and $w \in \Sigma^*$. One can conclude that $\sim_{L,D}$ is a right-congruence, as stated in the following proposition.

Proposition 4.2. For $L, D \subseteq \Sigma^\omega$ such that D has trivial right-congruence, $\sim_{L,D}$ is a right-congruence.

Proof:
That $\sim_{L,D}$ is an equivalence relation directly follows from the definition and does not require D to have trivial right-congruence. Now assume that $u \sim_{L,D} v$ and let $\sigma \in \Sigma$. For every $\alpha \in \Sigma^\omega \setminus D$ we also have $\sigma\alpha \in \Sigma^\omega \setminus D$ since D has trivial right-congruence. Hence $u\sigma\alpha \in L \Leftrightarrow v\sigma\alpha \in L$. □

Definition 4.3. (Informative D-congruence)
Let $\mathcal{T}_{L,D}$ be the transition system associated with $\sim_{L,D}$. A deterministic ω-automaton $\mathcal{A}$ with $L := L(\mathcal{A})$ possesses informative D-congruence if its transition system is isomorphic to $\mathcal{T}_{L,D}$. We also define $\mathbb{IB}(D)$ as the class of those ω-languages L for which there is a DBA $\mathcal{A}$ with informative D-congruence such that $L(\mathcal{A}) \equiv_D L$. The corresponding definitions for DCA, DPA, and DMA define the classes $\mathbb{IC}(D), \mathbb{IP}(D), \mathbb{IM}(D)$. Note that for $D = \emptyset$ we obtain the definitions of informative right-congruence.

If $\mathcal{A}$ is a DMA with $L(\mathcal{A}) \equiv_D L$, then clearly two words $w, w' \in \Sigma^*$ that lead to the same state are $\sim_{L,D}$-equivalent. This leads to the observation that a DMA with informative D-congruence is D-minimal (since DPA, DBA etc. can be viewed as DMA, this observation holds for all acceptance types considered in this paper). Thus, the classes $\mathbb{IB}(D), \mathbb{IC}(D), \mathbb{IP}(D), \mathbb{IM}(D)$ consist of those languages that have a D-minimal automaton with informative D-congruence (with the respective acceptance condition).

Based on the results from Section 3, we can show that it is decidable in polynomial time whether a given DPA has a D-minimal automaton with informative D-congruence.

Theorem 4.4. For given DPAs $\mathcal{A}$ and $\mathcal{D}$ where $D = L(\mathcal{D})$ has a trivial right-congruence, it is decidable in polynomial time whether $L(\mathcal{A})$ is in $\mathbb{IB}(D), \mathbb{IC}(D)$, or $\mathbb{IP}(D)$, and a corresponding DBA, DCA, or DPA $\mathcal{B}$ with informative D-congruence and $L(\mathcal{B}) \equiv_D L(\mathcal{A})$ can be constructed in polynomial time if one exists.

Proof:
Let $L = L(\mathcal{A})$. Given two states p, q of $\mathcal{A}$, it can be decided in polynomial time whether $p \not\sim_{L,D} q$: One can build the product $\hat{\mathcal{A}}$ of $\mathcal{A} \times \mathcal{A} \times \mathcal{D}$, and then check if there is a strongly connected set S in this product graph that is reachable from $(p, q, q_0^{\mathcal{D}})$ (where $q_0^{\mathcal{D}}$ is the initial state of $\mathcal{D}$) and on which

(a) the maximal priority of states in the first component has different parity than the maximal priority of states in the second component (the resulting word is accepted from p iff it is not accepted from q),

(b) the maximal priority of states in the third component is odd (the resulting word is not a don't care word).

An ω-word that reaches S from $(p, q, q_0^{\mathcal{D}})$ and has S as infinity set, is a word outside D that witnesses the difference between p and q.

The existence of of such a set can be tested by first removing all states in $\hat{\mathcal{A}}$ that are not reachable from $(p, q, q_0^{\mathcal{D}})$, leading to $\hat{\mathcal{A}}'$. Then one iterates through all triples (i, j, k) of priorities that satisfy properties (a) and (b). For each such triple one removes all states from $\hat{\mathcal{A}}'$ with priority triple (i', j', k') and $i' > i$ or $j' > j$ or $k' > k$, and then computes the MSCCs in the remaining graph. If one finds an MSCC with a state that has priority triple (i, j, k), then one has found a witness for $p \not\sim_{L,D} q$. If this procedure does not find a witness for $p \not\sim_{L,D} q$ for any of the considered triples (i, j, k), then the states are equivalent.

Once $\sim_{L,D}$ is computed, one can construct $\mathcal{T}_{L,D}$ and the corresponding homomorphism from the transition structure of $\mathcal{A}$ onto $\mathcal{T}_{L,D}$. By Theorem 3.1 one can decide in polynomial time whether there is a parity condition on $\mathcal{T}_{L,D}$ such that the resulting language is D-equivalent to $L(\mathcal{A})$.

Since furthermore the constructed parity condition is optimal, one can also deduce whether there is a Büchi or co-Büchi condition on $\mathcal{T}_{L,D}$ that leads to a language that is D-equivalent to $L(\mathcal{A})$. □

For $D = \emptyset$, this solves the minimization problem for IRC languages and DPAs, which has independently been obtained in [19, Theorem 15.2] with a different proof.

Corollary 4.5. Given a DPA $\mathcal{A}$, it is decidable in polynomial time whether $L(\mathcal{A})$ is in $\mathbb{IB}, \mathbb{IC}$, or $\mathbb{IP}$, and a corresponding DBA, DCA, or DPA with informative right-congruence for $L(\mathcal{A})$ can be constructed in polynomial time if one exists.

We have seen that one can efficiently check if a language of a DPA has a D-minimal automaton with informative D-congruence. We now analyze the question for which language classes D-minimal automata are guaranteed to have informative D-congruence. As a first observation, we can explain the result from [14] that for each WDBAs there is a unique D-minimal WDBA in terms of the D-congruence.

Theorem 4.6. Given a WDBA $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$ with $L := L(\mathcal{A})$ and a regular ω-language of don't care words $D \subseteq \Sigma^\omega$ with trivial right-congruence, $\mathcal{A}$ is D-minimal if and only if it has informative D-congruence.

Proof:
Clearly, for any $u, v \in \Sigma^*$, it holds that $\delta^*(q_0, u) = \delta^*(q_0, v)$ implies $u \sim_{L,D} v$. It follows then that $\mathcal{A}$ has at least $Ind(\sim_{L,D})$ states. Thus, we can conclude that any WDBA with informative D-congruence is D-minimal, as well.

We now show that for each equivalence class κ of $\sim_{L,D}$, all $\alpha \in \Sigma^\omega \setminus D$ with infinitely many prefixes in κ are in L, or all these α are outside L. This defines an acceptance status for each equivalence class such that for two equivalence classes that are strongly connected (in the transition system $\mathcal{T}_{L,D}$) the acceptance status is the same. Hence, $\mathcal{T}_{L,D}$ can be equipped with a weak acceptance condition such that the accepted language is D-equivalent to L.

As mentioned above, all finite words leading to the same state q are in the same equivalence class of $\sim_{L,D}$. If this equivalence class is κ, we call q a κ-state.

Let $\alpha, \beta \in \Sigma^\omega \setminus D$ be such that they both have infinitely many prefixes in κ. Assume that $\alpha \in L$. We show that also $\beta \in L$ (the other case is symmetric). Let $\alpha = u\alpha'$ and $\beta = v\beta'$, with $u, v \in \kappa$. Let q be a κ-state in $\mathcal{A}$ such that no other κ-state in a different MSCC is reachable from q, and let $w \in \Sigma^*$ such that $\delta^*(w) = q$. Then $w\alpha' \in L \setminus D$ because $\alpha' \notin D$ and $u \sim_{L,D} w$.

Since $u\alpha'$ and $v\beta'$ have infinitely many prefixes in κ, the same is true for $w\alpha'$ and $w\beta'$. Thus $Inf_{\mathcal{A}}(w\alpha'), Inf_{\mathcal{A}}(w\beta') \subseteq \text{MSCC}(q)$, by choice of q. Since $w\alpha' \in L$ as seen above, also $w\beta' \in L$ because $\mathcal{A}$ is a WDBA. We conclude that $\beta = v\beta' \in L$ because $v \sim_{L,D} w$. □

This result for WDBAs raises the question whether it can be extended to larger classes of automata. The next natural candidate is the class of deterministic ω-automata with informative right-congruence. Since these have unique transition graphs defined by the right-congruence of their language, one might hope that they behave similarly to WDBAs under don't care sets with trivial right-congruence. We show that this is not the case. Furthermore, D-minimal automata for deterministic ω-automata with informative right-congruence need not to be unique and do not necessarily have informative right-congruence.

Proposition 4.7. There is a don't care set D with trivial right-congruence, and a DBA $\mathcal{A}$ with informative right-congruence, such that there are several non-isomorphic D-minimal DBA for $\mathcal{A}$, and a D-minimal DBA for $\mathcal{A}$ does not necessarily have informative right-congruence.

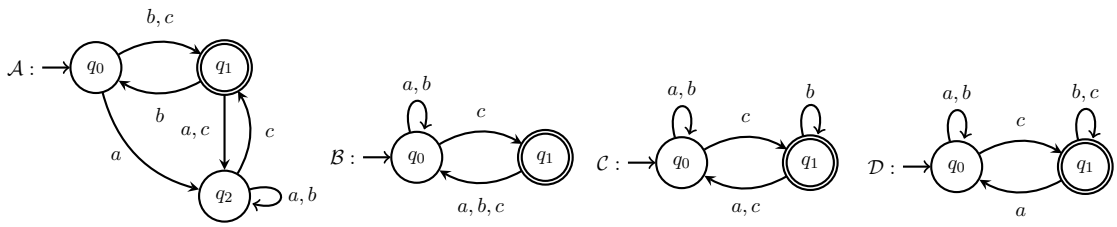


Figure 2. The DBA $\mathcal{A}$ has 3 different corresponding D-minimal DBA for the set of *don't care words* $D := \Sigma^* b^\omega$.

Proof:
Consider the DBA $\mathcal{A}$ in Figure 2 and let $L := L(\mathcal{A})$. Each state corresponds to a different equivalence-class in $\sim_L$, namely q_0 to $[\epsilon]$, q_1 to $[b]$ and q_2 to $[a]$. These equivalence classes are distinct since $b^\omega \in L$ and $ab^\omega \notin L$, showing $\epsilon \not\sim_L a$ and $b \not\sim_L a$, and $\epsilon \not\sim_L b$ is witnessed by $cb^\omega \in L$ and $bcb^\omega \notin L$. Thus, $L(\mathcal{A}) \in \mathbb{IB}$ and $\mathcal{A}$ is minimal for L. With $D = \Sigma^* b^\omega$, we have a set of *don't care words* with trivial right-congruence, and any D-minimal DBA for L has at least two states as $a^\omega \notin L(\mathcal{A}) \cup D$ and $c^\omega \in L(\mathcal{A}) \setminus D$. Figure 2 shows three different D-minimal DBA $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$, all with different transition graphs. Note that $L(\mathcal{C}), L(\mathcal{D}) \in \mathbb{IB}$ as $b^\omega \not\sim_{L'} cb^\omega$ for $L' \in \{L(\mathcal{C}), L(\mathcal{D})\}$, but $L(\mathcal{B})$ is the set of all words containing infinitely many c, which has trivial right-congruence, so $\mathcal{B}$ does not have informative right-congruence.

We used the set $D = \Sigma^* b^\omega \in \mathbb{DC}$ of don't care words, but the same results hold for the set of don't care words $D' = (\Sigma^* b)^\omega \in \mathbb{DB}$. □

We now consider the computational complexity of the D-minimization problem for automata with IRC, and show that it is NP-hard.

Definition 4.8. (**D-minimization Problem for IRC**)
Given a deterministic ω-automaton $\mathcal{A}$ with informative right-congruence, a DPA $\mathcal{D}$ accepting a set of don't care words $D = L(\mathcal{D})$ with trivial right-congruence, and $k \in \mathbb{N}$, does a corresponding D-minimal ω-automaton $\mathcal{B}$ of the same type as $\mathcal{A}$ with at most k states exist?

Theorem 4.9. The D-minimization Problem for IRC is NP-hard for DBA, DCA, and DPA.

We show NP-hardness via reduction from the NP-complete Vertex Coloring Problem [20]: Given an undirected graph $\mathcal{G} = (V, E)$ with $|E| > 0$ and $k \in \mathbb{N}$, does there exist a coloring $col : V \to \{1, \ldots, k\}$ of the vertices such that $col(v) \neq col(u)$ for all $(u, v) \in E$?

We first briefly describe the idea for the reduction, and then enter the formal details. For an undirected graph $\mathcal{G} = (V, E)$ with $|E| > 0$, we take the vertex set as alphabet, and consider the language of all $\alpha \in V^\omega$ in which only one vertex appears infinitely often. This language can be accepted by a DCA (a DPA with priorities 0 and 1) with one state of priority 0 for each vertex v, which is reached with and loops on v, and one extra state q_G of priority 1 that is reached whenever the next vertex in the sequence is different from the previous one. As an example, consider the DCA $\mathcal{A}_G$ in Figure 3 without the transition labels of the form x_v. These labels are added to ensure that the resulting language has informative right-congruence: the sequence x_v^ω is accepted only from the state for v.

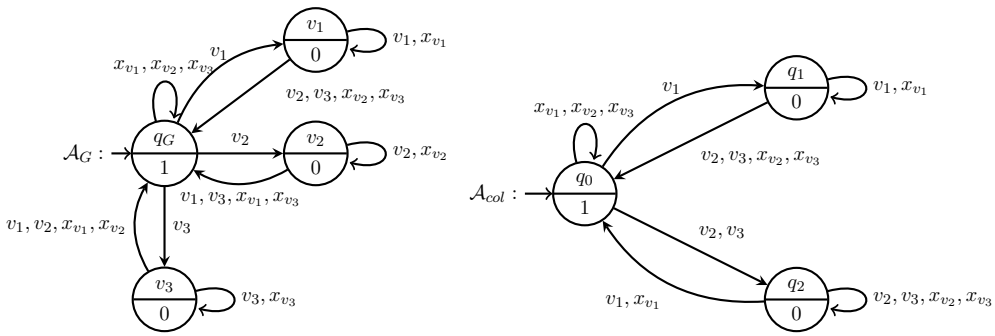


Figure 3. The D'-minimal DPA $\mathcal{A}_{col}$ and the DPA $\mathcal{A}_G$ for $\mathcal{G} = (\{v_1, v_2, v_3\}, \{(v_1, v_2), (v_1, v_3)\})$ recognize D'-equivalent ω-languages.

The don't care set then includes all words α in which labels of the form x_v occur infinitely often, or from some point on successive vertices vv' in α are not connected by an edge. With this don't care set, a D-minimal DPA $\mathcal{A}_{col}$ can use one state for each color class in a k-coloring instead of one state for each vertex v, as illustrated in Figure 3. This shows NP-hardness of D-minimization for DCA and DPA with IRC, and can easily be adapted to also work for DBA.

We now give the formal details of the reduction. For an undirected graph G with at least one edge, define the DPA $\mathcal{A}_G := (\Sigma, V \cup \{q_0\}, \delta_G, q_G, c_G)$ with $\Sigma := V \cup \{x_v \mid v \in V\}$ and for all $u, v \in V, u \neq v$:

- $\delta_G(q_G, u) = u$ and $\delta_G(q_G, x_u) = q_G$,

- $\delta_G(u, u) = u$, $\delta_G(u, x_u) = u$, $\delta_G(u, v) = q_G$ and $\delta_G(u, x_v) = q_G$,

- $c_G(q_G) = 1$ and $c_G(u) = 0$.

Furthermore, define the set of don't care words as $D := D' \cup D''$ with

- $D' := \{w\alpha \mid w \in \Sigma^*, \alpha \in V^\omega, \forall i \in \mathbb{N} : \exists j > i : \alpha(i) \neq \alpha(j)$ and $\forall i \in \mathbb{N} : \alpha(i) \neq \alpha(i+1) \to (\alpha(i), \alpha(i+1)) \notin E\}$
- $D'' := \{\alpha \in \Sigma^\omega \mid \exists v \in V : \alpha(i) = x_v$ for infinitely many $i \in \mathbb{N}\}$.

This set D is accepted by the DPA $(\Sigma, \{q_x\} \cup (V \times \{1,2,3\}), \delta_D, q_x, c_D)$ with

$$\delta_D((u,i),v) := \begin{cases} (v,1), & \text{if } u = v \\ (v,2), & \text{if } u \neq v \text{ and } (u,v) \notin E \\ (v,3), & \text{if } (u,v) \in E \end{cases}$$

for all $u, v \in V$ and $i \in \{1,2,3\}$, as well as $\delta_D(q_x, v) := (v, 2)$ and $\delta_D(q, x_v) = q_x$ for $q \in \{q_x\} \cup (V \times \{1,2,3\})$ and $v \in V$, and finally $c_D(v,i) = i$ and $c_D(q_x) = 4$.

Lemma 4.10. $\mathcal{A}_G$ has informative right-congruence.

Proof:
Let $L := L(\mathcal{A}_G)$. For every $u, v \in V$ with $u \neq v$, we have $ux_u^\omega \in L$ and $x_u^\omega, vx_u^\omega \notin L$. Consequently, for every $u, v \in V$ with $u \neq v$, it holds that $\epsilon \not\sim_L u$ and $u \not\sim_L v$. □

Lemma 4.11. G can be colored using at most k different colors if, and only if, any D-minimal DPA $\mathcal{B}$ with $L(\mathcal{A}_G) \equiv_D L(\mathcal{B})$ has at most $k+1$ states.

Proof:
First, we demonstrate that if G can be colored using at most k colors, this implies that any D-minimal DPA $\mathcal{B}$ with $L(\mathcal{A}_G) \equiv_D L(\mathcal{B})$ has at most $k+1$ states: We assume that a vertex-coloring $col : V \to \{1, \ldots, k\}$ on G exists. Then we construct a corresponding DPA $\mathcal{A}_{col} := (\Sigma, \{q_0, \ldots, q_k\}, \delta, q_0, c)$ with transitions $\delta(q_0, v) := q_{col(v)}$, $\delta(q_0, x_v) := q_0$,

$$\delta(q_i, v) := \begin{cases} q_i, & \text{if } i = col(v) \\ q_0, & \text{if } i \neq col(v), \end{cases} \text{ and } \delta(q_i, x_v) := \begin{cases} q_i, & \text{if } i = col(v) \\ q_0, & \text{if } i \neq col(v), \end{cases}$$

as well as $c(q_0) = 1$ and $c(q_i) = 0$ for all $i \in \{1, \ldots, k\}$ and $v \in V$. Since col is a valid vertex-coloring, it follows that $L(\mathcal{A}_{col})$ only differs from $L(\mathcal{A}_G)$ on words that are in D, and hence $L(\mathcal{A}_G) \equiv_D L(\mathcal{A}_{col})$.

Next we show that if a D-minimal DPA for $L(\mathcal{A}_G)$ has at most $k+1$ states, then G can be colored using at most k colors: Assume that there exists a DPA $\mathcal{C} := (\Sigma, \{0, \ldots, k\}, \delta, 0, c)$ with $L(\mathcal{A}_G) \equiv_D L(\mathcal{C})$. There is at least one state $q \in \{0, \ldots, k\}$ such that $c(q)$ is odd. W.l.o.g. let $c(0)$ be odd. Since $\mathcal{C}$ needs to accept each v^ω, we can define

$$col : V \to \{1, \ldots, k\} \text{ with } col(v) := \min\{q \in Inf_{\mathcal{C}}(v^\omega) \mid c(q) \text{ is even}\} \text{ for } v \in V.$$

For $(u, v) \in E$ any ω-word $\alpha \in (V^* uv)^\omega$ is rejected by $\mathcal{C}$. Thus, we have $Inf_{\mathcal{C}}(u^\omega) \cap Inf_{\mathcal{C}}(v^\omega) = \emptyset$. This mean that col is a valid vertex coloring on G which uses at most k different colors. □

As demonstrated in Lemma 4.11, we can reduce any instance of the Vertex Coloring Problem to an instance of the D-minimization Problem for IRC for DPA. Furthermore, the reduction is possible in polynomial time, as the alphabet has a size of $|\Sigma| = 2|V|$ and the number of states in $\mathcal{A}_G$ is $|V|+1$. It follows then that the D-minimization Problem with IRC for DPA is NP-hard.

The DPA used in the reduction are actually DCA, as we only utilize priorities 0 (final state) and 1 (non-final state). Consequently, the D-minimization Problem with IRC is NP-hard for DBA and DCA, as well. This finishes the proof of Theorem 4.9.

5. Active learning of WDBA

In [12] it is shown that the minimal WDBA for an unknown language U that is WDBA recognizable can be learned in polynomial time from membership and equivalence queries (the time is polynomial in the size of the minimal WDBA for U and the size of the counterexamples provided by the oracle). In this section, we consider the extension of this setting with an additional set D of don't care words with trivial right-congruence that is also unknown to the learner. We only consider WDBA in this context, because this is the only class of ω-automata for which a polynomial time active learning algorithm (with standard membership and equivalence queries) is known. The algorithm in [13] is not polynomial, and learns a different representation whose translation to deterministic ω-automata is exponential, in the worst case. Furthermore, in [18] it is shown that the IRC restriction does not help for active learning, in the sense that polynomial time active learning of regular ω-languages with IRC is not easier than polynomial time active learning of deterministic ω-automata in general.

As in [12], our algorithm can use membership and equivalence queries. Since there are don't care words, a membership query *MEMBER*(u, v) for $u \in \Sigma^*$ and $v \in \Sigma^+$ returns "yes" if $uv^\omega \in U \setminus D$, "no" if $uv^\omega \notin U \cup D$ and "don't care" if $uv^\omega \in D$. For a hypothesis WDBA $\mathcal{H}$, an equivalence query *EQUIV*$(\mathcal{H})$ either confirms that $L(\mathcal{H}) \equiv_D U$, or provides a counterexample, which is an ultimately periodic word $uv^\omega \in \Sigma^\omega \setminus D$ with $uv^\omega \in (L(\mathcal{H})\backslash U) \cup (U\backslash L(\mathcal{H}))$.

It turns out that it suffices to modify one part of the algorithm, which is called "conflict resolution" in [12], in order to deal with the don't care words. As in the original algorithm, we use an observation table in order to approximate $\sim_{U,D}$. An observation table consists of a prefix closed set $S \subseteq \Sigma^*$, a suffix closed set E of ultimately periodic words with $E \cap D = \emptyset$, and a function $f : (S \cup S\Sigma) \times E \to$ {"yes","no"} with $f(s, uv^\omega) :=$ *MEMBER*(su, v) for all $s \in S$ and $uv^\omega \in E$. Since D has trivial right-congruence and $D \cap E = \emptyset$, we know that *MEMBER*(su, v) is "yes" or "no" for all $s \in S$ and $uv^\omega \in E$, so we will never ask membership queries for don't care words.

In the table view, the rows are indexed with the elements from $S \cup S\Sigma$ and the columns are indexed with the elements from E. The entries in the row for $s \in S \cup S\Sigma$ are an approximation of the information required for identifying the equivalence class of s in $\sim_{L,D}$. This row is captured by the function $f_s : E \to$ {"yes","no"} with $f_s(uv^\omega) := f(s, uv^\omega)$ for each $s \in S \cup S\Sigma$. The observation tables are constructed in such a way that the rows of all elements in S are different, a property that is often referred to as reduced observation table. In the following we only work with reduced observation tables.

The elements from S correspond to the states of the WDBA, and the rows indexed by $S\Sigma$ are used for the transitions. We say that an observation table is closed if and only if for every $s \in S\Sigma$ there is

$t \in S$ with $f_s = f_t$. For a closed observation table we can then define the corresponding transition system $\mathcal{T}_{S,f} = (\Sigma, S, \delta_f, \epsilon)$ with $\delta_f(s, \sigma) := t$ for the unique $t \in S$ with $f_{s\sigma} = f_t$. The following lemma states that this transition system is isomorphic to the transition system induced by $\sim_{U,D}$ if S contains as many elements as there are equivalence classes.

Lemma 5.1. If $|S| = \mathit{Ind}(\sim_{U,D})$, then $\mathcal{T}_{S,f}$ is isomorphic to $\mathcal{T}_{U,D}$.

Proof:
Since $|S| = \mathit{Ind}(\sim_{U,D})$, there is one representative of each equivalence class in S. For $s \in S$ and $\sigma \in \Sigma$, the unique $t \in S$ with $f_{s\sigma} = f_t$ must be the representative of the equivalence class of $s\sigma$. □

The following lemma states that for ω-words that belong to the table, the transition system is always in a state that classifies the remaining suffix correctly.

Lemma 5.2. Let (S, E, f) be a closed observation table, $\alpha \in S \cdot E$, and $w \in \Sigma^*$ be a prefix of α with $\alpha = w\alpha'$. Further, let $r = \delta_f^*(\epsilon, w)$. Then $\alpha \in U$ iff $r\alpha' \in U$.

Proof:
We can write $\alpha = s\beta$ with $s \in S$ and $\beta \in E$. If w is a prefix of s, the claim follows since S is prefix-closed and thus $\delta_f^*(w) = w$. Otherwise, α' is a suffix of β and hence $\alpha' \in E$ since E is suffix-closed. Let $\alpha' = \sigma\alpha''$ and $r' = \delta_f(r, \sigma) = \delta_f^*(w\sigma)$. We show that $r\alpha' \in U$ iff $r'\alpha'' \in U$, which proves the claim by induction on the length of w. We have that $r\alpha' \in U$ iff $f(r, \alpha') =$ "yes" iff $f(r\sigma, \alpha'') =$ "yes" iff $f(r', \alpha'') =$ "yes" since r' is the unique element in S with $f_{r'} = f_{r\sigma}$. □

It remains to determine the accepting states of $\mathcal{T}_{S,f}$ in order to obtain a hypothesis WDBA. This is done by marking for each $\alpha \in S \cdot E$ the states in $\mathit{Inf}_{\mathcal{T}_{S,f}}(\alpha)$ with "yes" if $\alpha \in U$, and with "no" if $\alpha \notin U$. The marking fails if and only if there are two states s, t in the same MSCC such that s is marked "yes" and t is marked "no" (s and t can be the same if there is a state that is marked in both ways). Otherwise, the marking succeeds and we ask an equivalence query for $\mathcal{H} = (\mathcal{T}_{S,f}, F)$ with $F := S \setminus \{q \in S \mid$ a state in MSCC(q) is marked "no"$\}$ the set of states that are not in the MSCC of a state that is marked "no".

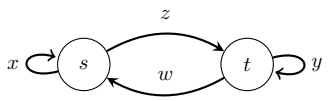


Figure 4. The situation described in Lemma 5.3 with $sx^\omega \in U \setminus D$ and $ty^\omega \notin U \cup D$.

It remains to deal with the case that the marking fails. The situation described in the following lemma is illustrated in Figure 4. The proof is based on Lemma 5.2.

Lemma 5.3. If the marking fails, there are $s, t \in S$, and $z, w \in \Sigma^*$ with $\delta_f^*(sz) = t$ and $\delta_f^*(tw) = s$, as well as $x^\omega, y^\omega \in E$ with $s \in \mathit{Inf}_{\mathcal{T}_{S,f}}(sx^\omega)$ and $t \in \mathit{Inf}_{\mathcal{T}_{S,f}}(ty^\omega)$ such that $f(s, x^\omega) =$ "yes" and $f(t, y^\omega) =$ "no".

Proof:
If the marking fails, there are $s, t \in S$ in the same MSCC such that s is marked "yes" and t is marked "no". Since s, t are in the same MSCC, the are words z, w connecting them. Furthermore, by definition of the marking, there are $\alpha, \beta \in S \cdot E$ with $\alpha \in U$ and $s \in Inf_{\mathcal{T}_{S,f}}(\alpha)$, and $\beta \notin U$ and $t \in Inf_{\mathcal{T}_{S,f}}(\beta)$. Since α, β are ultimately periodic, and s and t are visited infinitely often in the respective runs, we can decompose α and β as $\alpha = ux^\omega$ and $\beta = vy^\omega$ with $\delta_f^*(u) = s$, $\delta_f^*(v) = t$, and $x^\omega, y^\omega \in E$. By Lemma 5.2 we obtain that $sx^\omega \in U$ and $ty^\omega \notin U$, and thus $f(s, x^\omega) =$ "yes" and $f(t, y^\omega) =$ "no". □

From the situation described in Lemma 5.3 we want to extend the observation table such that a new state is discovered. This is done by finding a *distinguishing experiment*, which is of the form $rv\alpha \in \Sigma^\omega$ with $r \in S$ such that for $r' := \delta_f^*(rv)$ we have $r'\alpha \notin U \iff rv\alpha \in U$. Adding $v\alpha$ and its suffixes to E ensures by Lemma 5.2 that $\delta_{f'}^*(rv) \neq r'$ in the next hypothesis with table (S', E', f'). Hence $\delta_{f'}^*(rv) \in S' \setminus S$, which means that the number of discovered states increases. The algorithm in [12] can produce a distinguishing experiment with suffix $(zw)^\omega$, which might be a don't care word in our setting. We therefore have to identify a distinguishing experiment with a suffix that already appears in the table and thus cannot be a don't care word.

Lemma 5.4. For $n := Ind(\sim_{U,D})$ and the setting in Lemma 5.3, there is $i \leq n$ such that one of the following is a distinguishing experiment: $s(x^n zy^n w)^i x^\omega$, $s(x^n zy^n w)^i x^n zy^\omega$, $t(y^n wx^n z)^i y^\omega$, or $t(y^n wx^n z)^i y^n wx^\omega$.

Proof:
The D-minimal WDBA $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$ for U has at most n MSCCs, thus for all $w_1, w_2 \in \Sigma^*$, there is an integer $i \leq n$ such that $\delta^*(w_1 w_2^j) \in \text{MSSC}(\delta^*(w_1 w_2^i))$ for all $j \geq i$. Since every MSCC in $\mathcal{A}$ either consists entirely of final states or contains no final states, it follows that there must be an integer $i \leq n$ with

$$\begin{aligned} s(x^n zy^n w)^i x^\omega \in U \quad &\iff \quad s(x^n zy^n w)^i x^n zy^\omega \in U \text{ or} \\ t(y^n wx^n z)^i y^\omega \in U \quad &\iff \quad t(y^n wx^n z)^i y^n wx^\omega \in U. \end{aligned}$$

To see that, note that $\delta^*(s(x^n zy^n w)^i x^n)$ is in the MSCC of the infinity set of $s(x^n zy^n w)^i x^\omega$ by the above observation. So if, for example, $s(x^n zy^n w)^i x^\omega \in U$ and $s(x^n zy^n w)^i x^n zy^\omega \notin U$, then the infinity set of $s(x^n zy^n w)^i x^n zy^\omega$ must be in another MSCC than the one of $s(x^n zy^n w)^i x^\omega$. This change of MSCC can happen at most n times.

We obtain the distinguishing experiment as follows. If $s(x^n zy^n w)^i x^\omega \notin U$, then $rv\alpha$ with $r = s$, $v = (x^n zy^n w)^i$ and $\alpha = x^\omega$ satisfies the conditions with $r' = \delta_f^*(rv) = s$. If $s(x^n zy^n w)^i x^n zy^\omega \in U$, then choose $r = s$, $v = (x^n zy^n w)^i x^n z$ and $\alpha = y^\omega$ with $r' = \delta_f^*(rv) = t$. Similarly for the other cases. □

According to Lemma 5.4, the following procedure finds a distinguishing experiment in time $\mathcal{O}(n^2)$. Since we do not know the value of n, we have to check the candidates from Lemma 5.4 for increasing values, called k in the procedure. The procedure returns the word for which all suffixes need to be added to E:

1. Initialize $k = 1$, then repeat the following steps:
2. Let $z' := x^k z$ and $w' := y^k w$.
3. If $sz'y^\omega \in U$, terminate and return $x^k z y^\omega$
4. If $tw'x^\omega \notin U$, terminate and return $y^k w x^\omega$
5. For $i \in \{1, \ldots, k\}$ do
   (a) If $s(z'w')^i x^\omega \notin U$, terminate and return $(z'w')^i x^\omega$
   (b) If $t(w'z')^i w'x^\omega \notin U$, terminate and return $(w'z')^i w'x^\omega$
   (c) If $s(z'w')^i z'y^\omega \in U$, terminate and return $(z'w')^i z'y^\omega$
   (d) If $t(w'z')^i y^\omega \in U$, terminate and return $(w'z')^i y^\omega$
6. Set $k := k + 1$ and go back to step 2.

We now have all the ingredients for the modified learning algorithm: The algorithm initializes an observation table with $S = \{\epsilon\}$, $E = \emptyset$ and goes through the following steps:

1. Add $t \in S\Sigma$ with $\forall s \in S : f_t \neq f_s$ to S until the table is closed, and then construct $\mathcal{T}_{S,f}$.
2. Mark the states in $Inf_{\mathcal{T}_{S,f}}(s\alpha)$ according to $f(s, \alpha)$ for every $s \in S$ and $\alpha \in E$. If the marking fails, find a distinguishing experiment, add its suffixes to E and go back to step 1, else let $\mathcal{H} = (\mathcal{T}_{S,f}, F)$ with $F := S \setminus \{q \in S \mid$ a state in MSCC(q) is marked "no"$\}$.
3. If *EQUIV*$(\mathcal{H})$ returns a counterexample α, add α and its suffixes to E and go back to step 1, else return $\mathcal{H}$.

An example run of the algorithm can be found in the appendix.

Theorem 5.5. Let $n := Ind(\sim_{U,D})$. The algorithm learns a D-minimal WDBA $\mathcal{H}$ with $L(\mathcal{H}) \equiv_D U$ in time that is polynomial in n and the length of a longest counterexample.

Proof:
The algorithm starts with $E = \emptyset$ and thus the table is closed since all the functions f_w have empty domain. By definition, the first hypothesis then accepts all words and the equivalence query returns a word outside U and D (if there is any, otherwise a D-minimal WDBA has been found). By Lemma 5.1 the transition system constructed by the algorithm is isomorphic to $\mathcal{T}_{U,D}$ once $|S| = n$. Then the marking will not fail anymore, so the algorithm will terminate with a correct hypothesis once it has for each MSCC at least one word with infinity set in that MSCC.

It remains to show that the algorithm keeps discovering new states. If the table is not closed, a new element is added to S. So eventually, a transition system for a hypothesis will be constructed. If the marking fails, then adding the suffixes of a distinguishing experiment ensures that a new state is discovered (this follows from Lemma 5.2, as explained earlier).

Otherwise, the suffixes of a counterexample are added to E. If the table remains closed, then the next marking either fails or in the resulting hypothesis at least one MSCC is marked differently than in the previous hypotheses for this transition system. This can happen at most n times before the marking fails or the hypothesis is correct. This means that the main loop terminates after at most $\mathcal{O}(n^2)$ rounds. A distinguishing experiment can be found with at most $\mathcal{O}(n^2)$ membership queries (see Lemma 5.4 and the algorithm thereafter). The number of rows in the table is at most $n + n|\Sigma|$, and the number of columns is linear in the sum of the lengths of the counterexamples and the distinguishing experiments. By construction, the length of the distinguishing experiments is polynomial in n and the length of the longest element in E (see Lemmas 5.3 and 5.4). □

Note that the size of an automaton for D does not play any role for the complexity of the learning algorithm. The learner does not know D, it only knows that D has a trivial right-congruence. Based on this knowledge, the algorithm is designed in such a way that all membership queries are made for words outside D. The only way how D affects the complexity of the learner is by the number of classes in $\sim_{U,D}$, which is at most the number of classes in $\sim_U$ (if two ω-words are equivalent for $\sim_U$, they are clearly also equivalent for $\sim_{U,D}$). The counterexamples for equivalence queries are chosen by the oracle, so their lengths are not under control of the learner. However, since the hypothesis automata used in equivalence queries have at most $Ind(\sim_{U,D})$ many states, there always exists a counterexample of size polynomial in $Ind(\sim_{U,D})$.

6. Conclusion

We have shown that the problem of priority optimization for DPA under a given don't care set can be solved efficiently, and that active learning of WDBA under a don't care set with trivial right-congruence is possible in polynomial time. Minimization of automata with IRC under such a don't care set D is NP-hard, and D-minimal automata are not unique, which shows that DPA with IRC do not inherit all the good properties of WDBA. However, minimization without don't cares for DPAs that accept languages in $\mathbb{IP}$ is possible in polynomial time.

Since DPAs are a model that is used in synthesis problems (see [21] for a recent tool based on parity automata), it would be interesting to see if one can identify synthesis problems with don't cares in which the priority minimization can improve the results of such synthesis algorithms.

References

[1] Hopcroft JE, Ullman JD. Introduction to Automata Theory, Languages, and Computation. Addison Wesley, 1979.

[2] Angluin D. Learning regular sets from queries and counterexamples. *Information and Computation*, 1987. **75**(2):87 – 106. doi:https://doi.org/10.1016/0890-5401(87)90052-6. URL `http://www.sciencedirect.com/science/article/pii/0890540187900526`.

[3] Pfleeger CP. State Reduction in Incompletely Specified Finite-State Machines. *IEEE Transactions on Computers*, 1973. **C-22**(12):87–106.

[4] Schewe S. Beyond Hyper-Minimisation—Minimising DBAs and DPAs is NP-Complete. In: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2010). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2010 .

[5] Krishnan SC, Puri A, Brayton RK. Deterministic w Automata vis-a-vis Deterministic Buchi Automata. In: Algorithms and Computation, 5th International Symposium, ISAAC '94, Beijing, P. R. China, August 25-27, 1994, Proceedings, volume 834 of *Lecture Notes in Computer Science*. Springer, 1994 pp. 378–386. doi:10.1007/3-540-58325-4_202. URL `https://doi.org/10.1007/3-540-58325-4_202`.

[6] Thomas W. Automata on Infinite Objects. In: Leeuwen JV (ed.), Formal Models and Semantics, Handbook of Theoretical Computer Science, pp. 133–191. Elsevier, Amsterdam. ISBN 978-0-444-88074-1, 1990. doi:https://doi.org/10.1016/B978-0-444-88074-1.50009-3. URL `https://www.sciencedirect.com/science/article/pii/B9780444880741500093`.

[7] Grädel E, Thomas W, Wilke T (eds.). Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001], volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002. ISBN 3-540-00388-6. doi:10.1007/3-540-36387-4. URL `https://doi.org/10.1007/3-540-36387-4`.

[8] Löding C. Efficient minimization of deterministic weak ω-automata. *Information Processing Letters*, 2001. **79**(3):105 – 109. doi:https://doi.org/10.1016/S0020-0190(00)00183-6. URL `http://www.sciencedirect.com/science/article/pii/S0020019000001836`.

[9] L Staiger KW. Automatentheoretische und Automatenfreie Charakterisierung Toplogischer Klassen Regulärer Folgenmengen. *Elektron. Informationsverarbeitung und Kybernetik EIK 10*, 1974. p. 379 – 392.

[10] Maler O, Staiger L. On syntactic congruences for omega-languages. *Theoretical Computer Science*, 1997. **183**(1):93–112. doi:https://doi.org/10.1016/S0304-3975(96)00312-X. Formal Language Theory, URL `https://www.sciencedirect.com/science/article/pii/S030439759600312X`.

[11] Wagner K. On omega-regular sets. *Information and Control*, 1979. **43**(2):123–177. doi:https://doi.org/10.1016/S0019-9958(79)90653-3. URL `https://www.sciencedirect.com/science/article/pii/S0019995879906533`.

[12] Maler O, Pnueli A. On the Learnability of Infinitary Regular Sets. *Information and Computation*, 1995. **118**(2):316 – 326. doi:https://doi.org/10.1006/inco.1995.1070. URL `http://www.sciencedirect.com/science/article/pii/S089054018571070X`.

[13] Angluin D, Fisman D. Learning regular omega languages. *Theor. Comput. Sci.*, 2016. **650**:57–72. doi:10.1016/j.tcs.2016.07.031. URL `https://doi.org/10.1016/j.tcs.2016.07.031`.

[14] Eisinger J, Klaedtke F. Don't Care Words with an Application to the Automata-Based Approach for Real Addition. In: Computer Aided Verification, 18th International Conference, CAV 2006, volume 4144 of *Lecture Notes in Computer Science*. Springer. ISBN 3-540-37406-X, 2006 pp. 67–80. doi:10.1007/11817963. URL `https://doi.org/10.1007/11817963`.

[15] Angluin D, Fisman D. Regular omega-Languages with an Informative Right Congruence. *Electronic Proceedings in Theoretical Computer Science*, 2018. **277**:265–279. doi:10.4204/eptcs.277.19. URL `http://dx.doi.org/10.4204/EPTCS.277.19`.

[16] Carton O, Maceiras R. Computing the Rabin Index of a Parity Automaton. *RAIRO Theoretical Informatics and Applications*, 1999. **33**(6):495–506.

[17] Löding C, Pirogov A. New Optimizations and Heuristics for Determinization of Büchi Automata. In: Proceedings of the 17th International Symposium on Automated Technology for Verification and Analysis (ATVA'19), volume 11781 of *Lecture Notes in Computer Science*. Springer, 2019 .

[18] Bohn L, Löding C. Constructing Deterministic ω-Automata from Examples by an Extension of the RPNI Algorithm. In: 46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, volume 202 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021 pp. 20:1–20:18. doi:10.4230/LIPIcs.MFCS.2021.20. URL `https://doi.org/10.4230/LIPIcs.MFCS.2021.20`.

[19] Angluin D, Fisman D, Shoval Y. Polynomial Identification of omega-Automata. *CoRR*, 2022. **abs/2209.09336**. doi:10.48550/arXiv.2209.09336. `2209.09336`, URL `https://doi.org/10.48550/arXiv.2209.09336`.

[20] Karp RM. Reducibility among Combinatorial Problems, pp. 85–103. Springer US, Boston, MA. ISBN 978-1-4684-2001-2, 1972.

[21] Meyer PJ, Sickert S, Luttenberger M. Strix: Explicit Reactive Synthesis Strikes Back! In: Computer Aided Verification - 30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, Part I. 2018 pp. 578–586. doi: 10.1007/978-3-319-96145-3_31. URL `https://doi.org/10.1007/978-3-319-96145-3_31`.

A. Appendix

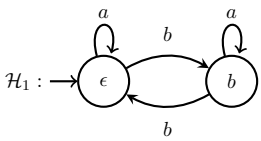
A.1. Example Run of Learning Algorithm from Section 5

We illustrate the learning algorithm with an example execution. Let $\Sigma := \{a, b, c\}$, $U := ab^\omega + ba^\omega + (ab)^\omega$ and $D := \Sigma^* b^\omega$. We show the closed observation tables and the hypotheses the algorithm produces. The entries in the observation table are denoted as 1 for "yes" and 0 for "no".

The first hypothesis for $S = \{\epsilon\}$ and $E = \emptyset$ is the WDBA accepting everything. Let a^ω be the counterexample, which is added to E:

	a^ω
ϵ	0
b	1
a	0
ba	1
bb	0

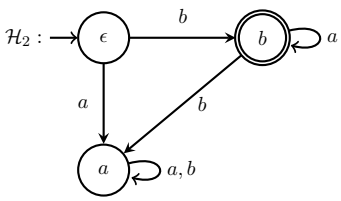
$\rightarrow$



The marking fails and we have $s = b$, $t = \epsilon$, $x = a = y$, $z = b = w$. For $k = 1$ and $w' = ab$, we find that $\epsilon aba^\omega \notin U$. As such, we add aba^ω and its suffixes to the observation table:

	a^ω	ba^ω	aba^ω
ϵ	0	1	0
a	0	0	0
b	1	0	0
aa	0	0	0
ab	0	0	0
ba	1	0	0
bb	0	0	0

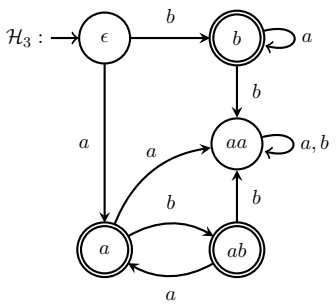
$\rightarrow$



The marking succeeds and we get $(ab)^\omega \in U$ as a counterexample. We extend the observation table accordingly and repeat the algorithm:

	a^ω	ba^ω	aba^ω	$(ab)^\omega$	$(ba)^\omega$
ϵ	0	1	0	1	0
a	0	0	0	0	1
b	1	0	0	0	0
aa	0	0	0	0	0
ab	0	0	0	1	0
ba	1	0	0	0	0
bb	0	0	0	0	0
aaa	0	0	0	0	0
aab	0	0	0	0	0
aba	0	0	0	0	1
abb	0	0	0	0	0

$\rightarrow$



The algorithm terminates and outputs $\mathcal{H}_3$ with $L_\omega(\mathcal{H}_3)$. $\mathcal{H}_3$ has informative D-congruence and is therefore D-minimal.