INFORMATIQUE THÉORIQUE ET APPLICATIONS

G. BUNTROCK
F. DREWES
C. LAUTEMANN
T. MOSSAKOWSKI

Some modifications of auxiliary pushdown automata

Informatique théorique et applications, tome 25, n° 6 (1991), p. 545-556

<http://www.numdam.org/item?id=ITA_1991__25_6_545_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 25, n° 6, 1991, p. 545 à 556)

SOME MODIFICATIONS OF AUXILIARY PUSHDOWN AUTOMATA (*)

by G. BUNTROCK (1), F. DREWES (2),
C. LAUTEMANN (3) and T. MOSSAKOWSKI (4)

Communicated by J. E. PIN

Abstract. – *We define a number of complexity classes by modified pushdown automata. These modifications are obtained through several restrictions on the way in which such a machine can access its input, or make use of its pushdown store or of nondeterminism. Comparison of these complexity classes provides new insight into some open problems in complexity theory, in particular the LBA problem.*

Résumé. – *A l'aide des machines à pile auxiliaires modifiées, on définit quelques classes de complexité. Ces modifications sont obtenues par certaines restrictions concernant la méthode par laquelle la machine peut se servir de l'entrée, ou par laquelle elle utilise sa pile ou le non-déterminisme. Une comparaison de ces classes de complexité éclaire des problèmes ouverts de la théorie de complexité, en particulier le problème LBA.*

1. INTRODUCTION

Many central problems in complexity theory concern the effect of adding certain capabilities to resource bounded computing devices. Much research has focussed on nondeterminism, in particular on the most famous problems of this type, namely the question whether P = NP [5], and the LBA problem (*i.e.*, the question whether LIN = NLIN, *cf.* Kuroda [11]). But there are also many other important problems of this type, for instance, log-space bounded Turing machines with an additional unbounded pushdown store are known to recognise exactly P [5], but whether L = P is an open problem. With an additional polynomial time bound these machines recognise exactly those sets

(*) Received May 1989, revised May 1991.
(1) Universität, Würzburg, Berlin, Germany.
(2) Universität, Bremen, Germany.
(3) Joh.-Gutenberg-Universität, Mainz, Germany.
(4) Universität, Bremen, Germany.

Informatique théorique et Applications/Theoretical Informatics and Applications
0988-3754/91/06 545 12/$3.20/

which are log-space reducible to contex-free languages [14], but, again, whether L = LOGCFL is an open problem.

In general, these problems seem very difficult; for more restricted models, however, such as one-way machines, separation results seem easier obtainable, and indeed a number of such results have been proved.

In this paper, we combine two approaches to problems of this type. On one hand, we define intermediate machine models which can use their extra power only in a restricted way. On the other hand, we study the effect of such limited additional power on one-way machines, and relate the results obtained here to the two-way case.

More precisely, extending an idea of Hartmanis and Mahaney [8], we define machine models with restricted availability of nondeterminism and/or of an additional pushdown store. In a uniform way, these definitions yield several hierarchies of complexity classes, relating the new ones to the well-studied classes L, NL, LIN, NLIN, LOGDCFL, and LOGCFL. Separation results are proved between the one-way and the two-way versions of these classes, and among the one-way classes. As our main result, we then show that many open inclusions among these classes are equivalent to each other, thereby obtaining several new formulations of the LBA problem. These equivalent formulations shed new light on this long-standing open problem [HH74] and provide a more precise understanding of its difficulty.

2. DEFINITIONS AND BASIC FACTS

Our basic computational model is the deterministic off-line Turing machine (TM), as defined in, e. g., [9], with the slight difference that we consider the work tapes of an $s(n)$-space bounded machine to have, on input x, $s(|x|)$ tape cells marked off at the beginning of the computation. This commonly made convention enables a one-way machine to use its full work tape even before it has read the complete input string, *cf.*, e. g., [7]. We enhance Turing machines with two additional capabilities: nondeterminism, and auxiliary pushdown store.

Nondeterministic Turing machines (NTMs) are defined in the usual way (*cf.* Hopcroft, Ullman [9]). A deterministic (nondeterministic) *auxiliary pushdown automaton*, DauxPDA (NauxPDA), is a Turing machine with an additional, unbounded work tape which can only be accessed as a pushdown store, *i. e.*, the head on this tape can move to the right only when reading a

blank (that is after popping the leftmost symbol). For a more detailed description of auxPDA see, again, Hopcroft and Ullman [9].

2.1. DEFINITION: Let $s, t : \mathbb{N} \to \mathbb{N}$ be any functions.

DTM$[s, t]$ is the class of all those sets which can be recognised by an s-space and t-time bounded DTM.

Similarly, we define the classes NTM$[s, t]$, DauxPDA$[s, t]$ and NauxPDA$[s, t]$.

The machine models defined above have either no or full access to their additional capabilities nondeterminism or auxiliary pushdown store, respectively. Investigations of unary languages complete for NL made by Hartmanis and Mahaney [8] led naturally to a machine model which could use nondeterminism only after having scanned the input. Exploiting this idea further, we now consider Turing machines which are not allowed to use their additional resources while scanning the input, but only either before or after that. (This idea is due to Inga Niepel, [13]. Using *restricted* models in the sense of Hartmanis and Mahaney [8] would be too unspecific in our context.)

2.2. DEFINITION: Let C be one of NTM$[s, t]$, DauxPDA$[s, t]$, Naux PDA$[s, t]$. Then EC ("End-C") is the class of all languages which can be recognised by a Turing machine of type C which uses its additional power (*i.e.* nondeterminism and/or the pushdown) only after the last move of the input head. Similarly, BC ("Begin-C") consists of all those languages which are recognised by a C-type Turing machine which uses its additional power only before moving its input head.

In the usual way, we extend these definitions to situations where s and t are classes of functions, e. g., ENauxPDA$[O(n), 2^{O(n)}]$, etc.

2.3. DEFINITION: For any of the classes C defined above, the class OW-C is defined like C with the additional restriction that the recognising machine is one-way (that is, it is allowed to scan the input only once).

We make use of the following common abbreviations.

L = DTM[log, −],

NL = NTM[log, −],

LIN = DTM[lin, −],

NLIN = NTM[lin, −],

and similarly for modifications by the prefixes "E", "B", and "OW".

Here and in the following, log denotes $O(\log n)$, poly denotes $n^{O(1)}$, lin denotes $O(n)$ and a dash indicates that no bound is employed.

Note that classes defined with at least linear work space are not changed by the "E" and "OW" modifications because a linear space bounded machine can copy the input onto one of its work tapes. This does also hold for the "B" case, because we can simply guess the input before starting the computation.

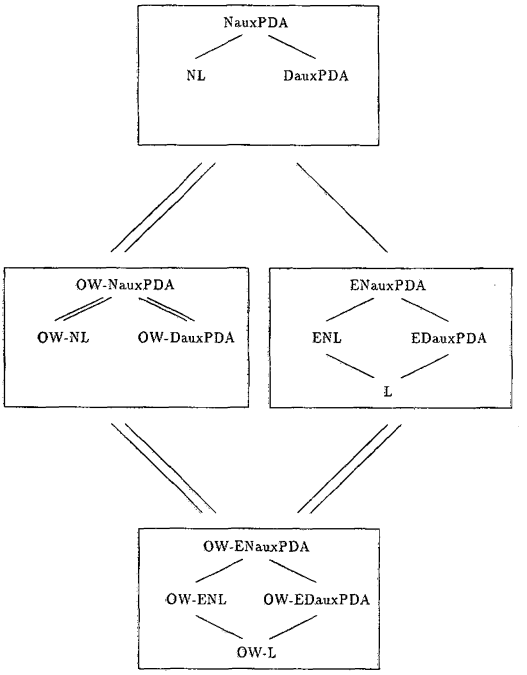

Figure 1. – Inclusions among several classes. Every edge indicates inclusion of the lower class in the diagram in the upper class. Edges between diamonds denote inclusion of every class in the diamond below the edge in the corresponding class above the edge. Strict inclusions are indicated by double edges. AuxPDA [log, poly] **is abbreviated by** AuxPDA.

Figure 1 shows the main inclusions among the log-space and polynomial time bounded complexity classes defined above. They are all immediate from the definitions. The diagram also shows strict inclusions; their strictness will be proved in Section 3.

For the sake of clarity, we have left out the "B"-classes in the figure. There is an interesting relation between the "B"- and the "E"-classes through logarithmically bounded projections.

2.4. DEFINITION: If C is any complexity class, then

$$\exists_{\log} \mathrm{C} = \{ L \subseteq \Sigma^* \mid L = \{ x \mid \exists y \, |y| \leq \log |x| \text{ and } yx \in L' \} \text{ for some } L' \in \mathrm{C} \}.$$

The "B"- and the "E"-classes are related by the $\exists_{\log}$-quantifier as follows.

2.5. THEOREM:

1. $\exists_{\log}$ OW-ENL = OW-BNL.
2. $\exists_{\log}$ OW-ENauxPDA [log, poly] = OW-BNauxPDA [log, poly].
3. $\exists_{\log}$ ENL = BNL.
4. $\exists_{\log}$ ENauxPDA [log, poly] = BNauxPDA [log, poly].

Proof: As all proofs have the same shape, we only prove the first equality.

Suppose $L \in \exists_{\log}$ OW-ENL. Then there is a OW-ENTM M which recognises L' according to Definition 2.4. L can be recognised by a OW-BNTM as follows. First, the configuration that M enters when having processed the input is guessed. Then it is verified that this configuration leads to acceptance. Finally, a string s of length at most log n is guessed, and the start of M's computation on s, followed by the original input is simulated until the end of the input is reached. It is checked if this simulation stops with the guessed configuration.

Conversely, suppose $L \in$ OW-BNL via M. Then L is in $\exists_{\log}$ OW-ENL via a OW-ENTM which interprets the first log n symbols of the input as a description of the configuration M is in when leaving the nondeterministic phase. The rest of the input is then processed normally, and finally it is checked that the configuration given by the first part of the input is indeed reachable. □

2.6. COROLLARY:

1. OW-ENL $\subseteq$ OW-BNL.
2. OW-ENauxPDA [log, poly] $\subseteq$ OW-BNauxPDA [log, poly].
3. ENL $\subseteq$ BNL.
4. ENauxPDA [log, poly] $\subseteq$ BNauxPDA [log, poly].

Proof: By Theorem 2.5, because $\exists_{\log}$ C contains C. □

Unfortunately, we do not know if a similar relation does hold between BDauxPDA [log, poly] and EDauxPDA [log, poly].

3. SEPARATION OF VARIOUS COMPLEXITY CLASSES

The computational power especially of the one-way machine models introduced in Section 2 is very limited. Due to this fact, we manage to demonstrate that some of the inclusions between the complexity classes shown in figure 1 are proper, namely those marked by double edges.

Most of these separations follow from the following theorem, which states that even very simple languages cannot be recognised by one-way machines.

3.1. THEOREM: *The language* $\{xx \mid x \in \Sigma^*\}$ *cannot be recognised by a* OW-NauxPDA *in* $o(n)$ *space.*

Proof: See [3] or [16] for a proof using formal language-theoretic arguments. Another, more direct proof can be found in [12]. □

3.2. COROLLARY: *In figure* 1, *any inclusion between a one-way and a two-way class is proper. (These are exactly the inclusions denoted by the upper left and lower right double edges in the figure.)*

Proof: Note that the largest one-way class and the smallest two-way class we consider are OW-NauxPDA [log, poly] and L respectively. By Theorem 3.1 above, $EQ := \{xx \mid x \in \Sigma^*\}$ is outside the class OW-NauxPDA [log, poly], but clearly it is in L. So the result follows. □

For the remaining separations, we need some more preliminaries:

3.3. LEMMA: OW-ENL *is closed under complement.*

Proof: Since, after entering the nondeterministic phase, the computation has no longer access to the input, one can use the method of Szelepcsényi [15] and Immerman [10] for recognising the complement. □

In contrast to OW-ENL, OW-NL is *not* closed under complement, since it does not contain EQ, but the complement of EQ is easily recognised by a one-way NTM M on logarithmic space: First M guesses the input length, n. If n is odd, M verifies that n is indeed the input length and accepts. If n is even, M guesses some $i \leqq n/2$, verifies both that the input string is of length n and that its ith and $(n/2)+i$th position differ, and accepts.

3.4. LEMMA: *The language* $\{xx^R \mid x \in \Sigma^*\}$ *(the palindromes) can neither be recognised in* $o(n)$ *space by a* OW-ENauxPDA *nor is it in* OW-NL.

Proof: Similar to Theorem 3.1, but much simpler. Let $s \in o(n)$ and M be any $s(n)$-space bounded OW-ENauxPDA or OW-NTM. Fix n such that

$|\Sigma|^{2n} > c^{s(2n)}$, where $c^{s(2n)}$ is the number of configurations of M when processing inputs of length $2n$ (not counting the pushdown).

Now, if M accepts all palindromes, then for each x of length n, there must be a computation for M which accepts xx^R. Consider the map which assigns to each such x a configuration of the corresponding computation, namely the configuration after having read the first half of the input (that is, x). Because $|\Sigma|^{2n} > c^{s(2n)}$ (the cardinality of the domain is greater than that of the range), this map cannot be injective, so there are different strings x, y of length n which are mapped to the same configuration. Hence, M accepts both xy^R and yx^R and therefore does not recognise $\{xx^R \mid x \in \Sigma^*\}$.

(Note that we did not have to take account of the pushdown because it can be used only after having read the entire input.) □

3.5. COROLLARY:

1. OW-DauxPDA [log, poly] $\subsetneq$ OW-NauxPDA [log, poly].
2. OW-ENL $\subsetneq$ OW-NL.
3. OW-NL $\subsetneq$ OW-NauxPDA [log, poly].
4. OW-EDauxPDA [log, poly] $\subsetneq$ OW-DauxPDA [log, poly].
5. OW-ENauxPDA [log, poly] $\subsetneq$ OW-NauxPDA [log, poly].

Proof: First two inclusions: The class OW-DauxPDA [log, poly] is clearly closed under complement, OW-ENL is so by Lemma 3.3. Since both do not contain the language EQ, they do not contain the complement of EQ either. But a OW-NauxPDA or a OW-NTM can recognise the complement of EQ.

Last three inclusions: Clearly, $\{xx^R \mid x \in \Sigma^*\}$ can be recognised by a OW-D(N)auxPDA, but neither by a OW-EN(D)auxPDA or by a OW-NTM (on log space), as Lemma 3.4 shows. □

4. EQUIVALENCES BETWEEN OPEN PROBLEMS

In the previous section, we have shown some of the inclusions in figure 1 to be proper, while properness of the other ones remains an open problem. In this section, we relate these problems to other open questions concerning the power of nondeterminism and additional pushdowns, especially to the LBA problem.

A well-known technique for relating machine models with different resource bounds is padding. For example, a language is padded exponentially as follows:

4.1. DEFINITION: Let $L \subseteq \Sigma^*$ be any language. Then ExpPad(L) is defined as $\{x \#^{2^{|x|}} \mid x \in L\}$.

With padding one can prove e.g. the following:

4.2. PROPOSITION: L = NL *implies* LIN = NLIN.

Proof: Suppose L = NL. Let L ∈ NLIN be recognised by a linear space bounded NTM M. The language ExpPad(L) can be recognised by a log-space bounded NTM M', which – after testing if the input has the correct form – just simulates M on the non-redundant part of the input. By assumption, ExpPad(L) ∈ L via a log-space bounded DTM M'', so that L is recognised by a linear space bounded DTM which supplies the redundant part of the input while simulating M''. □

The converse of Proposition 4.2 cannot be proved as easily. NL = L seems to be a stronger proposition than LIN = NLIN. In fact, if we examine the proof of Proposition 4.2, we notice that actually not the full power of a log-space bounded NTM is used when simulating the linear space bounded NTM. So we can weaken the hypothesis of the theorem above, which leads to the following.

4.3. THEOREM: OW-ENL ⊆ L *implies* LIN = NLIN.

Proof: Almost exactly as for Proposition 4.2. The only step that has to be checked is the simulation of the linear bounded NTM in order to recognise ExpPad(L). For this, a OW-ENTM suffices. First, it copies the non-redundant part of the input on its work tape and then skips the redundant part. Here we need the slight modification of the standard Turing machine model made in Section 2: Before the machine starts its computation, the available space is marked on the work tape. So the machine can reject when exceeding the space bound.

After having read the input, the OW-ENTM can start its simulation, because now it has full access to its resources (here: to nondeterminism). □

This result is stronger than Proposition 4.2, because the hypothesis was weakened. Is it now weak enough so that the converse also holds? Indeed, this is the case, as is shown below.

4.4. THEOREM: *The five diamonds in figure* 2 *are equivalent in the following sense: For any two diamonds, each inclusion in the first one is an equality if and only if the corresponding inclusion in the other diamond is an equality.*

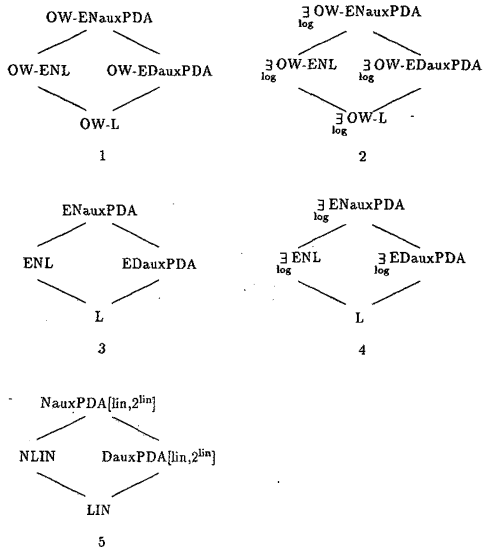

Figure 2. – Equivalences among open inclusions. Again, AuxPDA[log, poly] is abbreviated by AuxPDA. Two of the diamonds (No. 1 and 3) do already appear in figure 1, one (No. 5) is the same thing pushed onto the level of linear space, and the two remaining ones (Nos. 2 and 4) result from applying the $\exists_{\log}$-operation to Diamond 1 and to Diamond 3, respectively. (The linear space classes are closed under $\exists_{\log}$.)

Proof: We prove this with two chains of implications: "1 implies 2 implies 5 implies 1" and "3 implies 4 implies 5 implies 3". ("a implies b" is to be understood as: "Every equality of an inclusion in diamond a implies that the corresponding inclusion in diamond b is an equality".)

1. "1 implies 2": Trivial.

2. "2 implies 5": Similar to theorem 4.3. We prove as an example that $\exists_{\log}$ OW-L = $\exists_{\log}$ OW-EDauxPDA[log, poly] implies LIN = DauxPDA[lin, 2^{lin}].

If $L \in$ DauxPDA[lin, 2^{lin}], then ExpPad$(L) \in \exists_{\log}$ OW-EDauxPDA[log, poly] by a simulation on the non-redundant part of the input, which takes place when the entire input has been read and so the pushdown is available. Thus, using the hypothesis of the implication, ExpPad$(L) \in \exists_{\log}$ OW-L. Therefore, $L \in$ LIN, because the logarithmically bounded quantifier (which in the simulation becomes a linear bounded one) can be resolved by a deterministic search.

3. "5 implies 1": Again, we take an example, namely LIN = NLIN implies OW-L = OW-ENL.

Let $L \in$ OW-ENL via M. Consider the language conf(M) consisting of all configurations of M which have their input pointer pointing to the end of input-symbol and which lead to acceptance (through at least one computation). conf(M) is easily seen to be in NLIN, because relative to its configuration size, each space-bounded TM may use linear space. Now by assumption, conf(M) $\in$ LIN via M'. So there is a log-space bounded OW-DTM which recognises L by simulating M until the end of the input is reached and then uses M' to decide if the current configuration leads to acceptance.

4. "3 implies 4": Trivial, as "1 implies 2".

5. "4 implies 5": Similar to "2 implies 5".

6. "5 implies 3": Similar to "5 implies 1". □

Because of the importance of the LBA-Problem, we state the following equivalences separately:

4.5. COROLLARY: *The following propositions are equivalent*

- LIN = NLIN.
- OW-L = OW-ENL.
- $\exists_{\log}$ OW-L = OW-BNL.
- L = ENL.
- L = BNL.

Proof: By Theorem 2.5, $\exists_{\log}$ OW-ENL = OW-BNL and $\exists_{\log}$ ENL = BNL. Furthermore, $\exists_{\log}$ L = L: Simply simulate the quantifier by a deterministic search. Modulo these equalities, the equivalences are proven in the theorem. □

Note that by further examining the simulations used in the theorem, one can prove the following inclusions being equivalent to the above equalities, too: OW-ENL $\subseteq$ L and OW-BNL $\subseteq$ L.

5. DISCUSSION

In order to study the effect of adding both nondeterminism and an unlimited pushdown store to space bounded Turing machines, we have defined intermediate models in which pushdown store and/or nondeterminism are avilable only in a restricted way. We believe, and this view is confirmed by our results, that the computational power of machines, which can make use of their additional capabilities only when having no access to input, lies

strictly between that of deterministic Turing machines and that of machines with unrestricted additional capabilities. There are, however, several ways of defining such models. Originally, we used the model of *restricted auxiliary pushdown automaton* (resPDA), as defined by Buntrock [2]. A resPDA is an auxiliary pushdown automaton without input tape. Instead, the input is given in its pushdown store. This model is closely related to OW-auxPDA with restricted use of pushdown store, and in fact, we can show results about resPDA very similar to the results presented here. However, with the definitions given in this paper, restrictions in access to input and to pushdown store are clearly separated, and consequently, we find a more uniform spectrum of complexity classes and somewhat stronger results.

In conclusion, we feel that approaches such as this, namely qualitative restrictions in the use of additional capabilities will deepen our understanding of fundamental problems in complexity theory, and will, hopefully, eventually lead to separation results.

ACKNOWLEDGEMENTS

We thank Birgit Jenner and Bernd Kirsig for pointing out to us and discussing with us the notions of "begin" and "end" nondeterminism.

REFERENCES

1. F.-J. Brandenburg, On One-Way Auxiliary Pushdown Automata, *Proc. 3rd GI Conf., Lecture Notes in Comput. Sci.*, 1977, *48*, pp. 132-144.
2. G. Buntrock, On the Robustness of the Polynomial Time Hierarchy, *Technical report*, No. 87-11, Technische Universität Berlin, 1987.
3. M. P. Chytil, Analysis of the Non-Context-Free Component of Formal Languages, *Lecture Notes in Comput. Sci.*, 1976, *45*, pp. 230-236.
4. S. A. Cook, The Complexity of Theorem Proving Procedures. *3rd STOC*, 1971, pp. 151-158.
5. S. A. Cook, Characterizations of Pushdown Machines in Terms of Time-Bounded Computers, *J. Assoc. Comput. Mach.*, 1971, *18*, pp. 4-18.
6. J. Hartmanis and H. B. Hunt III, The LBA Problem and Its Importance in the Theory of Computing, *S.I.A.M.-A.M.S. Proc.*, 1974, *7*, pp. 1-26.
7. J. Hartmanis, N. Immerman and S. Mahaney, One-Way Log-Tape Reductions, *19th F.O.C.S.*, 1978, pp. 65-71.
8. J. Hartmanis and S. Mahaney, Languages Simultaneously Complete for One-Way and Two-Way Log-Tape Automata, *S.I.A.M.J. Comput.*, 1981, *10*, pp. 383-390.
9. J. E. Hopcroft and D. Ullman, Introduction to Automata Theory, Languages, and Computation, *Addison-Wesley*, 1979.

10. N. IMMERMAN, Nondeterministic Space is Closed Under Complement, *S.I.A.M. J. Comput.*, 1988, *17*, pp. 935-938.
11. S.-Y. KURODA, Classes of Languages and Linear-Bounded Automata, *Inform. and Contr.*, 1964, *7*, pp. 207-223.
12. C. LAUTEMANN, One Pushdown and a Small Tape. In K. W. WAGNER Ed., Dirk Siefkes zum 50. Geburtstag, *Technische Universität Berlin/Universität Augsburg*, 1988, pp. 42-47.
13. I. NIEPEL, Logarithmisch platzbeschränkte Komplexitätsklassen: Charakterisierungen und offene Fragen, *Diplomarbeit*, Universität Hamburg, 1987.
14. I. H. SUDBOROUGH, On the Tape Complexity of Deterministic Context-Free Languages, *J. Assoc. Comput. Mach.*, 1987, *25*, pp. 405-414.
15. R. SZELEPCSÉNYI, The Method of Forced Enumeration for Nondeterministic Automata, *Acta Inform.*, 1988, *26*, pp. 279-284.
16. K. W. WAGNER and G. WECHSUNG, Computational Complexity, *Reidel Verlag*, Dordrecht, and *V.E.B. Deutscher Verlag der Wissenschaften*, Berlin, 1986.