Cut Hierarchies for Restarting Automata and Marcus *t*-Contextual Grammars*

T. Jurdziński[1], **F. Mráz**[2], **F. Otto**[3], and **M. Plátek**[2]

[1] Institute of Computer Science, University of Wrocław
51-151 Wrocław, Poland
tju@ii.uni.wroc.pl

[2] Charles University, Faculty of Mathematics and Physics
Department of Computer Science, Malostranské nám. 25
118 00 PRAHA 1, Czech Republic
mraz@ksvi.ms.mff.cuni.cz, platek@ksi.ms.mff.cuni.cz

[3] Fachbereich Mathematik/Informatik, Universität Kassel
34109 Kassel, Germany
otto@theory.informatik.uni-kassel.de

Abstract. *t-contextual grammars* with regular selection are considered for which all t insertions are performed in the same neighbourhood. The languages generated by these grammars are accepted by restarting automata with *cut-index t*. Here the classes of languages accepted by certain variants of restarting automata with limited cut-index are studied.

1 Introduction

The motivation for Marcus contextual grammars [8] as well as for restarting automata [2, 15] comes mainly from linguistics. A *Marcus contextual grammar* is a generative device that describes a language through the process of inserting contexts into strings, and that works completely without nonterminals. Which strings are inserted at which places is controlled by the so-called *selection mapping*, which can be very general. Accordingly, different restrictions on this mechanism have been considered (see, e.g., [13]). A particular type of restriction are the *contextual grammars with regular selection* [10], in which the selection functions are described by means of regular languages.

A *restarting automaton*, on the other hand, is a model of an analytical device. It has a finite-state control and a scanning window of a fixed size that works on a flexible tape delimited by sentinels. Such an automaton works in cycles. In each cycle it starts in its initial state with the scanning window in the leftmost position. It can move the scanning window on the tape one cell at a time

* The work of the first and third authors was supported by a grant from the Deutsche Forschungsgemeinschaft. It was performed while T. Jurdziński was visiting the University of Kassel. The second and forth authors were partially supported by the Grant Agency of the Czech Republic under Grant-No. 201/04/2102 and by the program 'Information Society' under project 1ET100300517.

by performing move-right and move-left steps until, at some place, it decides (nondeterministically) to rewrite the part of the tape content in its window by a shorter string. After that it may perform some more move-right and move-left steps, until eventually the automaton restarts, that is, it reenters its initial state and places its window into the leftmost position. Then the next cycle starts on the now shortened tape. The automaton halts by either performing an *accept* operation, or by entering a configuration for which its control unit has no further instructions, in which case it *rejects*. Restarting automata which are restricted in their rewrite operations to only delete some symbols from the content of the window (called RL-*automata*) can be used as recognizers for languages that are generated by certain contextual grammars with regular selection [7, 10].

In some respect, however, RL-automata are more general than contextual grammars with regular selection, as in each cycle such an automaton can delete more than two subwords simultaneously. This corresponds to the notion of *t-contextual grammar with regular selection*, which is a contextual grammar with regular selection that inserts t ($t \geq 1$) subwords in each derivation step [13]. It is known that the expessive power of these grammars increases with the number t. Here we study this phenomenon in combination with the additional restriction that the places of insertion are close to each other, that is, the size of the part of the string that is changed in one derivation step by inserting t subwords is bounded by a constant. This restriction is also quite natural from a linguistic point of view.

It is easily seen that a language which is generated by a t-contextual grammar G that is restricted in this way can be recognized by an RL-automaton M which can delete at most t factors of the content of its window in each cycle. We say that such an RL-automaton has *cut-index t*. Each derivation step of G corresponds to one cycle of M, and the base set of words of G corresponds to the set of words that are accepted by M without a restart. Hence, we can even require that M is in *weak cyclic form* (see [1]), which means that M can accept without a restart only words of length not exceeding the size of M's scanning window. On the other hand, for each RL-automaton M in weak cyclic form with cut-index t, a restricted t-contextual grammar can be constructed that generates the language accepted by M. In this way the classification of classes of RL-automata in weak cyclic form with respect to the cut-index induces a classification of the generative power of those t-contextual grammars that are restricted in the aforementioned way.

For various subclasses of RL-automata we study the influence of the cut-index on their expressive power. In particular, we consider deterministic and nondeterministic variants, one-way variants, called RR-*automata*, and one-way variants that restart immediately after performing a rewrite step, called R-*automata*. Observe that the latter perform a rewrite in combination with a restart without being able to see the tape content to the right of the position where the rewrite operation is performed. It turns out that already R-automata with cut-index 1 are quite expressive, as they can recognize some NP-complete languages. Hence, we consider further restrictions – right- and left-monotone variants of restarting automata. This enables us to relate the corresponding language classes to the lower classes of the Chomsky hierarchy.

We will use the following notation. The empty word is denoted by λ, $\subseteq$ denotes the subset relation, and $\subset$ denotes the proper subset relation. $\mathbb{N}$ and $\mathbb{N}_+$ denote the sets of non-negative and of positive integers, respectively. FIN, REG, LIN, CFL, DCFL denote the classes of finite, regular, linear, context-free, and deterministic context-free languages, respectively, and DLIN denotes the class of deterministic linear languages, which is the class of languages that are accepted by deterministic one-turn pushdown automata. Further, for any set S, we will denote the power set of S by 2^S, and for any class A of grammars, $\mathcal{L}(A)$ will denote the class of languages that are generated by grammars from A.

This paper is an extended abstract, not containing full proofs for the results presented. They can be found in [5].

2 Contextual grammars

The t-contextual grammars, where t is a positive integer, are generalizations of Marcus contextual grammars [13]. In order to obtain selection mappings that are efficiently decidable we consider a restricted variant only.

Definition 1 *Let t be a positive integer. A* t-contextual grammar with regular selection *is a $(t+5)$-tuple $G = (V, B, C, \mathcal{L}_1, \ldots, \mathcal{L}_{t+1}, f)$, where V is a finite alphabet, B is a finite language over V, C is a finite subset of $(V^*)^t$, $\mathcal{L}_1, \ldots, \mathcal{L}_{t+1}$ are finite sets of regular languages over V, and $f : \mathcal{L}_1 \times \mathcal{L}_2 \times \ldots \times \mathcal{L}_{t+1} \to 2^C$ is the (regular) t-selection mapping.*

By $x \Rightarrow_G y$ we denote the derivation *of y from x in G, that is, $x \Rightarrow_G y$ holds if there exist languages $P_1 \in \mathcal{L}_1, P_2 \in \mathcal{L}_2, \ldots, P_{t+1} \in \mathcal{L}_{t+1}$ and words $x_1 \in P_1$, $x_2 \in P_2, \ldots, x_{t+1} \in P_{t+1}$ such that*

$$x = x_1 x_2 x_3 \ldots x_t x_{t+1} \quad \textit{and} \quad y = x_1 u_1 x_2 u_2 x_3 \ldots x_t u_t x_{t+1}$$

for some context $(u_1, \ldots, u_t) \in f(P_1, \ldots, P_{t+1})$. The language generated by G is the set $L(G) = \{\, v \mid \exists u \in B : u \Rightarrow_G^* v \,\}$, and $\mathsf{RS}(G) = (V^*, \Rightarrow_G^{-1})$ is the reduction system induced by G, *where $u \Rightarrow_G^{-1} v$ if and only if $v \Rightarrow_G u$.*

It is easily verified that this definition is equivalent to the definition of t-contextual grammars with selection of type REG of [13]. Thus, the expressive power of t-contextual grammars with regular selection stricly increases with the value of the parameter t.

We use the notation t-CGR to denote the class of all t-contextual grammars with regular selection. Apparently, 2-CGR coincides with the class of contextual grammars with regular selection from [10]. Restrictions that are imposed on the sets of strings describing the places of insertion will be put in parentheses after t-CGR. We will in particular be interested in the class $\mathsf{CGR}_{\mathsf{fin}}[t] :=$ t-$\mathsf{CGR}(\mathsf{REG}, \mathsf{FIN}, \ldots, \mathsf{FIN}, \mathsf{REG})$, which denotes the class of t-contextual grammars of the form $G = (V, B, C, \mathcal{L}_1, \ldots, \mathcal{L}_{t+1}, f)$, where $\mathcal{L}_1$ and $\mathcal{L}_{t+1}$ are finite sets of regular languages, and $\mathcal{L}_2, \ldots, \mathcal{L}_t$ are finite sets of finite languages. Thus, for a grammar of this form all t insertions are performed close to each other in each derivation step.

3 Two-way restarting automata

Here we describe in short the type of restarting automaton we will be dealing with. Details can be found in [11].

A *two-way restarting automaton without rewriting*, RL-automaton for short, is a nondeterministic machine M with a finite-state control Q, a finite input alphabet Σ, a flexible tape, and a read/write window of a fixed size $k \geq 1$. The work space is limited by the left sentinel ¢ and the right sentinel \$, which cannot be removed from the tape. The behaviour of M is described by a transition relation δ that associates to a pair (q, u) consisting of a state q and a possible content u of the read/write window a finite set of possible transition steps. There are five types of transition steps:

1. A *move-right step* (MVR) causes M to shift the read/write window one position to the right and to change the state. However, the read/write window cannot move across the right sentinel \$.
2. A *move-left step* (MVL) causes M to shift the read/write window one position to the left and to change the state. However, the window cannot move across the left sentinel ¢.
3. A *rewrite step* causes M to replace the content u of the read/write window by a proper scattered subword v of u, thereby shortening the tape, and to change the state. Further, the read/write window is placed immediately to the right of the string v.
4. A *restart step* causes M to place its read/write window over the left end of the tape, and to reenter the initial state q_0.
5. An *accept step* causes M to halt and accept.

If $\delta(q, u) = \emptyset$ for some pair (q, u), then M necessarily halts, and we say that M *rejects* in this situation. In addition, it is required that in each computation of M rewrite steps and restart steps *alternate* with a rewrite step coming first.

A *configuration* of M is a string $\alpha q \beta$ where $q \in Q$, and either $\alpha = \lambda$ and $\beta \in \{¢\} \cdot \Gamma^* \cdot \{\$\}$ or $\alpha \in \{¢\} \cdot \Gamma^*$ and $\beta \in \Gamma^* \cdot \{\$\}$; here q represents the current state, $\alpha\beta$ is the current content of the tape, and it is understood that the head scans the first k symbols of β or all of β when $|\beta| \leq k$. A *restarting configuration* is of the form $q_0¢w\$$, where $w \in \Sigma^*$. Thus, each computation of M can be described by a sequence of *cycles*, where a cycle begins with a restarting configuration and ends with the next restarting configuration. The part of the computation after the last restart operation is called the *tail* of the computation. We use the notation $u \vdash_M^c v$ to denote a cycle of M that begins with the restarting configuration $q_0¢u\$$ and ends with the restarting configuration $q_0¢v\$$; the relation $\vdash_M^{c^*}$ is the reflexive and transitive closure of $\vdash_M^c$. The pair $\mathsf{RS}(M) = (\Gamma^*, \vdash_M^c)$ is the *reduction system induced by M*.

An input $w \in \Sigma^*$ is *accepted* by M, if there is a computation which, starting with the *initial configuration* $q_0 ¢ w \$$, finishes by executing an accept instruction. By $L(M)$ we denote the language consisting of all words accepted by M; we say that M *recognizes (accepts) the language* $L(M)$.

In general, the automaton M is *nondeterministic*, that is, there can be two or more instructions with the same left-hand side (q, u). If this is not the case, the automaton is *deterministic*.

Now we define those subclasses of RL-automata that are relevant for our investigation. An RR-*automaton* is an RL-automaton which does not use any MVL instructions, and an R-*automaton* is an RR-automaton which restarts immediately after rewriting, that is, for such an automaton each rewrite transition is immediately followed by a restart transition. By det-RL we denote the class of *deterministic* RL-automata, and analogously for the other types of restarting automata. Further, for each type X of automata, we denote the class of languages that are accepted by automata from that class by $\mathcal{L}(\mathsf{X})$.

Next we turn to the various notions of monotonicity for restarting automata [2, 3, 16, 17]. Each cycle C of a computation of a restarting automaton contains a unique configuration $\alpha q \beta$ in which a rewrite instruction is applied. The number $|\beta|$ is called the *right distance* of C, denoted by $D_r(C)$, and $|\alpha|$ is the *left distance* of C, denoted by $D_l(C)$.

We say that a *sequence of cycles* $S = (C_1, C_2, \cdots, C_n)$ is *right-monotone* if $D_r(C_1) \geq D_r(C_2) \geq \ldots \geq D_r(C_n)$, and that it is *left-monotone* if $D_l(C_1) \geq D_l(C_2) \geq \ldots \geq D_l(C_n)$. A computation is *right-monotone* or *left-monotone*, respectively, if the corresponding sequence of cycles is right-monotone or left-monotone. Further, a computation is *right-left-monotone*, if it is simultaneously right-monotone and left-monotone. Observe that the tail of the computation does not play any role here. Finally, a restarting automaton M is called *right-monotone*, *left-monotone* or *right-left-monotone*, respectively, if all its computations that begin with an initial configuration are right-, left-, or right-left-monotone, respectively. Right-monotone restarting automata were introduced in [2] as monotone restarting automata. The prefixes mon-, left-mon-, and right-left-mon- are used to indicate the various classes of monotone, left-monotone and right-left-monotone restarting automata, respectively.

4 Cut-index and *t*-contextual grammars

Each application of the rewrite relation $\Rightarrow_G^{-1}$ corresponding to a t-contextual grammar G removes (at most) t factors from the current word, hence we relate them to RL-automata which can delete at most t factors in a rewrite step. We say that a rewrite step

$$x_1 y_1 x_2 y_2 x_3 \ldots x_t y_t x_{t+1} \to x_1 x_2 x_3 \ldots x_t x_{t+1}$$

has t 'cuts,' if the factors x_i are non-empty for all $2 \leq i \leq t$, and the factors y_j are non-empty for all $1 \leq j \leq t$. We say that an RL-automaton has *cut-index* t, if all its rewrite instructions have at most t cuts. For a class of automata X, we denote by $\mathsf{cut}(t)\text{-}\mathsf{X}$ the set of all automata from X with cut-index t.

Naturally, each language that is recognized by some R-, RR-, or RL-automaton has a finite cut-index. In particular, the $\mathsf{cut}(2)$-RL-automaton is just the *normal* RL-automaton introduced in [7] as a generalization of the normal R-automaton from [1]. Hence, the cut-index can be seen as a generalization of the notion of normality.

Further, a restarting automaton M with a read/write window of size k is said to be in *weak cyclic form* if M immediately (that is, without a restart) accepts or rejects any word of length less than or equal to k, and M performs at least one restart step or rejects for any word of length exceeding k. The property of being in weak cyclic form will be denoted by the prefix wcf-.

The following result generalizes a result for $\mathsf{CGR}_{\mathrm{fin}}[2]$ established in [7].

Lemma 2 *For each $G \in \mathsf{CGR}_{\mathrm{fin}}[t]$, a* $\mathsf{wcf}\text{-}\mathsf{cut}(t)\text{-}\mathsf{RL}$*-automaton M can be constructed such that $L(M) = L(G)$ and $\mathsf{RS}(M) = \mathsf{RS}(G)$.*

The converse transformation of $\mathsf{cut}(t)$-RL-automata in weak cyclic form to t-CGRs is also possible (it is a minor generalization of the corresponding statement for $\mathsf{CGR}_{\mathrm{fin}}[2]$ given in [7]).

Lemma 3 *For each* $\mathsf{wcf}\text{-}\mathsf{cut}(t)\text{-}\mathsf{RL}$*-automaton M, a t-contextual grammar $G \in \mathsf{CGR}_{\mathrm{fin}}[t]$ can be constructed such that $L(G) = L(M)$ and $\mathsf{RS}(G) = \mathsf{RS}(M)$.*

For some types of restarting automata it is possible to transform a given automaton of that type into another automaton of the same type that is in weak cyclic form. For example, each monotone RR-automaton can be transformed into an equivalent monotone RR-automaton that is in weak cyclic form [9], while it is shown in [14] that monotone R-automata in weak cyclic form are strictly less expressive than monotone R-automata that are not in weak cyclic form.

Remark 4 *In what follows we will establish various hierarchy results for restarting automata. All these results will be formulated in terms of restarting automata in weak cyclic form, but they all carry over to the case of restarting automata that are not in weak cyclic form.*

For each RL-automaton M, a (nondeterministic) RR-automaton M' can be constructed such that, for all words u, v, $u \vdash_M^c v$ if and only if $u \vdash_{M'}^c v$, and the right distance (as well as the left distance) is the same in both cycles [16]. Hence, in the nondeterministic case, RL- and RR-automata are equivalent. In particular, this yields the following consequence, where ε is used to denote the empty prefix.

Theorem 5 *For each $t \geq 1$ and each $\mathsf{Y} \in \{\varepsilon, \mathsf{mon}\text{-}, \mathsf{left}\text{-}\mathsf{mon}\text{-}, \mathsf{right}\text{-}\mathsf{left}\text{-}\mathsf{mon}\text{-}\}$,*

$$\mathcal{L}(\mathsf{wcf}\text{-}\mathsf{Y}\text{-}\mathsf{cut}(t)\text{-}\mathsf{RL}) = \mathcal{L}(\mathsf{wcf}\text{-}\mathsf{Y}\text{-}\mathsf{cut}(t)\text{-}\mathsf{RR}).$$

Nevertheless, we can show that by increasing the value of the cut-index we increase the expressive power of deterministic as well as nondeterministic RL-, RR- and R-automata. In order to establish these hierarchies we will construct a sequence of sample languages that will be based on the following language

$$\begin{aligned}L := \{\, x^{i_0}y^{i_1}x^{i_2}y^{i_3}\ldots y^{i_m} \mid\ & x,y \in \{a,b\}, x \neq y, m > 0, i_0,\ldots,i_{m-1} > 0, i_m \geq 0,\\ & \exists p \geq 0 : 2^p = i_0 + 2\cdot i_1 + 4\cdot i_2 + \ldots + 2^m \cdot i_m \,\}.\end{aligned}$$

For each $t \geq 1$, let $\varphi_t : \{a,b\}^* \to \{0,1\}^*$ be the morphism defined by

$$\varphi_t(a) := (001)^t \quad \text{and} \quad \varphi_t(b) := (01)^t,$$

and let L_t denote the language $L_t := \varphi_t(L)$.

Theorem 6 *For each $t > 1$, $L_t \in \mathcal{L}(\textsf{wcf-det-cut}(t)\textsf{-R}) \smallsetminus \mathcal{L}(\textsf{cut}(t-1)\textsf{-RL})$.*

First we establish the following result, where an RW-automaton is an R-automaton with rewriting, that is, its rewrite instructions are of the more general form $u \to v$, where v is any string from Σ^* satisfying $|v| < |u|$.

Lemma 7 *The language L is accepted by a deterministic* RW*-automaton M that is in weak cyclic form and that has a read/write window of size 3.*

Proof. Let $M = (Q, \{a,b\}, \{a,b\}, ¢, \$, q_0, 3, \delta)$ be the RW-automaton that is given by the following description:

- M immediately accepts the words a and b.
- For a word w of length at least 2, let $x \in \{a,b\}$ be the first symbol of w and let $y \in \{a,b\}$, $y \neq x$. If w does not start with xx, then M rejects, otherwise M moves to the right until the rightmost symbol in its read/write window is different from x. Now, if the content of the read/write window is
  - $xx\$$, then M rewrites it into $y\$$ and restarts,
  - xxy, then M rewrites it into yy and restarts.

Obviously, M is deterministic and in weak cyclic form. It remains to show that $L(M) = L$.

Let $w = x^{i_0}y^{i_1}x^{i_2}y^{i_3}\ldots y^{i_m}$ such that $x,y \in \{a,b\}$, $x \neq y$, $m > 0$, $i_m \geq 0$ and $i_0,\ldots,i_{m-1} > 0$. If $w \in L$, then there exists an integer $p \geq 0$ such that $2^p = i_0 + \sum_{j=1}^{m} 2^j \cdot i_j$. If $p = 0$, then $i_0 = 1$, $m = 1$, and $i_1 = 0$, that is, $w = x \in \{a,b\}$. Hence, w is immediately accepted by M.

If $p > 0$, then $i_0 > 0$ is even, and by performing $\frac{i_0}{2}$ cycles, M will rewrite w into the word

$$w' = y^{\frac{i_0}{2}+i_1}x^{i_2}y^{i_3}\ldots y^{i_m} = y^{i'_0}x^{i'_1}y^{i'_2}\ldots y^{i'_{m-1}},$$

where

$$\sum_{j=0}^{m-1} 2^j \cdot i'_j = (\frac{i_0}{2} + i_1) + \sum_{j=1}^{m-1} 2^j \cdot i_{j+1} = 2^{p-1}.$$

Hence, after a finite number of cycles, M obtains a word w'' of the form $w'' = x^{2^r}$ or $w'' = y^{2^r}$ for some $r \geq 0$. If $r > 0$, then $x^{2^r} \vdash_M^{c^*} y^{2^{r-1}}$ and $y^{2^r} \vdash_M^{c^*} x^{2^{r-1}}$. Thus, after a finite number of cycles M obtains the tape content a or b and accepts.

On the other hand, let w be a word that is accepted by M in a sequence of $n \geq 0$ cycles

$$w = w_0 \vdash_M^c w_1 \vdash_M^c \ldots \vdash_M^c w_n,$$

where $w_0, \ldots, w_n \in \{a, b\}^*$, which is followed by an accepting tail computation. As a, b are the only words accepted by M in tail computations, $w_n \in \{a, b\}$. By induction on the number n of cycles it can be shown that the following property holds:

$$\exists x, y \in \{a, b\} \, \exists m > 0 \, \exists p \geq 0 \, \exists i_0, i_1, \ldots, i_{m-1} > 0 \, \exists i_m \geq 0 : \qquad (*)$$
$$w = x^{i_0} y^{i_1} x^{i_2} y^{i_3} \ldots y^{i_m} \text{ and } 2^p = \sum_{j=0}^{m} 2^j \cdot i_j.$$

This completes the proof that $L(M) = L$. □

Proof of Theorem 6. Based on M and φ_t, a deterministic R-automaton M_t for the language L_t can be constructed. The rewrite instructions of M_t correspond to the rewrite instructions of M, which actually rewrite aa into b and bb into a. In order to rewrite $\varphi_t(aa) = (001)^{2t}$ into $\varphi_t(b) = (01)^t$, we can delete the prefix $(001)^t0$ and delete one occurrence of the symbol 0 from each of the remaining $t-1$ factors 001. Similarly, to rewrite $\varphi_t(bb) = (01)^{2t}$ into $\varphi_t(a) = (001)^t$, we can delete every odd-numbered occurrence of the symbol 1 from $(01)^t$. Hence, it is easily seen that M_t has cut-index t. Further, with M also M_t is in weak cyclic form.

It remains to prove that L_t is not accepted by any RL-automaton with cut-index $t-1$.

Assume that L_t is accepted by some RL-automaton M_t' with a read/write window of size k'. Then, for a sufficiently large integer $n > k'$, the word $w := \varphi_t(a^{2^n}) = (001)^{2^n \cdot t} \in L_t$ is not accepted by M_t' in a tail computation, as otherwise, using pumping techniques, we can easily construct a word outside L_t which is also accepted by M_t'. Thus, each accepting computation of M_t' on input w starts with a cycle $w \vdash_{M_t'}^c w'$ for some word $w' \in L_t$. As any word in $L_t \cap \varphi_t(a^+)$ that is shorter than the word $\varphi_t(a^{2^n})$ is of length at most

$$|\varphi_t(a^{2^{n-1}})| = |\varphi_t(a^{2^n})| - 2^{n-1} \cdot 3t < |\varphi_t(a^{2^n})| - k',$$

the word w' must contain at least one factor of the form $\varphi_t(b)$. The first such factor of w' is either preceded by ¢ , if it is a prefix of w', or it is preceded by a factor of the form $\varphi_t(a) = (001)^t$ that ends with the symbol 1. Hence, ¢w'\$ either contains the factor ¢$(01)^t$ or the factor $1(01)^t$. In either case this factor cannot be obtained from ¢$(001)^{2^n \cdot t}$\$ by deleting less than t subwords. Hence, M_t' must have cut-index larger than or equal to t, which completes the proof. □

As an immediate consequence we obtain the following hierarchies.

Corollary 8 *For each $t \geq 1$ and each $\mathsf{X} \in \{\mathsf{R}, \mathsf{RR}, \mathsf{RL}\}$,*

(a) $\mathcal{L}(\text{wcf-det-cut}(t)\text{-X}) \subset \mathcal{L}(\text{wcf-det-cut}(t+1)\text{-X})$,

(b) $\mathcal{L}(\text{wcf-cut}(t)\text{-X}) \subset \mathcal{L}(\text{wcf-cut}(t+1)\text{-X})$.

5 On the power of cut(1)-R-automata

The language $L_1 = \varphi_1(L)$ defined above is not context-free, in fact, it is not even semi-linear, as $L_1 \cap (001)^* = \{\, (001)^n \mid \exists p \geq 0 : n = 2^p \,\}$, but it is accepted by a wcf-det-cut(1)-R-automaton, implying the following result.

Theorem 9 $\mathcal{L}(\text{wcf-det-cut}(1)\text{-R})$ *contains a language which is not semi-linear.*

Syntactically nondeterministic cut(1)-R-automata are very restricted, but surprisingly they are still quite expressive. We demonstrate this by showing that cut(1)-R-automata can even recognize NP-complete languages.

Theorem 10 $\mathcal{L}(\text{wcf-cut}(1)\text{-R})$ *contains* NP*-complete languages.*

Proof. In [4] a log-space reduction is presented that, for any NP-complete language L, yields an NP-complete language L_1 which is accepted by a shrinking RWW-automaton M. A shrinking RWW-automaton is a generalization of an RWW-automaton that uses rewrite steps that are weight-reducing with respect to some weight function instead of being required to be length-reducing (see [4] for details). Our result is obtained by modifying this construction in two steps.

First M is transformed into a shrinking RWW-automaton M' such that $L(M') = L(M)$ and all rewrite instructions of M' replace a nonempty subword by a single symbol or by the empty word. Then a morphism ϑ is introduced such that $\vartheta(L(M'))$ is still an NP-complete language, and there exists an R-automaton M'' such that $L(M'') \cap \vartheta(\Sigma^*) = \vartheta(L(M'))$, which implies that $L(M'')$ is NP-complete, too. Because of the restricted form of the rewrite instructions of M', M'' has cut-index 1. □

6 Cut hierarchies for monotone automata

As seen in the previous section, already cut(1)-R-automata are quite powerful. Hence, it is worth to also study some other restricted variants of restarting automata. Here we concentrate on the various monotone versions, which enables us to relate the language classes obtained to the Chomsky hierarchy.

All the relations shown in this and the previous sections are summarized in Fig. 1–4. In these figures, an arrow $\mathsf{A} \xrightarrow{j} \mathsf{B}$ between two types of restarting automata A and B means that the proper inclusion relation $\mathcal{L}(\mathsf{A}) \subset \mathcal{L}(\mathsf{B})$, and also $\mathcal{L}(\text{wcf-A}) \subset \mathcal{L}(\text{wcf-B})$ (see Remark 4), is shown in statement j. Similarly, $\mathsf{A} \overset{j}{=\!=\!=} \mathsf{B}$ states that the equality $\mathcal{L}(\mathsf{A}) = \mathcal{L}(\mathsf{B})$, and also $\mathcal{L}(\text{wcf-A}) = \mathcal{L}(\text{wcf-B})$,

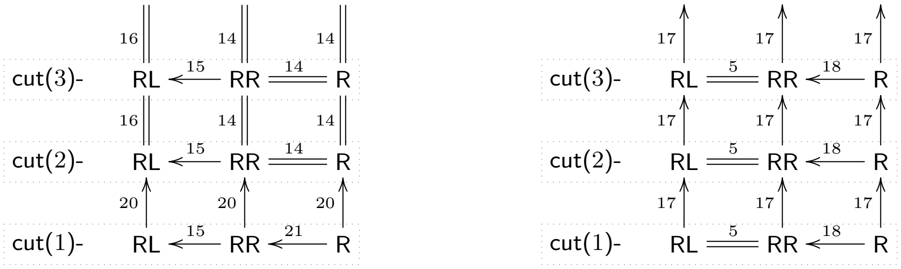

Fig. 1. Hierarchies for deterministic (left diagram) and nondeterministic (right diagram) right-left-monotone restarting automata.

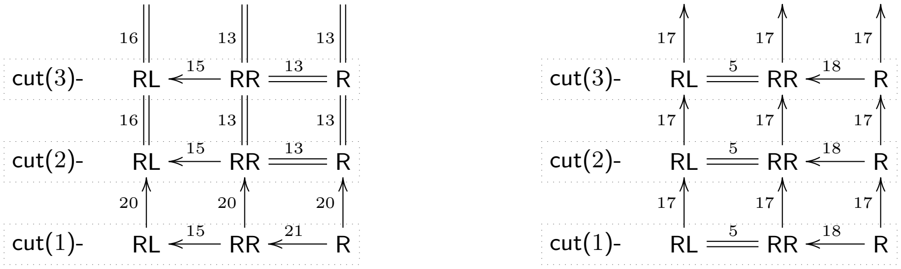

Fig. 2. Hierarchies for deterministic (left diagram) and nondeterministic (right diagram) (right-) monotone restarting automata.

is shown in statement j. Hence, each figure represents two diagrams, where the second one is obtained by adding the prefix wcf- to all the classes of restarting automata depicted.

Each regular language is accepted by some R-automaton without rewriting (deleting). On the other hand, the non-regular language $\{ a^n b^n \mid n \geq 0 \}$ is recognized by a deterministic wcf-right-left-monotone R-automaton with cut-index 1, implying the following proper inclusion.

Theorem 11 $\mathsf{REG} \subset \mathcal{L}(\mathsf{wcf\text{-}det\text{-}right\text{-}left\text{-}mon\text{-}cut(1)\text{-}R})$.

On the other hand, Theorem 5 and the results of [3, 17] yield the following.

Theorem 12 *For each $t \geq 1$ the following proper inclusions hold:*

(a) $\mathcal{L}(\mathsf{mon\text{-}cut}(t)\text{-}\mathsf{RL}) \subset \mathsf{CFL}$,

(b) $\mathcal{L}(\mathsf{left\text{-}mon\text{-}cut}(t)\text{-}\mathsf{RL}) \subset \mathsf{CFL}$,

(c) $\mathcal{L}(\mathsf{right\text{-}left\text{-}mon\text{-}cut}(t)\text{-}\mathsf{RL}) \subset \mathsf{LIN}$.

For deterministic (right-) monotone R- and RR-automata it follows from results of [1, 3] that the cut-hierarchies collapse at level 2 into DCFL.

Theorem 13 *For each $t \geq 2$,* $\mathcal{L}(\mathsf{wcf\text{-}det\text{-}mon\text{-}cut}(t)\text{-}\mathsf{R(R)}) = \mathsf{DCFL}$.

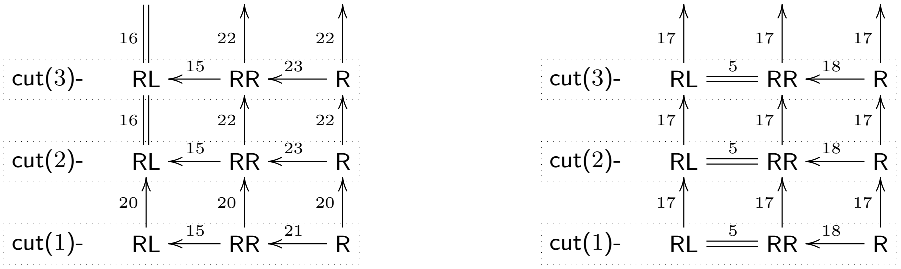

Fig. 3. Hierarchies for deterministic (left diagram) and nondeterministic (right diagram) left-monotone restarting automata.

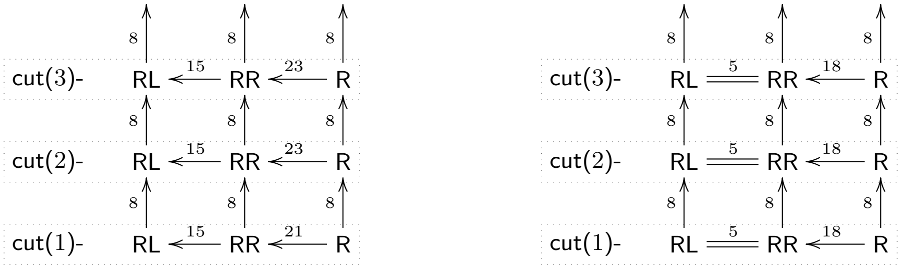

Fig. 4. Hierarchies for deterministic (left diagram) and nondeterministic (right diagram) restarting automata.

A similar collapse occurs for deterministic right-left-monotone R- and RR-automata, this time into the class DLIN of deterministic linear languages.

Theorem 14 *For each* $t \geq 2$, $\mathcal{L}(\text{wcf-det-right-left-mon-cut}(t)\text{-R(R)}) = \text{DLIN}$.

Deterministic RL-automata are more powerful than deterministic RR-automata. This remains true even when we consider automata with restricted cut-index, as the language $\{ a^n b^n c \mid n \geq 0 \} \cup \{ a^n b^{2n} d \mid n \geq 0 \}$ is accepted by a wcf-det-right-left-mon-cut(1)-RL-automaton, but it cannot be accepted by any det-RR-automaton. This result yields the following proper inclusions.

Corollary 15 *For each* $t \geq 1$ *and each* $\mathsf{Y} \in \{\varepsilon, \text{mon-}, \text{left-mon-}, \text{right-left-mon-}\}$,

$$\mathcal{L}(\text{wcf-det-Y-cut}(t)\text{-RR}) \subset \mathcal{L}(\text{wcf-det-Y-cut}(t)\text{-RL}).$$

On the other hand, a statement similar to Theorem 13 also holds for any monotone type of deterministic RL-automaton.

Theorem 16 *For each* $t \geq 2$ *and each* $\mathsf{Y} \in \{\text{mon-}, \text{left-mon-}, \text{right-left-mon-}\}$,

$$\mathcal{L}(\text{wcf-det-Y-cut}(t)\text{-RL}) = \mathcal{L}(\text{wcf-det-Y-cut}(2)\text{-RL}).$$

Proof. The first part was proved in [7] in the form

$$\mathcal{L}(\textsf{det-right-left-mon-RL}) = \mathcal{L}(\textsf{det-cut(2)-right-left-mon-RL}).$$

Again, the det-cut(2)-right-left-mon-RL-automaton constructed for a given deterministic right-left-RL-automaton is in weak cyclic form.

The second statement follows from the following observation. In [6] it is shown that, for each det-left-mon-RLWW-automaton M, there exists a deterministic generalized sequential machine G such that $G(L)^R \in \textsf{DCFL}$. Here we use the notation w^R to denote the mirror image of a word w, which is extended to languages L and language classes $\mathcal{L}$ by taking $L^R := \{ w^R \mid w \in L \}$ and $\mathcal{L}^R := \{ L^R \mid L \in \mathcal{L} \}$, respectively. Hence, the language $G(L)^R$ is accepted by a deterministic monotone R-automaton M_1 with cut-index 2 (Theorem 13). Further, it is shown in [6] that the language $L(M)$ is accepted by a deterministic RL-automaton of a very special form. In each cycle this RL-automaton first scans its tape from left to right (without performing a rewrite step), and then it scans it again from right to left using M_1 to identify the place and the form of the actual rewrite transition to be applied. Thus, the resulting automaton is a deterministic RL-automaton which is left-monotone and which has cut-index 2. Also it is in weak cyclic form.

The third equality follows from the fact that

$$\mathcal{L}(\textsf{det-mon-RL}) = \mathcal{L}(\textsf{det-left-mon-RL})^R$$

([6] Lemma 1), and the observation that this equality remains valid for all levels of the cut-hierarchy and also for automata in weak cyclic form. □

In contrast to the above results the cut-hierarchy does not collapse for any monotone type of nondeterministic R-, RR-, or RL-automaton. This is based on the fact that, for each $t \geq 2$, a properly encoded version of the language

$$\{ a^n b^n, a^n c b^n \mid n \geq 0 \} \cup \{ a^m b^n, a^m d b^n \mid m > 2n \geq 0 \}$$

is accepted by a wcf-right-left-mon-cut(t)-R-automaton, but it cannot be accepted by any cut($t-1$)-RL-automaton. As a consequence we obtain the following hierarchy results.

Corollary 17 *For each* $t \geq 1$, *each* $\textsf{Y} \in \{\textsf{mon-}, \textsf{left-mon-}, \textsf{right-left-mon-}\}$, *and each* $\textsf{X} \in \{\textsf{R}, \textsf{RR}, \textsf{RL}\}$, $\mathcal{L}(\textsf{wcf-Y-cut}(t)\textsf{-X}) \subset \mathcal{L}(\textsf{wcf-Y-cut}(t+1)\textsf{-X})$.

Next we separate the cut-hierarchies for nondeterministic RR-automata from those for nondeterministic R-automata. This is based on the observation that the language $\{ a^n b^n c \mid n \geq 0 \} \cup \{ a^n b^{2n} d \mid n \geq 0 \}$ cannot be accepted by any R-automaton.

Corollary 18 *For each* $t \geq 1$ *and each* $\textsf{Y} \in \{\varepsilon, \textsf{mon-}, \textsf{left-mon-}, \textsf{right-left-mon-}\}$,

$$\mathcal{L}(\textsf{wcf-Y-cut}(t)\textsf{-R}) \subset \mathcal{L}(\textsf{wcf-Y-cut}(t)\textsf{-RR}).$$

With this we have completely determined the relationships with respect to the cut-index between the various monotone types of nondeterministic R-, RR-, and RL-automata.

The language $\{ a^n b^m \mid 0 \leq n \leq m \leq 2n \}$ cannot be accepted by any det-RL-automaton, but it is accepted by a wcf-right-left-mon-R-automaton with cut-index 1, which gives the following separation results.

Corollary 19 *For each* $t \geq 1$*, each* $\mathsf{Y} \in \{\varepsilon, \mathsf{mon\text{-}}, \mathsf{left\text{-}mon\text{-}}, \mathsf{right\text{-}left\text{-}mon\text{-}}\}$*, and each* $\mathsf{X} \in \{\mathsf{R}, \mathsf{RR}, \mathsf{RL}\}$*,* $\mathcal{L}(\mathsf{wcf\text{-}det\text{-}Y\text{-}cut}(t)\text{-}\mathsf{X}) \subset \mathcal{L}(\mathsf{wcf\text{-}Y\text{-}cut}(t)\text{-}\mathsf{X})$.

The language $\{ a^n c b^n \mid n \geq 0 \}$ cannot be accepted by any RL-automaton with cut-index 1, while it is accepted by some deterministic right-left-mon-cut(2)-R-automaton. This yields the following results.

Corollary 20 *For each* $\mathsf{Y} \in \{\mathsf{mon\text{-}}, \mathsf{left\text{-}mon\text{-}}, \mathsf{right\text{-}left\text{-}mon\text{-}}\}$ *and each* $\mathsf{X} \in \{\mathsf{R}, \mathsf{RR}, \mathsf{RL}\}$*,* $\mathcal{L}(\mathsf{wcf\text{-}det\text{-}Y\text{-}cut}(1)\text{-}\mathsf{X}) \subset \mathcal{L}(\mathsf{wcf\text{-}det\text{-}Y\text{-}cut}(2)\text{-}\mathsf{X})$.

As $\{ a^n b^i c b^j \mid n, i, j \geq 0, n = i + j \} \cup \{ a^n b^n \mid n \geq 0 \}$ is accepted by a deterministic wcf-right-left-mon-RR-automaton with cut-index 1, while it cannot be accepted by any R-automaton with cut-index 1, we obtain the following proper inclusions.

Corollary 21 *For each* $\mathsf{Y} \in \{\varepsilon, \mathsf{mon\text{-}}, \mathsf{left\text{-}mon\text{-}}, \mathsf{right\text{-}left\text{-}mon\text{-}}\}$*,*

$$\mathcal{L}(\mathsf{wcf\text{-}det\text{-}Y\text{-}cut}(1)\text{-}\mathsf{R}) \subset \mathcal{L}(\mathsf{wcf\text{-}det\text{-}Y\text{-}cut}(1)\text{-}\mathsf{RR}).$$

In contrast to the situation for the other types of monotone deterministic restarting automata, the cut-hierarchies for left-monotone deterministic R- and RR-automata are infinite, as, for each $t \geq 3$, the language

$$\{ a^{m+n} (bc)^n b^m d \mid m, n \geq 0 \} \cup \{ a^{m+n} (bc)^{t \cdot n} c^{t \cdot m} e \mid m, n \geq 0 \}$$

is accepted by a wcf-det-left-mon-R-automaton that has cut-index t, while it cannot be accepted by any det-RR-automaton with cut-index less than t. This has the following consequences.

Corollary 22 *For each* $t \geq 2$*,*

$$\mathcal{L}(\mathsf{wcf\text{-}det\text{-}left\text{-}mon\text{-}cut}(t)\text{-}\mathsf{R(R)}) \subset \mathcal{L}(\mathsf{wcf\text{-}det\text{-}left\text{-}mon\text{-}cut}(t+1)\text{-}\mathsf{R(R)}).$$

Finally, we consider the language $L := L_1 \cup L_2 \cup L_3 \cup L_4$, where

$$\begin{array}{ll}
L_1 := \{ a^m (1010)^n (100)^i & \mid m = n + i, \text{ where } m, i > 0, n \geq 0 \}, \\
L_2 := \{ a^m (1010)^n (100)^j 1 (100)^i & \mid m = n + j + i, \text{ where } m, i, j > 0, n \geq 0 \}, \\
L_3 := \{ a^m (1010)^n 1 (100)^j 1 (100)^i & \mid m = n + j + i, \text{ where } m, i, j > 0, n \geq 0 \}, \\
L_4 := \{ a^m (1010)^n 0^i & \mid 2m = n + i, \text{ where } m, i > 0, n \geq 0 \}.
\end{array}$$

This language is accepted by a wcf-det-left-mon-RR-automaton M with cut-index 1. However, it cannot be accepted by any R-automaton, which yields the following separation results.

Corollary 23 *For each $t \geq 1$:*

(a) $\mathcal{L}(\mathsf{wcf\text{-}det\text{-}left\text{-}mon\text{-}cut}(t)\text{-}\mathsf{R}) \subset \mathcal{L}(\mathsf{wcf\text{-}det\text{-}left\text{-}mon\text{-}cut}(t)\text{-}\mathsf{RR})$,

(b) $\mathcal{L}(\mathsf{wcf\text{-}det\text{-}cut}(t)\text{-}\mathsf{R}) \subset \mathcal{L}(\mathsf{wcf\text{-}det\text{-}cut}(t)\text{-}\mathsf{RR})$.

7 Conclusions

We have studied the influence of the cut-index on the expressive power of various types of restarting automata. Because of the correspondence between restarting automata in weak cyclic form and t-contextual grammars with regular selection for which the places of insertion are required to be close to each other, our hierarchy results translate into corresponding results for these classes of t-contextual grammars. It is of interest to abstract from the locality restriction, and to study the correspondence of t-contextual grammars with regular selection to certain types of restarting automata. In order to obtain such classes of automata, the restarting automaton must be allowed to perform t $\mathsf{cut}(1)$-deletions at arbitrary places in each cycle. This line of research has been followed in [12].

References

1. P. Jančar, F. Mráz, M. Plátek, M. Procházka, and J. Vogel. Restarting automata, Marcus grammars and context-free languages. In J. Dassow, G. Rozenberg, and A. Salomaa, editors, *Developments in Language Theory II, Proc.*, World Scientific Publ., Singapore, 1996, 102–111.
2. P. Jančar, F. Mráz, M. Plátek, J. Vogel: Restarting automata. In H. Reichel, editor, *FCT'95, Proc.*, *LNCS 965*, Springer, Berlin, 1995, 283–292.
3. P. Jančar, F. Mráz, M. Plátek, and J. Vogel. On monotonic automata with a restart operation. *J. Autom. Lang. Comb.*, 4 (1999) 287–311.
4. T. Jurdziński, F. Mráz, F. Otto, and M. Plátek. On the complexity of 2-monotone restarting automata. *Mathematische Schriften Kassel* 4/04, Universität Kassel, 2004. An extended abstract appeared in C.S. Calude, E. Calude and M.J. Dinneen, editors, *DLT'2004, Proc.*, *LNCS 3340*, Springer, Berlin, 2004, 237–248.
5. T. Jurdziński, F. Mráz, F. Otto, and M. Plátek. Cut Hierarchies for Restarting Automata and Marcus t-Contextual Grammars. *Mathematische Schriften Kassel* 21/04, Universität Kassel, 2004.
6. T. Jurdziński, F. Mráz, F. Otto, and M. Plátek. Monotone deterministic RL-automata don't need auxiliary symbols. In C. De Felice and A. Restivo, editors, *DLT'2005, Proc.*, *LNCS 3572*, Springer, Berlin, 2005, 284–295.
7. T. Jurdziński, F. Otto, F. Mráz, and M. Plátek. Deterministic two-way restarting automata and Marcus contextual grammars. *Fund. Inform.*, 64 (2005) 217–228.
8. S. Marcus. Contextual grammars. *Revue Roum. Mathy. Pures Appl.*, 14 (1969) 1525–1534.
9. F. Mráz, M. Plátek, and M. Procházka. On special forms of restarting automata. *Grammars*, 2 (1999) 223–233.
10. F. Mráz, M. Plátek, and M. Procházka. Restarting automata, deleting, and Marcus grammars. In C. Martín-Vide and G. Păun, editors, *Recent Topics in Mathematical and Computational Linguistics*, The Publishing House of the Romanian Academy, Bucharest, 2000, 218–233.

11. F. Otto. Restarting Automata - Notes for a Course at the 3rd International PhD School in Formal Languages and Applications. *Mathematische Schriften Kassel* 6/04, Universität Kassel, 2004.
12. F. Otto, F. Mráz, and M. Plátek. Degrees of monotonicity and Marcus t-contextual grammars. In Z. Esik and Z. Fülöp, editors, *AFL'2005, Proc.*, University of Szeged, 2005.
13. G. Păun. *Marcus Contextual Grammars*, Studies in Linguistics and Philosophy, vol. 67. Kluwer Academic Publishers, Dordrecht/Boston/London, 1997.
14. M. Plátek. Weak cyclic forms of restarting automata. In G. Rozenberg and W. Thomas, editors, *DLT'99, Proceedings*, World Scientific, 2000, 115-124.
15. M. Plátek, M. Lopatková, and K. Oliva. Restarting automata: motivations and applications. In M. Holzer, editor, *Workshop "Petrinetze" und 13. Theorietag "Automaten und Formale Sprachen"*, Institut für Informatik, Technische Universität München, 2003, 90–96.
16. M. Plátek, F. Otto, and F. Mráz. Restarting automata and variants of j-monotonicity. In E. Csuhaj-Varjú, C. Kintala, D. Wotschke, and G. Vaszil, editors, *DCFS 2003 Descriptional Complexity of Formal Systems, Proc.*, MTA SZTAKI, Budapest, 2003, 303–312.
17. M. Plátek, F. Otto, F. Mráz, and T. Jurdzinski. Restarting automata and variants of j-monotonicity. *Mathematische Schriften Kassel* 9/03, Universität Kassel, 2003.