On the Complexities of Deterministic Indexed Grammars*

(Extended Abstract)

Markus Holzer and Klaus-Jörn Lange

Wilhelm-Schickard Institut für Informatik
Universität Tübingen
Sand 13, D-72076 Tübingen
Germany

There are many close connections between complexity theory and the area of formal languages and automata. Typical relations of this kind are completeness results for families of formal languages. Well-known examples are:

1. *REG*, the family of regular languages, is NC^1-complete (Barrington [3]),
2. *LIN*, the family of linear context-free languages, is $\mathit{NSpace}(\log)$-complete (Sudborough [7]),
3. *CFL*, the family of context-free languages is $\mathit{NAuxPDASpaceTime}(\log n, \mathit{poly})$-complete (Sudborough [8]), and
4. *IND*, the family of indexed languages, is *NP*-complete (Fischer [4]).

Several restrictions of the family of context-free languages, which are motivated from compiler construction, have been extensively studied in the literature. Here we are interested in LL(1) and LR(1) restrictions only.

The idea underlying both LL(1) and LR(1) conditions is the same, i.e., to make a language deterministic parsable. But the crucial difference between the conditions is how to obtain this determinism. In the case of LL(1) grammars the parsing process works in a top-down manner, whereas the process works bottom-up for LR(1) grammars. A third method, which is in fact the most natural, is to define determinism via the corresponding automaton model, and not via a grammar type.

In formal languages it is well-known that context-free LR(1) grammars generate exactly the deterministic context-free languages and form a proper superset of the context-free LL(1) languages (see Hopcroft and Ullman [6]). Nevertheless all these classes are complete for $\mathit{DAuxPDASpaceTime}(\log n, \mathit{poly})$ (Sudborough [8]).

Here, one could ask whether determinism of the automaton model, LL(1), and LR(1) conditions always behaves similar with respect to the complexity theoretical point of view. Unfortunately, such a general statement is not true, which was shown by Holzer and Lange [5] considering linear context-free languages. It was shown that linear context-free LR(1) grammars generate exactly the deterministic linear context-free languages and that the complexity theoretical equivalence of LL(1) and LR(1) conditions as in the context-free language case are no longer valid in the presence of linear context-free grammars, since here their equivalence would imply $NC^1 = \mathit{DSpace}(\log n)$.

*Part of the work was done during the authors worked at Institut für Informatik, Technische Universität München, Arcisstr. 21, D-80290 München, Germany.

Based on context-free grammars Aho [1] introduced indexed grammars, languages, respectively. The main difference between context-free grammars, indexed grammars, respectively, is that a nonterminal can be viewed as a function call without parameter, with parameter (evaluated in a call-by-value like way), respectively. For a precise definition see Aho [1]. In addition, Aho [2] also found a nice characterization of indexed languages in terms of stack automata. This automaton provides a mathematical model for a restricted form of embedded list structure, and is therefore called nested stack automaton.

The situation for deterministic versions of indexed languages is more involved as in the context-free case. It is easy to define determinism via the automaton and via LL(1) conditions on the grammar type. A condition playing the role of LR(1) as in the context-free case is up to now not known. Here bottom-up parsing is much more complicated, because of the above mentioned function calls with parameters.

We show that both deterministic indexed languages, i.e., languages accepted by deterministic nested stack automata, and indexed LL(1) languages are complete for the deterministic counterpart of NP, i.e., we show P-completeness. In addition we study variations of indexed languages. One variation consists of restricting the underlying grammar to regular or linear indexed grammars. The other variation is to restrict the usage of the indices, i.e., the values of the parameters during a function call (see Aho [1]).

In most cases, the complexity of indexed and indexed LL(1) languages is well determined, and behaves as we expect. This means that whenever a class of indexed languages is $\mathcal{A}$-complete, then the LL(1) counterpart is complete for the deterministic counterpart of $\mathcal{A}$.

References

[1] A. V. Aho. Indexed grammars—an extension of context-free grammars. *Journal of the ACM*, 15:647–671, 1968.

[2] A. V. Aho. Nested stack automata. *Journal of the ACM*, 16(3):383–406, July 1969.

[3] D. A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in $\mathcal{NC}^1$. *Journal of Computer and System Sciences*, 38:150–164, 1989.

[4] M. J. Fischer. Grammars with macro-like productions. Ph.D. thesis, Harvard University, 1968.

[5] M. Holzer and K.-J. Lange. On the complexities of linear LL(1) and LR(1) grammars. In *Proceedings of the 9th International Conference on Fundamentals of Computation Theory*, number 710 in LNCS, pages 299–308. Springer, August 1993.

[6] J. E. Hopcroft and J. D. Ullman. *Formal Languages and Their Relation to Automata*. Addison-Wesley, 1968.

[7] I. H. Sudborough. A note on tape-bounded complexity classes and linear context-free languages. *Journal of the ACM*, 22(4):499–500, October 1975.

[8] I. H. Sudborough. On the tape complexity of deterministic context-free languages. *Journal of the ACM*, 25:405–414, 1978.