Linear-Time Limited Automata
Extended Abstract*

Bruno Guillon and Luca Prigioniero

Dipartimento di Informatica, Università degli Studi di Milano, Italy
{guillonb, prigioniero}@di.unimi.it

Abstract. The time complexity of 1-limited automata is investigated from a descriptional complexity view point. Though the model recognizes regular languages only, it may use quadratic time in the input length. We show that, with a polynomial increase in size and preserving determinism, each 1-limited automaton can be transformed into an halting linear-time equivalent one. We also obtain polynomial transformations into related models, including weight-reducing Hennie machines, and we show exponential gaps for converse transformations in the deterministic case.

1 Introduction

One classical topic of computer science is the investigation of computational models operating under restrictions. Finite automata or pushdown automata, for instance, can be considered as particular Turing machines in which the access to memory storage is limited. Other kinds of restrictions follow from finer analysis of the computational resources an abstract device requires to recognize certain languages. For example, in the case of Turing machines, classical complexity classes such as P, NP, LogSpace, *etc.* are defined by introducing a limit on the amount of resources, namely time or space, at disposal of the model.

Usually, such limitations reduce the expressive power. For instance, it is well-known that one-tape nondeterministic Turing machines operating within a space bounded by the length of the input, namely *linear bounded automata*, capture exactly the class of *context-sensitive languages*, *e.g.* [5]. Phenomena like this, where limiting an abstract model reduces its expressiveness to the level of some standard class, are of great interest, as they provide alternative characterizations of standard classes. Another example of this kind has been observed by Hennie in 1965. He indeed proved that deterministic one-tape Turing machines operating in *linear time* (*i.e.*, time $O(m)$ over inputs of length m), which can be converted into linear bounded automata operating in linear time, recognize exactly the class of *regular languages* [3]. The result has then been extended to the nondeterministic case [14], see also [8] for further improvements.[1] As a consequence, each *Hennie machine*, namely nondeterministic linear bounded automata working in linear time, is equivalent to some finite automaton. From the opposite

* This work is an extended abstract of the conference paper [2].
[1] In nondeterministic linear-time devices each accepting computation has linear length.

point of view, this means that providing *two-way finite automata* (2NFAs) with the ability to overwrite the tape cells, does not extend the expressiveness of the model, as long as the time is linearly bounded in the length of the input.

Unfortunately, Hennie proved that it is undecidable, given a deterministic one-tape Turing machine, to check whether it works in linear time over all input strings, namely, whether it is actually a Hennie machine. To avoid this drawback, Průša proposed a variant of Hennie machine, called *weight-reducing Hennie machine*, in which the time limitation is syntactic [12]. In this model, each visit of a cell should overwrite its content with a symbol in a decreasing way, with respect to some fixed order on the working alphabet. As a consequence, the number of visits of a cell by the head is bounded by some constant (*i.e.*, not depending on the input length) whence the device works in linear time over every input string.

By contrast to Hennie machines, the *d-limited automata* (d-LA) introduced by Hibbard, restrict nondeterministic linear bounded automata by allowing overwriting of each tape cell during its first d visits only, for some fixed $d \geq 0$ [4]. Contrary to weight-reducing Hennie machines, the head is still allowed to visit a cell after the d-th visit, but it cannot rewrite its content anymore. This allows to use super-linear time, as shown in the following example.

Example 1. We consider the language

$$L_n = \{x_0 x_1 \cdots x_k \mid k \in \mathbb{N},\ x_i \in \{a,b\}^n,\ \#\{i > 0 \mid x_i = x_0\} \text{ is odd}\}.$$

A deterministic 1-LA $\mathcal{A}_n$ may recognize L_n as follows. It first overwrites the factor x_0, replacing each input symbol with a marked copy. Then, $\mathcal{A}_n$ repeats a subroutine which overwrites a factor x_i with some fixed symbol $\sharp$, while checking in the meantime whether x_i equals x_0 or not. This can be achieved as follows. Before overwriting the j-th symbol of x_i, first, $\mathcal{A}_n$, with the help of a counter modulo n, moves the head leftward to the position j of x_0 and stores the unmarked scanned symbol σ in its finite control; second, it moves the head rightward until reaching the position j of x_i, namely, the leftmost position that has not been overwritten so far. At this point, $\mathcal{A}_n$ compares the scanned symbol (*i.e.*, the j-th symbol of x_i) with σ (*i.e.*, the j-th symbol of x_0). By counting modulo 2 the number of factors equal to x_0, and finally checking that the input string has length multiple of n, $\mathcal{A}_n$ can decide the membership of the input to L_n.

It is possible to implement $\mathcal{A}_n$ with a number of states linear in n and $\#\Sigma+1$ working symbols. Since for each position of a factor x_i, $i > 0$, the head has to move back to the factor x_0, we observe that $\mathcal{A}_n$ works in quadratic time in the length of the input string.

For each $d \geq 2$, Hibbard proved that d-LA recognize exactly the class of *context-free languages*. He furthermore showed the existence of an infinite hierarchy of deterministic d-LA, whose first level (*i.e.*, corresponding to deterministic 2-LA) has been later proved to coincide with the class of *deterministic context-free languages* [10]. (See [7] and references therein for further connections between limited automata and context-free languages.)

Clearly, 0-limited automata are no more than two-way finite automata. Hence, they characterize the class of regular languages. Wagner and Wechsung extended

this result to the case $d = 1$: 1-LA recognize exactly the class of regular languages [15]. From that point, the question of the cost of their simulation by classical finite automata has been studied by Pighizzini and Pisoni in [9], where a tight doubly-exponential simulation by deterministic one-way finite automata is proved. This cost reduces to a single exponential when starting from a deterministic 1-LA. Also, an exponential lower bound, using a single-letter input alphabet, has been obtained in [11], for the simulation of deterministic 1-LA by 2NFA.

Like d-limited automata, 1-limited automata can operate in super-linear time (*cf.*, Example 1). This contrasts with Hennie machines which operate in linear time by definition. The question we address in this paper is whether this ability of 1-limited automata with respect to Hennie machines yields a gap between the two models in terms of the size of their representations.

2 Results

We show that, with a polynomial increase in size, each 1-limited automaton can be transformed into an halting linear-time 1-limited automaton. This is achieved by augmenting the exponential cost simulation of 1-LA by 2NFA given in [9] (which in turn augments Shepherdson's classic conversion of 2DFAs to 1DFAs [13]) with a method for storing and accessing a carefully chosen subcollection of the many "Shepherdson tables" that a 1NFA would need to remember in its states: the simulating automaton can both store the tables (despite the 1-limitation) and access them efficiently (in both size and time).

Theorem 1. *For each* 1-LA $\mathcal{A}$*, there exists an equivalent* 1-LA $\mathcal{A}'$ *satisfying:*

1. $\mathcal{A}'$ *has polynomial size with respect to* $\mathcal{A}$*;*
2. *in every computation of* $\mathcal{A}'$*, each tape cell is visited a number of times which is bounded by some polynomial in the size of* $\mathcal{A}$*;*
3. $\mathcal{A}'$ *works in linear time: on every input string* w*, it halts within* $O(|w|)$ *steps;*
4. *if* $\mathcal{A}$ *is deterministic, then so is* $\mathcal{A}'$*.*

Linear-time 1-LAs are particular cases of Hennie machines, hence, it follows from the above result that any 1-LA can be transformed into a Hennie machine of size polynomial in the size of the 1-LA. Using Item 2 we can actually strengthen the result: each 1-LA can be transformed into a *weight-reducing Hennie machine* of polynomial size.

Corollary 1. *For each* 1-LA $\mathcal{A}$*, there exists an equivalent weight-reducing Hennie machine* $\mathcal{A}'$ *of size polynomial in the size of* $\mathcal{A}$*. Furthermore, if* $\mathcal{A}$ *is deterministic, then so is* $\mathcal{A}'$*.*

We also observe that the 1-limited automaton resulting from the construction of Theorem 1 can be easily transformed into an equivalent one whose behavior can be divided into two phases: (1) an initial phase consisting in a left-to-right

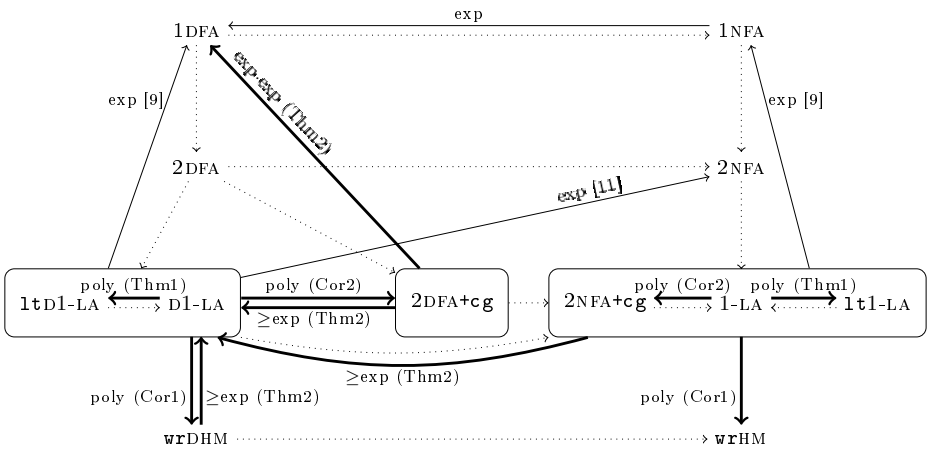

Fig. 1. Relationships between the main models studied in the paper. Here, `lt` and `wr` mean linear-time and weight-reducing, while D1-LA and (D)HM stand for deterministic 1-LA and (deterministic) Hennie machine, respectively. Dotted arrows indicate trivial connections while thick arrows indicate our results.

one-way traversal of the input, during which each input symbol is nondeterministically overwritten; (2) a second phase consisting in a read-only two-way computation. Similar behaviors have been considered in the context of *regular transductions* (*i.e.*, transductions computed by, for instance, two-way transducers), because of their correspondence with global existential quantification in *monadic second order logic*, see, *e.g.* [1]. Using terminology from [1], we define the model of *two-way automaton with common guess* (2NFA+cg) in order to capture these particular behaviors of 1-limited automata. Formally, such machines are not 1-limited automata, but the composition of an initial *common guess* (*i.e.*, a nondeterministic marking of the input symbols using symbols from a finite alphabet, computed, for instance, by a 1-state *letter-to-letter nondeterministic one-way transducer*) with a two-way automaton working on the enriched alphabet. Notice that a deterministic two-way automaton with common guess (2DFA+cg), is not a deterministic device, since it initially performs a common guess which is nondeterministic by definition. We also point out that 2DFA+cgs correspond to *synchronous two-way deterministic finite verifiers* [6].

Corollary 2. *For each* 1-LA *(resp., deterministic* 1-LA*), there exists an equivalent halting* 2NFA+cg *(resp.,* 2DFA+cg*) of polynomial size.*

A direct consequence of this last result, is that *reversing* a 1-limited automaton, *i.e.*, transforming it into another one recognizing the reverse of its accepted language, has polynomial cost. This fails in the deterministic case, for which we exhibit an exponential lower bound. As a consequence, we obtain exponential lower bounds for the simulation of deterministic weight-reducing Hennie

machines or deterministic two-way automata with common guess by deterministic 1-limited automata.

Theorem 2. *Let L_n be the language of Example 1. Hence*

$$L_n{}^{\mathrm{R}} = \{x_k x_{k-1} \cdots x_0 \mid k > 0,\ x_i \in \{a,b\}^n,\ \#\{i > 0 \mid x_i = x_0\} \text{ is odd}\}.$$

Then,

1. *$L_n{}^{\mathrm{R}}$ is accepted by a 2DFA+cg, a linear-time nondeterministic* 1-LA*, or a deterministic weight-reducing Hennie machine of size polynomial in n;*
2. *any* 1DFA *recognizing $L_n{}^{\mathrm{R}}$ requires 2^{2^n} states;*
3. *any deterministic* 1-LA *recognizing $L_n{}^{\mathrm{R}}$ requires $O(2^n)$ states.*

The results are summarized in Figure 1.

Acknowledgement. We are very indebted to Giovanni Pighizzini for suggesting the problem and for many stimulating conversations.

References

1. Bojańczyk, M., Daviaud, L., Guillon, B., Penelle, V.: Which classes of origin graphs are generated by transducers. In: ICALP 2017. LIPIcs, vol. 80, pp. 114:1–13 (2017)
2. Guillon, B., Prigioniero, L.: Linear-time limited automata. In: DCFS 2018. LNCS, vol. 10952. Springer (2018), to appear.
3. Hennie, F.C.: One-tape, off-line Turing machine computations. Information and Control **8**(6), 553–578 (1965)
4. Hibbard, T.N.: A generalization of context-free determinism. Information and Control **11**(1/2), 196–238 (1967)
5. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages and Computation. Addison-Wesley (1979)
6. Kapoutsis, C.A.: Predicate Characterizations in the Polynomial-Size Hierarchy. In: Conference on Computability in Europe. pp. 234–244. Springer (2014)
7. Kutrib, M., Pighizzini, G., Wendlandt, M.: Descriptional complexity of limited automata. Inf. Comput. **259**(2), 259–276 (2018)
8. Pighizzini, G.: Nondeterministic one-tape off-line Turing machines. Journal of Automata, Languages and Combinatorics **14**(1), 107–124 (2009)
9. Pighizzini, G., Pisoni, A.: Limited automata and regular languages. International Journal of Foundations of Computer Science **25**(07), 897–916 (2014)
10. Pighizzini, G., Pisoni, A.: Limited automata and context-free languages. Fundamenta Informaticae **136**(1-2), 157–176 (2015)
11. Pighizzini, G., Prigioniero, L.: Limited automata and unary languages. In: DLT 2017. LNCS, vol. 10396, pp. 308–319 (2017)
12. Průša, D.: Weight-reducing Hennie machines and their descriptional complexity. In: LATA 2014. LNCS, vol. 8370, pp. 553–564 (2014)
13. Shepherdson, J.C.: The reduction of two-way automata to one-way automata. IBM J. Res. Dev. **3**(2), 198–200 (1959)
14. Tadaki, K., Yamakami, T., Lin, J.C.H.: Theory of one-tape linear-time Turing machines. Theor. Comput. Sci. **411**(1), 22–43 (2010)
15. Wagner, K.W., Wechsung, G.: Computational Complexity. D. Reidel Publishing Company, Dordrecht (1986)