# Regular $\omega$-Languages with an Informative Right Congruence

Dana Angluin
Yale University
angluin@cs.yale.edu

Dana Fisman
Ben-Gurion University
dana@cs.bgu.ac.il

A regular language is almost fully characterized by its right congruence relation. Indeed, a regular language can always be recognized by a DFA isomorphic to the automaton corresponding to its right congruence, henceforth the *rightcon automaton*. The same does not hold for regular $\omega$-languages. The right congruence of a regular $\omega$-language is not informative enough; many regular $\omega$-languages have a trivial right congruence, and in general it is not always possible to define an $\omega$-automaton recognizing a given language that is isomorphic to the rightcon automaton.

The class of *weak regular $\omega$-languages* does have an informative right congruence. That is, any weak regular $\omega$-language can always be recognized by a deterministic Büchi automaton that is isomorphic to the rightcon automaton. Weak regular $\omega$-languages reside in the lower levels of the expressiveness hierarchy of regular $\omega$-languages. Are there more expressive sub-classes of regular $\omega$-languages that have an informative right congruence? Can we fully characterize the class of languages with a trivial right congruence? In this paper we try to place some additional pieces of this big puzzle.

## 1 Introduction

Regular $\omega$-languages play a key role in reasoning about reactive systems. Algorithms for verification and synthesis of reactive system typically build on the theory of $\omega$–automata. The theory of $\omega$-automata enjoys many properties that the theory of automata on finite words enjoys. These make it amenable for providing the basis for analysis algorithms (e.g. model checking algorithms rely on the fact that emptiness can be checked in nondeterministic logarithmic space). However, in general, the theory of $\omega$-automata is much more involved than that of automata on finite words, and many fundamental questions, such as minimization, are still open.

One of the fundamental theorems of regular languages on finite words is the Myhill-Nerode theorem stating a one-to-one correspondence between the state of the minimal deterministic finite automaton (DFA) for a language $L$ and the equivalence classes of the right congruence of $L$.[1] When moving to $\omega$-words, there is no similar theorem, and there are many regular $\omega$-languages where any minimal automaton requires more states than the number of equivalence classes in the right congruence of the language. For instance, consider the $\omega$-language $L = (a+b)^*a^\omega$. Its right congruence has only one equivalence class. That is, for any finite words $x$ and $y$ and any $\omega$-word $w$ we have that $xw \in L$ iff $yw \in L$ as membership in $L$ is determined only by the suffix. We say that the right congruence for $L$ is not *informative enough*.

The tight relationship between the equivalence classes of the right congruence and the states of a minimal DFA is at the heart of minimization and learning algorithms for regular languages of finite words, and seems to be a severe obstacle in obtaining efficient minimization and learning algorithms for regular $\omega$-languages. For this reason, we set ourselves to study classes of regular $\omega$-languages that do have a right congruence that is fully informative.

[1] Formal definitions are deferred to Section 2.

Submitted to:
SOS 2007

Several acceptance criteria are in use for $\omega$-automata, in particular, Büchi, co-Büchi, Muller and Parity. There are differences in the expressiveness of the corresponding deterministic automata. We use $\mathbb{DB}$, $\mathbb{DC}$, $\mathbb{DP}$ and $\mathbb{DM}$ to denote the classes of languages accepted by deterministic Büchi, co-Büchi, Muller and Parity automata, respectively. We also consider a version of Muller automata where acceptance is defined using transitions, refer to this acceptance criterion as Transition-Table, and use $\mathbb{DT}$ for the corresponding class of languages. The classes $\mathbb{DT}$, $\mathbb{DM}$ and $\mathbb{DP}$ accept all regular $\omega$-languages whereas $\mathbb{DB}$ and $\mathbb{DC}$ are strictly less expressive and are dual to each other. The intersection of $\mathbb{DB}$ and $\mathbb{DC}$ is the class of *weak* regular $\omega$-languages. This class does have the property that any language in $\mathbb{DB} \cap \mathbb{DC}$ has a fully informative right congruence. The regular $\omega$-languages can be arranged in an infinite hierarchy of expressive power as suggested by Wagner [33] and the class $\mathbb{DB} \cap \mathbb{DC}$ corresponds to one of the lowest levels of the hierarchy.

We define the classes $\mathbb{IB}$, $\mathbb{IC}$, $\mathbb{IP}$, $\mathbb{IM}$, $\mathbb{IT}$ to be the class of regular $\omega$-languages that can be accepted by a Büchi, co-Büchi, Parity and Muller and Transition-Table automata, respectively, whose number of states equals the number of equivalence classes in the right congruence of the language. We show that these form a strictly inclusive hierarchy of expressiveness as shown in Fig. 1 on the left, and moreover in every class of the infinite Wagner hierarchy of regular $\omega$-languages there are languages whose right congruence is fully informative.

Noting another difficulty in inferring a regular $\omega$-language from examples of $\omega$-words in the language, we consider a further restriction on languages, that if $ux^\omega$ is accepted by a minimal automaton for the language, then that automaton has a loop of size at most $|x|$ in which $ux^\omega$ loops. We term this property, being *respective* of the right congruence. This property is reminiscent of the property of being non-counting [13], and we show that a language that is non-counting is respective of its right congruence but the other direction does not necessarily hold. We define the classes $\mathbb{RB}$, $\mathbb{RC}$, $\mathbb{RP}$, $\mathbb{RM}$, $\mathbb{RT}$ of languages in $\mathbb{IB}$, $\mathbb{IC}$, $\mathbb{IP}$, $\mathbb{IM}$, $\mathbb{IT}$ that are respective of their right congruence. We show that these classes constitute a further restriction in terms of expressive power as shown in Fig. 1 on the right, and yet here as well, in every class of the infinite Wagner hierarchy of regular $\omega$-languages there are languages which are respective of their fully informative right congruence.

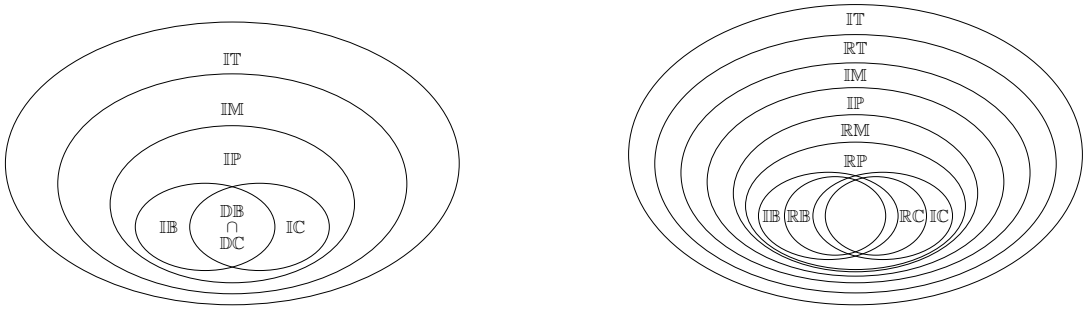


Figure 1: On the left, a summary of inclusion of the classes of $\omega$-automata that are isomorphic to their rightcon automaton. On the right a summary of inclusion of the classes of $\omega$-automata that are isomorphic to their rightcon automaton, as well as those that are in addition respective of their right congruence.

The rest of the paper is organized as follows. In Section 2 we provide the necessary definitions of the $\omega$-automata that we consider and of right congruence, state the well known results about their expressiveness, and briefly summarize the importance of the relation between the syntactic right congruence of a language and its minimal acceptor in learning algorithms. In Section 2 we state the relations between states of arbitrary $\omega$-automata for a regular $\omega$-language $L$ and its syntactic right congruence. In

Section 3 we provide a full characterization of $\omega$-languages $L$ for which $\sim_L$ is trivial. In Section 4 we explore expressiveness results related to the classes of languages with an informative right congruence. In Section 5 we explore expressiveness results related to the classes of languages that not only have an informative right congruence, but are also respective of their right congruence. In Section 6 we explore closure properties of these classes, and in Section 7 we conclude. Due to lack of space some proofs are deferred to the appendix.

## 2 Preliminaries

An *alphabet* $\Sigma$ is a finite set of symbols. The set of finite words over $\Sigma$ is denoted by $\Sigma^*$, and the set of infinite words, termed $\omega$-words, over $\Sigma$ is denoted by $\Sigma^\omega$. We use $\varepsilon$ for the empty word and $\Sigma^+$ for $\Sigma^* \setminus \{\varepsilon\}$. A *language* is a set of finite words, that is a subset of $\Sigma^*$, while an $\omega$-language is a set of $\omega$-words, that is a subset of $\Sigma^\omega$. For natural numbers $i$ and $j$ and a word $w$, we use $[i..j]$ for the set $\{i, i+1, \ldots, j\}$, $w[i]$ for the $i$-th letter of $w$, and $w[i..j]$ for the subword of $w$ starting at the $i$-th letter and ending at the $j$-th letter, inclusive.

**Automata and Acceptors** An *automaton* is a tuple $\mathcal{A} = \langle \Sigma, Q, \lambda, \delta \rangle$ consisting of an alphabet $\Sigma$, a finite set $Q$ of states, an initial state $\lambda \in Q$, and a transition function $\delta : Q \times \Sigma \to 2^Q$. A run of an automaton on a finite word $v = a_1 a_2 \ldots a_n$ is a sequence of states $q_0, q_1, \ldots, q_n$ such that $q_0 = \lambda$, and for each $i \geq 0$, $q_{i+1} \in \delta(q_i, a_{i+1})$. A run on an infinite word is defined similarly and results in an infinite sequence of states. The transition function is naturally extended to a function from $Q \times \Sigma^*$, by defining $\delta(q, \varepsilon) = q$ and $\delta(q, av) = \delta(\delta(q, a), v)$ for $q \in Q$, $a \in \Sigma$ and $v \in \Sigma^*$. We often use $\mathcal{A}(v)$ as a shorthand for $\delta(\lambda, v)$ and $|\mathcal{A}|$ for the number of states in $Q$. We use $\mathcal{A}_q$ to denote the automaton $\langle \Sigma, Q, q, \delta \rangle$ obtained from $\mathcal{A}$ by replacing the initial state with $q$. We say that $\mathcal{A}$ is *deterministic* if $|\delta(q, a)| \leq 1$ and *complete* if $|\delta(q, a)| \geq 1$, for every $q \in Q$ and $a \in \Sigma$.

By augmenting an automaton with an acceptance condition $\alpha$, obtaining a tuple $\langle \Sigma, Q, \lambda, \delta, \alpha \rangle$, we get an *acceptor*, a machine that accepts some words and rejects others. An acceptor accepts a word if at least one of the runs on that word is accepting. For finite words the acceptance condition is a set $F \subseteq Q$ and a run on a word $v$ is accepting if it ends in an accepting state, i.e., if $\delta(\lambda, v)$ contains an element of $F$. For infinite words, there are various acceptance conditions in the literature; we consider five: Büchi, co-Büchi, parity, Muller and Transition-Table [29]. The Büchi and co-Büchi acceptance conditions are also a set $F \subseteq Q$. A run of a Büchi automaton is accepting if it visits $F$ infinitely often. A run of a co-Büchi is accepting if it visits $F$ only finitely many times. A parity acceptance condition is a map $\kappa : Q \to [1..k]$ (for some $k \in \mathbb{N}$) assigning each state a color (or priority). A run is accepting if the minimal color visited infinitely often is odd. A Muller acceptance condition is a set of sets of states $\alpha = \{F_1, F_2, \ldots, F_k\}$ for some $k \in \mathbb{N}$ and $F_i \subseteq Q$ for $i \in [1..k]$. A run of a Muller automaton is accepting if the set $S$ of states visited infinitely often in the run is in $\alpha$. A Transition-Table acceptance condition is a set $\alpha = \{T_1, T_2, \ldots, T_k\}$ of sets of transitions, where a transition is a tuple in $Q \times \Sigma \times Q$. A run of a Transition-Table automaton is accepting if the set $T$ of transitions visited infinitely often in the run is in $\alpha$. We use $[\![\mathcal{A}]\!]$ to denote the set of words accepted by a given acceptor $\mathcal{A}$.

We use three letter acronyms to describe acceptors, where the first letter is in $\{\text{D}, \text{N}\}$ and denotes if the automaton is *deterministic* or *nondeterministic*. The second letter is one of $\{\text{B},\text{C},\text{P},\text{M},\text{T}\}$ for the first letter of the acceptance condition: Büchi, co-Büchi, parity, Muller or Transition-Table. The third letter is always A for acceptor. Figure 2 gives examples for a DBA, DCA, DPA, DMA, and DTA, and specifies the accepted languages. A language is said to be *regular* if it is accepted by a DFA. An $\omega$-language is said

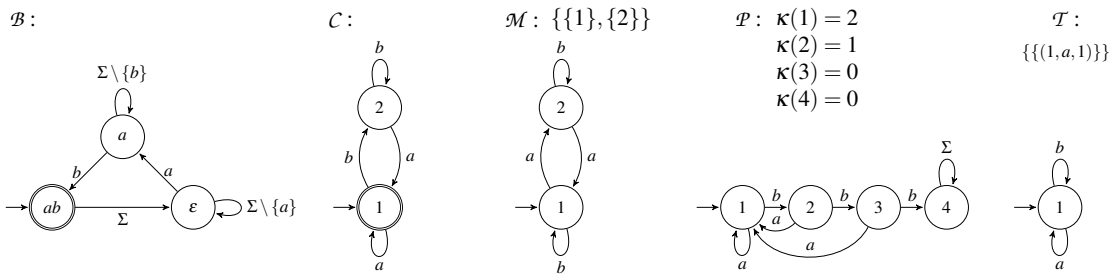


Figure 2: A DBA $\mathcal{B}$ accepting $(\Sigma^* a \Sigma^* b)^\omega$ where $\Sigma = \{a,b,c\}$, a DCA $\mathcal{C}$ accepting $(a+b)^* b^\omega$, a DMA $\mathcal{M}$ accepting $(a+b)^* b^\omega$, a DPA $\mathcal{P}$ accepting $(a+ba+bba)^*(a^*ba)^\omega$, and a DTA $\mathcal{T}$ accepting $(a+b)^* a^\omega$.

to be *regular* if it is accepted by a DMA. An $\omega$-language is said to be *weak* if it is accepted by a DBA as well as by a DCA.

**Complexity and expressiveness of sub-classes of regular $\omega$-languages** We use $\mathbb{DB}$, $\mathbb{DC}$, $\mathbb{DP}$, $\mathbb{DM}$ and $\mathbb{DT}$ to denote the classes of languages accepted by DBA, DCA, DPA, DMA and DTA, respectively, and $\mathbb{NB}$, $\mathbb{NC}$, $\mathbb{NP}$, $\mathbb{NM}$ and $\mathbb{NT}$ for the class of languages accepted by NBA, NCA, NPA, NMA and NTA, respectively. The classes $\mathbb{NB}$, $\mathbb{NP}$, $\mathbb{NM}$, $\mathbb{NT}$, $\mathbb{DP}$, $\mathbb{DM}$ and $\mathbb{DT}$ are equi-expressive and contain all $\omega$-regular languages. The classes $\mathbb{DB}$ and $\mathbb{DC}$ are strictly less expressive and are dual to each other in the sense that $L \in \mathbb{DB}$ iff $L^c \in \mathbb{DC}$ where $L^c$ is the complement language of $L$, i.e. $\Sigma^\omega \setminus L$. The classes $\mathbb{NC}$ and $\mathbb{DC}$ are equi-expressive.

A subset $S$ of the automaton states, where for every $s_1, s_2 \in S$ there exists a string $x \in \Sigma^+$ such that $\delta(s_1, x) = s_2$ is termed an SCC (abbreviating strongly connected component).[2] A Muller automaton $\mathcal{M}$ can be seen as classifying its SCCs into *accepting* and *rejecting*. An important measure of the complexity of a Muller automaton is the number of alternations between accepting and rejecting SCCs along an inclusion chain. For instance, the Muller automaton $\mathcal{M}$ in Figure 2 has an inclusion chain of SCCs with one alternation: $\{1\}$ (accepting) $\subseteq \{1,2\}$ (rejecting); and the Muller automaton $\mathcal{T}$ in Figure 3 whose only accepting SCC is $\{1,\lambda\}$ has an inclusion chain of SCCs with two alternations: $\{1\}$ (rejecting) $\subseteq \{1,\lambda\}$ (accepting) $\subseteq \{1,\lambda,0\}$ (rejecting). Wagner [33] has shown that this complexity measure is language-specific and is invariant over all Muller automata accepting the same language. Under this view, $\mathbb{DB}$ is the class of languages where no superset of an accepting SCC can be rejecting, $\mathbb{DC}$ is the class of languages where no subset of an accepting SCC can be rejecting, and $\mathbb{DB} \cap \mathbb{DC}$ is the class of languages where no alternation between accepting and rejecting SCCs is allowed along any inclusion chain. Thus, the language $[\![\mathcal{M}]\!]$ is not in $\mathbb{DB}$ and the language $[\![\mathcal{T}]\!]$ is not in $\mathbb{DB} \cup \mathbb{DC}$.

**Right congruence and the rightcon automaton** An equivalence relation $\sim$ on $\Sigma^*$ is a *right congruence* if $x \sim y$ implies $xv \sim yv$ for every $x, y, v \in \Sigma^*$. The *index* of $\sim$, denoted $|\sim|$ is the number of equivalence classes of $\sim$. Given a language $L$ its *canonical right congruence* $\sim_L$ is defined as follows: $x \sim_L y$ iff $\forall v \in \Sigma^*$ we have $xv \in L \iff yv \in L$. For a word $v \in \Sigma^*$ the notation $[v]$ is used for the class of $\sim$ in which $v$ resides.

A right congruence $\sim$ can be naturally associated with an automaton $\mathcal{M}_\sim = \langle \Sigma, Q, \lambda, \delta \rangle$ as follows: the set of states $Q$ consists of the equivalence classes of $\sim$. The initial state $\lambda$ is the equivalence class

[2] Note that there is no requirement for $S$ to be maximal in this sense.

$[\varepsilon]$. The transition function $\delta$ is defined by $\delta([u],a) = [ua]$. In the sequel, we use $\mathcal{R}_L$ for the automaton $M_{\sim_L}$ associated with the right congruence of a given language $L$, and call it the *rightcon automaton of* $L$.

Similarly, given an automaton $\mathcal{M} = \langle \Sigma, Q, \lambda, \delta \rangle$ we can naturally associate with it a right congruence as follows: $x \sim_{\mathcal{M}} y$ iff $\mathcal{M}(x) = \mathcal{M}(y)$. The Myhill-Nerode Theorem states that a language $L$ is regular iff $\sim_L$ is of finite index. Moreover, if $L$ is accepted by a DFA $\mathcal{A}$ then $\sim_{\mathcal{A}}$ refines $\sim_L$. Finally, the index of $\sim_L$ gives the size of the minimal DFA for $L$.

For $\omega$-languages, the right congruence $\sim_L$ is defined similarly, by quantifying over $\omega$-words. That is, $x \sim_L y$ iff $\forall w \in \Sigma^\omega$ we have $xw \in L \iff yw \in L$. Given a deterministic automaton $\mathcal{M}$ we can define $\sim_{\mathcal{M}}$ exactly as for finite words. However, for $\omega$-regular languages, right congruence alone does not suffice to obtain a "Myhill-Nerode" characterization. As an example consider the language $L = (a+b)^*(aba)^\omega$. We have that $\sim_L$ consists of just one equivalence class, since for any $x \in \Sigma^*$ and $w \in \Sigma^\omega$ we have that $xw \in L$ iff $w$ has $(aba)^\omega$ as a suffix. But an acceptor recognizing $L$ obviously needs more than a single state. Note that the other side of the story entails that there are $\omega$-automata that are minimal, although two different states recognize the same language. For instance, the DBA $\mathcal{B}$ in Figure 2 is minimal for $[\![\mathcal{B}]\!]$ but $[\![\mathcal{B}_\varepsilon]\!] = [\![\mathcal{B}_a]\!] = [\![\mathcal{B}_{ab}]\!]$, and the DMA $\mathcal{M}$ of Figure 2 is minimal for $[\![\mathcal{M}]\!]$ but $[\![\mathcal{M}_1]\!] = [\![\mathcal{M}_2]\!]$. In general the problem of minimizing DBAs and DPAs is known to be NP-complete [30].

**Grammatical Inference** *Grammatical inference* or *automata learning* refers to the problem of designing algorithms for inferring an unknown language from good and bad examples, i.e. from words labeled by their membership in the language. The learning algorithm is required to return some concise representation of the language, typically an automaton. The task of a learning algorithm can thus be thought of as trying to distinguish the different necessary states of an automaton recognizing the language and establishing the transitions between them. For a regular language, by the Myhill-Nerode theorem, $\sim_L$ can be used to distinguish states. Indeed, if the algorithm learns that $u_1 v \in L$ and $u_2 v \notin L$ for some $u_1, u_2, v \in \Sigma^*$ then $u_1 \not\sim_L u_2$ and the words $u_1$ and $u_2$ must reach two different states of the minimal DFA for $L$. Once all equivalence classes of $\sim_L$ are discovered, the automaton $\mathcal{M}_{\sim_L}$ can be extracted, and by setting the state corresponding to the empty word to be the initial state, and states corresponding to positive examples as accepting, the minimal DFA is obtained. Many learning algorithms, e.g. Bierman and Feldman's algorithm [9] for learning a regular language from a finite sample, and Angluin's $L^*$ [2] algorithm for learning a regular language using membership and equivalence queries build on this idea.

The idea of trying to distinguish states using right congruence relations is in the essence of many learning algorithms for formalisms richer than regular language (c.f. [1, 10, 11, 14, 21, 25]). For instance, learning of deterministic weighted automata [7, 8] is founded on Fliess's theorem [17] which is a generalization of the Myhill-Nerode theorem to the weighted automata setting.

For $\omega$-regular languages, learning algorithms encounter the problem that the right congruence is not informative enough. Maler and Pnueli [22] give a polynomial time algorithm for learning the class $\mathbb{DB} \cap \mathbb{DC}$ using membership and equivalence queries. Their algorithm also works by trying to distinguish all equivalence classes of $\sim_L$ for the unknown language $L$, and relies on the fact that $\sim_L$ is informative enough for the class $\mathbb{DB} \cap \mathbb{DC}$. The problem of learning the full class of regular $\omega$-languages via membership and equivalence queries was considered open for many years [20]. It was then suggested by Farzan et al. [15] to use the reduction to finite words [12], building on the fact that two $\omega$-languages are equivalent iff they agree on the set of ultimately periodic words. We followed a different route [5, 6], and devised an algorithm for learning the full class of $\omega$-regular languages using families of DFAs as acceptors [3, 4]. Families of DFAs build on the notion of families of right congruences [23], a set of right congruence relations for a given regular $\omega$-language $L$, which are enough to fully characterize $L$.

Both solutions, however, may encounter big automata in intermediate stages. The solution in [15] may involve DFAs of size $2^n + 2^{2n^2+n}$ where $n$ is the number of states in a non-deterministic Büchi automaton for the language [12], and the solution in [5] may involve families of DFAs of size $m2^m$ where $m$ is the number of states in a minimal deterministic parity automaton for $L$ [16]. We do not go into further details here since they are not used in the current work, which focuses on languages for which the right congruence is informative enough (in hopes of providing a basis for more efficient learning algorithms for these restricted classes). The reader interested in these details is referred to [16].

**Refinement of the right congruence** We show that as is the case in finitary regular languages, every deterministic $\omega$-automaton $\mathcal{D}$ refines the rightcon automaton of the respective language $[\![\mathcal{D}]\!]$, and the automaton for the powerset construction of a given non-deterministic $\omega$-automaton $\mathcal{N}$ refines the right congruence of the respective language $[\![\mathcal{N}]\!]$.

**Proposition 1** *Let $\mathcal{D}$ be a deterministic $\omega$-automaton. Then $\sim_{\mathcal{D}}$ refines $\sim_{[\![\mathcal{D}]\!]}$.*

Let $\mathcal{N}$ be a non-deterministic automaton. We use $\mathcal{P}_{\mathcal{N}}$ to denote the deterministic automaton obtained from $\mathcal{N}$ by applying the powerset construction to $\mathcal{N}$. That is, if $\mathcal{N} = (\Sigma, Q, q_0, \delta, \alpha)$ then $\mathcal{P}_{\mathcal{N}} = (\Sigma, 2^Q, \{q_0\}, \delta')$ where $\delta'(S,a) = \cup_{q\in S}\delta(q,a)$ for any $S \subseteq Q$ and $a \in \Sigma$.

**Proposition 2** *Let $\mathcal{N}$ be a non-deterministic $\omega$-automaton. Then $\sim_{\mathcal{P}_{\mathcal{N}}}$ refines $\sim_{[\![\mathcal{N}]\!]}$.*

## 3 $\omega$-Languages with a trivial right congruence

If $L$ is a regular $\omega$-language such that $|\sim_L| = 1$, we say that the rightcon automaton of $L$ is *trivial*. In this case, the rightcon automaton conveys almost no information about $L$. It was shown in [31] that there are $2^{2^{\aleph_0}}$ $\omega$-languages for which the rightcon is trivial. In Proposition 4 we characterize those regular $\omega$-languages that have a trivial rightcon automaton.

If $w_1$ and $w_2$ are $\omega$-words, then $w_1$ is a *finite variant* of $w_2$, denoted $w_1 =_\infty w_2$, if there exist finite words $x_1$ and $x_2$ and an $\omega$-word $w$ such that $w_1 = x_1 w$ and $w_2 = x_2 w$. The following shows that languages with a trivial rightcon automaton ignore differences between finite variants.

**Proposition 3** *Let $L$ be a regular $\omega$-language such that $|\sim_L| = 1$. Let $w_1, w_2 \in \Sigma^\omega$. If $w_1 =_\infty w_2$ then $w_1 \in L$ iff $w_2 \in L$.*

**Proof.** Because $w_1 =_\infty w_2$, there exist $x_1$, $x_2$ and $w$ such that $w_1 = x_1 w$ and $w_2 = x_2 w$. Because $|\sim_L| = 1$, $[x_1]_{\sim_L} = [x_2]_{\sim_L}$. Thus $x_1 w \in L$ iff $x_2 w \in L$, so $w_1 \in L$ iff $w_2 \in L$. □

Clearly if $L = \Sigma^* v^\omega$ for some $v \in \Sigma^*$ then its rightcon automaton is trivial. Does this hold in general when $L = \Sigma^* L'$ for some $\omega$-regular language $L'$? The following example shows that the answer is negative. Let $\Sigma = \{a,b\}$ and $L_1 = \Sigma^* L'$ for $L' = (ab^\omega + ba^\omega)$ then $\sim_{L_1}$ has four equivalence classes.

However, if $L = \Sigma^* R^\omega$ for some regular language $R$ we can show that then $|\sim_L| = 1$. Is this also a necessary condition for having a trivial right congruence? The answer again is negative. For example, the language $L_2 = (a+b)^*(a^\omega + b^\omega)$ has $|\sim_{L_2}| = 1$, but it is not of the form $(a+b)^* R^\omega$ for any regular set $R$.

The following proposition provides a full characterization of regular $\omega$-language with a trivial rightcon automaton.

**Proposition 4** *A regular $\omega$-language has a trivial rightcon automaton iff $L = \Sigma^*(R_1^\omega + R_2^\omega + \ldots + R_k^\omega)$ for some regular languages $R_1, \ldots, R_k$.*

**Proof.** Let $\mathcal{M} = (\Sigma, Q, \lambda, \delta, \alpha)$ be a DMA for $L$ with no unreachable states. Let $\alpha = \{S_1, S_2, \ldots, S_k\}$. We can assume wlog that all $S_i$'s are strongly connected (because an $S_i$ that is not an SCC can be omitted). For $i \in [1..k]$ let $s_i$ be some state in $S_i$ and let $R_i$ be the regular set of finite words that traverse $\mathcal{M}$ starting at $s_i$, ending in $s_i$, and visiting all states in $S_i$ and no other states.

Assume that $|\sim_L| = 1$. Let $L' = \Sigma^*(R_1^\omega + R_2^\omega + \ldots + R_k^\omega)$. We claim that $L = L'$. Let $w \in L$. Then $w = xw'$, where for some $i$, $x \in \Sigma^*$ reaches $s_i \in S_i$ and $w' \in R_i^\omega$, so $w \in L'$. Conversely, if $w \in L'$ then $w = xw'$, where $x \in \Sigma^*$ and for some $i$, $w' \in R_i^\omega$. Because $M$ contains no unreachable states, there exists $y \in \Sigma^*$ such that $y$ reaches state $s_i \in S_i$. Then $yw' \in L$, so by Proposition 3, $xw' \in L$.

For the converse, suppose that $R_1, \ldots, R_k$ are regular languages and $L = \Sigma^*(R_1^\omega + \ldots + R_k^\omega)$. If $|\sim_L| > 1$ then there exist $x, y \in \Sigma^*$ such that $[x]_L \neq [y]_L$, so wlog assume $w \in [x]_L$ and $w \notin [y]_L$. Because $xw \in L$, there exists an $i$ such that $xw \in \Sigma^* R_i^\omega$. Thus for some $x' \in \Sigma^*$ and elements $w_1, w_2, \ldots$ of $R_i$, $xw = x'w_1w_2\cdots$. Hence there exists $x'' \in \Sigma^*$ and $w'' \in R_i^\omega$ such that $xw = xx''w''$. But then $yw = yx''w''$, and $yx''w'' \in \Sigma^* R_i^\omega$, which implies that $yw \in L$, a contradiction. Thus we must have $|\sim_L| = 1$. □

## 4 $\omega$-Languages with an informative right congruence

We turn to examine the cases where the right congruence is as informative as it can be; that is the rightcon automaton is isomorphic to an $\omega$-automaton recognizing the respective language. We use $\mathbb{IB}$ (resp. $\mathbb{IC}$, $\mathbb{IP}$, $\mathbb{IM}$, $\mathbb{IT}$) to denote the class of languages for which the rightcon automaton $\mathcal{R}_L$ is isomorphic to a DBA (resp. DCA, DPA, DMA, DTA) accepting the language $L$.

**A small experiment** We were curious to see what are the odds that a randomly generated Muller automaton will be isomorphic to its rightcon automaton, i.e. fully informative. We ran a small experiment in which we generated a random Muller automaton over an alphabet of cardinality 3, with 2 accepting strongly connected sets, and tested whether it turned out to be isomorphic to its rightcon automaton. The procedure was to try to distinguish states of the random DMA using 100,000 random ultimately periodic $\omega$-words. If all states were successfully distinguished then the DMA is certainly isomorphic to its rightcon automaton, and was declared as such. If we failed to distinguish at least 2 states, we declared the DMA as non-isomorphic, though it might be that more tests would distinguish the undistinguished states and the DMA may in fact be isomorphic. So the probability of a randomly generated DMA being isomorphic to its rightcon automaton may be higher than what is suggested by our results.

We generated DMAs with 5, 6, 7, 8, 9, and 10 states; 100 of each size. The results are summarized in the following table. We find it interesting that in most of the cases a randomly generated DMA turns out to be isomorphic to its rightcon automaton, suggesting that this property is not rare. We defer a more careful study of the extent to which random automata have informative right congruences for further research.

| Number of states | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|
| Isomorphic | 85 | 93 | 88 | 96 | 96 | 94 |
| Not Isomorphic | 15 | 7 | 12 | 4 | 4 | 6 |

**Expressiveness results** As mentioned earlier, all weak regular $\omega$-languages, i.e. all languages that are in $\mathbb{DB} \cap \mathbb{DC}$ are isomorphic to their right congruence. We turn to the question of whether there exist languages outside this class that are isomorphic to their right congruence.

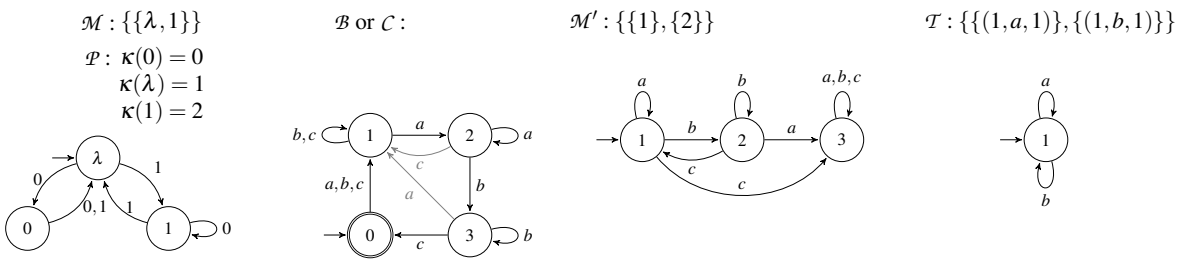


Figure 3: A DMA $\mathcal{M}$ and an equivalent DPA $\mathcal{P}$ for a language $L_{PM}$ such that $L_{PM} \in \mathbb{IM} \setminus \mathbb{DB} \cap \mathbb{DC}$ and $L_{PM} \in \mathbb{IP} \setminus \mathbb{DB} \cap \mathbb{DC}$. Second from the left, when regarded as a DBA $\mathcal{B}$ we have $[\![\mathcal{B}]\!] \in \mathbb{IB} \setminus \mathbb{DC}$. When regarded as a DCA $\mathcal{C}$ we have $[\![\mathcal{C}]\!] \in \mathbb{IC} \setminus \mathbb{DB}$. A DMA $\mathcal{M}'$ such that $[\![\mathcal{M}']\!] \in \mathbb{IM} \setminus \mathbb{IP}$, and a DTA $\mathcal{T}$ such that $[\![\mathcal{T}]\!] \in \mathbb{RT} \setminus \mathbb{RM}$.

Staiger [31] has shown that $\mathbb{DB} \cap \mathbb{DC} \subseteq \mathbb{IM}$. It is easy to see that this entails $\mathbb{DB} \cap \mathbb{DC} \subseteq \mathbb{IP}$. We show that both inclusions are strict.

**Proposition 5** $\mathbb{DB} \cap \mathbb{DC} \subsetneq \mathbb{IM}$ *and* $\mathbb{DB} \cap \mathbb{DC} \subsetneq \mathbb{IP}$

**Proof.** In [31, Thm. 24] Staiger showed that $\mathbb{DB} \cap \mathbb{DC} \subseteq \mathbb{IM}$. From this, since any DBA can be recognized by an isomorphic DPA (by setting the accepting states color 1 and the non-accepting states color 2) and a DCA can be recognized by an isomorphic DPA (by setting the $F$ states color 0 and the non-$F$ states color 1) it follows that $\mathbb{DB} \cap \mathbb{DC} \subseteq \mathbb{IP}$.

To show that the inclusion is strict we show that the language $L_{PM}$ recognized by the automaton in Fig. 3 on the left, either when regarded as a DMA $\mathcal{M}$ or as a DPA $\mathcal{P}$ is in $\mathbb{IM} \cap \mathbb{IP}$ but not in $\mathbb{DB} \cap \mathbb{DC}$. One can verify that $\sim_{[\![\mathcal{M}]\!]}$ is isomorphic to $\mathcal{M}$. (Note that $(0011)^\omega$ and $(0110)^\omega$ are sufficient to distinguish $\{\lambda,0,1\}$.) As mentioned in the preliminaries, this language is not in $\mathbb{DB} \cap \mathbb{DC}$ since it has alternation between accepting and rejecting SCCs along the inclusion chain $\{1\} \subseteq \{1,\lambda\} \subseteq \{1,\lambda,0\}$. □

It is easy to see that both $\mathbb{IM}$ and $\mathbb{IP}$ subsume both $\mathbb{IB}$ and $\mathbb{IC}$. The same example used in the proof of proposition 5 can be used to show that the inclusion is strict.

**Proposition 6** $\mathbb{IB} \cup \mathbb{IC} \subsetneq \mathbb{IM}$ *and* $\mathbb{IB} \cup \mathbb{IC} \subsetneq \mathbb{IP}$

It is thus interesting to see whether there are any languages in $\mathbb{IB}$ (or $\mathbb{IC}$) that are not already in $\mathbb{DB} \cap \mathbb{DC}$. The answer is affirmative.

**Proposition 7** $\mathbb{DB} \cap \mathbb{DC} \subsetneq \mathbb{IB}$ *and* $\mathbb{DB} \cap \mathbb{DC} \subsetneq \mathbb{IC}$

**Proof.** Assume $L \in \mathbb{DB} \cap \mathbb{DC}$. By proposition 5 we have $L \in \mathbb{IM}$. Suppose $\mathcal{M}$ is a minimal DMA for $L$. It follows, as explained in the preliminaries, that in $\mathcal{M}$ there is no alternation between accepting and rejecting SCCs along any inclusion chain. Therefore, defining a DBA $\mathcal{B}_\mathcal{M}$ from $\mathcal{M}$ by changing the acceptance condition to $\{q \mid q \in S$ for some accepting SCC $S\}$ gives $[\![\mathcal{B}_\mathcal{M}]\!] = [\![\mathcal{M}]\!]$ and thus $L \in \mathbb{IB}$. Similarly, defining a DCA $\mathcal{C}_\mathcal{M}$ from $\mathcal{M}$ by changing the acceptance condition to $\{q \mid q \in S$ for some rejecting SCC $S\}$ gives $[\![\mathcal{C}_\mathcal{M}]\!] = [\![\mathcal{M}]\!]$ and thus $L \in \mathbb{IC}$. This completes the inclusion part.

To see that the inclusion is strict for $\mathbb{IB}$ consider the DBA $\mathcal{B}$ from Fig. 3. By [19, Lemma 2] for every language $L$ recognized by a DBA $\mathcal{B}$, if $L$ is also in $\mathbb{DC}$, then a DCA embodied in the structure of $\mathcal{B}$ can be defined. Since none of the DCAs embodied in $\mathcal{B}$ accepts the same language as $[\![\mathcal{B}]\!]$, it follows that $[\![\mathcal{B}]\!] \in \mathbb{DB} \setminus \mathbb{DC}$. Thus $\mathcal{B} \notin \mathbb{DB} \cap \mathbb{DC}$.

Next we show that $\sim_{[\![\mathcal{B}]\!]}$ has at least 4 equivalence classes. The $\omega$-word $(ababc)^\omega$ is accepted only from states 0 and 3 and the $\omega$-word $(babca)^\omega$ is accepted only from states 0 and 1; these two experiments distinguish all 4 states. Since by Proposition 1 $\mathcal{B}$ refines $\sim_{[\![\mathcal{B}]\!]}$, it follows that the automaton for $\sim_{[\![\mathcal{B}]\!]}$ is isomorphic to $\mathcal{B}$, thus $[\![\mathcal{B}]\!] \in \mathbb{IB}$.

The proof for strictness for $\mathbb{IC}$ is dual, using the DCA $\mathcal{C}$ in Fig. 3 and [19, Lemma 2] which states also that for every language $L$ recognized by a DCA $\mathcal{C}$ if $L$ is also in $\mathbb{DB}$ then a DBA embodied in the structure of $\mathcal{C}$ can be defined. □

Since $\mathbb{IB} \subseteq \mathbb{DB}$ and $\mathbb{IC} \subseteq \mathbb{DC}$, from $\mathbb{DB} \cap \mathbb{DC} \subseteq \mathbb{IB} \cap \mathbb{IC}$ we get that $\mathbb{DB} \cap \mathbb{DC} = \mathbb{IB} \cap \mathbb{IC}$.

**Corollary 8** $\mathbb{DB} \cap \mathbb{DC} = \mathbb{IB} \cap \mathbb{IC}$

It is shown in [32] that any DMA can be defined on a DTA with the same structure, and that the converse is not true. For instance, the language $(a+b)^*a^\omega$ can be defined by the one-state DTA $\mathcal{T}$ in Fig. 2, but no DMA with one state accepts it. This shows $\mathbb{IM}$ is strictly contained in $\mathbb{IT}$.

**Proposition 9 ([32])** $\mathbb{IM} \subsetneq \mathbb{IT}$

The last missing part of the puzzle of the inclusions of the subsets $\mathbb{IB}$, $\mathbb{IC}$, $\mathbb{IP}$, $\mathbb{IM}$, $\mathbb{IT}$ is provided in the following proposition showing that $\mathbb{IP}$ is strictly contained in $\mathbb{IM}$.

**Proposition 10** $\mathbb{IP} \subsetneq \mathbb{IM}$

**Proof.** Inclusion follows since any DPA can be converted into a DMA on the same structure (by setting an SCC to accepting iff the minimal color in it is odd). We claim that the DMA $\mathcal{M}'$ in Figure 3 is in $\mathbb{IM} \setminus \mathbb{IP}$. To see that it is in $\mathbb{IM}$, note that $ca^\omega$ distinguishes state 2 from the other states, and $a^\omega$ distinguishes states 1 and 3. Thus the rightcon automaton has 3 states, and since by proposition 1, $\mathcal{M}'$ refines $\mathcal{R}_{[\![\mathcal{M}']\!]}$ we get that they are isomorphic. To see that it is not in $\mathbb{IP}$, note that to define a DPA on the same structure we need to give state 1 an odd color, so that when the set of states visited inf. often is $\{1\}$ it will accept. For the same reason we need to give state 2 an odd color. But then, when the set of states visited infinitely often is $\{1,2\}$, the automaton will accept as well, while it needs to reject. □

These relations are summarized in Figure 1 on the left. The last question is then how complex can a language in $\mathbb{IP}$, $\mathbb{IM}$, or $\mathbb{IT}$ be? We show that such languages can be arbitrarily complex. That is, for every class $\mathbb{DM}^p_{n,m}$ of the Wagner Hierarchy, there is a language $L \in \mathbb{IM} \cap \mathbb{IP} \cap \mathbb{IT}$ that is in $\mathbb{DM}^p_{n,m}$ and not in any proper subclass of the Wagner Hierarchy. Thus, $\mathbb{IT}$, $\mathbb{IM}$ and $\mathbb{IP}$ include classes as complex as can be[3], as measured by the Wagner Hierarchy.[4]

**Proposition 11** *Let $n,m$ be two natural numbers and $p \in \{+,-,\pm\}$. Let $\mathbb{DM}^p_{n,m}$ denote the Wagner Hierarchy class with a maximum of n alternations in each inclusion chain starting with polariy p, and a sequence of at most m chains of alternating polarity. Then exists a language $L \in \Sigma^\omega$ for $\Sigma = \{a,b\}$ such that $L \in \mathbb{IT} \cap \mathbb{IM} \cap \mathbb{IP} \cap \mathbb{DM}^p_{n,m}$ and $L \notin \mathbb{DM}^p_{n-1,m}$ and $L \notin \mathbb{DM}^p_{n,m-1}$.*

[3] A similar result is mentioned without a proof in a footnote in [23], with credit to "N. Gutleben (personal communication)".

[4] Due to lack of space we do not include the full definition of the Wagner Hierarchy. For details, we refer the reader to [33],[26, Chapter V].

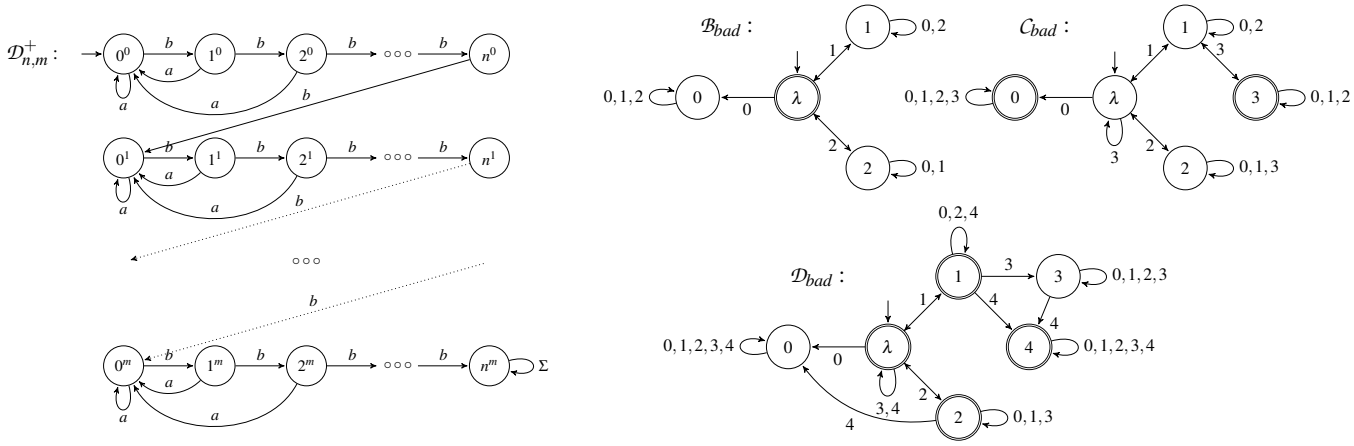


Figure 4: On the left, a representative example for the Wagner class $\mathbb{DM}^+_{n,m}$. The acceptance condition is $\mathcal{F} = \{\{0^i, 1^i, \ldots, j^i\} \mid j \text{ is odd iff } i \text{ is odd}\}$. On the right a DBA $\mathcal{B}_{bad}$ and DCA $\mathcal{C}_{bad}$ in $\mathbb{IB} \setminus \mathbb{RB}$ and $\mathbb{IC} \setminus \mathbb{RC}$, respectively, and a DBA $\mathcal{D}_{bad}$ in $\mathbb{IB} \cap \mathbb{IC}$ that is not respective of its right congruence.

**Proof.** Consider the DMA $\mathcal{D}^+_{n,m}$ in Figure 4, with acceptance condition $\mathcal{F} = \{\{0^i, 1^i, \ldots, j^i\} \mid j \text{ is odd iff } i \text{ is odd }\}$. For instance, $\{0^0\} \in \mathcal{F}$, $\{0^0, 1^0, 2^0\} \in \mathcal{F}$, and $\{0^3, 1^3\} \in \mathcal{F}$ but $\{1^0\} \notin \mathcal{F}$ and $\{0^0, 1^0\} \notin \mathcal{F}$. It strictly belongs to the Wagner hierarchy class $\mathbb{DM}^+_{n,m}$. To see that it is in $\mathbb{IM}$, we show for each state, a word that distinguishes it from other states. We fix an order between the states: a state $k^\ell$ is smaller than $k'^{\ell'}$ if either $\ell < \ell'$ or $\ell = \ell'$ and $k < k'$. Thus the last state $n^m$ is the biggest in this order. The word $a^\omega$ distinguishes the last state $n^m$ from all states on odd rows, and the word $(ab)^\omega$ distinguishes $m^n$ from all states on even rows. For $k \in [0..n]$, the word $b^{n-k}a^\omega$ distinguishes state $k^m$ from all smaller states on odd rows, and the word $b^{n-k}(ab)^\omega$ distinguishes it from all smaller states on even rows. Finally, for $k \in [0..n]$ and $\ell \in [0..m]$ the word $(b^{n+1})^{m-\ell}b^{n-k}a^\omega$ distinguishes state $k^\ell$ from all smaller states on odd rows, and the word $(b^{n+1})^{m-\ell}b^{n-k}(ab)^\omega$ distinguishes it from all smaller states on even rows. This shows that the rightcon automaton has $(n+1)(m+1)$ states, and thus $[\![\mathcal{D}^+_{n,m}]\!] \in \mathbb{IM}$. It is also in $\mathbb{IP}$, since we can define a DPA on the same structure, by assigning state $k^\ell$ the color $k$ if $\ell$ is odd, and $k+1$ if $\ell$ is even. It is in $\mathbb{IT}$ since $\mathbb{IM} \subset \mathbb{IT}$.

The proof for $\mathbb{DM}^-_{n,m}$ is symmetric, and the proof for $\mathbb{DM}^\pm_{n,m}$ can be easily deduced from this. □

## 5 Respective of the right congruence

As mentioned above, one of the motivations for studying classes of languages that are isomorphic to the right congruence is in the context of learning an unknown language. In this context, positive and negative examples ($\omega$-words labeled by their membership in the language) should help a learning algorithm to infer an automaton for the language. Consider the positive example $(ab)^\omega$ for an unknown language $L$. Intuitively, we expect that a minimal automaton for $L$ would have a loop of size 2 in which the word $ab$ cycles. This is not necessarily the case, as shown by the language $L = (aba + bab)^\omega$, whose minimal DBA $\mathcal{B}_{\bowtie}$ is given in Fig 6. In the case of regular languages of finite words, if we regard the automaton $\mathcal{B}_{\bowtie}$ as a DFA, we note that $ab$, $abab$ are negative examples, while $ababab$ is a positive example. From this a learning algorithm can clearly infer the smallest loop on which $ab$ cycles is of length 6, and not 2. But in the case of $\omega$-languages there are no negative $\omega$-words that can provide such information. We

thus define a class of languages in which if $uv^\omega$ is a positive example for $L$, then a minimal automaton for $L$ has a cycle of length at most $|v|$ in which $uv^\omega$ loops.

**Definition 12 (respective of $\sim_L$)** *A language $L$ is said to be* respective *of its right congruence if* $\exists n_0 \in \mathbb{N}.\ \forall n > n_0.\ \forall x,u \in \Sigma^*.\ xu^\omega \in L$ *implies* $xu^n \sim_L xu^{n+1}$.

Intuitively, a language that is respective of its right congruence, can "delay" entering a loop as much as needed, but once it loops on a periodic part, it loops on the smallest period possible.

Being respective of the right congruence does not entail having an $\omega$-automaton that is isomorphic to the right congruence. Any language $L$ with $|\sim_L| = 1$ is (trivially) respective of its right congruence. By Proposition 4, $L = (a+b)^*(aba)^\omega$ has $|\sim_L| = 1$, but $L$ is not in $\mathbb{IT}$, $\mathbb{IM}$, $\mathbb{IP}$, $\mathbb{IB}$ or $\mathbb{IC}$, because every $\omega$-automaton accepting $L$ requires more than one state. We thus concentrate on languages which are both isomorphic to the rightcon automaton and respective of their right congruence. We use $\mathbb{RB}$, $\mathbb{RC}$, $\mathbb{RP}$, $\mathbb{RM}$ and $\mathbb{RT}$ for the classes of languages that are respective of their right congruence and reside in $\mathbb{IB}$, $\mathbb{IC}$, $\mathbb{IP}$, $\mathbb{IM}$ and $\mathbb{IT}$, respectively. By definition, thus, $\mathbb{IX} \supseteq \mathbb{RX}$ for $\mathbb{X} \in \{\mathbb{B},\mathbb{C},\mathbb{P},\mathbb{M},\mathbb{T}\}$. We show that these inclusions are strict.

**Proposition 13** $\mathbb{IB} \supsetneq \mathbb{RB}$, $\mathbb{IC} \supsetneq \mathbb{RC}$, $\mathbb{IP} \supsetneq \mathbb{RP}$, $\mathbb{IM} \supsetneq \mathbb{RM}$ *and* $\mathbb{IT} \supsetneq \mathbb{RT}$.

**Proof.** Consider the DBA $\mathcal{B}_{bad}$ in Fig. 4 on the right. The language $B_{bad}$ accepted by $\mathcal{B}_{bad}$ is in $\mathbb{IB} \subset \mathbb{IP} \subset \mathbb{IM} \subset \mathbb{IT}$ but it is not respective of its right congruence. To see that it is in $\mathbb{IB}$ take $\varepsilon$, 0, 1 and 2 as the representative words for states $\lambda$, 0, 1 and 2 respectively. Note that $(011)^\omega$ distinguishes 1 from the rest of the representative words, and $(022)^\omega$ distinguishes 2 from the rest of the representative words. Finally, $(11)^\omega$ distinguishes $\varepsilon$ from 0. The pair $(\varepsilon, 1012)$ shows that $B_{bad}$ is not respective of its right congruence since $(1012)^\omega \in B_{bad}$ yet for all $n \in \mathbb{N}$ we have that $(1012)^{n+1} \not\sim_{B_{bad}} (1012)^n$.

Consider the DCA $\mathcal{C}_{bad}$ in Fig. 4. The language $C_{bad}$ accepted by $\mathcal{C}_{bad}$ is in $\mathbb{IC}$ but it is not respective of its right congruence. To see that it is in $\mathbb{IC}$ take $\varepsilon$, 0, 1, 2 and 13 as the representative words for states $\lambda$, 0, 1, 2 and 3 respectively. Note that $0^\omega$ distinguishes 1 and 2 from the rest of the representative words; $3^\omega$ distinguishes 1 from 2 and it distinguishes $\varepsilon$ from 0 and 13; and $30^\omega$ distinguishes 13 from 0. The pair $(\varepsilon, 1012)$ shows that $C_{bad}$ is not respective of its right congruence since $(1012)^\omega \in C_{bad}$ yet for all $n \in \mathbb{N}$ we have that $(1012)^{n+1} \not\sim_{C_{bad}} (1012)^n$. □

Recall that $\mathbb{DB} \cap \mathbb{DC} = \mathbb{IB} \cap \mathbb{IC}$. The DBA $\mathcal{D}_{bad}$ in Fig. 4 can be used to show a language in $\mathbb{IB} \cap \mathbb{IC}$ that is not respective of its right congruence.

**Proposition 14** *There exists languages in* $\mathbb{IB} \cap \mathbb{IC}$ *that are not respective of their right congruences.*

To complete the picture of inclusions between the $\mathbb{RX}$ classes, we establish that $\mathbb{RM} \supsetneq \mathbb{RP}$ and $\mathbb{RT} \supsetneq \mathbb{RM}$.

**Proposition 15** $\mathbb{RM} \supsetneq \mathbb{RP}$ *and* $\mathbb{RT} \supsetneq \mathbb{RM}$.

Figure 1 on the right summarizes the above results. While it is notable that the requirement of being respective of the right congruence constitutes a restriction, there exist languages which are respective of their right congruence in every class of the Wagner Hierarchy.

**Proposition 16** *Let $n,m$ be two natural numbers and $p \in \{+,-,\pm\}$. Let $\mathbb{DM}^p_{n,m}$ denote the Wagner Hierarchy class with a maximum of n alternations in each inclusion chain, and a sequence of at most m chains of alternating polarity. Then there exists a language $L \in \Sigma^\omega$ for $\Sigma = \{a,b\}$ such that $L \in \mathbb{DM}^p_{n,m} \cap \mathbb{RM} \cap \mathbb{RP} \cap \mathbb{RT}$.*

**Proof.** Consider again the DMA $\mathcal{D}^+_{n,m}$ in Fig. 4 and let $L^+_{n,m}$ be the language that it recognizes. We have established in the proof of Proposition 11 that $[\![\mathcal{D}^+_{n,m}]\!] \in \mathbb{IM} \cap \mathbb{IP} \cap \mathbb{IT}$. Take $n_0 = (m+1)(n+1)$. Consider the state that $\mathcal{D}^+_{n,m}$ reaches after reading $xu^{n_0}$. If this state is $n^m$ then clearly for any $n' > n_0$, $xu^{n'}$ also reaches states $n^m$. Otherwise there is an $a$ following the longest subsequence of $b^*$ in $u$. The state that $\mathcal{D}^+_{n,m}$ reaches after reading $xu^{n_0}$ depends on (a) the the longest subsequence of $b$'s in $x$ (b) the longest subsequence of $b$'s in $u$ and (c) the number of consecutive $b$'s in the rightmost subsequence of $b$'s in $u$. Since the parameter (a) depends only on $x$ and the parameters (b) and (c) remain the same in $u^i$ for any $i \in \mathbb{N}$ we have that $xu^{n_0+i} \sim_{L^+_{n,m}} xu^{n_0+i+1}$. Thus, the accepted language is respective of its right congruence. □

**Relation to Non-Counting Languages** The definition of respective of $\sim_L$ is reminiscent of the definition of non-counting languages [13]. A language $L \subseteq \Sigma^\omega$ is said to be *non-counting* iff $\exists n_0 \in \mathbb{N}.\ \forall n > n_0.\ \forall u, v \in \Sigma^*,\ w \in \Sigma^\omega . uv^n w \in L \iff uv^{n+1}w \in L$.

**Proposition 17** *If L is non-counting then L is respective of its right-congruence.*

The converse does not hold. The set $(aa)^*b^\omega$ is respective of $\sim_L$ but is not non-counting.

**Proposition 18** *There exist languages that are respective of $\sim_L$ but are not non-counting.*

One of the most commonly used temporal logics is Linear temporal logic (LTL) [27]. LTL formulas are non-counting [18, 34, 13, 28]. But, there are LTL formulas that characterize languages that are not in $\mathbb{IT}$ (and thus not in any of the $\mathbb{I}$ classes). Indeed, the formula $FG(a \vee Xa)$ characterizes the language $L = \Sigma^*(a + \Sigma a)^\omega$ and by Proposition 4, $|\sim_L| = 1$.

## 6 Closure properties

We examine what Boolean closure properties hold or do not hold for these classes of languages recognizable by an automaton isomorphic to the rightcon automaton, and by automata that are also respective of the right congruence.

It is a well known result that weak regular $\omega$-languages are closed under all Boolean operations as stated by the following proposition.

**Proposition 19 (c.f. [24])** *The class $\mathbb{DB} \cap \mathbb{DC}$ is closed under the Boolean operations complementation, union and intersection.*

The classes $\mathbb{IT}$, $\mathbb{IM}$ and $\mathbb{IP}$ are closed under complementation. The other classes that we consider are not closed under complementation. See Fig. 5 for counterexamples.

**Proposition 20** *The classes $\mathbb{IT}$, $\mathbb{IM}$ and $\mathbb{IP}$ are closed under complementation.*
*The classes $\mathbb{IB}$ and $\mathbb{IC}$ are not closed under complementation.*

**Proposition 21** *$\mathbb{RB}$, $\mathbb{RC}$, $\mathbb{RP}$, $\mathbb{RM}$ and $\mathbb{RT}$ are not closed under complementation.*

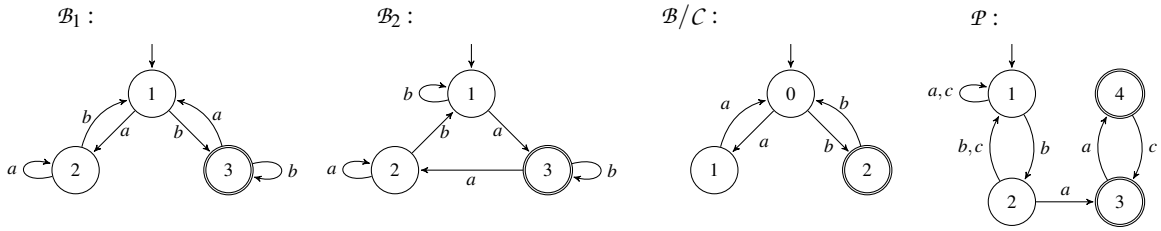


Figure 5: On the left examples for non-closure of union for $\mathbb{IB}$. On the right a DBA $\mathcal{B}$ and a DCA $\mathcal{C}$ showing $\mathbb{IB}$ and $\mathbb{IC}$ are not closed under complementation (the sink state is not shown).

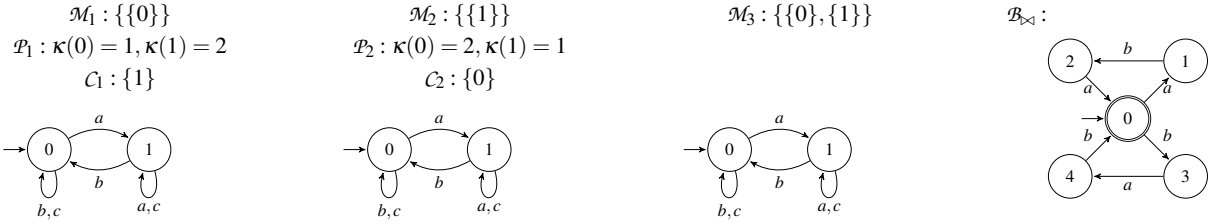


Figure 6: On the left, examples for non closure of union for $\mathbb{IM}$, $\mathbb{IP}$, and $\mathbb{IC}$. On the right $\mathcal{B}_{\bowtie}$.

**Proof.** Consider the DBA $\mathcal{P}$ in Fig. 5. The $\omega$-words $ba(ac)^\omega$, $a(ac)^\omega$, $(ac)^\omega$ and $(ca)^\omega$ distinguish its five states (the four shown in the figure, and the sink state). This shows that $\mathcal{P} \in \mathbb{IB} \subset \mathbb{IP} \subset \mathbb{IM} \subset \mathbb{IT}$. To see that it is respective of its right congruence, if $xy^\omega$ is in $[\![\mathcal{P}]\!]$ then $y$ must traverse the cycle of states 3 and 4. Thus for some $n_0$, $\mathcal{P}(xy^{n_0}) = 3$ and $y = (ac)^k$ for some $k \geq 1$, or $\mathcal{P}(xy^{n_0}) = 4$ and $y = (ca)^k$ for some $k \geq 1$. In either case, $\mathcal{P}(xy^n) = \mathcal{P}(xy^{n+1})$ for all $n \geq n_0$. However, its complement $\mathcal{P}^c$ accepts the word $b^\omega$. But for every $n \in \mathbb{N}$ we have $b^n \not\sim_{[\![\mathcal{P}^c]\!]} b^{n+1}$.

This shows that $\mathbb{RB}$, $\mathbb{RP}$, $\mathbb{RM}$ and $\mathbb{RT}$ are not closed under complementation. Consider the DCA $\mathcal{P}'$ obtained from $\mathcal{P}$ by marking the states $\{1,2,0\}$ where 0 is the sink state. Then $\mathcal{P}'$ recognizes the same language as $\mathcal{P}$. We get that $\mathcal{P}'$ is in $\mathbb{IC}$, is respective of its right congruence, yet its complement is not respective of its right congruence. □

Aside from the $\mathbb{DB} \cap \mathbb{DC}$ none of the classes are closed under union or intersection. The automata in Figures 5 and 6 are used to refute these closures. The complete proofs are given in the appendix.

**Proposition 22** *The classes $\mathbb{IB}$, $\mathbb{IC}$, $\mathbb{IP}$, $\mathbb{IM}$ and $\mathbb{IT}$ are not closed under union or intersection. The classes $\mathbb{RB}$, $\mathbb{RC}$, $\mathbb{RP}$, $\mathbb{RM}$ and $\mathbb{RT}$ are not closed under union or intersection.*

# 7 Discussion

We have explored properties of the right congruences of regular $\omega$-languages, characterized when a language has a trivial right congruence, defined classes of languages that have a fully informative right congruence, and defined an orthogonal property of a language being respective of its right congruence, which is implied by but does not imply the property of being non-counting. We have shown that there are languages with fully informative right congruences in every class of the infinite Wagner hierarchy, and that this remains true if we consider languages that are also respective of their right congruences. The (mostly) non-closure results under Boolean operations are not necessarily inimical to learnability. Our

hope is that future research will be able to take advantage of these properties in the search for efficient minimization and learning algorithms for regular $\omega$-languages.

## References


[1] F. Aarts & F.W. Vaandrager (2010): *Learning I/O Automata*. In: *CONCUR 2010 - Concurrency Theory, 21th Inter. Conf., CONCUR 2010, Paris, France, August 31-September 3, 2010. Proc.*, pp. 71–85, doi:10.1007/978-3-642-15375-4_6.

[2] D. Angluin (1987): *Learning Regular Sets from Queries and Counterexamples*. *Inf. Comput.* 75(2), pp. 87–106, doi:10.1016/0890-5401(87)90052-6.

[3] D. Angluin, U. Boker & D. Fisman (2016): *Families of DFAs as Acceptors of omega-Regular Languages*. In: *41st Inter. Symp. on Mathematical Foundations of Computer Science, MFCS*, pp. 11:1–11:14, doi:10.4230/LIPIcs.MFCS.2016.11.

[4] D. Angluin, U. Boker & D. Fisman (2018): *Families of DFAs as Acceptors of ω-Regular Languages*. *Logical Methods in Computer Science* Volume 14, Issue 1, doi:10.23638/LMCS-14(1:15)2018.

[5] D. Angluin & D. Fisman (2014): *Learning Regular Omega Languages*. In: *Algorithmic Learning Theory - 25th Inter. Conf., ALT Proc.*, pp. 125–139, doi:10.1007/978-3-319-11662-4_10.

[6] D. Angluin & D. Fisman (2016): *Learning regular omega languages*. *Theor. Comput. Sci.* 650, pp. 57–72, doi:10.1016/j.tcs.2016.07.031.

[7] B. Balle & M. Mohri (2015): *Learning Weighted Automata*. pp. 1–21, doi:10.1007/978-3-319-23021-4_1.

[8] F. Bergadano & S. Varricchio (1996): *Learning Behaviors of Automata from Multiplicity and Equivalence Queries*. *SIAM J. Comput.* 25(6), pp. 1268–1280, doi:10.1137/S009753979326091X.

[9] A. W. Biermann & J. A. Feldman (1972): *On the Synthesis of Finite-State Machines from Samples of Their Behavior*. *IEEE Trans. Comput.* 21(6), pp. 592–597, doi:10.1109/TC.1972.5009015.

[10] M. Bojańczyk (2014): *Transducers with Origin Information*. In: *Automata, Languages, and Programming - 41st Inter. Colloq., ICALP*, pp. 26–37, doi:10.1007/978-3-662-43951-7_3.

[11] M. Botincan & D. Babic (2013): *Sigma*: symbolic learning of input-output specifications*. In: *40th ACM SIGPLAN-SIGACT Symp. on Principles of Programming Language (POPL13)*, pp. 443–456, doi:10.1145/2429069.2429123.

[12] H. Calbrix, M. Nivat & A. Podelski (1994): *Ultimately Periodic Words of Rational w-Languages*. In: *Proc. of the 9th Inter. Conf. on Mathematical Foundations of Programming Semantics*, Springer-Verlag, pp. 554–566, doi:10.1007/3-540-58027-1_27.

[13] V. Diekert & P. Gastin (2008): *First-order definable languages*. In: *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*., pp. 261–306.

[14] D. Drews & L. D'Antoni (2017): *Learning Symbolic Automata*. In: *Tools and Algorithms for the Construction and Analysis of Systems - 23rd Inter. Conf., TACAS*, pp. 173–189, doi:10.1007/978-3-662-54577-5_10.

[15] A. Farzan, Y. Chenand E.M. Clarke, Y. Tsay & B. Wang (2008): *Extending Automated Compositional Verification to the Full Class of Omega-Regular Languages*. In: *Tools and Algorithms for the Construction and Analysis of Systems, LNCS* 4963, Springer Berlin Heidelberg, pp. 2–17, doi:10.1007/978-3-540-78800-3_2.

[16] D. Fisman (2018): *Inferring Regular Languages and ω-Languages*. *Journal of Logical and Algebraic Methods in Programming* 98C, pp. 27–49, doi:10.1016/j.jlamp.2018.03.002.

[17] M. Fliess (1974): *Matrices de Hankel*. *Journal de Mathemátiques Pureset Appliqueés*, pp. 197–224.

[18] D. M. Gabbay, A. Pnueli, S. Shelah & J. Stavi (1980): *On the Temporal Basis of Fairness*. In: *Conf. Record of the 7th Annual ACM Symp. on Principles of Programming Languages*, pp. 163–173, doi:10.1145/567446.567462.

[19] O. Kupferman, G. Morgenstern & A. Murano (2006): *Typeness for omega-regular Automata*. *Int. J. Found. Comput. Sci.* 17(4), pp. 869–884, doi:10.1142/S0129054106004157.

[20] M. Leucker (2006): *Learning Meets Verification*. In: *Formal Methods for Components and Objects, 5th Inter. Symp., FMCO*, pp. 127–151, doi:10.1007/978-3-540-74792-5_6.

[21] O. Maler & I. Mens (2017): *A Generic Algorithm for Learning Symbolic Automata from Membership Queries*. In: *Models, Algorithms, Logics and Tools - Essays Dedicated to Kim Guldstrand Larsen on the Occasion of His 60th Birthday*, pp. 146–169, doi:10.1007/978-3-319-63121-9_8.

[22] O. Maler & A. Pnueli (1995): *On the Learnability of Infinitary Regular Sets*. *Inf. Comput.* 118(2), pp. 316–326, doi:10.1006/inco.1995.1070.

[23] O. Maler & L. Staiger (1997): *On Syntactic Congruences for Omega-Languages*. *Theor. Comput. Sci.* 183(1), pp. 93–112, doi:10.1016/S0304-3975(96)00312-X.

[24] Z. Manna & A. Pnueli (1990): *A Hierarchy of Temporal Properties*. In: *Proc. of the Ninth Annual ACM Symp. on Principles of Distributed Computing*, pp. 377–410, doi:10.1145/93385.93442.

[25] I. Mens & O. Maler (2015): *Learning Regular Languages over Large Ordered Alphabets*. *Logical Methods in Computer Science* 11(3), doi:10.2168/LMCS-11(3:13)2015.

[26] D. Perrin & J-E. Pin (2004): *Infinite Words: Automata, Semigroups, Logic and Games*. *Pure and Applied Math* 141, Elsevier.

[27] A. Pnueli (1977): *The Temporal Logic of Programs*. In: *FOCS*, pp. 46–57, doi:10.1109/SFCS.1977.32.

[28] A. Rabinovich (2014): *A Proof of Kamp's theorem*. *Logical Methods in Computer Science* 10(1), doi:10.2168/LMCS-10(1:14)2014.

[29] B. Le Saëc (1990): *Saturating right congruences*. *ITA* 24, pp. 545–560.

[30] S. Schewe (2010): *Beyond Hyper-Minimisation—Minimising DBAs and DPAs is NP-Complete*. In: *IARCS Annual Conf. on Foundations of Software Technology and Theor. Comp. Science, FSTTCS*, pp. 400–411, doi:10.4230/LIPIcs.FSTTCS.2010.400.

[31] L. Staiger (1983): *Finite-State omega-Languages*. *J. Comput. Syst. Sci.* 27(3), pp. 434–448, doi:10.1016/0022-0000(83)90051-X.

[32] D. Long Van, B. Le Saëc & I. Litovsky (1995): *Characterizations of Rational omega-Languages by Means of Right Congruences*. *Theor. Comput. Sci.* 143(1), pp. 1–21, doi:10.1016/0304-3975(95)80022-2.

[33] K. W. Wagner (1975): *A Hierarchy of Regular Sequence Sets*. In: *MFCS*, pp. 445–449, doi:10.1007/3-540-07389-2_231.

[34] P. Wolper (1983): *Temporal Logic Can Be More Expressive*. *Information and Control* 56(1/2), pp. 72–99, doi:10.1016/S0019-9958(83)80051-5.

# A Appendix

## A.1 Proofs omitted from Section 2

**Proof of Proposition 1** Proposition 1 states the following:

Let $\mathcal{D}$ be a deterministic $\omega$-automaton. Then $\sim_{\mathcal{D}}$ refines $\sim_{[\![\mathcal{D}]\!]}$.

**Proof.** Assume $u \sim_{\mathcal{D}} v$ for some $u, v \in \Sigma^*$. Thus $\mathcal{D}(u) = \mathcal{D}(v)$. That is, there exists a state $q$ of $\mathcal{D}$ such that reading $u$ or $v$ from the initial state of $\mathcal{D}$ reaches $q$. Therefore, for every $x \in \Sigma^*$ we have $\mathcal{D}(ux) = \mathcal{D}(vx)$. In particular, for every $w \in \Sigma^\omega$ and every prefix $y \in \Sigma^*$ of $w$, $\mathcal{D}(uy) = \mathcal{D}(vy)$. Therefore exactly the same set of states are visited infinitely often by $\mathcal{D}$ when reading $uw$ or $vw$. Thus $uw \in [\![\mathcal{D}]\!]$ iff $vw \in [\![\mathcal{D}]\!]$. (Note that this is true for all the acceptance condition considered by the different $\omega$-automata).

Thus $u \sim_{[\![\mathcal{D}]\!]} v$. □

**Proof of Proposition 2** Proposition 2 states the following:

Let $\mathcal{N}$ be a non-deterministic $\omega$-automaton. Then $\sim_{\mathcal{P}_\mathcal{N}}$ refines $\sim_{[\![\mathcal{N}]\!]}$.

**Proof.** Suppose $u \sim_{\mathcal{P}_\mathcal{N}} v$. Then $\mathcal{P}_\mathcal{N}$ reaches the same set $S$ of states of $\mathcal{N}$ on $u$ and on $v$. If $w \in \Sigma^\omega$ is such that $uw$ is in the language of $\mathcal{N}$, then there exists a state $q$ in $S$ such that there is an accepting computation of $\mathcal{N}$ from $q$ on $w$. Thus $vw$ is also in the language of $\mathcal{N}$, so $u \sim_{[\![\mathcal{N}]\!]} v$. □

## A.2 Proofs omitted from Section 4

**Proof of Proposition 6** Proposition 6 states the following:

$$\mathbb{IB} \cup \mathbb{IC} \subsetneq \mathbb{IM} \text{ and } \mathbb{IB} \cup \mathbb{IC} \subsetneq \mathbb{IP}$$

**Proof.** Inclusion follows since any DBA or DCA can be transformed into a DMA over the same structure (by defining $\alpha = \{S \subseteq Q : S \cap F \neq \emptyset$ for DBA, and dually for DCA) and to a DPA over the same structure (as mentioned in the proof of Proposition 5). The language $L_{PT}$ accepted by the DMA $\mathcal{T}$ in Fig. 3 and the equivalent DPA $\mathcal{P}$ is not in $\mathbb{IB}$ since it cannot be accepted by a DBA because it has an accepting SCC ($\{\lambda, 1\}$) which is a subset of a rejecting SCC ($\{\lambda, 1, 0\}$). It is not in $\mathbb{IC}$ since it cannot be accepted by a DCA because it has a rejecting SCC ($\{1\}$) which is a subset of an accepting SCC ($\{\lambda, 1\}$). □

## A.3 Proofs omitted from Section 5

**Proof of Proposition 14** Proposition 14 states the following:

There exists languages in $\mathbb{IB} \cap \mathbb{IC}$ that are not respective of their right congruence.

**Proof.** The DBA $\mathcal{D}_{bad}$ in Fig. 4 is in $\mathbb{IB} \cap \mathbb{IC}$ since it has no alternations between accepting and rejecting SCCs along an inclusion chain. To see that it is isomorphic to its right congruence, take $\varepsilon$, 0, 1, 2 13 and 14 as the representative words for states $\lambda$, 0, 1, 2, 3 and 4 respectively. The words $14^\omega$, $0^\omega$, $3^\omega$, $1^\omega$, $04^\omega$ distinguish the representative words. Last, as in the proof of Proposition 13 the pair of words $(\varepsilon, 1012)$ shows that its language, $D_{bad}$, is not respective of its right congruence. □

**Proof of Proposition 15** Proposition 15 states the following:

$$\mathbb{RM} \supsetneq \mathbb{RP} \text{ and and } \mathbb{RT} \supsetneq \mathbb{RM}$$

**Proof.** The language $M = [\![\mathcal{M}]\!]$ accepted by the DMA $\mathcal{M}$ in Fig. 3 is respective of its right congruence, but as shown in the proof of Proposition 10 it is not in $\mathbb{IP}$. To see that $M$ is respective of its right congruence note that $M = (a+bb^*c)^*(a^\omega + b^\omega)$. In particular, $xu^\omega \in M$ iff $x \in (a+bb^*c)^*$ and $u \in \{a^k \mid k \in \mathbb{N}\} \cup \{b^k \mid k \in \mathbb{N}\}$, and for any such $x$ and $u$ we have $xu^n \sim_M xu^{n+1}$.

The language accepted by the DMA $\mathcal{T}$ in Fig. 3 is $(a+b)^*(a^\omega + b^\omega)$. Since it is accepted by a one state DTA, by Proposition 1, it is in $\mathbb{IT}$. It is also respective of its right congruence using similar reasoning to the above. Thus it is in $\mathbb{RT}$. However, it cannot be recognized by a DMA with one state and it is thus not in $\mathbb{IM} \supset \mathbb{RM}$. □

**Proof of Proposition 17**

If $L$ is non-counting then $L$ is respective of its right-congruence.

**Proof.** Let $L \subseteq \Sigma^\omega$ be non-counting. Then $\exists n_0 \in \mathbb{N}.\ \forall n > n_0.\ \forall u, v \in \Sigma^*.\ \forall w \in \Sigma^\omega.\ uv^n w \in L \iff uv^{n+1}w \in L$. Assume towards contradiction that $L$ is not respective of its right congruence. Thus, there exists $x, u \in \Sigma^*$ such that $xu^\omega \in L$ yet $\forall n_0 \in \mathbb{N}$ there exists $n > n_0$ such that $xu^n \not\sim_L xu^{n+1}$. Hence, there exists $w \in \Sigma^\omega$ such that $xu^n w \in L$ iff $xu^{n+1}w \notin L$, contradicting that $L$ is non-counting. □

**Proof of Proposition 18**

There exist languages that are respective of $\sim_L$ but are not non-counting.

**Proof.** The language $(aa)^*b^\omega$ is respective of $\sim_L$ but is not non-counting. To see that it is not non-counting note that for every even $n$ we have that $a^n b^\omega \in L$ while $a^{n+1}b^\omega \notin L$. To see that it is respective of $\sim_L$ note that all words in $L$ are of the form $(aa)^n(b^k)^\omega$, and for all of these it holds that $(aa)^n b^k \sim_L (aa)^n b^{k+1}$ since $(aa)^n b^k w \in L$ iff $w = b^\omega$ iff $(aa)^n b^{k+1} w \in L$. □

## A.4 Proofs omitted from Section 6

**Lemma 23** *Let $L$ be a regular $\omega$-language, and let $L^c = \Sigma^\omega \setminus L$. Then $\mathcal{R}_{[\![L]\!]}$ is isomorphic to $\mathcal{R}_{[\![L^c]\!]}$.*

**Proof.** $u \sim_L v$ iff $\forall w \in \Sigma^\omega$, $uw \in L \iff vw \in L$ iff
$\forall w \in \Sigma^\omega$, $uw \notin L \iff vw \notin L$ iff $u \sim_{L^c} v$.
Thus $\mathcal{R}_{[\![L]\!]}$ and $\mathcal{R}_{[\![L^c]\!]}$ are isomorphic. □

The propositions below partition the results summarized in Propositions 20 to 22 in a slightly different manner.

**Proposition 24** *$\mathbb{IT}$, $\mathbb{IM}$ and $\mathbb{IP}$ are closed under complementation.*

**Proof.** The result for $\mathbb{IM}$ was established in [32].

Given a DTA $\mathcal{T} = \langle \Sigma, Q, \lambda, \delta, \alpha \rangle \in \mathbb{IT}$ we can define the DTA $\mathcal{T}^c = \langle \Sigma, Q, \lambda, \delta, \alpha^c \rangle$ where $\alpha^c = \{T \subseteq \delta \mid T \notin \alpha\}$. The DTA $\mathcal{T}^c$ recognizes $\Sigma^\omega \setminus [\![\mathcal{T}]\!]$ and is isomorphic to $\mathcal{T}$ per Lemma 23.

Similarly, given a DPA $\mathcal{P} = \langle \Sigma, Q, \lambda, \delta, \kappa \rangle \in \mathbb{IP}$ we can define the DPA $\mathcal{P}^c = \langle \Sigma, Q, \lambda, \delta, \kappa^c \rangle$ by assigning $\kappa^c(q) = \kappa(q) + 1$. The DPA $\mathcal{P}^c$ recognizes $\Sigma^\omega \setminus [\![\mathcal{P}]\!]$ and is isomorphic to $\mathcal{P}$ per Lemma 23. □

**Proposition 25** *$\mathbb{IB}$ and $\mathbb{IC}$ are not closed under complementation.*

**Proof.** Consider the language $L = ((aa + bb)^* bb)^\omega$. It is accepted by the DBA $\mathcal{B}$ in Fig. 5. Its rightcon automaton is isomorphic to $\mathcal{B}$. To see this, consider $\varepsilon$, $a$, $b$ and $ab$ as representative words for the states 0, 1, 2 and the sink state. The words $a(bb)^\omega$, $aa(bb)^\omega$ and $baa(bb)^\omega$ distinguish the four states.

The rightcon automaton for its complement is also isomorphic to $\mathcal{B}$ but it is not in $\mathbb{DB}$. Thus $\mathbb{IB}$ is not closed under complementation. Likewise, if we regard this automaton as a DCA $\mathcal{C}$ we obtain a language in $\mathbb{IC}$, whose complement is not in $\mathbb{IC}$ since it is not in $\mathbb{DC}$. □

**Proposition 26** *$\mathbb{IM}$ and $\mathbb{IP}$ are not closed under union or intersection.*

**Proof.** It was established in [32] that $\mathbb{IM}$ is not closed under intersection, and since it is closed under complementation, its non-closure under union follows.

Consider the DPAs $\mathcal{P}_1$ and $\mathcal{P}_2$ from Figure 6. The word $c^\omega$ differentiates state 0 from state 1 in both $\mathcal{P}_1$ and $\mathcal{P}_2$. Thus both $\mathcal{P}_1$ and $\mathcal{P}_2$ are in $\mathbb{IP}$. The DPA $\mathcal{P}_3$ recognizes the union of $[\![\mathcal{P}_1]\!] \cup [\![\mathcal{P}_2]\!]$. However, the respective language $[\![\mathcal{P}_3]\!]$ has one equivalence class in its right congruence, and is thus not in $\mathbb{IP}$. This shows that $\mathbb{IP}$ is not closed under union.

Non-closure under intersection follows since by Proposition 20 $\mathbb{IP}$ is closed under complementation. □

**Proposition 27** *$\mathbb{IC}$ is not closed under union and $\mathbb{IB}$ is not closed under intersection.*

**Proof.** The proof that $\mathbb{IC}$ is not closed under union follows the same reasoning as the respective proof for $\mathbb{IM}$ in Proposition 26 since equivalent DCA $\mathcal{C}_1$ and $\mathcal{C}_2$ can be defined on the same structure.

Non-closure of $\mathbb{IB}$ under union follows, since $L \in \mathbb{IC} \iff L^c \in \mathbb{IB}$. Thus, $L_1 \in \mathbb{IC} \wedge L_2 \in \mathbb{IC} \wedge L_1 \cup L_2 \notin \mathbb{IC}$ implies $L_1^c \in \mathbb{IB} \wedge L_2^c \in \mathbb{IB} \wedge (L_1 \cup L_2)^c \notin \mathbb{IB}$. And the last conjunct is equivalent to $L_1^c \cap L_2^c \notin \mathbb{IB}$. □

**Proposition 28** *$\mathbb{IB}$ is not closed under union and $\mathbb{IC}$ is not closed under intersection.*

**Proof.** We show that $\mathbb{IB}$ is not closed under union. From the duality of $\mathbb{IC}$ and $\mathbb{IB}$ it follows that $\mathbb{IC}$ is not closed under intersection.

Consider the languages $L_1$ and $L_2$ accepted by the DBA $\mathcal{B}_1$ and $\mathcal{B}_2$ in Figure 5, respectively. We claim that $L_1, L_2 \in \mathbb{IB}$. To see that note that $(ba)^\omega$ and $(ab)^\omega$ distinguish all states of $\mathcal{B}_1$, and $b^\omega$, $ab^\omega$ distinguish all states of $\mathcal{B}_2$.

We claim that $L_1 \cup L_2$ consists of the language of all omega words with infinitely many $b$'s, and therefore has one right congruence class and is not in $\mathbb{IB}$. To see this, note first that $\mathcal{B}_1$ and $\mathcal{B}_2$ both reject words with a suffix $a^\omega$ (a run on such words eventually loops in state 2 in both machines.) If a word $w$

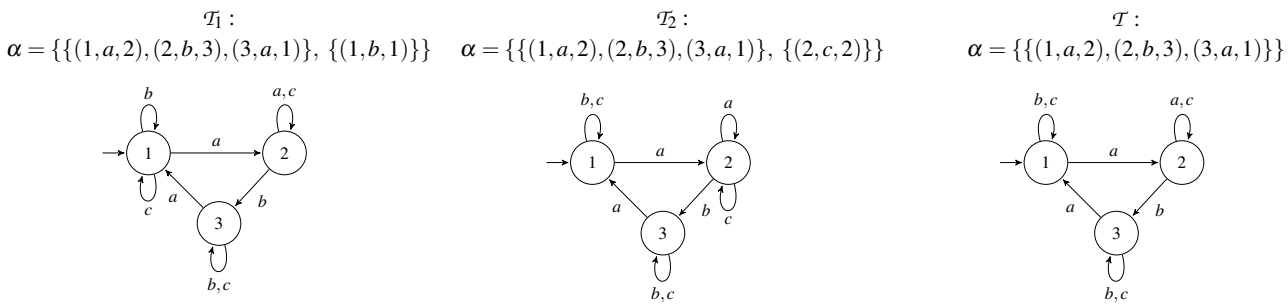


Figure 7: Example for non closure of intersection for $\mathbb{IT}$ and $\mathbb{RT}$.

has infinitely many $b$'s, then either (1) it ends in $b^\omega$, and is accepted by $\mathcal{B}_1$, or (2) it has infinitely many $a$'s and infinitely many $b$'s, and is accepted by $\mathcal{B}_2$. □

**Proposition 29** *The classes* $\mathbb{RM}, \mathbb{RP}, \mathbb{RC}$ *are not closed under union.*

**Proof.** Consider the DMAs $\mathcal{M}_1$ and $\mathcal{M}_2$ in Fig. 6. They are in $\mathbb{RM}$, $\mathbb{RP}$ and $\mathbb{RC}$. As shown in the proof of Proposition 26 their union (given by $\mathcal{M}_3$ in Fig. 6) is not in $\mathbb{IM}$ or $\mathbb{IP}$ or $\mathbb{IC}$ and thus not in $\mathbb{RM}$ or $\mathbb{RP}$ or $\mathbb{RC}$. □

**Proposition 30** *The classes* $\mathbb{RM}, \mathbb{RP}, \mathbb{RB}$ *are not closed under intersection.*

**Proof.** Consider the DMAs $\mathcal{M}_1'$ and $\mathcal{M}_2'$ obtained from $\mathcal{M}_1$ and $\mathcal{M}_2$ by changing the acceptance condition to $\{\{0\},\{0,1\}\}$ and $\{\{1\},\{0,1\}\}$, respectively. Both $\mathcal{M}_1'$ and $\mathcal{M}_2'$ are in $\mathbb{IM}$ since $c^\omega$ distinguishes the two states. In addition they are respective of their right congruence and are thus in $\mathbb{RM}$. Their intersection is the DMA $\mathcal{M}_3'$ obtained from $\mathcal{M}_3$ by changing the acceptance condition to $\{\{0,1\}\}$. The language $[\![\mathcal{M}_3']\!]$ however has only one equivalence class in its right congruence, implying $\mathcal{M}_3'$ is not in $\mathbb{RM}$.

The proof for $\mathbb{RB}$ follows since $\mathcal{M}_1'$ and $\mathcal{M}_2'$ can also be considered as DBAs with accepting sets $\{0\}$ and $\{1\}$, respectively.

The proof for $\mathbb{RP}$ follows since $\mathcal{M}_1'$ and $\mathcal{M}_2'$ can also be considered as DPAs with coloring functions $\kappa_1(0,1) = (1,2)$ and $\kappa_2(0,1) = (2,1)$, respectively. □

**Proposition 31** *The classes* $\mathbb{IT}$ *and* $\mathbb{RT}$ *are not closed under intersection.*

**Proof.** Consider the DTAs $\mathcal{T}_1$ and $\mathcal{T}_2$ in Fig. 7. We claim that $[\![\mathcal{T}_1]\!]$ and $[\![\mathcal{T}_2]\!]$ are in $\mathbb{RT}$. The words $b^\omega$ and $(aba)b^\omega$ distinguish all states of $\mathcal{T}_1$ showing it is in $\mathbb{IT}$ and the words $c^\omega$, $(aba)c^\omega$ distinguish all states of $\mathcal{T}_2$ showing it is in $\mathbb{IT}$. To see that $\mathcal{T}_1$ is respective of its right congruence note that $xy^\omega \in [\![\mathcal{T}_1]\!]$ if either $y$ is a cyclic permutation of $aba$ or if $y$ is in $b^+$ and $x \sim_{[\![\mathcal{T}_1]\!]} \varepsilon$. In bot cases for any $n$ we have $xy^n \sim_{[\![\mathcal{T}_1]\!]} xy^{n+1}$. The argument for $\mathcal{T}_2$ being respective is similar. Clearly, $\mathcal{T}$ of the same figure recognizes the intersection of $\mathcal{T}_1$ and $\mathcal{T}_2$. However $\sim_{[\![\mathcal{T}]\!]}$ has only one equivalence class, showing $\mathcal{T} \notin \mathbb{IT} \supset \mathbb{RT}$.

□

**Proposition 32** *The classes* $\mathbb{IT}$ *and* $\mathbb{RT}$ *are not closed under union.*

**Proof.** Non-closure of $\mathbb{IT}$ under union follows from its non-closure under intersection (Prop. 31) and its closure under complementation (Prop. 20).

To see that $\mathbb{RT}$ is not closed under union consider the languages $T_1 = \Sigma^*(ba)^\omega + \Sigma^* ac^\omega$ and $T_1 = \Sigma^*(ba)^\omega + \Sigma^* bc^\omega$. It is easy to see that both $T_1$ and $T_2$ are in $\mathbb{IT}$ and are respective of their right congruence. However, their union $T = \Sigma^*((ab)^\omega + c^\omega))$ is not in $\mathbb{RT}$ since $\sim_T$ has a trivial right congruence as per Prop. 4 but a DTA recognizing it requires at least 2 states. □

**Proposition 33** *The class $\mathbb{RB}$ is not closed under union and the class $\mathbb{RC}$ is not closed under intersection.*

**Proof.** Consider the DBAs $\mathcal{B}_1$ and $\mathcal{B}_2$ in Fig 5. As shown in the proof of Proposition 28 they are both in $\mathbb{IB}$ but their union is not in $\mathbb{IB}$. It is easy to see that they are respective of their right congruence and thus in $\mathbb{RB}$, but since their union is not in $\mathbb{IB}$ it is not in $\mathbb{RB}$ either. □