



A Myhill-Nerode Theorem for Generalized Automata, with Applications to Pattern Matching and Compression

Nicola Cotumaccio  

Gran Sasso Science Institute, L'Aquila, Italy
Dalhousie University, Halifax, Canada

Abstract

The model of generalized automata, introduced by Eilenberg in 1974, allows representing a regular language more concisely than conventional automata by allowing edges to be labeled not only with characters, but also strings. Giammaresi and Montalbano introduced a notion of determinism for generalized automata [STACS 1995]. While generalized deterministic automata retain many properties of conventional deterministic automata, the uniqueness of a minimal generalized deterministic automaton is lost.

In the first part of the paper, we show that the lack of uniqueness can be explained by introducing a set $\mathcal{W}(\mathcal{A})$ associated with a generalized automaton \mathcal{A} . The set $\mathcal{W}(\mathcal{A})$ is always trivially equal to the set of all prefixes of the language recognized by the automaton, if \mathcal{A} is a conventional automaton, but this need not be true for generalized automata. By fixing $\mathcal{W}(\mathcal{A})$, we are able to derive for the first time a full Myhill-Nerode theorem for generalized automata, which contains the textbook Myhill-Nerode theorem for conventional automata as a degenerate case.

In the second part of the paper, we show that the set $\mathcal{W}(\mathcal{A})$ leads to applications for pattern matching and data compression. Wheeler automata [TCS 2017, SODA 2020] are a popular class of automata that can be compactly stored using $e \log \sigma(1 + o(1)) + O(e)$ bits (e being the number of edges, σ being the size of the alphabet) in such a way that pattern matching queries can be solved in $\tilde{O}(m)$ time (m being the length of the pattern). In the paper, we show how to extend these results to generalized automata. More precisely, a Wheeler generalized automata can be stored using $\epsilon \log \sigma(1 + o(1)) + O(e + rn)$ bits so that pattern matching queries can be solved in $\tilde{O}(rm)$ time, where ϵ is the total length of all edge labels, r is the maximum length of an edge label and n is the number of states.

2012 ACM Subject Classification Theory of computation → Regular languages; Theory of computation → Pattern matching; Theory of computation → Data compression

Keywords and phrases Generalized Automata, Myhill-Nerode Theorem, Regular Languages, Wheeler Graphs, FM-index, Burrows-Wheeler Transform

Digital Object Identifier 10.4230/LIPIcs.STACS.2024.26

Related Version *Full Version:* <https://arxiv.org/abs/2302.06506>

Funding This work was partially funded by Dante Labs.

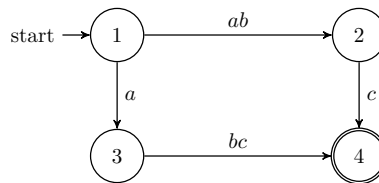
1 Introduction

The class of regular languages can be defined starting from *non-deterministic finite automata* (NFAs). In his monumental work [18] on automata theory (which dates back to 1974), Eilenberg proposed a natural generalization of NFAs where edges can be labeled not only with characters but with (possibly empty) finite strings, the so-called *generalized non-deterministic finite automata* (GNFAs). While classical automata are only a special case of generalized automata, it is immediate to realize that generalized automata can only recognize regular languages, because it is well-known that epsilon transitions do not add expressive



power [31], and a string-labeled edge can be decomposed into a path of edges labeled only with characters. However, generalized automata can represent regular languages more concisely than classical automata. A standard measure of the complexity of a regular language is the minimum number of states of some automaton recognizing the language, and generalized automata may have fewer states than conventional automata. In generalized automata, we assume that both the number of states and the number of edges are finite, but the number of edges cannot be bounded by some function of the number of states and the size of the finite alphabet (and so edge labels may be arbitrarily long). As a consequence, in principle it is not clear whether the problem of determining the minimum number states of some generalized automaton recognizing a given language is decidable. In [29], Hashiguchi showed that the problem is decidable by proving that there must exist a state-minimal generalized automaton for which the lengths of edge labels can be bounded by a function that only depends on the size of the syntactic monoid recognizing the language.

An NFA is a *deterministic finite automaton (DFA)* if no state has two distinct outgoing edges with the same label. This local notion of determinism extends to global determinism, that is, given a string α , there exists at most one path labeled α that can be followed starting from the initial state. However, this is not true for generalized automata (see Figure 1). When considering generalized automata, we must add the additional requirement that no state has two distinct outgoing edges such that one edge label is a prefix of the other edge label. By adding this requirement, we retrieve global determinism, thus obtaining *generalized deterministic finite automata (GDFAs)*.



■ **Figure 1** No state has two distinct outgoing edges with the same label, but there are two distinct paths labeled abc from the initial state.

For every regular language, there exists a *unique* deterministic automaton recognizing the language and having the minimum number of states among all deterministic automata recognizing the language, the *minimal DFA* of the language. More generally, a classical textbook result in automata theory is the *Myhill-Nerode theorem*. Let $\text{Pref}(\mathcal{L})$ be the set of all strings prefixing at least one string in the language \mathcal{L} . We have the following result.

► **Theorem 1** (Myhill-Nerode theorem). *Let $\mathcal{L} \subseteq \Sigma^*$. The following are equivalent:*

1. \mathcal{L} is recognized by an NFA.
2. The Myhill-Nerode equivalence $\equiv_{\mathcal{L}}$ has finite index.
3. There exists a right-invariant equivalence relation \sim on $\text{Pref}(\mathcal{L})$ of finite index such that \mathcal{L} is the union of some \sim -classes.
4. \mathcal{L} is recognized by a DFA.

Moreover, if one of the above statements is true (and so all the above statements are true), then there exists a unique minimal DFA recognizing \mathcal{L} (that is, two DFAs having the minimum number of states among all DFAs that recognize \mathcal{L} must be isomorphic).

The problem of studying the notion of determinism in the setting of generalized automata was approached by Giammaresi and Montalbano [28, 27]. The notion of isomorphism can be naturally extended to GDFAs (intuitively, two GDFAs are isomorphic if they are the same

G DFA up to renaming the states), and the natural question is whether one can analogously define the minimal G DFA of a regular language. This is not possible: in general, there can exist two or more non-isomorphic state-minimal G DFAs recognizing a given regular language. Consider the two distinct G DFAs in Figure 2. It is immediate to check that the two (non-isomorphic) G DFAs recognize the same language, and it can be shown that the no G DFA with less than three states can recognize the same language [28].



■ **Figure 2** Two state-minimal G DFAs recognizing the same regular language.

The non-uniqueness of a state-minimal G DFAs seems to imply a major difference in the behavior of generalized automata compared to conventional automata, so it looks like there is no hope to derive a structural result like the Myhill-Nerode theorem in the model of the generalized automata. It is natural to wonder whether the lack of uniqueness should be interpreted as a weakness of the model of generalized automata, or rather as a consequence of some deeper property. Consider a conventional automaton \mathcal{A} that recognizes a language \mathcal{L} . As typical in automata theory, we can assume that all states are reachable from the initial state, and all states are either final or they allow reaching a final state. Then, the set $\mathcal{W}(\mathcal{A})$ of all strings that can be read starting from the initial state and reaching some state is exactly equal to $\text{Pref}(\mathcal{L})$. However this is no longer true in the model of generalized automata: typically, we do not have $\mathcal{W}(\mathcal{A}) = \text{Pref}(\mathcal{L})$, but only $\mathcal{W}(\mathcal{A}) \subseteq \text{Pref}(\mathcal{L})$. For example, consider Figure 2. In both automata we have $a^3 \in \text{Pref}(\mathcal{L})$, but we have $a^3 \in \mathcal{W}(\mathcal{A})$ only for the G DFA on the left.

Given $\mathcal{W} \subseteq \text{Pref}(\mathcal{L})$, we say that a GNFA \mathcal{A} recognizing \mathcal{L} is a \mathcal{W} -GNFA if $\mathcal{W}(\mathcal{A}) = \mathcal{W}$. We will show that, if \mathcal{L} is recognized by a \mathcal{W} -G DFA, then there exists a *unique* state-minimal \mathcal{W} -G DFA recognizing \mathcal{L} . In particular, our result will imply the uniqueness of the minimal automaton for standard DFAs, because for DFAs it must necessarily be $\mathcal{W} = \text{Pref}(\mathcal{L})$.

We will actually prove much more. We will show that, once we fix \mathcal{W} , then nondeterminism and determinism still have the same expressive power, and it is possible to derive a characterization in terms of equivalence relations. In other words, we will prove a *Myhill-Nerode theorem for generalized automata*. To this end, we will introduce the notion of *locally bounded set* (Definition 8), which we can use to prove the following result.

► **Theorem 2** (Myhill-Nerode theorem for generalized automata). *Let $\mathcal{L} \subseteq \Sigma^*$ and let $\mathcal{W} \subseteq \Sigma^*$ be a locally bounded set such that $\mathcal{L} \cup \{\epsilon\} \subseteq \mathcal{W} \subseteq \text{Pref}(\mathcal{L})$. The following are equivalent:*

1. \mathcal{L} is recognized by a \mathcal{W} -GNFA.
2. The Myhill-Nerode equivalence $\equiv_{\mathcal{L}, \mathcal{W}}$ has finite index.
3. There exists a right-invariant equivalence relation \sim on \mathcal{W} of finite index such that \mathcal{L} is the union of some \sim -classes.
4. \mathcal{L} is recognized by a \mathcal{W} -G DFA.

Moreover, if one of the above statements is true (and so all the above statements are true), then there exists a unique minimal \mathcal{W} -G DFA recognizing \mathcal{L} (that is, two \mathcal{W} -G DFAs having the minimum number of states among all \mathcal{W} -G DFAs that recognize \mathcal{L} must be isomorphic).

In particular, we will show that there is no loss of generality in assuming that $\mathcal{W} \subseteq \Sigma^*$ is a locally bounded set such that $\mathcal{L} \cup \{\epsilon\} \subseteq \mathcal{W} \subseteq \text{Pref}(\mathcal{L})$, because these are *necessary* conditions for the existence of a \mathcal{W} -GNFA. We conclude that our Myhill-Nerode theorem for GNFA provides the first structural result for the model of generalized automata.

In the second part of the paper, we show that the set $\mathcal{W}(\mathcal{A})$ sheds new light on the *String Matching in Labeled Graphs (SMLG)* problem. The SMLG problem has a fascinating history that dates back to more than 30 years ago. Loosely speaking, the SMLG problem can be defined as follows: given a directed graph whose nodes are labeled with nonempty strings and given a pattern string, decide whether the pattern can be read by following a path on the graph and concatenating the node labels. The SMLG problem is a natural generalization of the classical pattern matching problem on texts (which requires determining whether a pattern occurs in a text) because texts can be seen as graphs consisting of a single path. The pattern matching problem on text can be efficiently solved in $O(n + m)$ time (n being the length of the text, m being the length of the pattern) by using the Knuth-Morris-Pratt algorithm [34]. The SMLG problem is more challenging, and the complexity can be affected by the specific variant of the pattern matching problem under consideration or the class of graphs to which the problem is restricted. For example, in the (approximate) variant where one allows errors in the graph the problem becomes NP-hard [5], so generally errors are only allowed in the pattern. The SMLG problem was studied extensively during the nineties [35, 1, 39, 5, 41, 37]; Amir et al. showed how to solve the (exact) SMLG problem on arbitrary graphs in $O(\epsilon + me)$ time [5], where e is the number of edges in the graph, m is the length of the pattern, and ϵ is the total length of all labels in the graph. Recently, the SMLG problem has been back in the spotlight. Equi et al. [21, 19, 20] showed that, *on arbitrary graphs*, for every $\epsilon > 0$ the SMLG cannot be solved in $O(me^{1-\epsilon})$ or $O(m^{1-\epsilon}e)$ time, unless the Orthogonal Vectors hypothesis fails. In applications (especially in bioinformatics) we often need faster algorithms, so the SMLG problem has been restricted to class of graphs on which it can be solved more efficiently. For example, *Elastic Founder graphs* can be used to represent multiple sequence alignments (MSA), a central model of biological evolution, and on Elastic Founder graphs the SMLG problem can be solved in linear time under a number of assumptions which only have a limited impact on the generality of the model [23, 22, 42].

The pattern matching problem on texts has been revolutionized by the invention of the Burrows-Wheeler Transform [10] and the FM-index [24, 25], which allow solving pattern matching queries efficiently on *compressed* text, thus establishing a new paradigm in bioinformatics (where the huge increase of genomic data requires the development of space-efficient algorithms) [43]. Recently, these ideas were extended to NFAs. In particular, *Wheeler NFAs* are a popular class of automata on which the SMLG problem can be solved in linear time, while only storing a *compact* representation of the Wheeler NFA [26, 3]. A special case of Wheeler NFAs are de Bruijn graphs [9], which are used to perform Eulerian sequence assembly [32, 40, 7]. Wheeler NFAs are also of relevant theoretical interest: for example, the powerset construction applied to a Wheeler NFA leads to a linear blow-up in the number of states of the equivalent DFA, and the equivalent DFA is Wheeler [4]; on arbitrary NFAs, the blow-up can be exponential.

The missing step is to determine whether it is possible to generalize the Burrows-Wheeler Transform and the FM-index to GNFA, so that the resulting data structures can also be applied to Elastic Founder graphs and other classes of graphs where labels can be arbitrary strings. Indeed, in data compression it is common to consider edge-labeled graphs where one compresses unary paths in the graph to save space and the path is replaced by a single edge labeled with the concatenation of all labels. For example, some common data structures that

are stored using this mechanism are Patricia trees, suffix trees and pangenomes [38, 6, 36]. In the following, we only consider GNFA without ϵ -transitions (that is, GNFA where no edge is labeled with the empty string ϵ) because in general a GNFA may contain arbitrarily long paths consisting of adjacent ϵ -transitions, so the SMLG problem becomes more difficult and our mechanism, based on the FM-index, fails. We say that a GNFA is an r -GNFA if all edge labels have length at most r (so a GNFA is a conventional NFA if and only if it is a 1-GNFA). Let m , e and ϵ as above, let n be the number of states, and let $\sigma = |\Sigma|$. In Section 4, we will extend the notion of Wheelerness to GNFA. The key ingredient will be the same set $\mathcal{W}(\mathcal{A})$ that we use in our Myhill-Nerode theorem: we will consider a partial order $\preceq_{\mathcal{A}}$ which sorts the set of all states with respect to the *co-lexicographical order* of the strings in $\mathcal{W}(\mathcal{A})$ (See Definition 21). We will then prove the following result.

► **Theorem 3** (FM-index of generalized automata). *Let \mathcal{A} be a Wheeler r -GNFA. Then, we can encode \mathcal{A} by using $\epsilon \log \sigma(1 + o(1)) + O(e + rn)$ bits so that later on, given a pattern $\alpha \in \Sigma^*$ of length m , we can solve the SMLG problem on \mathcal{A} in $O(m \log \log \sigma)$ time. Within the same time bound we can also decide whether $\alpha \in \mathcal{L}(\mathcal{A})$.*

If $r = 1$ (that is, if \mathcal{A} is a conventional Wheeler NFA), we conclude that we can encode \mathcal{A} by using $e \log \sigma(1 + o(1)) + O(e)$ bits (we assume that every state is reachable from the initial state, thus $e \geq n - 1$) so that we can solve the SMLG problem in $O(m \log \log \sigma)$ time, that is, we retrieve the time and space bound which were already known for Wheeler automata [26, 15]. If $r = O(1)$, we can still solve pattern matching queries in linear time (for constant alphabets), thus breaking the lower bound by Equi et al., while only storing a compact representation of \mathcal{A} .

Due to space constraints, some proofs and some auxiliary results can be found in the full version [14].

2 Preliminary Definitions

Let Σ be a finite alphabet. We denote by Σ^* the set of all finite strings on Σ . We denote by ϵ the empty string, and $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$ is the set of all nonempty finite strings on Σ . For $i \geq 0$, let $\Sigma^i \subseteq \Sigma^*$ be the set of all strings of length i (we will often interpret Σ^i as a new alphabet). If $\mathcal{L} \subseteq \Sigma^*$, let $\text{Pref}(\mathcal{L})$ be the set of all prefixes of some string in \mathcal{L} . Note that if $\mathcal{L} \neq \emptyset$, then $\epsilon \in \text{Pref}(\mathcal{L})$. We say that $\mathcal{L} \subseteq \Sigma^*$ is *prefix-free* if no string in \mathcal{L} is a strict prefix of another string in \mathcal{L} . Note that if \mathcal{L} is prefix-free and $\epsilon \in \mathcal{L}$, then $\mathcal{L} = \{\epsilon\}$. If $\mathcal{L} \subseteq \Sigma^*$, the *prefix-free kernel* of \mathcal{L} is the set $\mathcal{K}(\mathcal{L})$ of all strings in \mathcal{L} whose strict prefixes are all not in \mathcal{L} . Note that $\mathcal{K}(\mathcal{L})$ is always prefix-free, and \mathcal{L} is prefix-free if and only if $\mathcal{L} = \mathcal{K}(\mathcal{L})$.

Let us recall the definition of generalized deterministic finite automaton (GDFA) [27, 28].

► **Definition 4.** *A generalized non-deterministic finite automaton (GNFA) is a 4-tuple $\mathcal{A} = (Q, E, s, F)$, where Q is a finite set of states, $E \subseteq Q \times Q \times \Sigma^*$ is a finite set of string-labeled edges, $s \in Q$ is the initial state and $F \subseteq Q$ is a set of final states. Moreover, we assume that each $u \in Q$ is reachable from the initial state and is co-reachable, that is, it is either final or allows reaching a final state.*

A generalized deterministic finite automaton (GDFA) is a GNFA such that for every $u \in Q$ no edge leaving u is labeled with ϵ , distinct edges leaving u have distinct labels, and the set of all strings labeling some edge leaving u is prefix-free.

The assumption that every state is reachable and co-reachable is standard in automata theory because all states that do not satisfy this requirement can be removed without changing the recognized language. A conventional NFA (DFA, respectively) is a GNFA

(G DFA, respectively) where all edges are labeled with characters from Σ . Note that we explicitly require a GNFA to have finitely many edges (in conventional NFAs, the finiteness of the number of states automatically implies the finiteness of the number of edges, the alphabet being finite). If we allowed a GNFA to have infinitely many edges, then any nonempty (possibly non-regular) language would be recognized by a GNFA with two states, where the first state is initial, the second state is final, all edges go from the first state to the second state, and a string labels an edge if and only if it is in the language. By requiring a GNFA to have finitely many edges, the class of recognized languages is exactly the class of regular languages, because it is easy to transform a GNFA into a NFA with ϵ -transitions that recognizes the same language by proceeding as follows: for every edge $(u', u, \rho) \in E$, with $\rho = r_1, \dots, r_{|\rho|} \in \Sigma^+$, where $r_1, \dots, r_{|\rho|} \in \Sigma$ and $|\rho| \geq 2$, we delete the edge (u', u, ρ) , we add $|\rho| - 1$ new states $z_1, \dots, z_{|\rho|-1}$ and then we add the edges $(u', z_1, r_1), (z_1, z_2, r_2), \dots, (z_{|\rho|-1}, u, r_{|\rho|})$ (none of the new states is made initial or final).

Let us introduce some notation that will be helpful in the paper.

► **Definition 5.** Let $\mathcal{A} = (Q, E, s, F)$ be a GNFA.

1. For every $\alpha \in \Sigma^*$, let I_α be the set of all states that can be reached from the initial state by following edges whose labels, when concatenated, yield α . In other words, for some $t \geq 0$ there exist edges $(s, u_1, \alpha_1), (u_1, u_2, \alpha_2), (u_2, u_3, \alpha_3), \dots, (u_{t-1}, u_t, \alpha_t)$ such that $\alpha = \alpha_1 \alpha_2 \alpha_3 \dots \alpha_t$.
 2. Let $\mathcal{L}(\mathcal{A})$ be the language recognized by \mathcal{A} , that is, $\mathcal{L}(\mathcal{A}) = \{\alpha \in \Sigma^* \mid I_\alpha \cap F \neq \emptyset\}$.
 3. For every $u \in Q$, let I_u be the set of all strings that can be read from the initial state to u by concatenating edge labels, that is, $I_u = \{\alpha \in \Sigma^* \mid u \in I_\alpha\}$. Note that for every $u \in Q$ we have $\emptyset \subsetneq I_u \subseteq \text{Pref}(\mathcal{L}(\mathcal{A}))$ because every state is reachable and co-reachable.
- When \mathcal{A} is not clear from the context, we write $I_\alpha^{\mathcal{A}}$ and $I_u^{\mathcal{A}}$.

Lastly, following the introduction of the paper, we can naturally define the SMLG problem for GNFA's.

► **Definition 6.** Let \mathcal{A} be a GNFA. The String Matching in Labeled Graphs (SMLG) problem for GNFA's is defined as follows: build a data structure that encodes \mathcal{A} such that, given a string α , we can efficiently compute the set of all states reached by a path suffixed by α .

3 The Myhill-Nerode Theorem for Generalized Automata

The Myhill-Nerode theorem for conventional automata (Theorem 1) provides some algebraic properties that $\text{Pref}(\mathcal{L})$ must satisfy for $\mathcal{L} \subseteq \Sigma^*$ to be a regular language. Intuitively, the reason why $\text{Pref}(\mathcal{L})$ captures the regularity of a regular language (that is, the link between the algebraic characterization and the automata characterization of regular languages) is that, given an NFA $\mathcal{A} = (Q, E, s, F)$ that recognizes \mathcal{L} , we have $\bigcup_{u \in Q} I_u = \text{Pref}(\mathcal{L})$ because if $\alpha \in \text{Pref}(\mathcal{L})$, one can read α on \mathcal{A} starting from the initial state. However, if more generally $\mathcal{A} = (Q, E, s, F)$ is a GNFA that recognizes \mathcal{L} , then we only have $\bigcup_{u \in Q} I_u \subseteq \text{Pref}(\mathcal{L})$, because if $\alpha \in \text{Pref}(\mathcal{L})$, then one can read α on \mathcal{A} starting from the initial state, but it may happen that we have read only a strict prefix of the label of the last edge, if the label is a string of length at least two.

Let us give the following definition.

► **Definition 7.** Let $\mathcal{A} = (Q, E, s, F)$ be a GNFA. Define:

$$\mathcal{W}(\mathcal{A}) = \bigcup_{u \in Q} I_u.$$

We say that \mathcal{A} is a $\mathcal{W}(\mathcal{A})$ -GNFA.

Note that for every $\alpha \in \Sigma^*$ we have $I_\alpha \neq \emptyset$ if and only if $\alpha \in \mathcal{W}(\mathcal{A})$. Moreover, $\mathcal{L}(\mathcal{A}) \cup \{\epsilon\} \subseteq \mathcal{W}(\mathcal{A}) \subseteq \text{Pref}(\mathcal{L}(\mathcal{A}))$, because (i) if $\alpha \in \mathcal{L}(\mathcal{A})$, then $I_\alpha \cap F \neq \emptyset$ and in particular $\alpha \in \mathcal{W}(\mathcal{A})$, (ii) $\epsilon \in I_s$, (iii) $I_u \subseteq \text{Pref}(\mathcal{L}(\mathcal{A}))$ for every $u \in Q$. Let us prove an additional property of $\mathcal{W}(\mathcal{A})$. Pick $\alpha \in \mathcal{W}(\mathcal{A})$, and consider the set $T_\alpha = \{\rho \in \Sigma^+ \mid \alpha\rho \in \mathcal{W}(\mathcal{A})\}$. Consider the prefix-free kernel $\mathcal{K}(T_\alpha)$. If $\rho \in \mathcal{K}(T_\alpha)$, then $I_{\alpha\rho} \neq \emptyset$, but for every $\rho' \in \Sigma^+$ being a strict nonempty prefix of ρ we have $I_{\alpha\rho'} = \emptyset$. This implies that $|\rho| \leq r$, where r is the maximum of the lengths of edge labels. We conclude that $\mathcal{K}(T_\alpha)$ must be finite, because Σ is finite. This leads to the following definition.

► **Definition 8.**

1. Let $\mathcal{W} \subseteq \Sigma^*$. We say that \mathcal{W} is locally bounded if for every $\alpha \in \mathcal{W}$ we have that $\mathcal{K}(T_\alpha)$ is finite, where $T_\alpha = \{\rho \in \Sigma^+ \mid \alpha\rho \in \mathcal{W}\}$.
2. Let $\mathcal{A} = (Q, E, s, F)$ be a GNFA, and let $\mathcal{W} \subseteq \Sigma^*$ be a locally bounded set such that $\mathcal{L}(\mathcal{A}) \cup \{\epsilon\} \subseteq \mathcal{W} \subseteq \text{Pref}(\mathcal{L}(\mathcal{A}))$. We say that \mathcal{A} is a \mathcal{W} -GNFA if $\mathcal{W}(\mathcal{A}) = \mathcal{W}$.

► **Remark 9.** Let $\mathcal{A} = (Q, E, s, F)$ be a GDFA. Let $\alpha \in \mathcal{W}(\mathcal{A})$. If $\alpha = \epsilon$, then $I_\epsilon = \{s\}$ because no edge is labeled with ϵ . If $|\alpha| > 1$, then there exist a prefix $\alpha_1 \in \Sigma^*$ of α and $u_1 \in Q$ such that $(s, u_1, \alpha_1) \in E$, and since \mathcal{A} is a GDFA, we have $\alpha_1 \in \Sigma^+$, and both α_1 and u_1 are unique. In particular, $\alpha_1 \in \mathcal{W}(\mathcal{A})$. If α_1 is a strict prefix of α , we can repeat the argument starting from u_1 . We conclude that for every $\alpha \in \mathcal{W}(\mathcal{A})$ we have $|I_\alpha| = 1$. As a consequence, if $u, v \in Q$ are distinct, then $I_u \cap I_v = \emptyset$. In the following, if \mathcal{A} is a GDFA and $\alpha \in \mathcal{W}(\mathcal{A})$, we will identify I_α and the state being its unique element.

Moreover, our argument shows that, if \mathcal{A} is a GDFA, then for every $\alpha \in \mathcal{W}(\mathcal{A})$ such that $|\alpha| > 0$, the longest strict prefix of α in $\mathcal{W}(\mathcal{A})$ is the unique strict prefix α' of α in $\mathcal{W}(\mathcal{A})$ such that, letting $\rho \in \Sigma^+$ be the string for which $\alpha = \alpha'\rho$, we have $(I_{\alpha'}, I_\alpha, \rho) \in E$. This implies that if \mathcal{A} is a GDFA and $\alpha \in \mathcal{W}(\mathcal{A})$, then $\mathcal{K}(T_\alpha) = \{\rho \in \Sigma^+ \mid \alpha\rho \in \mathcal{W}(\mathcal{A}), (I_\alpha, I_{\alpha\rho}, \rho) \in E\}$.

In the classical Myhill-Nerode we consider equivalence relations defined on $\text{Pref}(\mathcal{L})$. In our setting, we will need to define equivalence relations on subsets of $\text{Pref}(\mathcal{L})$. This leads to the following general definition.

► **Definition 10.** Let $\mathcal{W} \subseteq \Sigma^*$ and let \sim be an equivalence relation on \mathcal{W} . We say that \sim is right-invariant if:

$$(\forall \alpha, \beta \in \mathcal{W})(\forall \phi \in \Sigma^*)(\alpha \sim \beta \rightarrow ((\alpha\phi \in \mathcal{W} \iff \beta\phi \in \mathcal{W}) \wedge (\alpha\phi \in \mathcal{W} \implies \alpha\phi \sim \beta\phi))).$$

► **Remark 11.** Notice that the property defining a right-invariant equivalence relation is trivially true if ϕ is the empty string, so it can be rephrased as follows:

$$(\forall \alpha, \beta \in \mathcal{W})(\forall \phi \in \Sigma^+)(\alpha \sim \beta \rightarrow ((\phi \in T_\alpha \iff \phi \in T_\beta) \wedge (\phi \in T_\alpha \implies \alpha\phi \sim \beta\phi))).$$

Let us prove that \sim is right-invariant if and only if:

$$(\forall \alpha, \beta \in \mathcal{W})(\forall \phi \in \Sigma^+)(\alpha \sim \beta \rightarrow ((\phi \in \mathcal{K}(T_\alpha) \iff \phi \in \mathcal{K}(T_\beta)) \wedge (\phi \in \mathcal{K}(T_\alpha) \implies \alpha\phi \sim \beta\phi))).$$

(\implies) Let $\alpha, \beta \in \mathcal{W}$ such that $\alpha \sim \beta$, and let $\phi \in \mathcal{K}(T_\alpha)$. We must prove that $\phi \in \mathcal{K}(T_\beta)$ and $\alpha\phi \sim \beta\phi$. Since $\phi \in \mathcal{K}(T_\alpha) \subseteq T_\alpha$, we immediately obtain $\phi \in T_\beta$ and $\alpha\phi \sim \beta\phi$, so we only have to prove that $\phi \in \mathcal{K}(T_\beta)$. Since for every $\phi' \in \Sigma^+$ we have $\phi' \in T_\alpha$ if and only if $\phi' \in T_\beta$, then $T_\alpha = T_\beta$, and so $\mathcal{K}(T_\alpha) = \mathcal{K}(T_\beta)$. As a consequence, from $\phi \in \mathcal{K}(T_\alpha)$ we conclude $\phi \in \mathcal{K}(T_\beta)$.

(\impliedby) Let $\alpha, \beta \in \mathcal{W}$ such that $\alpha \sim \beta$, and let $\phi \in T_\alpha$. We must prove that $\phi \in T_\beta$ and $\alpha\phi \sim \beta\phi$. Let ϕ_1, \dots, ϕ_s be all prefixes of ϕ such that $\alpha\phi_i \in \mathcal{W}$ for every $1 \leq i \leq s$, where ϕ_i is a strict prefix of ϕ_{i+1} for every $1 \leq i \leq s-1$. Note that $s \geq 2$, $\phi_1 = \epsilon$ and $\phi_s = \phi$. For

every $1 \leq i \leq s-1$, let $\psi_i \in \Sigma^+$ be such that $\phi_{i+1} = \phi_i \psi_i$. Notice that by definition we have $\psi_i \in \mathcal{K}(T_{\alpha\phi_i})$ for every $1 \leq i \leq s-1$. Since $\alpha, \beta \in \mathcal{W}$, $\alpha \sim \beta$ and $\psi_1 \in \mathcal{K}(T_{\alpha\phi_1}) = \mathcal{K}(T_\alpha)$, we obtain $\psi_1 \in \mathcal{K}(T_\beta)$ and $\alpha\phi_2 = \alpha\psi_1 \sim \beta\psi_1 = \beta\phi_2$. Since $\alpha\phi_2, \beta\phi_2 \in \mathcal{W}$, $\alpha\phi_2 \sim \beta\phi_2$ and $\psi_2 \in \mathcal{K}(T_{\alpha\phi_2})$, we obtain $\psi_2 \in \mathcal{K}(T_{\beta\phi_2})$ and $\alpha\phi_3 = \alpha\phi_2\psi_2 \sim \beta\phi_2\psi_2 = \beta\phi_3$. By continuing like that, we conclude that $\phi \in T_\beta$ and $\alpha\phi = \alpha\phi_s \sim \beta\phi_s = \beta\phi$.

In general, an equivalence relation is not right-invariant. Let us show how to define a canonical right-invariant equivalence relation starting from *any* equivalence relation.

► **Lemma 12.** *Let $\mathcal{W} \subseteq \Sigma^*$ and let \sim be an equivalence relation on \mathcal{W} . For every $\alpha, \beta \in \mathcal{W}$, let:*

$$\alpha \sim_r \beta \iff (\forall \phi \in \Sigma^*)((\alpha\phi \in \mathcal{W} \iff \beta\phi \in \mathcal{W}) \wedge (\alpha\phi \in \mathcal{W} \implies \alpha\phi \sim \beta\phi)).$$

Then \sim_r is an equivalence relation on \mathcal{W} , it is right-invariant and it is the coarsest right-invariant equivalence relation on \mathcal{W} refining \sim . We say that \sim_r is the right-invariant refinement of \sim .

The *Myhill-Nerode equivalence* plays a major role in the classical Myhill-Nerode theorem. Let us show how we can extend it when \mathcal{W} is not necessarily equal to $\text{Pref}(\mathcal{L})$.

► **Definition 13.** *Let $\mathcal{L}, \mathcal{W} \subseteq \Sigma^*$. The Myhill-Nerode equivalence on \mathcal{L} and \mathcal{W} is the equivalence relation $\equiv_{\mathcal{L}, \mathcal{W}}$ on \mathcal{W} defined as the right-invariant refinement of $\sim_{\mathcal{L}, \mathcal{W}}$, where $\sim_{\mathcal{L}, \mathcal{W}}$ is the equivalence relation on \mathcal{W} such that for every $\alpha, \beta \in \mathcal{W}$:*

$$\alpha \sim_{\mathcal{L}, \mathcal{W}} \beta \iff (\alpha \in \mathcal{L} \iff \beta \in \mathcal{L}).$$

If $\mathcal{W} = \text{Pref}(\mathcal{L})$, then we retrieve the classical Myhill-Nerode equivalence relation for \mathcal{L} . Let us describe some elementary properties of $\equiv_{\mathcal{L}, \mathcal{W}}$.

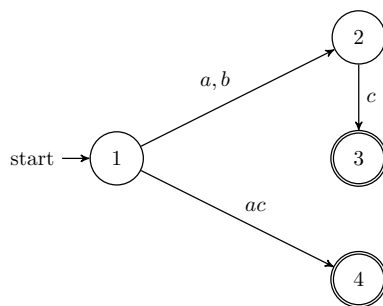
► **Lemma 14.** *Let $\mathcal{L}, \mathcal{W} \subseteq \Sigma^*$. Then $\equiv_{\mathcal{L}, \mathcal{W}}$ is right-invariant and \mathcal{L} is the union of some $\equiv_{\mathcal{L}, \mathcal{W}}$ -classes.*

Proof. First, $\equiv_{\mathcal{L}, \mathcal{W}}$ is right-invariant because it is a right-invariant refinement by definition. Moreover, \mathcal{L} is the union of some $\sim_{\mathcal{L}, \mathcal{W}}$ -classes, and so also of some $\equiv_{\mathcal{L}, \mathcal{W}}$ -classes because $\equiv_{\mathcal{L}, \mathcal{W}}$ refines $\sim_{\mathcal{L}, \mathcal{W}}$. ◀

Let \mathcal{A} be a conventional NFA, and define the equivalence relation $\sim_{\mathcal{A}}$ on $\text{Pref}(\mathcal{L}(\mathcal{A}))$ as follows: for every $\alpha, \beta \in \text{Pref}(\mathcal{L}(\mathcal{A}))$, let $\alpha \sim_{\mathcal{A}} \beta$ if and only if $I_\alpha = I_\beta$. This equivalence relation is an intermediate tool in the Myhill-Nerode theorem for conventional automata, and it can be also defined for a generalized automata \mathcal{A} by considering the equivalence relation $\sim_{\mathcal{A}}$ on $\mathcal{W}(\mathcal{A})$ such that for every $\alpha, \beta \in \mathcal{W}(\mathcal{A})$ we have $\alpha \sim_{\mathcal{A}} \beta$ if and only if $I_\alpha = I_\beta$. If \mathcal{A} is an NFA (or an NFA with ϵ -transitions), then $\sim_{\mathcal{A}}$ is right-invariant, because for every $\alpha \in \text{Pref}(\mathcal{L}(\mathcal{A}))$ and for every prefix α' of α , any path from the initial state to a node in I_α must go through a node in $I_{\alpha'}$. However, in general this property is not true if \mathcal{A} is a GNFA, so $\sim_{\mathcal{A}}$ need not be right-invariant if \mathcal{A} is a GNFA (see Figure 3). Since right-invariance is crucial in the Myhill-Nerode theorem, we will consider the right-invariant refinement of $\sim_{\mathcal{A}}$.

► **Definition 15.** *Let $\mathcal{A} = (Q, E, s, F)$ be a GNFA. Let $\equiv_{\mathcal{A}}$ be the right-invariant refinement of $\sim_{\mathcal{A}}$, where $\sim_{\mathcal{A}}$ is the equivalence relation on $\mathcal{W}(\mathcal{A})$ such that for every $\alpha, \beta \in \mathcal{W}(\mathcal{A})$:*

$$\alpha \sim_{\mathcal{A}} \beta \iff I_\alpha = I_\beta.$$



■ **Figure 3** A GNFA \mathcal{A} such that $\sim_{\mathcal{A}}$ is not right-invariant. Indeed, $a, b, ac, bc \in \mathcal{W}(\mathcal{A})$ and $a \sim_{\mathcal{A}} b$, but $ac \not\sim_{\mathcal{A}} bc$.

► **Remark 16.** Let us prove that if \mathcal{A} is GDFA, then $\sim_{\mathcal{A}}$ is right-invariant. Let $\alpha, \beta \in \mathcal{W}$ be such that $I_{\alpha} = I_{\beta}$, and let $\phi \in \mathcal{K}(T_{\alpha})$. By Remark 11, we only have to prove that $\phi \in \mathcal{K}(T_{\beta})$ and $I_{\alpha\phi} = I_{\beta\phi}$. By Remark 9, we have $\alpha\phi \in \mathcal{W}(\mathcal{A})$ and $(I_{\alpha}, I_{\alpha\phi}, \phi) \in E$. Hence $(I_{\beta}, I_{\alpha\phi}, \phi) \in E$, we obtain $I_{\alpha\phi} = I_{\beta\phi}$ and again by Remark 9, we conclude $\phi \in \mathcal{K}(T_{\beta})$. Notice that, in fact, the generalized automaton \mathcal{A} in Figure 3 is not a GDFA.

Since $\equiv_{\mathcal{A}}$ is the right-invariant refinement of $\sim_{\mathcal{A}}$, we conclude that, if \mathcal{A} is a GDFA, then $\equiv_{\mathcal{A}}$ and $\sim_{\mathcal{A}}$ are the same equivalence relation.

Let us study the properties of $\equiv_{\mathcal{A}}$.

► **Lemma 17.** *Let $\mathcal{A} = (Q, E, s, F)$ be a GNFA. Then, $\equiv_{\mathcal{A}}$ is right-invariant, it refines $\equiv_{\mathcal{L}(\mathcal{A}), \mathcal{W}(\mathcal{A})}$, it has finite index and $\mathcal{L}(\mathcal{A})$ is the union of some $\equiv_{\mathcal{A}}$ -classes.*

The following lemma is crucial to derive our Myhill-Nerode theorem for generalized automata.

► **Lemma 18.** *Let $\mathcal{L} \subseteq \Sigma^*$ and let $\mathcal{W} \subseteq \Sigma^*$ be a locally bounded set such that $\mathcal{L} \cup \{\epsilon\} \subseteq \mathcal{W} \subseteq \text{Pref}(\mathcal{L})$. Assume that \mathcal{L} is the union of some classes of a right-invariant equivalence relation \sim on \mathcal{W} of finite index. Then, \mathcal{L} is recognized by a \mathcal{W} -GDFA $\mathcal{A}_{\sim} = (Q_{\sim}, E_{\sim}, s_{\sim}, F_{\sim})$ such that:*

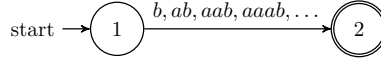
1. $|Q_{\sim}|$ is equal to the index of \sim .
2. $\equiv_{\mathcal{A}_{\sim}}$ and \sim are the same equivalence relation.

Moreover, if \mathcal{B} is a \mathcal{W} -GDFA that recognizes \mathcal{L} , then $\mathcal{A}_{\equiv_{\mathcal{B}}}$ is isomorphic to \mathcal{B} .

Proof (Sketch). The desired \mathcal{W} -GDFA $\mathcal{A}_{\sim} = (Q_{\sim}, E_{\sim}, s_{\sim}, F_{\sim})$ is defined as follows.

- $Q_{\sim} = \{[\alpha]_{\sim} \mid \alpha \in \mathcal{W}\}$.
- $s_{\sim} = [\epsilon]_{\sim}$.
- $E_{\sim} = \{([\alpha]_{\sim}, [\alpha\rho]_{\sim}, \rho) \mid \alpha \in \mathcal{W}, \rho \in \mathcal{K}(T_{\alpha})\}$.
- $F_{\sim} = \{[\alpha]_{\sim} \mid \alpha \in \mathcal{L}\}$. ◀

► **Remark 19.** In the statement of Lemma 18 we cannot remove the assumption that \mathcal{W} is locally bounded, because we have shown that if \mathcal{A} is a GNFA, then $\mathcal{W}(\mathcal{A})$ is locally bounded. However, if \mathcal{W} is not locally bounded, then \mathcal{A}_{\sim} is still a well-defined automaton with finitely many states, but it has infinitely many edges. For example, $\mathcal{W} = \{\epsilon\} \cup a^*b$ is not locally bounded because $T_{\epsilon} = a^*b$ and $\mathcal{K}(T_{\epsilon}) = a^*b$ is an infinite set. If $\mathcal{L} = a^*b$, then $\mathcal{L} \cup \{\epsilon\} \subseteq \mathcal{W} \subseteq \text{Pref}(\mathcal{L})$. Moreover, $\equiv_{\mathcal{L}, \mathcal{W}}$ has finite index (the equivalence classes are $\{\epsilon\}$ and a^*b), and by Lemma 14 we know that $\equiv_{\mathcal{L}, \mathcal{W}}$ is right-invariant and \mathcal{L} is the union of some $\equiv_{\mathcal{L}, \mathcal{W}}$ -classes. We conclude that $\mathcal{A}_{\equiv_{\mathcal{L}, \mathcal{W}}}$ is well-defined, but it has infinitely many edges (see Figure 4).



■ **Figure 4** An example where \mathcal{W} is not locally bounded.

We can now state our Myhill-Nerode theorem for generalized automata.

► **Theorem 20** (Myhill-Nerode theorem for generalized automata). *Let $\mathcal{L} \subseteq \Sigma^*$ and let $\mathcal{W} \subseteq \Sigma^*$ be a locally bounded set such that $\mathcal{L} \cup \{\epsilon\} \subseteq \mathcal{W} \subseteq \text{Pref}(\mathcal{L})$. The following are equivalent:*

1. \mathcal{L} is recognized by a \mathcal{W} -GNFA.
2. The Myhill-Nerode equivalence $\equiv_{\mathcal{L}, \mathcal{W}}$ has finite index.
3. There exists a right-invariant equivalence relation \sim on \mathcal{W} of finite index such that \mathcal{L} is the union of some \sim -classes.
4. \mathcal{L} is recognized by a \mathcal{W} -GDFA.

Moreover, if one of the above statements is true (and so all the above statements are true), then there exists a unique minimal \mathcal{W} -GDFA recognizing \mathcal{L} (that is, two \mathcal{W} -GDFA having the minimum number of states among all \mathcal{W} -GDFA that recognize \mathcal{L} must be isomorphic).

Proof.

- (1) \rightarrow (2) Let \mathcal{A} be a \mathcal{W} -GDFA recognizing \mathcal{L} . By Lemma 17 we have that $\equiv_{\mathcal{A}}$ has finite index and it refines $\equiv_{\mathcal{L}, \mathcal{W}}$, so also $\equiv_{\mathcal{L}, \mathcal{W}}$ has finite index.
- (2) \rightarrow (3) By Lemma 14 the desired equivalence relation is $\equiv_{\mathcal{L}, \mathcal{W}}$.
- (3) \rightarrow (4) It follows from Lemma 18.
- (4) \rightarrow (1) Every \mathcal{W} -GDFA is a \mathcal{W} -GNFA.

Now, let us prove that the minimum automaton is $\mathcal{A}_{\equiv_{\mathcal{L}, \mathcal{W}}}$ as defined in Lemma 18. First, $\mathcal{A}_{\equiv_{\mathcal{L}, \mathcal{W}}}$ is well-defined because $\equiv_{\mathcal{L}, \mathcal{W}}$ is right-invariant and \mathcal{L} is the union of some $\equiv_{\mathcal{L}, \mathcal{W}}$ -classes by Lemma 14, and $\equiv_{\mathcal{L}, \mathcal{W}}$ has finite index by one of the statements that we assume to be true. Now, by Lemma 18 the number of states of $\mathcal{A}_{\equiv_{\mathcal{L}, \mathcal{W}}}$ is equal to the index of $\equiv_{\mathcal{L}, \mathcal{W}}$, or equivalently, of $\equiv_{\mathcal{A}_{\equiv_{\mathcal{L}, \mathcal{W}}}}$. On the other hand, let \mathcal{B} be any \mathcal{W} -GDFA recognizing \mathcal{L} non-isomorphic to $\mathcal{A}_{\equiv_{\mathcal{L}, \mathcal{W}}}$. Then $\equiv_{\mathcal{B}}$ is a refinement of $\equiv_{\mathcal{L}, \mathcal{W}}$ by Lemma 17, and it must be a strict refinement of $\equiv_{\mathcal{L}, \mathcal{W}}$, otherwise $\mathcal{A}_{\equiv_{\mathcal{L}, \mathcal{W}}}$ would be equal to $\mathcal{A}_{\equiv_{\mathcal{B}}}$, which by Lemma 18 is isomorphic to \mathcal{B} , a contradiction. We conclude that the index of $\equiv_{\mathcal{L}, \mathcal{W}}$ is smaller than the index of $\equiv_{\mathcal{B}}$, so again by Lemma 18 the number of states of $\mathcal{A}_{\equiv_{\mathcal{L}, \mathcal{W}}}$ is smaller than the number of states of $\mathcal{A}_{\equiv_{\mathcal{B}}}$ and so of \mathcal{B} . ◀

Myhill-Nerode theorem for conventional automata (Theorem 1) is a special case of Theorem 20, because if \mathcal{A} is an NFA, then $\mathcal{W}(\mathcal{A}) = \text{Pref}(\mathcal{L}(\mathcal{A}))$. Moreover, Theorem 20 is consistent with the example in Figure 2, because, calling \mathcal{A}_1 and \mathcal{A}_2 the two GDFA in Figure 2, we have shown that $\mathcal{W}(\mathcal{A}_1) \neq \mathcal{W}(\mathcal{A}_2)$.

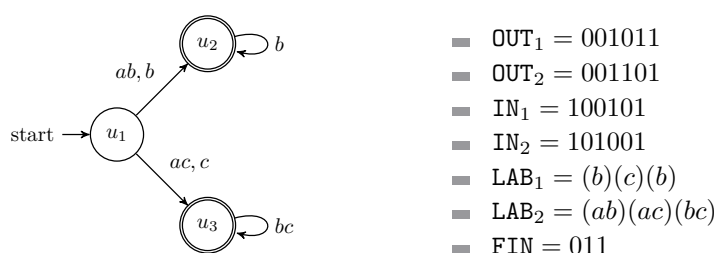
4 The FM-index of Generalized Automata

In this section, we prove Theorem 3. In order to present the main ideas, it will suffice to consider GDFA. The more general case of GNFA without ϵ -transitions requires minor technical modifications and is considered in the extended version [14].

Let V be a set. We say that a (binary) relation \leq on V is a *partial* order if \leq is reflexive, antisymmetric and transitive. A partial order \leq is a *total* order if for every $u, v \in V$ we have $(u \leq v) \vee (v \leq u)$. We say that $U \subseteq V$ is \leq -convex if for every $u, v, z \in V$, if $u \leq v \leq z$ and $u, z \in U$, then $v \in U$. For every $u, v \in V$, we write $u < v$ if $(u \leq v) \wedge (u \neq v)$.

Let us define Wheeler GDFA. As customary in the literature on Wheeler automata [3, 17], we assume that there exists a fixed total order \preceq on Σ (in our examples, we always assume $a \prec b \prec c \prec \dots$), and \preceq is extended *co-lexicographically* to Σ^* (that is, for every $\alpha, \beta \in \Sigma^*$ we have $\alpha \prec \beta$ if the reverse string α^R is lexicographically smaller than the reverse string β^R). Let $\mathcal{A} = (Q, E, s, F)$ be a GDFA. Let $\preceq_{\mathcal{A}}$ be the reflexive relation on Q such that, for every $u, v \in Q$ with $u \neq v$, we have $u \prec_{\mathcal{A}} v$ if and only if $(\forall \alpha \in I_u)(\forall \beta \in I_v)(\alpha \prec \beta)$. Since each I_u is nonempty, it is immediate to realize that $\preceq_{\mathcal{A}}$ is a partial order, but in general it is not a total order. We can then give the following definition (see Figure 5 for an example).

► **Definition 21.** Let $\mathcal{A} = (Q, E, s, F)$ be a GDFA. We say that \mathcal{A} is Wheeler if $\preceq_{\mathcal{A}}$ is a total order.



■ **Figure 5** Left: A Wheeler GDFA \mathcal{A} . States are numbered following the total order $\preceq_{\mathcal{A}}$. Right: The Burrows-Wheeler Transform (BWT) of \mathcal{A} (see Definition 29).

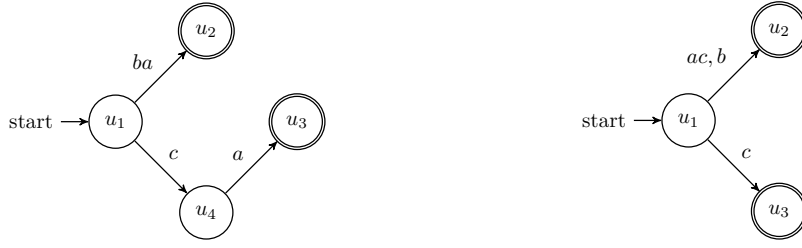
If \mathcal{A} is a conventional DFA, it is not immediately clear that Definition 21 is equivalent to the *local* definition of Wheeler DFA commonly used in the literature [3, 11, 16]. According to the local definition, a DFA $\mathcal{A} = (Q, E, s, F)$ is Wheeler if there exists a total order \leq on Q such that (i) s comes first in the total order, (ii) for every $(u', u, a), (v', v, b) \in E$, if $u < v$, then $a \preceq b$ and (iii) for every $(u', u, a), (v', v, a) \in E$, if $u < v$, then $u' < v'$. Alanko et al. [3, Corollary 3.1] proved that, if such a total order \leq exists, then it is unique and it is equal to $\preceq_{\mathcal{A}}$, so we only have to prove that if $\preceq_{\mathcal{A}}$ is a total order, then it satisfies properties (i), (ii), (iii). This follows from the following lemma.

► **Lemma 22.** Let $\mathcal{A} = (Q, E, s, F)$ be a Wheeler GDFA. Then:

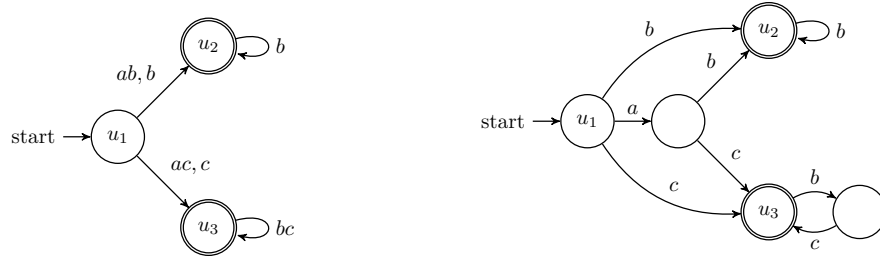
1. s comes first in the total order $\preceq_{\mathcal{A}}$.
2. For every $(u', u, \rho), (v', v, \rho') \in E$, if $u \prec_{\mathcal{A}} v$ and ρ' is not a strict suffix of ρ , then $\rho \preceq \rho'$.
3. For every $(u', u, \rho), (v', v, \rho) \in E$, if $u \prec_{\mathcal{A}} v$, then $u' \prec_{\mathcal{A}} v'$.

In case 2 of Lemma 22 we cannot remove the assumption that ρ' is not a strict prefix of ρ ; as a consequence, we cannot use Lemma 22 to provide an equivalent, local definition of Wheeler GDFA (see Figure 6). The local definition of Wheeler DFA easily implies that the problem of deciding whether a given DFA is Wheeler can be solved in polynomial time [3], but since we do not have a local definition of Wheeler GDFA, it is not clear whether the corresponding problem on GDFA is also solvable in polynomial time (and we saw in the introduction that computational problems on generalized automata are usually hard). However, we can prove that the problem is still tractable by reducing it to computing the *partial* order $\preceq_{\mathcal{A}^*}$ on a conventional DFA \mathcal{A}^* equivalent to a given GDFA \mathcal{A} .

► **Lemma 23.** Let $\mathcal{A} = (Q, E, s, F)$ be a GDFA, and let ϵ be the total length of all edge labels. In $O(\epsilon \log \epsilon)$ time we can decide whether \mathcal{A} is Wheeler and, if so, we can compute $\preceq_{\mathcal{A}}$.



■ **Figure 6** *Left:* A Wheeler G DFA $\mathcal{A} = (Q, E, s, F)$. States are numbered following the total order $\preceq_{\mathcal{A}}$. Note that $(u_1, u_2, ba), (u_4, u_3, a) \in E$, $u_2 \prec_{\mathcal{A}} u_3$, a is a strict suffix of ba and $a \prec ba$. *Right:* A G DFA $\mathcal{A} = (Q, E, s, F)$ such that states are numbered following a total order \leq such that (i) s comes first in the total order \leq , (ii) for every $(u', u, \rho), (v', v, \rho') \in E$, if $u < v$ and ρ' is not a strict suffix of ρ , then $\rho \preceq \rho'$ and (iii) for every $(u', u, \rho), (v', v, \rho) \in E$, if $u < v$, then $u' < v'$. Note that \mathcal{A} is not Wheeler because u_2 and u_3 are not $\preceq_{\mathcal{A}}$ -comparable, since $b \prec c < ac$, $c \in I_{u_3}$ and $b, ac \in I_{u_2}$.



■ **Figure 7** *Left:* The G DFA \mathcal{A} in Figure 5. *Right:* The DFA \mathcal{A}^* built starting from \mathcal{A} in the proof of Lemma 23.

Proof. Let us build a DFA \mathcal{A}^* starting from \mathcal{A} (see Figure 7 for an example). In general, if $C \subseteq \Sigma^+$, we can build a trie such that the set of all strings that can be read following a path starting from the root is equal to the set of all strings prefixing at least one element in C . If C is prefix-free, then a string is in C if and only if it can be read from the root to a leaf; we say that such a leaf *spells* the considered string. For every $u \in Q$, let C_u be the prefix-free set of all strings labeling some edge leaving u . We build an automaton $\mathcal{A}^* = (Q^*, E^*, s^*, F^*)$ by picking \mathcal{A} , removing all edges, and building a trie for every nonempty C_u in such a way that (a) the root of the trie is u , (b) the leaf that spells ρ is the unique $v \in Q$ such that $(u, v, \rho) \in E$ and (c) every internal state of the trie is a new state. Notice that $Q \subseteq Q^*$; we define $s^* = s$ and $F^* = F$. By construction, \mathcal{A}^* is a DFA, and for every $u \in Q$ we have $I_u^{\mathcal{A}} = I_u^{\mathcal{A}^*}$ (so $\mathcal{L}(\mathcal{A}^*) = \mathcal{L}(\mathcal{A})$). As a consequence, \mathcal{A} is Wheeler if and only if the restriction of the partial order $\preceq_{\mathcal{A}^*}$ to Q is a total order. Since \mathcal{A}^* is a conventional DFA, we can compute the partial order $\preceq_{\mathcal{A}^*}$ in polynomial time [17, 33, 13, 8]. For example, the algorithm in [8] runs in $O(|E^*| \log |Q^*|)$ time, and the claimed time bound follows because $|Q^*| - 1 \leq |E^*| = \epsilon$. ◀

The remaining of the section is devoted to proving that the SMLG problem can be solved efficiently on Wheeler G DFAs. If $\mathcal{A} = (Q, E, s, F)$ is a Wheeler G DFA, we write $Q = \{Q[1], Q[2], \dots, Q[n]\}$, where $Q[1] \prec_{\mathcal{A}} Q[2] \prec_{\mathcal{A}} \dots \prec_{\mathcal{A}} Q[n]$; if $1 \leq i \leq j \leq n$, let $Q[i, j] = \{Q[i], Q[i+1], \dots, Q[j-1], Q[j]\}$. If $\alpha, \beta \in \Sigma^*$, we write $\alpha \dashv \beta$ if and only if α is a suffix of β .

► **Definition 24.** Let $\mathcal{A} = (Q, E, s, F)$ be a Wheeler G DFA, and let $\alpha \in \Sigma^*$. Define:

- $G^{\prec}(\alpha) = \{u \in Q \mid (\forall \beta \in I_u)(\beta \prec \alpha)\}$;
- $G_{\dashv}(\alpha) = \{u \in Q \mid (\exists \beta \in I_u)(\alpha \dashv \beta)\}$;
- $G_{\dashv}^{\prec}(\alpha) = G^{\prec}(\alpha) \cup G_{\dashv}(\alpha) = \{u \in Q \mid (\forall \beta \in I_u)(\beta \prec \alpha) \vee (\exists \beta \in I_u)(\alpha \dashv \beta)\}$.

Intuitively, the set $G_{\downarrow}(\alpha)$ is the set of states that the SMLG problem must return on input α , and $G^{\prec}(\alpha)$ is the set of all states reached only by strings smaller than α . The following lemma shows that, as in the case of conventional Wheeler automata, both $G^{\prec}(\alpha)$ and $G_{\downarrow}(\alpha)$ are intervals with respect to the total order $\preceq_{\mathcal{A}}$, and there is no state between $G^{\prec}(\alpha)$ and $G_{\downarrow}(\alpha)$.

► **Lemma 25.** *Let $\mathcal{A} = (Q, E, s, F)$ be a Wheeler GDFA, and let $\alpha \in \Sigma^*$. Then:*

1. $G^{\prec}(\alpha) \cap G_{\downarrow}(\alpha) = \emptyset$.
2. $G_{\downarrow}(\alpha)$ is $\preceq_{\mathcal{A}}$ -convex.
3. If $u, v \in Q$ are such that $u \prec_{\mathcal{A}} v$ and $v \in G^{\prec}(\alpha)$, then $u \in G^{\prec}(\alpha)$. In other words, $G^{\prec}(\alpha) = Q[1, |G^{\prec}(\alpha)|]$.
4. If $u, v \in Q$ are such that $u \prec_{\mathcal{A}} v$ and $v \in G_{\downarrow}^{\prec}(\alpha)$, then $u \in G_{\downarrow}^{\prec}(\alpha)$. In other words, $G_{\downarrow}^{\prec}(\alpha) = Q[1, |G_{\downarrow}^{\prec}(\alpha)|]$.

In order to compute $G_{\downarrow}(\alpha)$, it will suffice to compute $G^{\prec}(\alpha)$ and $G_{\downarrow}^{\prec}(\alpha)$. Let us show how to compute $G^{\prec}(\alpha)$ and $G_{\downarrow}^{\prec}(\alpha)$; to this end, we will constantly use property 3 in Lemma 22, which is also crucial for conventional Wheeler automata (it is a generalization of the LF mapping in the FM-index [25]). First, let $G^*(\alpha)$ be the set of all states reached by an edge labeled with a string suffixed by α . Formally:

$$G^*(\alpha) = \{v \in Q \mid (\exists v' \in Q)(\exists \rho \in \Sigma^*)((v', v, \rho) \in E \wedge (\alpha \dashv \rho))\}.$$

Clearly, $G^*(\alpha) \subseteq G_{\downarrow}(\alpha)$. Now, let us give the following definition.

► **Definition 26.** *Let $\mathcal{A} = (Q, E, s, F)$ be a Wheeler GDFA. Let $U \subseteq Q$ and $\rho \in \Sigma^+$. We denote by $\text{out}(U, \rho)$ the number of edges labeled with ρ that leave states in U , and we denote by $\text{in}(U, \rho)$ the number of edges labeled with ρ that enter states in U .*

If $\alpha \in \Sigma^*$ and $0 \leq k \leq |\alpha|$, let $p(\alpha, k)$ and $s(\alpha, k)$ be the prefix and the suffix of α of length k , respectively. If $\mathcal{A} = (Q, E, s, F)$ is a GDFA and $u \in Q$, let $\lambda(u)$ be the set of all strings in Σ^+ labeling an edge reaching u ; we denote by $\min_{\lambda(u)}$ and $\max_{\lambda(u)}$ the (co-lexicographically) smallest and largest strings in $\lambda(u)$, respectively.

The next lemma formalizes the following intuition: to compute $G^{\prec}(\alpha)$, we have to consider all states whose incoming edges have a label smaller than α ; moreover, if the label is $s(\alpha, k)$ (for some k), then the start state must be in $G^{\prec}(p(\alpha, |\alpha| - k))$.

► **Lemma 27.** *Let $\mathcal{A} = (Q, E, s, F)$ be a Wheeler GDFA, and let $\alpha \in \Sigma^*$. For $0 < k < |\alpha|$, let $f_k = \text{out}(Q[1, |G^{\prec}(p(\alpha, |\alpha| - k))|], s(\alpha, k))$. Then, $|G^{\prec}(\alpha)|$ is the largest integer $0 \leq j \leq |Q|$ such that (i) $\text{in}(Q[1, j], s(\alpha, k)) \leq f_k$, for every $0 < k < |\alpha|$, and (ii) if $j \geq 1$, then $\max_{\lambda(Q[j])} \prec \alpha$.*

The following crucial lemma shows that, in order to compute $|G_{\downarrow}^{\prec}(\alpha)|$, we only have to consider $|G^{\prec}(\alpha)|$, the biggest (w.r.t $\preceq_{\mathcal{A}}$) state in $G^*(\alpha)$ and the states in $G_{\downarrow}^{\prec}(\alpha) \setminus G^*(\alpha)$.

► **Lemma 28.** *Let $\mathcal{A} = (Q, E, s, F)$ be a Wheeler GDFA, and let $\alpha \in \Sigma^*$. For $0 < k < |\alpha|$, let $f_k = \text{out}(Q[1, |G^{\prec}(p(\alpha, |\alpha| - k))|], s(\alpha, k))$ and $g_k = \text{out}(Q[1, |G_{\downarrow}^{\prec}(p(\alpha, |\alpha| - k))|], s(\alpha, k))$. Then, $g_k \geq f_k$ for every $0 < k < |\alpha|$. Moreover, $|G_{\downarrow}^{\prec}(\alpha)|$ is equal to the maximum among:*

1. $|G^{\prec}(\alpha)|$;
2. the largest integer $0 \leq i \leq |Q|$ such that, if $i \geq 1$, then $Q[i] \in G^*(\alpha)$;
3. the smallest integer $0 \leq j \leq |Q|$ such that, for every $0 < k < |\alpha|$ such that $g_k > f_k$, we have $\text{in}(Q[1, j], s(\alpha, k)) \geq g_k$.

We are now ready to generalize the Burrows-Wheeler Transform and the FM-index to Wheeler GDFAs. Let $\mathcal{A} = (Q, E, s, F)$ be a G DFA. We say that \mathcal{A} is a r -G DFA if all edge labels have length at most r . Fix $1 \leq i \leq r$. Let $E(i) = \{(u', u, \rho) \in E \mid \rho \in \Sigma^i\}$ be the number of edges labeled with a string of length i , and let $\Sigma(i) = \{\rho \in \Sigma^i \mid (\exists u', u \in Q)((u', u, \rho) \in E(i))\}$ be the set of all strings of length i labeling some edge. Let $e_i = |E(i)|$ and $\sigma_i = |\Sigma(i)|$; we have $\sigma_i \leq \min\{\sigma^i, e_i\}$. The i -outdegree (i -indegree, respectively) of a state is equal to the number of edges in $E(i)$ leaving (reaching, respectively) the state. The sum of the i -outdegrees of all the states and the sum of the i -indegrees of all the states are both equal to e_i . Lastly, $\sum_{i=1}^r e_i = e$ and the total length of all edge labels is $\epsilon = \sum_{i=1}^r e_i i$.

Let us define the Burrows-Wheeler Transform of a Wheeler G DFA (see Figure 5 for an example).

► **Definition 29** (Burrows-Wheeler Transform of a Wheeler G DFA). *Let $\mathcal{A} = (Q, E, s, F)$ be a Wheeler G DFA. The Burrows-Wheeler Transform $BWT(\mathcal{A})$ of \mathcal{A} consists of the following strings.*

1. For every $1 \leq i \leq r$, the bit string $\text{OUT}_i \in \{0, 1\}^{e_i+n}$ that stores the i -outdegrees in unary. More precisely, (i) OUT_i contains exactly n characters equal to 1, (ii) OUT_i contains exactly e_i characters equal to 0, and (iii) the number of zeros between the $(\ell - 1)$ -th character equal to one (or the beginning of the sequence if $\ell = 1$) and the ℓ -th character equal to 1 yields the i -outdegree of $Q[\ell]$.
2. For every $1 \leq i \leq r$, the bit string $\text{IN}_i \in \{0, 1\}^{e_i+n}$ that stores the i -indegrees in unary. More precisely, (i) IN_i contains exactly n characters equal to 1, (ii) IN_i contains exactly e_i characters equal to 0, and (iii) the number of zeros between the $(\ell - 1)$ -th character equal to one (or the beginning of the sequence if $\ell = 1$) and the ℓ -th character equal to 1 yields the i -indegree of $Q[\ell]$.
3. For every $1 \leq i \leq r$, the string $\text{LAB}_i \in (\Sigma^i)^{e_i}$ that stores the edge labels of length i (with their multiplicities). More precisely, we sort of all edges in $E(i)$ by the index of the start states (w.r.t to $\preceq_{\mathcal{A}}$). Edges with the same start state are sorted by label. Lastly, we obtain LAB_i by concatenating the labels of all the edges following this edge order.
4. The bit string $\text{FIN} \in \{0, 1\}^n$ that marks the final states, that is, the i -th bit of FIN is equal to 1 if and only if $Q[i] \in F$.

We can now prove that the BWT of a Wheeler G DFA \mathcal{A} is a valid encoding of \mathcal{A} , just like the BWT of a string is a valid encoding of the string.

► **Theorem 30.** *Let $\mathcal{A} = (Q, E, s, F)$ be a Wheeler G DFA. If we only know $BWT(\mathcal{A})$, then we can retrieve \mathcal{A} (up to isomorphism). In other words, $BWT(\mathcal{A})$ is an encoding of \mathcal{A} .*

Proof. Consider states $Q[1], \dots, Q[n]$. By Lemma 22 we have $s = Q[1]$, and by using FIN we can retrieve the set of all final states F . We only have to show that we can retrieve E . In other words, for every $1 \leq i', i \leq n$ and for every $\rho \in \Sigma^+$ we must determine whether $(Q[i'], Q[i], \rho) \in E$. It will suffice to retrieve the set $E(i)$ for every $1 \leq i \leq r$, because E is the (disjoint) union of the $E(i)$'s. From LAB_i we can retrieve the labels all edges in $E(i)$, with their multiplicities. From IN_i we can retrieve the i -indegree of each $Q[\ell]$. By Lemma 22, for every $\rho \in \Sigma^i$ labeling some edge reaching some node $Q[\ell]$ and for every $\rho' \in \Sigma^i$ labeling some edge reaching $Q[\ell + 1]$ it must be $\rho \preceq \rho'$. Since we know the labels of all edges in $E(i)$ with multiplicities and we know the i -indegrees, then we can retrieve the labels of all edges entering each $Q[\ell]$, with multiplicities. From OUT_i we can retrieve the i -outdegrees of each $Q[\ell]$, and the order used in the definition of LAB_i implies that we can retrieve the labels of all edges leaving each $Q[\ell]$. Since we know the labels of all edges entering each $Q[\ell]$ and we

know the labels of all edges leaving each $Q[\ell]$, then for every $\rho \in \Sigma^i$ we know the set of all states reached by an edge labeled ρ , with multiplicities, and the set of all states having an outgoing edge labeled ρ . By Lemma 22, for every $1 \leq j < k \leq n$, if $Q[j]$ is reached by an edge labeled ρ leaving the state $Q[j']$ and $Q[k]$ is reached by an edge labeled ρ leaving the state $Q[k']$, then it must be $j' < k'$. As a consequence, we can retrieve the set $E(i)_\rho$ of all edges labeled ρ for every $\rho \in \Sigma^i$, and so we can retrieve $E(i)$, because $E(i)$ is the (disjoint) union of the $E(i)_\rho$'s. ◀

We are ready to prove the main theorem of this section (the full proof is in the extended version [14]). Recall that n is the number of vertices, e is the number of edges, and ϵ is the total length of all edge labels.

► **Theorem 31** (FM-index of Wheeler GDFAs). *Let \mathcal{A} be a Wheeler r -GDFA. Then, we can encode \mathcal{A} by using $\epsilon \log \sigma(1 + o(1)) + O(e + rn)$ bits so that later on, given a pattern $\alpha \in \Sigma^*$ of length m , we can solve the SMLG problem on \mathcal{A} in $O(m(\log r + \log \log \sigma))$ time. Within the same time bound we can also decide whether $\alpha \in \mathcal{L}(\mathcal{A})$.*

Proof (Sketch). By Lemma 25, we only need to compute $|G^{\prec}(\alpha)|$ and $|G_{\prec}^{\prec}(\alpha)|$. In $O(m)$ steps, we recursively compute $|G^{\prec}(p(\alpha, k))|$ and $|G_{\prec}(p(\alpha, k))|$ for every $0 \leq k \leq |\alpha|$. By Lemma 27 and Lemma 28, we can reduce this problem to the problem of solving a number of elementary queries, such as computing $f_k = \text{out}(Q[1, |G^{\prec}(p(\alpha, |\alpha| - k))|], s(\alpha, k))$ for every $0 < k < |\alpha|$. But we only need to compute f_k for $0 < k < r + 1$, because otherwise $|s(\alpha, k)| > r$ and all edge labels have length at most r . In general, we need to solve every elementary query at most $O(r)$ time. By augmenting the Burrows-Wheeler-Transform of \mathcal{A} with compact data structures for rank/select queries and succinct dictionaries, we can solve every elementary query in $O(\log r + \log \log \sigma)$ time. ◀

5 Conclusions and Future Work

In this paper, we considered the model of generalized automata, and we introduced the set $\mathcal{W}(\mathcal{A})$. We showed that $\mathcal{W}(\mathcal{A})$ plays the same role of $\text{Pref}(\mathcal{L}(\mathcal{A}))$ in conventional NFAs: the set $\mathcal{W}(\mathcal{A})$ can be used to derive a Myhill-Nerode theorem, and it represents the starting point for extending the FM-index to generalized automata.

Further lines of research include extending the Burrows-Wheeler Transform and the FM-index to arbitrary GNFA. Indeed, the Burrows-Wheeler Transform and the FM-index were recently generalized from Wheeler NFAs to arbitrary NFAs by means of the so-called *co-lex orders* [15, 17] and *co-lex relations* [12]. However, we remark that the efficient time bounds for the SMLG problem that we derived in this paper cannot hold for arbitrary GNFA due to the (conditional) lower bounds by Equi et al. that we recalled in the introduction.

Giammaresi and Montalbano described an effective procedure for computing a minimal GDFA equivalent to a given GDFA [28, 27], but we do not know if there exists an efficient algorithm for minimizing a GDFA. The Myhill-Nerode theorem for generalized automata implies that for every minimal GDFA \mathcal{A} there exists a set $\mathcal{W} \subseteq \Sigma^*$ such that \mathcal{A} is isomorphic to the minimal \mathcal{W} -GDFA recognizing $\mathcal{L}(\mathcal{A})$. At the same time, given a \mathcal{W} -GDFA recognizing \mathcal{L} , we may build the minimal \mathcal{W} -GDFA recognizing \mathcal{L} by extending Hopcroft's algorithm [30] to GDFAs. If we could prove that, for every admissible $\mathcal{W} \subseteq \Sigma^*$, the number of states of the minimal \mathcal{W} -GDFA recognizing \mathcal{L} is comparable to the number of states of a minimal GDFA recognizing \mathcal{L} , then we would obtain a fast algorithm that significantly reduces the number of states of a GDFA without changing the recognized language.

This paper leaves many questions of theoretical interest open. The class of *Wheeler languages* is the class of all regular languages that are recognized by some Wheeler NFA [4]. Wheeler languages enjoy several properties: for example, they admit a characterization in terms of *convex* equivalence relations [4]. In addition, every Wheeler language is also recognized by some DFA, and, in particular, there exists a unique state-minimal DFA recognizing a given Wheeler language [2]. The main limitation of Wheeler languages is that they represent only a small subclass of regular languages: for example, a unary language (that is, a language over an alphabet of size one) is Wheeler if and only if it is either finite or co-finite [4]. The intuitive reason why most regular languages are not Wheeler is that the definition of a Wheeler NFA \mathcal{A} implies strong requirements on the set $\text{Pref}(\mathcal{L}(\mathcal{A}))$ (which lead to the characterization in terms of convex equivalence relations). However, when we switch to GNFA, it is $\mathcal{W}(\mathcal{A})$ that plays the role of $\text{Pref}(\mathcal{L}(\mathcal{A}))$, and it may hold $\mathcal{W}(\mathcal{A}) \subsetneq \text{Pref}(\mathcal{L}(\mathcal{A}))$, thus now the same requirement only apply to a smaller subset. The natural question is whether Wheeler GNFA extend the class of Wheeler languages. The answer is affirmative: there exists a regular \mathcal{L} language such that \mathcal{L} is not Wheeler (that is, no Wheeler NFA recognizes \mathcal{L}), but \mathcal{L} is recognized by a Wheeler GDFA. Define $\mathcal{L} = \{a^{2^n} \mid n \geq 0\}$. Then, \mathcal{L} is not Wheeler [4], but \mathcal{L} is recognized by the GDFA consisting of a single state, both initial and final, with a self-loop labeled aa . As a consequence, the class of all languages recognized by a Wheeler GNFA is strictly larger than the class of Wheeler languages. We can call the languages in this new class *generalized Wheeler languages*: the next step is to understand which properties of Wheeler languages are still true and how it is possible to characterize this new class.

References

- 1 Tatsuya Akutsu. A linear time pattern matching algorithm between a string and a tree. In Alberto Apostolico, Maxime Crochemore, Zvi Galil, and Udi Manber, editors, *Combinatorial Pattern Matching*, pages 1–10, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.
- 2 Jarno Alanko, Nicola Cotumaccio, and Nicola Prezza. Linear-time minimization of wheeler dfas. In *2022 Data Compression Conference (DCC)*, pages 53–62, 2022. doi:10.1109/DCC52660.2022.00013.
- 3 Jarno Alanko, Giovanna D’Agostino, Alberto Policriti, and Nicola Prezza. *Regular Languages meet Prefix Sorting*, pages 911–930. SIAM, 2020. doi:10.1137/1.9781611975994.55.
- 4 Jarno Alanko, Giovanna D’Agostino, Alberto Policriti, and Nicola Prezza. Wheeler languages. *Information and Computation*, 281:104820, 2021. doi:10.1016/j.ic.2021.104820.
- 5 Amihood Amir, Moshe Lewenstein, and Noa Lewenstein. Pattern matching in hypertext. *Journal of Algorithms*, 35(1):82–99, 2000. doi:10.1006/jagm.1999.1063.
- 6 Jasmijn A. Baaijens, Paola Bonizzoni, Christina Boucher, Gianluca Della Vedova, Yuri Pirola, Raffaella Rizzi, and Jouni Sirén. Computational graph pangenomics: a tutorial on data structures and their applications. *Nat. Comput.*, 21(1):81–108, 2022. doi:10.1007/s11047-022-09882-6.
- 7 Anton Bankevich, Sergey Nurk, Dmitry Antipov, Alexey A. Gurevich, Mikhail Dvorkin, Alexander S. Kulikov, Valery M. Lesin, Sergey I. Nikolenko, Son Pham, Andrey D. Prjibelski, Alexey V. Pyshkin, Alexander V. Sirotkin, Nikolay Vyahhi, Glenn Tesler, Max A. Alekseyev, and Pavel A. Pevzner. SPAdes: A new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology*, 19(5):455–477, 2012. PMID: 22506599. doi:10.1089/cmb.2012.0021.
- 8 Ruben Becker, Manuel Cáceres, Davide Cenzato, Sung-Hwan Kim, Bojana Kodric, Francisco Olivares, and Nicola Prezza. Sorting Finite Automata via Partition Refinement. In Inge Li Gørtz, Martin Farach-Colton, Simon J. Puglisi, and Grzegorz Herman, editors, *31st Annual European Symposium on Algorithms (ESA 2023)*, volume 274 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:15, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ESA.2023.15.

- 9 Alexander Bowe, Taku Onodera, Kunihiko Sadakane, and Tetsuo Shibuya. Succinct de bruijn graphs. In Ben Raphael and Jijun Tang, editors, *Algorithms in Bioinformatics*, pages 225–235, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- 10 M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm, 1994.
- 11 Alessio Conte, Nicola Cotumaccio, Travis Gagie, Giovanni Manzini, Nicola Prezza, and Marinella Sciortino. Computing matching statistics on wheeler dfas. In *2023 Data Compression Conference (DCC)*, pages 150–159, 2023. doi:10.1109/DCC55655.2023.00023.
- 12 Nicola Cotumaccio. Graphs can be succinctly indexed for pattern matching in $o(|e|^2 + |v|^{5/2})$ time. In *2022 Data Compression Conference (DCC)*, pages 272–281, 2022. doi:10.1109/DCC52660.2022.00035.
- 13 Nicola Cotumaccio. Prefix Sorting DFAs: A Recursive Algorithm. In Satoru Iwata and Naonori Kakimura, editors, *34th International Symposium on Algorithms and Computation (ISAAC 2023)*, volume 283 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:15, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ISAAC.2023.22.
- 14 Nicola Cotumaccio. A myhill-nerode theorem for generalized automata, with applications to pattern matching and compression, 2024. arXiv:2302.06506.
- 15 Nicola Cotumaccio, Giovanna D’Agostino, Alberto Policriti, and Nicola Prezza. Co-lexicographically ordering automata and regular languages - part i. *J. ACM*, 70(4), August 2023. doi:10.1145/3607471.
- 16 Nicola Cotumaccio, Travis Gagie, Dominik Köppl, and Nicola Prezza. Space-time trade-offs for the lcp array of wheeler dfas. In Franco Maria Nardini, Nadia Pisanti, and Rossano Venturini, editors, *String Processing and Information Retrieval*, pages 143–156, Cham, 2023. Springer Nature Switzerland.
- 17 Nicola Cotumaccio and Nicola Prezza. On indexing and compressing finite automata. In *Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA ’21*, pages 2585–2599, USA, 2021. Society for Industrial and Applied Mathematics.
- 18 Samuel Eilenberg. *Automata, Languages, and Machines*. Academic Press, Inc., USA, 1974.
- 19 Massimo Equi, Roberto Grossi, Veli Mäkinen, and Alexandru I. Tomescu. On the complexity of string matching for graphs. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 55:1–55:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.55.
- 20 Massimo Equi, Veli Mäkinen, and Alexandru I. Tomescu. Graphs cannot be indexed in polynomial time for sub-quadratic time string matching, unless seth fails. In Tomáš Bureš, Riccardo Dondi, Johann Gamper, Giovanna Guerrini, Tomasz Jurdziński, Claus Pahl, Florian Sikora, and Prudence W.H. Wong, editors, *SOFSEM 2021: Theory and Practice of Computer Science*, pages 608–622, Cham, 2021. Springer International Publishing.
- 21 Massimo Equi, Veli Mäkinen, Alexandru I. Tomescu, and Roberto Grossi. On the complexity of string matching for graphs. *ACM Trans. Algorithms*, 19(3), April 2023. doi:10.1145/3588334.
- 22 Massimo Equi, Tuukka Norri, Jarno Alanko, Bastien Cazaux, Alexandru I. Tomescu, and Veli Mäkinen. Algorithms and complexity on indexing elastic founder graphs. In Hee-Kap Ahn and Kunihiko Sadakane, editors, *32nd International Symposium on Algorithms and Computation, ISAAC 2021, December 6-8, 2021, Fukuoka, Japan*, volume 212 of *LIPIcs*, pages 20:1–20:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ISAAC.2021.20.
- 23 Massimo Equi, Tuukka Norri, Jarno Alanko, Bastien Cazaux, Alexandru I. Tomescu, and Veli Mäkinen. Algorithms and complexity on indexing founder graphs. *Algorithmica*, 85(6):1586–1623, 2023. doi:10.1007/s00453-022-01007-w.

- 24 P. Ferragina and G. Manzini. Opportunistic data structures with applications. In *Proc. 41st Annual Symposium on Foundations of Computer Science (FOCS'00)*, pages 390–398, 2000. doi:10.1109/SFCS.2000.892127.
- 25 Paolo Ferragina and Giovanni Manzini. Indexing compressed text. *J. ACM*, 52(4):552–581, July 2005. doi:10.1145/1082036.1082039.
- 26 Travis Gagie, Giovanni Manzini, and Jouni Sirén. Wheeler graphs: A framework for bwt-based data structures. *Theoretical Computer Science*, 698:67–78, 2017. Algorithms, Strings and Theoretical Approaches in the Big Data Era (In Honor of the 60th Birthday of Professor Raffaele Giancarlo). doi:10.1016/j.tcs.2017.06.016.
- 27 Dora Giammarresi and Rosa Montalbano. Deterministic generalized automata. In Ernst W. Mayr and Claude Puech, editors, *STACS 95, 12th Annual Symposium on Theoretical Aspects of Computer Science, Munich, Germany, March 2-4, 1995, Proceedings*, volume 900 of *Lecture Notes in Computer Science*, pages 325–336. Springer, 1995. doi:10.1007/3-540-59042-0_84.
- 28 Dora Giammarresi and Rosa Montalbano. Deterministic generalized automata. *Theor. Comput. Sci.*, 215(1-2):191–208, 1999. doi:10.1016/S0304-3975(97)00166-7.
- 29 Kosaburo Hashiguchi. Algorithms for determining the smallest number of nonterminals (states) sufficient for generating (accepting) a regular language. In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez Artalejo, editors, *Automata, Languages and Programming*, pages 641–648, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.
- 30 John Hopcroft. An $n \log n$ algorithm for minimizing states in a finite automaton. In Zvi Kohavi and Azaria Paz, editors, *Theory of Machines and Computations*, pages 189–196. Academic Press, 1971. doi:10.1016/B978-0-12-417750-5.50022-1.
- 31 John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., USA, 2006.
- 32 Ramana M. Idury and Michael S. Waterman. A new algorithm for DNA sequence assembly. *Journal of computational biology : a journal of computational molecular cell biology*, 2 2:291–306, 1995.
- 33 Sung-Hwan Kim, Francisco Olivares, and Nicola Prezza. Faster prefix-sorting algorithms for deterministic finite automata. In Laurent Bulteau and Zsuzsanna Lipták, editors, *34th Annual Symposium on Combinatorial Pattern Matching, CPM 2023, June 26-28, 2023, Marne-la-Vallée, France*, volume 259 of *LIPICs*, pages 16:1–16:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.CPM.2023.16.
- 34 Donald E. Knuth, James H. Morris, Jr., and Vaughan R. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6(2):323–350, 1977. doi:10.1137/0206024.
- 35 Udi Manber and Sun Wu. Approximate string matching with arbitrary costs for text and hypertext. In *Advances In Structural And Syntactic Pattern Recognition*, pages 22–33. World Scientific, 1992.
- 36 Veli Mäkinen, Djamal Belazzougui, Fabio Cunial, and Alexandru I. Tomescu. *Genome-Scale Algorithm Design: Bioinformatics in the Era of High-Throughput Sequencing*. Cambridge University Press, 2 edition, 2023.
- 37 Gonzalo Navarro. Improved approximate pattern matching on hypertext. *Theor. Comput. Sci.*, 237(1–2):455–463, April 2000. doi:10.1016/S0304-3975(99)00333-3.
- 38 Gonzalo Navarro. *Compact Data Structures: A Practical Approach*. Cambridge University Press, 2016. doi:10.1017/CB09781316588284.
- 39 Kunsoo Park and Dong Kyue Kim. String matching in hypertext. In Zvi Galil and Esko Ukkonen, editors, *Combinatorial Pattern Matching*, pages 318–329, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.
- 40 Pavel A. Pevzner, Haixu Tang, and Michael S. Waterman. An Eulerian path approach to DNA fragment assembly. *Proceedings of the National Academy of Sciences*, 98(17):9748–9753, 2001. doi:10.1073/pnas.171285098.

- 41 Mikko Rautiainen and Tobias Marschall. Aligning sequences to general graphs in $o(v + me)$ time. *bioRxiv*, 2017. doi:10.1101/216127.
- 42 Nicola Rizzo and Veli Mäkinen. Linear time construction of indexable elastic founder graphs. In Cristina Bazgan and Henning Fernau, editors, *Combinatorial Algorithms*, pages 480–493, Cham, 2022. Springer International Publishing.
- 43 Jared T. Simpson and Richard Durbin. Efficient construction of an assembly string graph using the fm-index. *Bioinform.*, 26(12):367–373, 2010. doi:10.1093/bioinformatics/btq217.