The Algorithmics Column

by

Ioana O. Bercea and Thomas Erlebach

KTH, Stockholm, Sweden and Durham University, UK
bercea@kth.se and thomas.erlebach@durham.ac.uk

Data Compression Meets Automata Theory

Nicola Cotumaccio
University of Helsinki, Finland
nicola.cotumaccio@helsinki.fi

Abstract

I received my PhD in Computer Science on January 31, 2024, under a Joint PhD agreement between Gran Sasso Science Institute (L'Aquila, Italy) and Dalhousie University (Halifax, Canada). I was supervised by Travis Gagie, Nicola Prezza and Catia Trubiani. My PhD thesis, *Data Compression Meets Automata Theory*, was selected by the Italian Chapter of the EATCS for the Best PhD Thesis Award.

The thesis introduces a new paradigm for studying regular languages, establishing a connection between classical results in automata theory, such as the powerset construction, and the most important data structures for solving pattern matching queries on compressed strings, such as the Burrows-Wheeler transform. The results and the open problems should be of interest to both the algorithmic community and the formal language theory community.

In 2020, Alanko et al. introduced *Wheeler automata* [3]. Intuitively, a nondeterministic finite automaton (NFA) is Wheeler if there exists a total order on the set of all nodes that is consistent with the co-lexicographic order on the strings reaching each node (see Figure 1). We say that a regular language is Wheeler if it is recognized by some Wheeler NFA. Wheeler automata were initially motivated by classical results on compressed data structures [18] but, as a matter of fact, the class of Wheeler languages is a surprisingly rich and stable subclass of regular languages. Basic results in automata theory show that, if a language is recognized by a nondeterministic finite automaton (NFA), then the language is also recognized by a deterministic finite automaton (DFA), and up to isomorphism there exists a unique (state)-minimal DFA recognizing the language. Moreover, regular languages admit an algebraic characterization in terms of equivalence relations, and the minimal automaton can be described through the Myhill-Nerode equivalence. Analogous results hold for Wheeler languages: determinism and non-determinism have the same expressive power, there exists a unique minimal Wheeler DFA, and there exists a characterization in terms of *convex* equivalence relations [4].

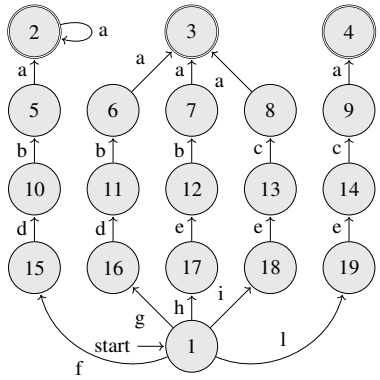

Figure 1: A Wheeler automaton. States are numbered according to their positions in the corresponding total order. Node 2 is reached by the string $fdba$, node 3 is reached by the string $gdba$, we have $2 < 3$ and consistently the string $fdba$ is co-lexicographically smaller than $gdba$.

Most automata are not Wheeler: every Wheeler language must be star-free. In [13], we show how to extend the idea behind Wheeler automata to *arbitrary* automata and *arbitrary* regular languages. The key idea is that we should not use a total order, but a *partial order* (see Figure 2): in this way, it is possible to capture every automaton. We can measure how far a partial order is from being a total order by the notion of *width*: a partial order has width p is it can be decomposed into p total orders, but not into $p-1$ total orders. Dilworth's theorem [14] shows that the width of a partial order is equal to the size of a largest set of pairwise incomparable elements. Wheeler automata correspond to the case $p = 1$.

The parameter p is a complexity parameter with multiple interpretations and applications.

- First, p is a nondeterminism parameter. The powerset construction [23] is a classical construction that converts an NFA into an equivalent DFA (see Figure 3). If the original NFA has n states, in the worst case the equivalent DFA can have up to $2^n - 1$ states. This exponential blow-up is unavoidable: there exist regular languages for which, if N is the number of states of a state-minimal NFA recognizing the language and D is the number of states of the minimal DFA for the language, then $D = 2^N - 1$ [20]. In the paper, we show that, in fact, the powerset construction is exponential only in p: if we start from an NFA with n states, then the equivalent DFA has at most $2^p(n-p+1)-1$ states. In the worst case, we have $p = n$ and we retrieve the bound $2^n - 1$ but, for example, a Wheeler NFA ($p = 1$) with n states can be converted into an equivalent DFA with at most $2n - 1$ states. This implies that several classical algorithmic problems that are computationally hard on

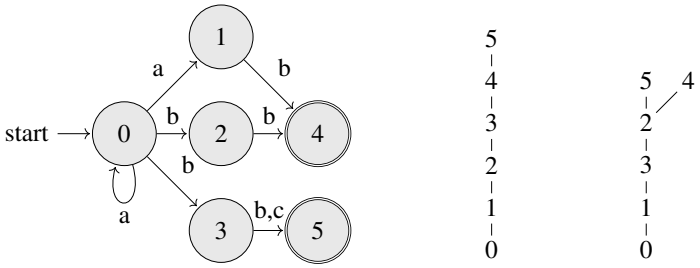

Figure 2: An automaton with the Hasse diagrams of two partial orders (on the set of all states), both respecting the co-lexicographic order on the strings reaching each state. Note that the first partial order is, in fact, a total order.

NFAs but easy on DFA are fixed-parameter tractable with respect to p. For example, the problem of deciding whether two NFAs recognize the same languages (*equivalence problem*) is PSPACE-complete [25], but the same problem can be solved efficiently if the input automata are DFAs. This implies that the equivalence problem is fixed-parameter tractable with respect to p, because one can convert the input NFAs into equivalent DFAs by the powerset construction and then test the resulting DFAs for equivalence.

- Second, p is a compression parameter. An NFA with n states and e edges on an alphabet of size σ can be stored by using only $e(2\log p + \log\sigma)(1 + o(1)) + O(e)$ bits. This is surprising because at the very least we need $e\log\sigma$ bits to store the edge labels, so with a small overhead we can also store the topology of the automaton. This result extends the Burrows-Wheeler transform [6] from strings to arbitrary automata.

- Third, p is an algorithmic parameter. By only querying our compressed representation, we can solve pattern matching on an NFA (including deciding whether a string in accepted by the NFA) in $O(mp^2 \log\log(p\sigma))$ time, where m is the length of the pattern. Our algorithm matches and parameterizes well-known conditional lower bounds on the problem [15]. This result extends the FM-index [17] from strings to arbitrary automata.

Since we can capture all automata, we can also capture all regular languages. In particular, we can define the deterministic width of a regular language as the minimum width of some DFA that recognizes the language. In our journal extension [8], we show that the width of a regular language is related to the notion of *entanglement*: intuitively, some states of an automaton (and in particular of the minimal automaton) are entangled if the strings reaching these states are inherently incomparable in *every* DFA recognizing the language. Based on the notion of entanglement, we define a new canonical automaton for each regular language,

the *Hasse automaton* of the language, and we show that the problem of computing the deterministic width of a language is decidable. The Hasse automaton captures the propensity of a regular language to be sorted.

The journal extension [8] contains the main ideas of the thesis. A short initial section presents the new paradigm; then, the readers can skip the automata theory results or the data compression results, based on their interests and preferences.

Computing p is NP-hard. In [10], I show that the hardness can be overcome by building a quotient automaton obtained by collapsing some states in the original automaton. The quotient automaton and the original automaton recognize the same language, and some correspondence theorems ensure that solving pattern matching queries on the original automaton is equivalent to solving pattern matching queries on the quotient automaton. This paper received the Best Student Paper Award at the 2022 Data Compression Conference (DCC).

In the deterministic case, the problem of computing p can be interpreted as a highly non-trivial extension of the problem of computing the suffix array of a string, a well-studied problems in string processing (see [21] for a survey). In the original paper [13] we gave an $O(m^2)$ algorithm, where m is the number of edges. In [11], I improve this running time to $O(m + n^2)$, where m is the number of edges and n is the number of states, by proposing a recursive algorithm based on induced sorting (a powerful and elegant algorithmic technique for sorting the suffixes of a string). This paper received the Best Student Paper Award at the 2023 International Symposium on Algorithms and Computation (ISAAC). Interestingly, it is an open problem to determine whether it is possible to achieve $O(m)$ time. There exists an alternative algorithm based on Paige and Tarjan's partition refinement algorithm [22], and the $O(m + n^2)$ algorithm suggests that induced sorting may outperfom approaches based on partition refinement. This could lead to unexpected consequences, because the most efficient algorithm for DFA minimization is still Hopcroft's algorithm [19], which is a partition refinement algorithm.

The same paradigm can be extended to more general computation models. In his monumental work on automata theory [16], Eilenberg proposed a natural generalization of NFAs where edges can be labeled not only with characters but with finite strings, the so-called *generalized automata*. String-labeled graphs appear in some classical data structures (such as suffix trees) and in the emerging field of pangenomics. In [12], I show that our new paradigm can be extended to generalized automata, and I describe a full Myhill-Nerode theorem, the first structural result for the class of generalized automata, which includes a sound notion of minimal DFA for generalized automata.

As mentioned earlier, Wheeler languages always admit a minimal Wheeler automata. A Wheeler DFA can be minimized in $O(n \log n)$ time by adapting Hopcroft's algorithm for DFA minimization. In [2], we show that, in the Wheeler case, we can do better: it is possible to achieve linear time minimization by ex-

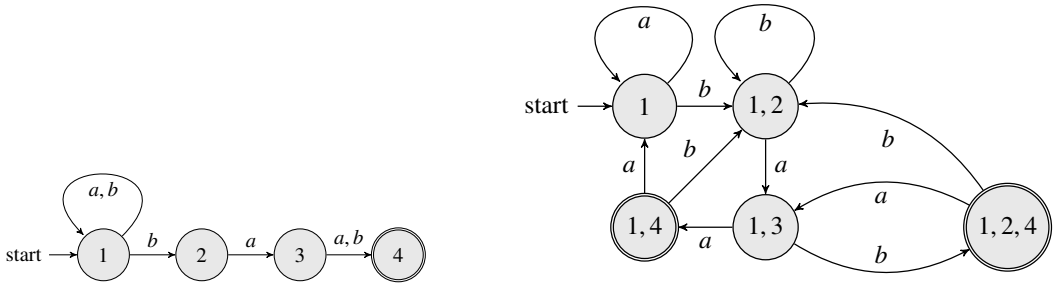

Figure 3: An NFA (left) and the equivalent DFA obtained by the powerset construction (right). Each state of the DFA corresponds to a nonempty set of states of the original NFA.

ploiting the co-lexicographic structure of a Wheeler DFA.

If we want to solve more complicated pattern matching queries on graphs and automata (for example, approximate pattern matching queries) we need a more powerful data structure. In the string setting, the most versatile data structure is the suffix tree [26], so we should extend suffix tree functionality to automata. In a chain of papers [7, 9, 1], we provide some partial results in this direction by showing how to extend the longest common prefix array to Wheeler DFAs. The longest common prefix array is a key data structure for simulating a traversal of a suffix tree [5].

In the last chapter of my thesis, I discuss some additional open problems. We outlined how Wheeler languages enjoy several properties, but two characterizations are missing: one in terms of regular expressions, and one through logic. Wheeler languages are star-free, so they capture a fragment of first-order logic: to obtain a logical characterization it may be necessary to go through difficult results such as the Schützenberger theorem for aperiodic monoids [24]. At the same time, the thesis introduces a very natural hierarchy of regular languages parameterized by p, so by exploring these characterizations and extending them to each level of the hierarchy we may shed new light on famous important open problems, such as the generalized star-height problem.

References

[1] Jarno N. Alanko, Davide Cenzato, Nicola Cotumaccio, Sung-Hwan Kim, Giovanni Manzini, and Nicola Prezza. Computing the LCP Array of a Labeled Graph. In Shunsuke Inenaga and Simon J. Puglisi, editors, *35th Annual Symposium on Combinatorial Pattern Matching (CPM 2024)*, volume 296 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:15,

Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[2] Jarno Alanko, Nicola Cotumaccio, and Nicola Prezza. Linear-time minimization of Wheeler DFAs. In *2022 Data Compression Conference (DCC)*, pages 53–62, 2022.

[3] Jarno Alanko, Giovanna D'Agostino, Alberto Policriti, and Nicola Prezza. Regular languages meet prefix sorting. In *Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '20, page 911–930, USA, 2020. Society for Industrial and Applied Mathematics.

[4] Jarno Alanko, Giovanna D'Agostino, Alberto Policriti, and Nicola Prezza. Wheeler languages. *Information and Computation*, 281:104820, 2021.

[5] Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. Replacing suffix trees with enhanced suffix arrays. *Journal of Discrete Algorithms*, 2(1):53–86, 2004. The 9th International Symposium on String Processing and Information Retrieval.

[6] Michael Burrows and David J Wheeler. A block-sorting lossless data compression algorithm. Technical Report 124, Digital Equipment Corporation, 1994.

[7] 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.

[8] Nicola Cotumaccio, Giovanna D'Agostino, Alberto Policriti, and Nicola Prezza. Co-lexicographically ordering automata and regular languages-part i. *Journal of the ACM*, 70(4):1–73, 2023.

[9] 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.

[10] 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.

[11] 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.

[12] Nicola Cotumaccio. A Myhill-Nerode Theorem for Generalized Automata, with Applications to Pattern Matching and Compression. In Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, and Daniel Lokshtanov, editors, *41st International Symposium on Theoretical Aspects of Computer Science (STACS 2024)*, volume 289 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:19, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[13] Nicola Cotumaccio and Nicola Prezza. On indexing and compressing finite automata. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2585–2599. SIAM, 2021.

[14] RP Dilworth. A decomposition theorem for partially ordered sets. *The Annals of Mathematics*, 51(1):161, 1950.

[15] 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)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 55:1–55:15, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[16] Samuel Eilenberg. *Automata, Languages, and Machines*. Academic Press, Inc., USA, 1974.

[17] Paolo Ferragina and Giovanni Manzini. Indexing compressed text. *J. ACM*, 52(4):552–581, July 2005.

[18] 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).

[19] John Hopcroft. An n log n algorithm for minimizing states in a finite automaton. In *Theory of machines and computations*, pages 189–196. Elsevier, 1971.

[20] F.R. Moore. On the bounds for state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way finite automata. *IEEE Transactions on Computers*, C-20(10):1211–1214, 1971.

[21] Simon J. Puglisi, W. F. Smyth, and Andrew H. Turpin. A taxonomy of suffix array construction algorithms. *ACM Comput. Surv.*, 39(2):4–es, July 2007.

[22] Robert Paige and Robert E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, 1987.

[23] M. O. Rabin and D. Scott. Finite automata and their decision problems. *IBM Journal of Research and Development*, 3(2):114–125, 1959.

[24] M.P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8(2):190–194, 1965.

[25] L. J. Stockmeyer and A. R. Meyer. Word problems requiring exponential time(preliminary report). In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, STOC '73, page 1–9, New York, NY, USA, 1973. Association for Computing Machinery.

[26] Peter Weiner. Linear pattern matching algorithms. In *14th Annual Symposium on Switching and Automata Theory (swat 1973)*, pages 1–11. IEEE, 1973.