CHURCH-ROSSER GROUPS AND GROWING CONTEXT-SENSITIVE GROUPS

Mark Kambites and Friedrich Otto

Fachbereich Mathematik / Informatik, Universität Kassel
34109 Kassel, Germany

[kambites|otto]@theory.informatik.uni-kassel.de

Abstract. A finitely generated group is called a *Church-Rosser group* (*growing context-sensitive group*) if it admits a finitely generated presentation for which the word problem is a Church-Rosser (growing context-sensitive) language. Although the Church-Rosser languages are incomparable to the context-free languages under set inclusion, they strictly contain the class of deterministic context-free languages. As each context-free group language is actually deterministic context-free, it follows that all context-free groups are Church-Rosser groups. As the free abelian group of rank 2 is a non-context-free Church-Rosser group, this inclusion is proper. On the other hand, we show that there are co-context-free groups that are not growing context-sensitive. Also some closure and non-closure properties are established for the classes of Church-Rosser and growing context-sensitive groups. More generally, we also establish some new characterizations and closure properties for the classes of Church-Rosser and growing context-sensitive languages.

1. Introduction

Let $\langle \Sigma; R\rangle$ be a presentation of a group G, that is, Σ is a finite set of generators and $R \subset \underline{\Sigma}^*$ is a set of defining relators, where $\overline{\Sigma}$ is a set of formal inverses that is in one-to-one correspondence to Σ, and $\underline{\Sigma} = \Sigma \cup \overline{\Sigma}$. With this presentation we can associate the language $\mathrm{WP}(G, \Sigma)$, which consists of all words over $\underline{\Sigma}$ that represent the identity element of the group G. Accordingly, this language is known as the *word problem* of G with respect to the generating set Σ. As $\mathrm{WP}(G, \Sigma)$ is a formal language in the sense of formal language theory, it is quite natural to classify group presentations (and groups) with respect to the form of this language. If $\mathcal{L}$ is a class of languages that is closed under inverse morphisms, then the word problem $\mathrm{WP}(G, \Sigma)$ belongs to $\mathcal{L}$ if and only if $\mathrm{WP}(G, \Sigma')$ belongs to $\mathcal{L}$, that is, this property is independent of the chosen finite generating set.

It is known that the word problem of a finitely generated group is a regular language if and only if this group is finite [1]. Further, it has been established that the word problem of a finitely generated group is context-free if and only if this group is a finitely generated virtually free group [8, 15].

Here we are interested in groups for which the word problem is a Church-Rosser language [14] or a growing context-sensitive language [7]. It is known that the class CRL of Church-Rosser languages and the class CFL of context-free languages are incomparable under set inclusion [6]. Further, CRL can be seen as the deterministic variant of the class GCSL of growing context-sensitive languages [16, 17], which in turn contains CFL properly. A finitely generated group G is called *Church-Rosser* (*growing context-sensitive*), if G admits a finitely generated presentation for which the word problem is a Church-Rosser (growing context-sensitive) language. As these language classes are closed under inverse morphisms, these properties do not depend on the choice of the finitely generated presentation, that is, they are actually properties of groups.

It is known that the word problem of a finitely generated group is in fact deterministic context-free, if it is context-free (see, e.g., [2]). As CRL contains the class DCFL of deterministic context-free languages [14], we see that we have the following sequence of inclusions, where CFG, CRG, and GCSG denote the classes of finitely generated groups that are context-free, Church-Rosser, or growing context-sensitive, respectively:

$$\mathsf{CFG} \subseteq \mathsf{CRG} \subseteq \mathsf{GCSG}.$$

A finitely generated group G is called *co-context-free* [11], if G admits a finitely generated presentation $\langle \Sigma; R\rangle$ for which the complement of the word problem, that is, the set $\text{co-WP}(G,\Sigma) := \underline{\Sigma}^* \smallsetminus \text{WP}(G,\Sigma)$, is context-free, and analogously, we obtain the co-Church-Rosser groups and the co-growing context-sensitive groups. Again these properties are independent of the chosen finitely generated presentation. The corresponding classes of groups are denoted by $\mathsf{co\text{-}CFG}$, $\mathsf{co\text{-}CRG}$, and $\mathsf{co\text{-}GCSG}$, respectively. However, as CRL is closed under complement, we see that $\mathsf{co\text{-}CRG} = \mathsf{CRG}$ holds. Thus, we have the inclusions

$$\mathsf{CRG} \subseteq \mathsf{GCSG} \cap \mathsf{co\text{-}GCSG} \quad \text{and} \quad \mathsf{co\text{-}CFG} \subseteq \mathsf{co\text{-}GCSG}.$$

Here we are concerned with the problem of which of the above inclusions are proper. As the membership problem for each growing context-sensitive language can be solved in polynomial time, the word problem for each (co-)growing context-sensitive group is solvable in polynomial time. Are there other interesting decision problems that can be solved efficiently, that is in polynomial time, for these groups?

This paper is structured as follows. In Section 2 we restate in short the definitions of the Church-Rosser and the growing context-sensitive languages, and we describe characterizations of these language classes in terms of a particular machine model. Further, some new closure properties of these language classes are stated. In Section 3 Goodman's and Shapiro's notions of *generalized Dehn algorithms* are described, and it is shown that these algorithms can be interpreted as descriptions of special types of Church-Rosser languages. Finally, in Section 4 we introduce the necessary notions on groups, and in Sections 5 and 6 we present our results on Church-Rosser groups, growing context-sensitive groups and co-growing context-sensitive groups. The paper closes with a discussion of various open problems.

2. Church-Rosser Languages and Growing Context-Sensitive Languages

Let Σ be a finite alphabet. Then Σ^* denotes the set of strings over Σ including the empty string ε, and $\Sigma^+ := \Sigma^* \smallsetminus \{\varepsilon\}$. A function $\varphi : \Sigma \to \mathbb{N}_+$

is called a *weight-function.* Its extension to Σ^*, which we will also denote by φ, is defined inductively through $\varphi(\varepsilon) := 0$ and $\varphi(wa) := \varphi(w) + \varphi(a)$ for all $w \in \Sigma^*$ and $a \in \Sigma$. A particular weight-function is the *length-function* $|\,.\,| : \Sigma \to \mathbb{N}_+$, which assigns each letter the weight (length) 1.

A *string-rewriting system* R on Σ is a subset of $\Sigma^* \times \Sigma^*$. An element $(\ell, r) \in R$ is called a *rewrite rule* or simply a *rule*, and it will usually be written as $(\ell \to r)$. In this paper we will only be dealing with finite string-rewriting systems.

The string-rewriting system R induces several binary relations on Σ^*, the simplest of which is the *single-step reduction relation* $\to_R := \{(u\ell v, urv) \mid u, v \in \Sigma^*, (\ell \to r) \in R\}$. Its reflexive and transitive closure is the *reduction relation* $\to_R^*$ induced by R, and its reflexive, symmetric, and transitive closure $\leftrightarrow_R^*$ is the *Thue congruence* generated by R. If $u \to_R^* v$, then u is an *ancestor* of v, and v is a *descendant* of u. By $\Delta_R^*(u)$ we denote the set of all descendants of u, and for a set $S \subseteq \Sigma^*$, $\Delta_R^*(S) = \bigcup_{u \in S} \Delta_S^*(u)$. If there is no $v \in \Sigma^*$ such that $u \to_R v$ holds, then the string u is called *irreducible* (mod R). By $\mathsf{IRR}(R)$ we denote the set of all irreducible strings. If R is finite, then $\mathsf{IRR}(R)$ is obviously a regular language. The string-rewriting system R is called

– *length-reducing* if $|\ell| > |r|$ holds for each rule $(\ell \to r) \in R$,
– *weight-reducing* if there exists a weight-function φ such that $\varphi(\ell) > \varphi(r)$ holds for each rule $(\ell \to r) \in R$,
– *confluent* if, for all $u, v, w \in \Sigma^*$, $u \to_R^* v$ and $u \to_R^* w$ imply that v and w have a common descendant.

If a string-rewriting system R is weight-reducing, then reduction sequences mod R are linearly bounded in length, that is, if $w_0 \to_R w_1 \to_R \cdots \to_R w_m$, then $m \leq \varphi(w_0)$. If, in addition, R is confluent, then each string $w \in \Sigma^*$ has a unique irreducible descendant $\hat{w} \in \mathsf{IRR}(R)$. Actually, in this situation $u \leftrightarrow_R^* v$ if and only if $\hat{u} = \hat{v}$. Since $\hat{u}$ can be determined from u in linear time, this shows that the Thue congruence $\leftrightarrow_R^*$ is decidable in linear time for each finite, weight-reducing, and confluent string-rewriting system.

Definition 2.1. *A language $L \subseteq \Sigma^*$ is a* Church-Rosser language (CRL) *if there exist an alphabet Γ properly containing Σ, a finite, length-reducing, and confluent string-rewriting system R on Γ, two strings $t_1, t_2 \in (\Gamma \smallsetminus \Sigma)^* \cap \mathsf{IRR}(R)$, and a letter $Y \in (\Gamma \smallsetminus \Sigma) \cap \mathsf{IRR}(R)$ such that, for all $w \in \Sigma^*$, $t_1 w t_2 \to_R^* Y$ if and only if $w \in L$.*

Instead of using length-reducing string-rewriting systems, one can also use weight-reducing string-rewriting systems in the definition above [17]. By CRL we denote the class of Church-Rosser languages.

Definition 2.2. *A language $L \subseteq \Sigma^*$ is* growing context-sensitive *if it is generated by a phrase-structure grammar $G = (N, \Sigma, S, P)$ that satisfies the following restrictions:*

(1) *The initial nonterminal S does not occur on the right-hand side of any production of G.*
(2) *For each production $(\ell, r) \in P$, if $\ell \neq S$, then $|\ell| < |r|$.*

As shown in [7], the membership problem for a growing context-sensitive language is solvable in deterministic polynomial time. By GCSL we denote the class of growing context-sensitive languages. It is shown in [5] that GCSL is an abstract family of languages, that is, it is closed under union, product, Kleene plus, intersection with regular languages, inverse morphisms, and ε-free morphisms. However, the class GCSL is not closed under projections or complement.

Next we introduce a machine model that yields characterizations for the language classes CRL and GCSL.

Definition 2.3. *A* two-pushdown automaton (TPDA) *with* pushdown windows of size k *is a nondeterministic automaton with two pushdown stores. Formally, it is defined as a 9-tuple* $M = (Q, \Sigma, \Gamma, \delta, q_0, \perp, t_1, t_2, F)$*, where*

- Q *is a finite set of states,*
- Σ *is a finite input alphabet,*
- Γ *is a finite tape alphabet with* $\Gamma \supsetneq \Sigma$ *and* $\Gamma \cap Q = \emptyset$*,*
- $q_0 \in Q$ *is an initial state,*
- $\perp \in \Gamma \smallsetminus \Sigma$ *is the bottom marker of the pushdown stores,*
- $t_1, t_2 \in (\Gamma \smallsetminus \Sigma)^*$ *are preassigned contents of the left/right pushdown store, respectively,*
- $F \subseteq Q$ *is a set of final (or halting) states, and*
- $\delta\colon Q \times {}_\perp\Gamma^{\leq k} \times \Gamma_\perp^{\leq k} \to \mathfrak{P}_{\mathit{fin}}(Q \times \Gamma^* \times \Gamma^*)$ *is a transition relation, where* ${}_\perp\Gamma^{\leq k} := \Gamma^k \cup \{\perp u \mid |u| \leq k-1\}$, $\Gamma_\perp^{\leq k} := \Gamma^k \cup \{v\perp \mid |v| \leq k-1\}$, *and* $\mathfrak{P}_{\mathit{fin}}(Q \times \Gamma^* \times \Gamma^*)$ *denotes the set of finite subsets of* $Q \times \Gamma^* \times \Gamma^*$.

The automaton M is a deterministic two-pushdown automaton (DTPDA)*, if δ is a (partial) function from* $Q \times {}_\perp\Gamma^{\leq k} \times \Gamma_\perp^{\leq k}$ *into* $Q \times \Gamma^* \times \Gamma^*$.

A configuration of a (D)TPDA is described as uqv, where $q \in Q$ is the actual state, $u \in \Gamma^*$ is the contents of the first pushdown store with the first letter of u at the bottom, and $v \in \Gamma^*$ is the contents of the second pushdown store with the first letter of v at the top. For an input string $w \in \Sigma^*$, the corresponding initial configuration is $\perp t_1 q_0 w t_2 \perp$. The (D)$\mathsf{TPDA}$ M induces a computation relation $\vdash_M^*$ on the set of configurations, which is the reflexive transitive closure of the single-step computation relation $\vdash_M$ (see, e.g., [12]). The (D)TPDA M accepts with empty pushdown stores, that is, the *language $N(M)$ accepted by M* is defined as $N(M) := \{ w \in \Sigma^* \mid \perp t_1 q_0 w t_2 \perp \ \vdash_M^* q \text{ for some } q \in F \}$.

Definition 2.4. *A* (D)TPDA *is* shrinking *if there exists a weight function* $\varphi : Q \cup \Gamma \to \mathbb{N}_+$ *such that, for all* $q \in Q$, $u \in {}_\perp\Gamma^{\leq k}$, *and* $v \in \Gamma_\perp^{\leq k}$, $(p, u', v') \in \delta(q, u, v)$ *implies that* $\varphi(u'pv') < \varphi(uqv)$. *By* sTPDA *and* sDTPDA *we denote the corresponding classes of shrinking automata.*

A (D)TPDA *is* length-reducing *if, for all* $q \in Q$, $u \in {}_\perp\Gamma^{\leq k}$, *and* $v \in \Gamma_\perp^{\leq k}$, $(p, u', v') \in \delta(q, u, v)$ *implies* $|u'v'| < |uv|$. *We denote the corresponding classes of length-reducing automata by* lrTPDA *and* $\mathsf{lrDTPDA}$*, respectively.*

Thus, if M is a shrinking TPDA with weight-function φ, then $\varphi(u_1q_1v_1) > \varphi(u_2q_2v_2)$ holds for all configurations $u_1q_1v_1$ and $u_2q_2v_2$ of M that satisfy $u_1q_1v_1 \vdash_M u_2q_2v_2$. If M is a length-reducing TPDA, then $|u_1q_1v_1| > |u_2q_2v_2|$ holds for all configurations $u_1q_1v_1$ and $u_2q_2v_2$ of M that satisfy $u_1q_1v_1 \vdash_M u_2q_2v_2$. Obviously, the length-reducing TPDA is a special case of the shrinking TPDA.

Observe that the input is provided to a TPDA as the initial contents of its second pushdown store, and that in order to accept a TPDA is required to empty its pushdown stores. Thus, it is forced to consume its input completely. The main results of [6] and [17] state the following. Here we use the notation $\mathcal{L}(\mathcal{A})$ to denote the class of languages that are accepted by automata from class $\mathcal{A}$.

Theorem 2.5. (a) $\mathsf{CRL} = \mathcal{L}(\mathsf{lrDTPDA}) = \mathcal{L}(\mathsf{sDTPDA})$.
(b) $\mathsf{GCSL} = \mathcal{L}(\mathsf{lrTPDA}) = \mathcal{L}(\mathsf{sTPDA})$.

Theorem 2.6. (a) *The class* CRL *is closed under reversal, intersection with regular languages, complement, and inverse morphisms.*
(b) *The class* CRL *is not closed under union, intersection, product, Kleene plus, or (ε-free) morphisms.*

In particular, $\mathsf{DCFL} \subset \mathsf{CRL} \subset \mathsf{GCSL}$, while CRL is incomparable under inclusion to the class CFL of context-free languages.

Next we observe that the requirement that the string-rewriting system used in specifying a Church-Rosser language must be confluent can be relaxed somewhat. We say that a string-rewriting system R on Γ is *confluent on* a set $S \subseteq \Gamma^*$, if R is confluent on $\Delta_R^*(S)$, that is, for all $w \in \Delta_R^*(S)$, if $w \to_R^* u$ and $w \to_R^* v$, then u and v have a common descendant mod R.

Theorem 2.7. *Let $L \subseteq \Sigma^*$ be a language, and suppose that there exist a finite, length-reducing string-rewriting system R on some alphabet Γ properly containing Σ, two strings $t_1, t_2 \in (\Gamma \smallsetminus \Sigma)^* \cap \mathsf{IRR}(R)$ and a letter $Y \in (\Gamma \smallsetminus \Sigma) \cap \mathsf{IRR}(R)$ such that*

(1) *R is confluent on the set $t_1 \cdot L \cdot t_2$; and*
(2) *$w \in L$ if and only if $t_1 w t_2 \to_R^* Y$.*

Then L is a Church-Rosser language.

Proof. A string-rewriting system R' is called *normalized* (or *interreduced*) if $r \in \mathsf{IRR}(R')$ and $\ell \in \mathsf{IRR}(R' \smallsetminus \{\ell \to r\})$ hold for each rule $\ell \to r$ of R'. First, we construct from R a normalized string-rewriting system R' as follows. If the right-hand side of a production in R is reducible, then we replace it with one of its irreducible descendants. If two or more productions in R have the same left-hand side, then we discard (arbitrarily) all but one of them. If two productions are such that the left-hand side of one is a factor of the left-hand side of the other, then we discard the one with the longer left-hand side.

Clearly, R' is length-reducing and normalized, and the reduction relation $\to_{R'}^*$ is contained in the reduction relation of R. Thus, for any word $w \in \Gamma^*$, if $w \to_{R'}^* Y$, then also $w \to_R^* Y$ holds. Further, a word is reducible mod R' if and only if it is reducible mod R, that is, the sets of irreducible words $\mathsf{IRR}(R')$ and $\mathsf{IRR}(R)$ coincide. Confluence of R on the set $t_1 \cdot L \cdot t_2$ ensures that Y is the only irreducible descendant for each word from the set $\Delta_R^*(t_1 \cdot L \cdot t_2)$. It follows that, for all $w \in \Sigma^*$, $w \in L$ if and only if $t_1 w t_2 \to_{R'}^* Y$ holds.

The proofs of [6, Theorems 4.5 and 4.6] now show that the language L is accepted by an sDTPDA. It follows by Theorem 2.5 (a) that L is a Church-Rosser language. □

Below we will need some additional closure properties of CRL and GCSL. A *generalized sequential machine* (GSM) is a deterministic finite-state automaton that is equipped with an output. With a GSM A with input alphabet Γ and output alphabet Ω, we associate the GSM-mapping $\psi_A : \Gamma^* \to \Omega^*$ that assigns, to each word $w \in \Gamma^*$, the output string $\psi_A(w)$ that A produces when processing the input w. By ψ_A^{-1} we denote the corresponding inverse mapping.

Proposition 2.8. (a) *The class* CRL *is closed under union with regular languages.*

(b) CRL *and* GCSL *are closed under inverse* GSM*-mappings.*

Proof. (a) Let L be a Church-Rosser language over Σ, and let E be a regular language over Σ. From a sDTPDA M with weight function φ for L we construct a sDTPDA M' with weight function φ' for the language $L' := L \cup E$. The machine M' has the same input alphabet Σ as M, but its tape alphabet contains two additional copies $\hat{\Sigma}$ and $\tilde{\Sigma}$ of Σ. The weight function φ' is defined by taking $\varphi'(a) := \varphi(a) + 2$, $\varphi'(\hat{a}) := \varphi(a) + 1$, $\varphi'(\tilde{a}) := \varphi(a)$ for all $a \in \Sigma$ and $\varphi'(b) := \varphi(b)$ for all non-input symbols of M. On input $w \in \Sigma^*$, M' shifts w from the right-hand pushdown to the left-hand pushdown, replacing each symbol $a \in \Sigma$ by the corresponding symbol $\hat{a}$. While doing so it checks whether $w \in E$ holds. In the affirmative, it halts and accepts, otherwise it shifts the string $\hat{w}$ back to the right-hand pushdown, replacing each symbol $\hat{a}$ by the corresponding symbol $\tilde{a}$. Then it simulates the computation of M on input w. It follows that $L(M') = L'$, and it is obvious that M' is shrinking with respect to the weight function φ'.

(b) Let L be a Church-Rosser language over Σ, and let A be a GSM with input alphabet Γ and output alphabet Σ. From a sDTPDA M with weight function φ for the language L and the GSM A we construct a sDTPDA M' with weight function φ' for the language $\psi_A^{-1}(L) = \{\, w \in \Gamma^* \mid \psi_A(w) \in L \,\}$. Without loss of generality we can assume that Σ and Γ are disjoint. The machine M' has input alphabet Γ, and its tape alphabet is the union of the tape alphabet of M with the input alphabet Γ and a copy $\hat{\Sigma}$ of Σ. The weight function φ' assigns to each symbol of the tape alphabet of M the same weight as the function φ, to each symbol $\hat{a} \in \hat{\Sigma}$ it assigns the weight $\varphi(a)+1$, and to each symbol $b \in \Gamma$ it assigns the weight $\varphi(\psi_A(b)) + |\psi_A(b)| + 1$.

Given an input $w \in \Gamma^*$, M' replaces w by the string $\psi_A(w)$ as follows. First it reads w from the right-hand pushdown, replacing each symbol $b \in \Gamma$ by the string $\hat{u} \in \hat{\Sigma}^*$ corresponding to $u := \psi_A(b)$. Then it shifts the contents of the left-hand pushdown to the right-hand pushdown, replacing each symbol $\hat{a}$ by the symbol $a \in \Sigma$. Now M' is in the start configuration of M for input $\psi_A(w)$, that is, M' can now simulate M on this input. It follows that M' accepts the language $\psi_A^{-1}(L)$, and it is verified easily that M' is shrinking with respect to the weight function φ'.

The proof for GCSL is completely analogous. □

3. Generalized Dehn Algorithms

In [10] Goodman and Shapiro study *generalized Dehn algorithms* for the word problem of finitely generated groups. These are reduction algorithms of a special form. As their definition does not refer to groups at all, we can describe them in general language theoretic terms. In the following we assume that, for each rewriting system considered, no two rules have the same left-hand side, that is, each rule is determined by its left-hand side.

Let Γ be a finite alphabet, let ¢ and \$ be two symbols not in Γ, let R_{in} be a finite string-rewriting system over Γ, and let $R_{pre} \subset (¢ \cdot \Gamma^+) \times (¢ \cdot \Gamma^*)$ be another finite rewriting system of *prefix rules.* We assume that in the combined system $R := R_{in} \cup R_{pre}$ each rule is length-reducing. The *incremental reduction relation* $\to_{inc}^*$ induced by R is the reflexive transitive closure of the single-step reduction $\to_{inc}$ on $¢ \cdot \Gamma^*$ that is defined as follows:

$$¢ \cdot u \to_{inc} ¢ \cdot v$$
$$\text{if and only if}$$
$$¢ \cdot u = x\ell y \text{ and } ¢ \cdot v = xry \text{ for some } x \in ¢ \cdot \Gamma^* \cup \{\varepsilon\},\ y \in \Gamma^*,\ (\ell, r) \in R$$
$$\text{such that } |x\ell| \text{ is minimal, and } |\ell| \text{ is maximal with these properties.}$$

Essentially this reduction relation coincides with the *leftmost reductions* considered in [3], the only difference being that here prefix-rules are considered as well. As the left-hand side uniquely determines the rule according to the assumptions above, the reduction relation $\to_{inc}$ is deterministic. Accordingly, we can associate with the system R a function $R_{inc} : \Gamma^* \to \Gamma^*$ that associates with a word $w \in \Gamma^*$ the word $\hat{w}$ such that $¢ \cdot \hat{w}$ is the unique irreducible descendant of $¢ \cdot w$ mod $\to_{inc}$. From the definition of $\to_{inc}$ it follows immediately that $R_{inc}(uv) = R_{inc}(R_{inc}(u)v)$ holds for all words $u, v \in \Gamma^*$.

Definition 3.1. *A language $L \subseteq \Sigma^*$ admits a* generalized Dehn algorithm *if there exist an alphabet Γ containing Σ and a rewriting system R over Γ of the form described above such that $L = \{ w \in \Sigma^* \mid R_{inc}(w) = \varepsilon \}$. By* GDL *we denote the class of all languages that admit a generalized Dehn algorithm.*

From the incremental property of R_{inc} we obtain the following property of languages that admit generalized Dehn algorithms.

Lemma 3.2. *Let $L \subseteq \Sigma^*$ admit a generalized Dehn algorithm.*

(1) *If $u, v \in L$, then also $uv \in L$.*
(2) *If $u, v \in L$ satisfying $u = vw$, then also $w \in L$.*

Proof. Let R be the rewriting system on which the generalized Dehn algorithm for L is based.

1. By the observation above, $R_{inc}(uv) = R_{inc}(R_{inc}(u)v) = R_{inc}(v) = \varepsilon$, if $u, v \in L$. Hence, also $uv \in L$.

2. If $u = vw$, then $R_{inc}(u) = R_{inc}(vw) = R_{inc}(R_{inc}(v)w)$. As $u, v \in L$, $R_{inc}(u) = \varepsilon = R_{inc}(v)$, which implies that $R_{inc}(w) = R_{inc}(R_{inc}(v)w) = R_{inc}(u) = \varepsilon$. Thus, $w \in L$. □

It follows that a non-empty language that admits a generalized Dehn algorithm is necessarily infinite.

Proposition 3.3. *If L admits a generalized Dehn algorithm, then L is a Church-Rosser language.*

Proof. It follows immediately from the definition of $\to_{inc}$ that the relation R_{inc} can be computed by a DTPDA (see, e.g., the proof of Theorem 2.2.9

(p. 44) of [3]). As each rule of the system R is length-reducing, the DTPDA can be realized in a weight-reducing fashion. The result now follows from Theorem 2.5. □

Theorem 3.4. (1) *There exists an infinite regular language L_r for which there is no generalized Dehn algorithm.*
(2) *There exists a language $L_d \in \mathsf{GDL}$ that is not context-free.*

Proof. 1. The infinite regular language $L_r := a \cdot b^+$ does not satisfy the properties of Lemma 3.2. Hence, $L_r \notin \mathsf{GDL}$.
2. Consider the language $L_d := \{ a^{2^{n_1}} b a^{2^{n_2}} b \cdots a^{2^{n_k}} b \mid n_1, n_2, \ldots, n_k \geq 0,\ k \geq 0 \}$. As L_d is not semilinear, it is not context-free. On the other hand, L_d admits a generalized Dehn algorithm which is specified by the following rewriting system on the alphabet $\Gamma := \{a, b, A\}$: $aab \to Ab$, $AAb \to ab$, $aaA \to AA$, $AAa \to aa$, $¢Ab \to ¢$, $¢ab \to ¢$. □

These results have the following consequences.

Corollary 3.5. (1) *The class of languages* GDL *that admit a generalized Dehn algorithm is incomparable to* REG *as well as to* CFL.
(2) $\mathsf{GDL} \subset \mathsf{CRL}$.

Goodman and Shapiro also introduce a second variant of generalized Dehn algorithms. Let R_{in} and R_{pre} be as above, let $R_{suf} \subset (\Gamma^+ \cdot \$) \times (\Gamma^* \cdot \$)$ be another finite rewriting system, and let $R_{fin} \subset (¢ \cdot \Gamma^+ \cdot \$) \times (¢ \cdot \Gamma^* \cdot \$)$ be a forth finite rewriting system. Again we assume that each rule of the combined system $R := R_{in} \cup R_{pre} \cup R_{suf} \cup R_{fin}$ is length-reducing.

The *non-incremental reduction relation* $\to_{non}^*$ induced by R is the reflexive transitive closure of the single-step reduction $\to_{non}$ on $¢ \cdot \Gamma^* \cdot \$$ that is defined as follows:

$$u \to_{non} v$$
if and only if
$$u = x\ell y \text{ and } v = xry \text{ for some } x \in ¢ \cdot \Gamma^* \cup \{\varepsilon\},\ y \in \Gamma^* \cdot \$ \cup \{\varepsilon\},\ (\ell, r) \in R$$
such that $|x|$ is minimal, and $|x\ell|$ is maximal with respect to these conditions.

Also this reduction relation is deterministic. Accordingly, we can associate with the system R a function $R_{non} : \Gamma^* \to \Gamma^*$ that associates with a word $w \in \Gamma^*$ the word $\hat{w}$ such that $¢ \cdot \hat{w} \cdot \$$ is the unique irreducible descendant of $¢ \cdot w \cdot \$$ mod $\to_{non}$.

Definition 3.6. *A language $L \subseteq \Sigma^*$ admits a* non-incremental Dehn algorithm *if there exist an alphabet Γ containing Σ and a rewriting system R over Γ of the form described above such that $L = \{ w \in \Sigma^* \mid R_{non}(w) = \varepsilon \}$.*

The following result, which is the main result of this section, relates non-incremental Dehn algorithms to Church-Rosser languages.

Theorem 3.7. *A language L admits a non-incremental Dehn algorithm if and only if L is a Church-Rosser language.*

Proof. As in the case of the reduction relation $\to_{inc}$ it is easily seen that the relation $\to_{non}$ can be computed by a shrinking DTPDA. Thus, if L admits a non-incremental Dehn algorithm, then it is a Church-Rosser language.

Conversely, assume that $L \subseteq \Sigma^*$ is a Church-Rosser language. Then there exists a lrDTPDA $M = (Q, \Sigma, \Gamma, \delta, q_0, \bot, t_1, t_2, F)$ that accepts the language L. Hence, if a word $w \in \Sigma^*$ belongs to L, then $\bot q_0 w \bot \vdash_M^* q_f$ for some final state q_f, and if w does not belong to L, then $\bot q_0 w \bot \vdash_M^* \bot q_f$ for some final state q_f. As M cannot execute any transition steps once a pushdown store is empty, we see that the removal of the bottom marker from one or both pushdowns must be done in the last step of the corresponding computation. Accordingly, we can even assume that M has a single final state q_f only, and that M contains certain final transitions that remove bottom markers and enter this final state. By Lemma 3.4 of [17] we can assume that $t_1 = t_2 = \varepsilon$. Further, according to Lemma 3.2.8 of [16] we can assume that each step of M that does not yet yield a halting configuration reduces the combined length of the contents of the two pushdowns by at least 2. In addition, we may assume that the initial state q_0 is not entered by any transition of M.

With M we now associate a string-rewriting system R_M. Let k be a constant such that, for each word $w \in L$ of length exceeding k, M executes at least 3 transition steps. We take

$$R_0 := \{\, \bar{\bot} u \bot \to \bar{\bot}\bot \mid w \in L,\ |w| \leq k \,\}.$$

Next we introduce rules for simulating the DTPDA M in the standard way, using a copy alphabet $\overline{\Gamma}$ for describing the contents of the lefthand pushdown, and prolonging those rules that involve the initial state q_0 sufficiently to make sure that the left-hand side $\bar{\bot} q_0 u$ $(u \in \Gamma^*)$ of each of these rules has length at least $k+3$. The system R_M is finite and length-reducing. It simulates M until one of the final transitions (see above) becomes applicable. Further, $|\ell| \geq |r| + 2$ for each rule of R_M involving the initial state q_0. As R_M does not have any critical pairs, it is confluent. Finally, for all $w \in \Sigma^*$ satisfying $|w| > k$, $\bar{\bot} q_0 w \bot \to_{R_M}^* \bar{\bot} \bar{u} q v \bot$ for some words $u, v \in \Gamma^*$ and some state q such that in the configuration $\bot u q v \bot$ a final transition of M is applicable. Now we take

$$\begin{aligned} R := R_0 \cup R_M \quad & \cup \quad \{\, \bar{\bot} u \to \bar{\bot} v \mid (\bar{\bot} q_0 u \to \bar{\bot} v) \in R_M \,\} \\ & \cup \quad \{\, \bar{\bot} \bar{u} q v \bot \to \bar{\bot}\bot \mid \bot u q v \bot \vdash_M q_f \,\}. \end{aligned}$$

Then R is length-reducing and confluent. Further, for all $w \in \Sigma^*$, $\bar{\bot} w \bot \to_R^* \bar{\bot}\bot$ if and only if $w \in L$. As R is confluent, and as a word is reducible mod $\to_R$ if and only if it is reducible mod $\to_{non}$, it follows that we can restrict $\to_R$ to $\to_{non}$ in the above characterization, implying that $L = \{\, w \in \Sigma^* \mid R_{non}(w) = \varepsilon \,\}$. Thus, L admits a non-incremental Dehn algorithm. $\square$

4. Groups

Let Σ be a finite alphabet, let $\overline{\Sigma}$ be a set of formal inverses that is in one-to-one correspondence to Σ, where this correspondence is expressed by the mapping $^- : \Sigma \to \overline{\Sigma}$, and let $\underline{\Sigma} := \Sigma \cup \overline{\Sigma}$. A *group presentation* $\langle \Sigma; R \rangle$ defines the group G that is given through the string-rewriting system $S(R) := \{\, u \to \varepsilon \mid u \in R \,\} \cup \{\, a\overline{a} \to \varepsilon, \overline{a}a \to \varepsilon \mid a \in \Sigma \,\}$. Thus, G is isomorphic to the quotient of the free monoid generated by $\underline{\Sigma}$ modulo the Thue congruence $\leftrightarrow_{S(R)}^*$. For simplicity we will write $u =_G v$ for $u \leftrightarrow^*_{S(R)} v$, and we use 1 for the group identity (which is obviously presented by ε).

A group G is called *finitely generated* if G has a group presentation with a finite generating set, and G is called *finitely presented* if it has a finite group presentation. With the group presentation $\langle \Sigma; R\rangle$ we associate the following languages:

(1) the *word problem* $\mathrm{WP}(G, \Sigma) := \{ w \in \underline{\Sigma}^* \mid w =_G 1 \}$, and
(2) the *co-word problem* $\text{co-WP}(G, \Sigma) := \underline{\Sigma}^* \smallsetminus \mathrm{WP}(G, \Sigma)$.

A group G is called *context-free* if it admits a finitely generated group presentation such that the corresponding language $\mathrm{WP}(G, \Sigma)$ is context-free, and it is called *co-context-free* if it admits a finitely generated group presentation such that $\text{co-WP}(G, \Sigma)$ is context-free. Analogously, *context-sensitive* and *co-context-sensitive* groups are defined. By CFG, $\mathsf{co\text{-}CFG}$, and CSG we denote the class of context-free, co-context-free, and context-sensitive groups, respectively. Observe that $\mathrm{WP}(G, \Sigma)$ is context-sensitive if and only if $\text{co-WP}(G, \Sigma)$ is, as the class CSL of context-sensitive languages is closed under complement [13]. Thus, we will not talk about co-context-sensitive groups anymore.

As the classes of context-free and context-sensitive languages are closed under inverse morphisms, it follows that the properties of being a context-free group, a co-context-free group, or a context-sensitive group are independent of the chosen finitely generated presentation. In combination with the results of [8], the main result of [15] shows that a group is context-free if and only if it is a finitely generated virtually free group. Further, it turned out that the language $\mathrm{WP}(G, \Sigma)$ of a context-free group is in fact an NTS-language, and so it is a congruential language that is deterministic context-free [2].

Concerning co-context-free groups it has been shown that the class $\mathsf{co\text{-}CFG}$ is closed under the operations of passing to finitely generated subgroups, of passing to finite index overgroups and under finite direct products. Further, a finitely generated nilpotent group, a Baumslag-Solitar group or a polycyclic group is co-context-free if and only if it is virtually abelian. Finally, every co-context-free group has a solvable order problem [11]. We see in particular that the proper inclusion $\mathsf{CFG} \subset \mathsf{co\text{-}CFG}$ holds.

5. Church-Rosser Groups

A group G is called a *Church-Rosser group* (*growing context-sensitive group*) if it admits a finitely generated group presentation such that the corresponding language $\mathrm{WP}(G, \Sigma)$ is Church-Rosser (growing context-sensitive). As CRL and GCSL are both closed under inverse morphisms, the properties of being Church-Rosser and of being growing context-sensitive are independent of the chosen finitely generated presentation. By GCSG and CRG we denote the corresponding classes of groups. A group G is *co-Church-Rosser* (*co-growing context-sensitive*) if it admits a finitely generated presentation such that the language $\text{co-WP}(G, \Sigma)$ is Church-Rosser (growing context-sensitive). Again these properties are independent of the chosen finitely generated presentation (see [11] Lemma 1). By $\mathsf{co\text{-}CRG}$ and $\mathsf{co\text{-}GCSG}$ we denote the corresponding classes of groups. From the results above we immediately obtain the following inclusions.

Proposition 5.1. (a) $\mathsf{CFG} \subseteq \mathsf{CRG} = \text{co-}\mathsf{CRG} \subseteq \mathsf{GCSG}$.
(b) $\text{co-}\mathsf{CFG} \cup \mathsf{CRG} \subseteq \text{co-}\mathsf{GCSG}$.

Next we present an example that will imply that the first of these inclusions is proper. Let A_2 be the free abelian group of rank 2, that is, A_2 is given through the presentation $\langle \Sigma_2; ab = ba \rangle$, where $\Sigma_2 := \{a, b\}$. Thus,

$$\mathrm{WP}(A_2, \Sigma_2) = \{\, w \in \underline{\Sigma}_2^* \mid |w|_a = |w|_{\bar{a}} \text{ and } |w|_b = |w|_{\bar{b}} \,\}.$$

As $\mathrm{WP}(A_2, \Sigma_2) \cap a^* \cdot b^* \cdot \bar{a}^* \cdot \bar{b}^* = \{\, a^n b^m \bar{a}^n \bar{b}^m \mid n, m \geq 0 \,\}$, it follows that $\mathrm{WP}(A_2, \Sigma_2)$ is not context-free, that is, A_2 is not a context-free group. However, co-$\mathrm{WP}(A_2, \Sigma_2)$ is context-free, implying that A_2 is a co-context-free group. The following proposition is actually a special case of a much more general result (see Theorem 5.9), but we present it here including a proof to illustrate the workings of the length-reducing DTPDA.

Proposition 5.2. *A_2 is a Church-Rosser group.*

Proof. We will present a $\mathsf{lrDTPDA}$ M for the language $\mathrm{WP}(A_2, \Sigma_2)$. Its tape alphabet is $\underline{\Sigma}_2$, and its window size is $k := 4$. Alternatingly it will shift the contents of the right-hand pushdown to the left-hand pushdown and back, shortening the combined contents in each step. Internally it will use four binary indicators that store the parities $p_a := |w|_a \mod 2$, $p_{\bar{a}} := |w|_{\bar{a}} \mod 2$, $p_b := |w|_b \mod 2$, and $p_{\bar{b}} := |w|_{\bar{b}} \mod 2$. If at the end of a round the parities p_a and $p_{\bar{a}}$ do not coincide, or if the parities p_b and $p_{\bar{b}}$ do not coincide, then M halts and rejects; otherwise the next round is executed. M accepts if it reaches a configuration in which the combined contents of the pushdowns is a word from $\mathrm{WP}(A_2, \Sigma_2)$ of length at most eight. Below we give a detailed description for those transition steps that shift the contents from the right-hand to the left-hand pushdown store; the description of the converse shift is completely symmetric to this one. The current state of M is described as a 4-tuple giving the values of the indicators $(p_a, p_{\bar{a}}, p_b, p_{\bar{b}})$. Here is the description of the transition function of M, where $u \in {}_\perp\Gamma^{\leq k}$, $p_i \in \{0, 1\}$, $\bar{p}_i := 1 - p_i$ $(1 \leq i \leq 4)$, $c^1 := c$, $c^{-1} := \bar{c}$ for all $c \in \{a, b\}$, and

$\mu, \nu \in \{1, -1\}$;

$$
\begin{array}{llll}
(1.) & ((p_1,p_2,p_3,p_4),u,cccc) & \to & ((p_1,p_2,p_3,p_4),ucc,\varepsilon) \quad \text{for all } c \in \underline{\Sigma}_2, \\
(2.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,p_2,p_3,p_4),ua^{\mu}b^{\nu},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{a^{\mu}} = 2 \text{ and } |v|_{b^{\nu}} = 2, \\
(3.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,p_2,p_3,p_4),u,\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{a} = 2 \text{ and } |v|_{\bar{a}} = 2, \\
(4.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,p_2,p_3,p_4),u,\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{b} = 2 \text{ and } |v|_{\bar{b}} = 2, \\
(5.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,p_2,p_3,p_4),u,\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{a} = |v|_{\bar{a}} = |v|_{b} = |v|_{\bar{b}} = 1, \\
(6.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,p_2,p_3,p_4),ua^{\mu},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{a^{\mu}} = 2 \text{ and } |v|_{b} = |v|_{\bar{b}} = 1, \\
(7.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,p_2,p_3,p_4),ub^{\nu},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{b^{\nu}} = 2 \text{ and } |v|_{a} = |v|_{\bar{a}} = 1, \\
(8.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,p_2,p_3,p_4),ua^{\mu},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{a^{\mu}} = 3 \text{ and } |v|_{a^{-\mu}} = 1, \\
(9.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,p_2,p_3,p_4),ub^{\nu},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{b^{\nu}} = 3 \text{ and } |v|_{b^{-\nu}} = 1, \\
(10.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((\bar{p}_1,p_2,\bar{p}_3,p_4),ua^{2-\bar{p}_1}b^{1-\bar{p}_3},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{a} = 3 \text{ and } |v|_{b} = 1, \\
(11.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((\bar{p}_1,p_2,p_3,\bar{p}_4),ua^{2-\bar{p}_1}\bar{b}^{1-\bar{p}_4},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{a} = 3 \text{ and } |v|_{\bar{b}} = 1, \\
(12.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,\bar{p}_2,\bar{p}_3,p_4),u\bar{a}^{2-\bar{q}_2}b^{1-\bar{p}_3},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{\bar{a}} = 3 \text{ and } |v|_{b} = 1, \\
(13.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,\bar{p}_2,p_3,\bar{p}_4),u\bar{a}^{2-\bar{p}_2}\bar{b}^{1-\bar{p}_4},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{\bar{a}} = 3 \text{ and } |v|_{\bar{b}} = 1, \\
(14.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((\bar{p}_1,p_2,\bar{p}_3,p_4),ua^{1-\bar{p}_1}b^{2-\bar{p}_3},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{b} = 3 \text{ and } |v|_{a} = 1, \\
(15.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,\bar{p}_2,\bar{p}_3,p_4),u\bar{a}^{1-\bar{p}_2}b^{2-\bar{p}_3},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{b} = 3 \text{ and } |v|_{\bar{a}} = 1, \\
(16.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((\bar{p}_1,p_2,p_3,\bar{p}_4),ua^{1-\bar{p}_1}\bar{b}^{2-\bar{p}_4},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{\bar{b}} = 3 \text{ and } |v|_{a} = 1, \\
(17.) & ((p_1,p_2,p_3,p_4),u,v) & \to & ((p_1,\bar{p}_2,p_3,\bar{p}_4),u\bar{a}^{1-\bar{p}_2}\bar{b}^{2-\bar{p}_4},\varepsilon) \quad \text{for all } v \in \underline{\Sigma}_2^4 \\
 & & & \text{satisfying } |v|_{\bar{b}} = 3 \text{ and } |v|_{\bar{a}} = 1,
\end{array}
$$

Starting from a configuration of the form $\perp(0,0,0,0)w\perp$, M will eventually reach a configuration of the form $\perp u(p_1,p_2,p_3,p_4)\perp$. From the form of the transition steps it follows that there exist nonnegative integers j_a and j_b such that u satisfies the following conditions:

$$
\begin{array}{rcl}
2 \cdot |u|_a + p_1 + j_a & = & |w|_a, \\
2 \cdot |u|_{\bar{a}} + p_2 + j_a & = & |w|_{\bar{a}}, \\
2 \cdot |u|_b + p_3 + j_b & = & |w|_b, \\
2 \cdot |u|_{\bar{b}} + p_4 + j_b & = & |w|_{\bar{b}}.
\end{array}
$$

Here j_a is the number of cancellations $a^{\mu}a^{-\mu}$ that took place during the above transitions, and j_b is the corresponding number of cancellations $b^{\nu}b^{-\nu}$. Thus, $w \in \mathrm{WP}(A_2, \Sigma_2)$ if and only if $p_1 = p_2$, $p_3 = p_4$, and $u \in \mathrm{WP}(A_2, \Sigma_2)$.

It follows that the $\mathsf{lrDTPDA}$ M accepts the language $\mathrm{WP}(A_2, \Sigma_2)$, which implies that A_2 is indeed a Church-Rosser group. □

As A_2 is not a context-free group, this yields the following consequence.

Corollary 5.3. $\mathsf{CFG} \subset \mathsf{CRG}$.

The proof of Proposition 5.2 easily extends to free abelian groups of any finite rank. Because of Proposition 2.8, Lemma 5 of [11] implies the following closure property for CRG.

Proposition 5.4. *The class* CRG *of Church-Rosser groups is closed under the operation of passing to finite index overgroups.*

As a consequence we see that the class CRG of all Church-Rosser groups contains all finitely generated virtually abelian groups. On the other hand, Lemma 2 of [11] implies the following closure property for CRG, as CRL is closed under inverse morphisms and under intersection with regular languages.

Proposition 5.5. *The class* CRG *of Church-Rosser groups is closed under the operation of passing to finitely generated subgroups.*

Next we establish a non-closure property for the class of Church-Rosser groups.

Theorem 5.6. *The class* CRG *of Church-Rosser groups is not closed under direct products.*

Proof. The free group F_2 of rank 2 is context-free, and therewith a Church-Rosser group. We now consider the direct product $G_{2,2}$ of two copies of the group F_2, that is, $G_{2,2}$ is given through the presentation $\langle a, b, c, d; ac = ca, ad = da, bc = cb, bd = db\rangle$.

We claim that the language $L := \mathrm{WP}(G_{2,2}, \Sigma_4)$, which is the shuffle product of the languages $\mathrm{WP}(F_2, \{a, b\})$ and $\mathrm{WP}(F_2, \{c, d\})$, is not a Church-Rosser language. Actually, we will establish the following stronger result.

Claim. The language L is not accepted by any one-way auxiliary pushdown automaton ($\mathsf{OW\text{-}auxPDA}$) with a sublinear space bound.

Proof. According to [4], the language $L_B := \{\, w_1\#w_2\#w_1^R\#w_2^R \mid w \in \{0,1\}^* \,\}$ is not accepted by any $\mathsf{OW\text{-}auxPDA}$ with a sublinear space bound. Now assume that the language L is accepted by a $\mathsf{OW\text{-}auxPDA}$ A with space bound $s(n)$. From A we obtain a $\mathsf{OW\text{-}auxPDA}$ A' with space bound $s(n)$ for the language $L' := L \cap \{a,b\}^* \cdot \{c,d\}^* \cdot \{\bar{a},\bar{b}\}^* \cdot \{\bar{c},\bar{d}\}^*$ by running A and a finite-state acceptor for the regular language $\{a,b\}^* \cdot \{c,d\}^* \cdot \{\bar{a},\bar{b}\}^* \cdot \{\bar{c},\bar{d}\}^*$ in parallel. Define a mapping $\iota : \{a,b,c,d\}^* \to \{\bar{a},\bar{b},\bar{c},\bar{d}\}^*$ through $\iota(\varepsilon) := \varepsilon$ and $\iota(w \cdot x) := \bar{x} \cdot \iota(w)$ for all $w \in \{a,b,c,d\}^*$ and $x \in \{a,b,c,d\}$. It is easily checked that a word $w \in \underline{\Sigma}_4^*$ belongs to the language L' if and only if w can be factored as $w = u \cdot v \cdot \iota(u) \cdot \iota(v)$ for some words $u \in \{a,b\}^*$ and $v \in \{c,d\}^*$.

Now let $\phi_1 : \{0,1\}^* \to \{a,b\}^*$ be the morphism that is given through $0 \mapsto a$ and $1 \mapsto b$, let $\phi_2 : \{0,1\}^* \to \{c,d\}^*$ be the morphism that is given through $0 \mapsto c$ and $1 \mapsto d$, let $\phi_3 : \{0,1\}^* \to \{\bar{a},\bar{b}\}^*$ be the morphism that is given through $0 \mapsto \bar{a}$ and $1 \mapsto \bar{b}$, and let $\phi_4 : \{0,1\}^* \to \{\bar{c}, \bar{d}\}^*$ be the morphism that is given through $0 \mapsto \bar{c}$ and $1 \mapsto \bar{d}$. Then $w_1\#w_2\#w_3\#w_4 \in L_B$, where $w_1, w_2, w_3, w_4 \in \{0,1\}^*$, if and only if $\phi_1(w_1) \cdot \phi_2(w_2) \cdot \phi_3(w_3) \cdot \phi_4(w_4) \in L'$.

We can easily define a GSM that computes the mapping

$$w_1\#w_2\#w_3\#w_4 \to \phi_1(w_1)\cdot\phi_2(w_2)\cdot\phi_3(w_3)\cdot\phi_4(w_4)\ (w_1, w_2, w_3, w_4 \in \{0,1\}^*),$$

and we can combine this GSM with A' into a $\mathsf{OW\text{-}auxPDA}$ B for the language L_B. As B will have space bound $s(n)$, it follows that $s(n)$ is not sublinear. □

As each growing context-sensitive language is accepted by a $\mathsf{OW\text{-}auxPDA}$ with logarithmic space bound [6], it follows that $\mathrm{WP}(G_{2,2}, \Sigma_4)$ is not even a growing context-sensitive language. Thus, the class CRG of Church-Rosser groups is not closed under direct products. □

An application of Theorem 2.7 shows that the word problem of a hyperbolic group is a Church-Rosser language. Actually, we have a much stronger result. A finitely generated group G is said to *admit a generalized Dehn algorithm* if G has a finite set of semigroup generators Σ such that the language $\mathrm{WP}(G, \Sigma)$ belongs to the class GDL. As an immediate consequence of Proposition 3.3 we obtain the following implication.

Theorem 5.7. *If a finitely generated group admits a generalized Dehn algorithm, then it is a Church-Rosser group.*

Concerning groups that admit generalized Dehn algorithms, Goodman and Shapiro have established the following positive results.

Proposition 5.8. [10]

(1) *The property of admitting a generalized Dehn algorithm is independent of the chosen finite set of generators, that is, it is a property of finitely generated groups.*
(2) *The class of groups that admit a generalized Dehn algorithm is closed under the operations of passing to finitely generated subgroups and of passing to finite index overgroups.*
(3) *The class of groups admitting a generalized Dehn algorithm is closed under free products.*

Theorem 5.9. [10]

(1) *If G is a finitely generated group that is virtually nilpotent, then G admits a generalized Dehn algorithm.*
(2) *If G is a geometrically finite hyperbolic group, then G admits a generalized Dehn algorithm.*
(3) *If M is a graph manifold each of whose pieces is hyperbolic, then its fundamental group $\pi_1(M)$ admits a generalized Dehn algorithm.*

Thus, all these groups are Church-Rosser groups. Goodman and Shapiro also present the following negative results. Recall from Theorem 3.7 that the word problem $\mathrm{WP}(G, \Sigma)$ of a group G admits a non-incremental Dehn algorithm if and only if G is a Church-Rosser group.

Theorem 5.10. [10]

(1) *If a group G has exponential growth and the center of G contains an infinite cyclic group, then G does not admit a non-incremental Dehn algorithm.*
(2) *The group $F_2 \times F_1$ does not admit a non-incremental Dehn algorithm.*
(3) *If G is a braid group of 3 or more strands, then G does not admit a non-incremental Dehn algorithm.*
(4) *Thompson's group F does not admit a non-incremental Dehn algorithm.*
(5) *If G contains an abelian subgroup which has exponential growth, then G does not admit a non-incremental Dehn algorithm.*
(6) *If G is a Baumslag-Solitar group $\langle a, t; ta^pt^{-1} = a^q\rangle$ with $p \neq \pm q$, then G does not admit a non-incremental Dehn algorithm.*

Thus, none of these groups is Church-Rosser.

6. Growing Context-Sensitive Groups

Finally we turn to the classes GCSG and co-GCSG of growing context-sensitive and co-growing context-sensitive groups.

Proposition 6.1. (1) *The classes* GCSG *and* co-GCSG *are closed under the operation of passing to finite index overgroups and under the operation of passing to finitely generated subgroups.*
(2) *The class* co-GCSG *is closed under direct products.*
(3) *The class* GCSG *is not closed under direct products.*

Proof. 1. As GCSL is closed under inverse morphisms and under intersections with regular languages, Lemma 2 of [11] applies, showing that GCSG as well as co-GCSG are closed under taking finitely generated subgroups. Further, as GCSL is closed under union and under inverse GSM-mappings (Proposition 2.8), Lemma 5 of [11] applies, showing that GCSG and co-GCSG are closed under the operation of passing to finite index overgroups.

2. As GCSL is closed under union and under shuffle with regular languages, Lemma 3 of [11] applies, showing that co-GCSG is closed under direct products.

3. This is an immediate consequence of the proof of Theorem 5.6. □

Actually, the proof of Theorem 5.6 shows that $G_{2,2} \in$ co-CFG $\smallsetminus$ GCSG, which implies the following results.

Corollary 6.2. CRG $\subset$ co-GCSG, *and* co-CFG $\not\subseteq$ GCSG.

Let $G_1 = \langle\Sigma_1; R_1\rangle$ and $G_2 = \langle\Sigma_2; R_2\rangle$ be two finitely generated groups. Without loss of generality we may assume that the alphabets $\underline{\Sigma}_1$ and $\underline{\Sigma}_2$ are disjoint. Then the group G presented by the group presentation $\langle\Sigma_1 \cup \Sigma_2; R_1 \cup R_2\rangle$ is the *free product* of G_1 and G_2, denoted by $G = G_1 * G_2$.

Proposition 6.3. *The class* GCSG *is closed under free products.*

Proof. Let $G_1 = \langle\Sigma_1; R_1\rangle$ and $G_2 = \langle\Sigma_2; R_2\rangle$, where $\underline{\Sigma}_1 \cap \underline{\Sigma}_2 = \emptyset$, and let $G = G_1 * G_2 = \langle\Sigma_1 \cup \Sigma_2; R_1 \cup R_2\rangle$. If G_1 and G_2 are growing context-sensitive, then there exist sTPDAs M_1 and M_2 such that $L(M_i) = \mathrm{WP}(G_i, \Sigma_i)$ ($i =$

$1, 2)$. From M_1 and M_2 we now construct a sTPDA M for the language $L := \mathrm{WP}(G, \Sigma_1 \cup \Sigma_2)$.

Let $w = w_1 w_2 w_3 \cdots w_n$, where all factors w_i are non-empty, and w_i and w_{i+1} are in different factors for all $1 \leq i < n$. Then $w =_G 1$ if and only if there exists an index $j \in \{1, \ldots, n\}$ such that $w_j =_{G_j} 1$ and $w_1 \cdots w_{j-1} w_{j+1} \cdots w_n =_G 1$, where G_j denotes the factor containing w_j (that is, $G_j = G_1$ if $w_j \in \underline{\Sigma}_1^+$, and $G_j = G_2$ if $w_j \in \underline{\Sigma}_2^+$). Given w as input, M will proceed as follows. It first guesses the smallest index j with the above properties, shifting the prefix $w_1 \cdots w_{j-1}$ to the left-hand pushdown store using a copy alphabet, and simulates the computation of the machine M_j on input w_j. Should the computation of M_j being simulated not lead to acceptance, then M simply halts without accepting. Otherwise, M has successfully verified that $w_j =_{G_j} 1$ holds. It then erases the remains of this syllable from its pushdown stores, and guesses the next index j' with the above properties. Observe that now $w_{j-1} w_{j+1}$ forms a single syllable, and that $j' \geq j - 1$ holds due to the choice of j as the minimal index with the required property. If $j' > j - 1$, then M simply shifts the preceding syllables to its left-hand pushdown, and if $j' = j - 1$, then M shifts the syllable w_{j-1} back to its right-hand pushdown (using a second copy alphabet). In each case M then simulates the computation of the corresponding machine $M_{j'}$ on input $w_{j'}$. This process continues until either M rejects or until all syllables have been processed successfully. In the latter case $w =_G 1$ has been verified. It follows that $L(M) = L$, and it is seen easily that M can be designed to be shrinking. □

Observe that in the proof above nondeterminism is used in an essential way. It is currently not clear whether the corresponding result also holds for the class CRG of Church-Rosser groups. The reason being that it is not clear how to construct a sDTPDA M for the free product $G_1 * G_2$ from sDTPDAs M_1 and M_2 for the groups G_1 and G_2, respectively.

7. Concluding remarks

According to Theorem 5.10 the group $F_2 \times F_1$ does not admit a (non-incremental) Dehn algorithm, and hence, it is not a Church-Rosser group by Theorem 3.7. Is $F_2 \times F_1$ a growing-context-sensitive group? Observe that by Proposition 6.1 it is co-growing context-sensitive. Further, it is easily seen that the language $\mathrm{WP}(F_2 \times F_1, \{a, b, c\})$ corresponding to the presentation $\langle a, b, c; ac = ca, bc = cb \rangle$ is accepted by a OW-auxPDA in polynomial time and logarithmic space. Thus, if $F_2 \times F_1$ is *not* growing context-sensitive, then this would show that the inclusion of the language class GCSL in $\mathcal{L}(\text{OW-auxPDA}(poly, log))$ established in [6] is proper. On the other hand, if $F_2 \times F_1$ is growing context-sensitive, then this would imply that CRG is properly contained in GCSG.

In [18] it is shown that each finitely generated subgroup of an automatic group has a deterministic context-sensitive word problem. Thus, each automatic group is (deterministic) context-sensitive. As $F_2 \times F_2$ is automatic, we see from the proof of Theorem 5.6 that in general automatic groups are

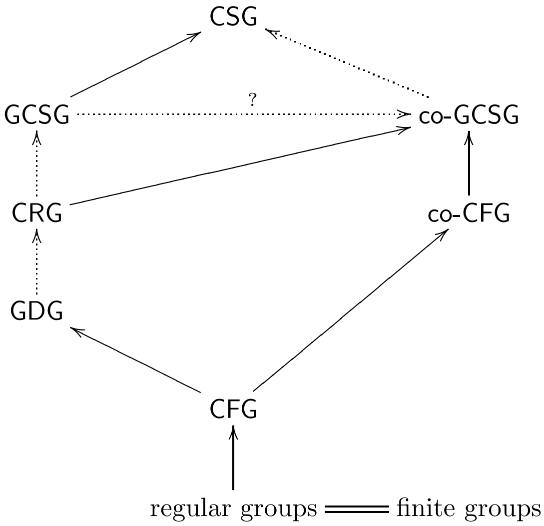

Figure 1. Inclusion relations between various classes of groups.

not growing context-sensitive. Is each automatic group co-growing context-sensitive? Observe that each automatic group is necessarily finitely presented [9], while it is known that there are even co-context-free groups that are not finitely presented [11].

The diagram in Figure 1 summarizes the inclusion relations between the various classes of groups considered in this paper. Here GDG denotes the class of groups that admit a generalized Dehn algorithm, $\mathcal{G}_1 \longrightarrow \mathcal{G}_2$ denotes a proper inclusion, $\mathcal{G}_1 \dashrightarrow \mathcal{G}_2$ denotes an inclusion for which it is not known whether it is proper, and $\mathcal{G}_1 =\!= \mathcal{G}_2$ denotes equality.

Acknowledgements

The research of the first author was supported by a Marie Curie Intra-European Fellowship within the 6th European Community Framework Programme.

References

[1] A.V. Anissimov: Group languages. *Cybernetics*, 7:594–601, 1971.
[2] J.-M. Autebert, L. Boasson, and G. Senizergues: Groups and NTS languages. *J. Comput. System Sci.*, 35:243–267, 1987.
[3] R.V. Book and F. Otto: *String-Rewriting Systems*. Springer, New York, 1993.
[4] F.J. Brandenburg: On one-way auxiliary pushdown automata. In H. Tzschach, H. Waldschmidt and H. Walter (eds.): *3. GI-Fachtagung 'Theoretische Informatik,' Proc.*, *Lect. Notes Comput. Sci.* 48: 132-144, Springer, Berlin, 1977.
[5] G. Buntrock and K. Loryś: On growing context-sensitive languages. In W. Kuich (ed.): *19. ICALP, Proc.*, *Lect. Notes Comput. Sci.* 623: 77–88, Springer, Berlin, 1992.
[6] G. Buntrock and F. Otto: Growing context-sensitive languages and Church-Rosser languages. *Inform. and Comput.*, 141:1–36, 1998.
[7] E. Dahlhaus and M. Warmuth: Membership for growing context-sensitive grammars is polynomial. *J. Comput. System Sci.*, 33:456–472, 1986.

[8] M.J. Dunwoody: The accessibility of finitely presented groups. *Invent. Math.*, 81:449–457, 1985.

[9] D.B.A. Epstein, J.W. Cannon, D.F. Holt, S.V.F. Levy, M.S. Paterson, and W.P. Thurston: *Word Processing In Groups*. Jones and Bartlett Publishers, Boston, 1992.

[10] O. Goodman and M. Shapiro: Dehn's algorithm for non-hyperbolic groups. *Preprint*, 2003.

[11] D. Holt, S. Rees, C. Röver, and R. Thomas: Groups with context-free co-word problem. *J. London Math. Soc.*, 71:643–657, 2005.

[12] J.E. Hopcroft and J.D. Ullman: *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, MA, 1979.

[13] N. Immerman: Languages that capture complexity classes. *SIAM J. on Computing*, 16:760–778, 1987.

[14] R. McNaughton, P. Narendran, and F. Otto: Church-Rosser Thue systems and formal languages. *J. Assoc. Comput. Mach.*, 35:324–344, 1988.

[15] D.E. Muller and P.E. Schupp: Groups, the theory of ends, and context-free languages. *J. Comput. System Sci.*, 26:295–310, 1983.

[16] G. Niemann: *Church-Rosser Languages and Related Classes*. Doctoral Dissertation, Fachbereich Mathematik/Informatik, Universität Kassel, 2002.

[17] G. Niemann and F. Otto: The Church-Rosser languages are the deterministic variants of the growing context-sensitive languages. *Inform. and Comput.*, 197:1–21, 2005. An extended abstract appeared in M. Nivat (ed.): *FoSSaCS 1998, Proc.*, *Lect. Notes Comput. Sci.* 1378, 243–257. Springer, Berlin, 1998.

[18] M. Shapiro: A note on context-sensitive languages and word problems. *Intern. J. Algebra Comput.*, 4:493–497, 1994.