Theoretical Computer Science 113 (1993) 119–165
Elsevier

On weakly confluent monadic string-rewriting systems*

K. Madlener

Fachbereich Informatik, Universität Kaiserslautern, Postfach 3049, 6750 Kaiserslautern, Germany

P. Narendran

Department of Computer Science, State University of New York, Albany, NY 12222, USA

F. Otto

Fachbereich Mathematik/Informatik, Gesamthochschule Kassel, Postfach 101380, 3500 Kassel, Germany

L. Zhang

Department of Computer Science, University of Waterloo, Waterloo, Ont., N2L 3G1, Canada

Abstract

Madlener, K., P. Narendran, F. Otto and L. Zhang, On weakly confluent monadic string-rewriting systems, Theoretical Computer Science 113 (1993) 119–165.

It is investigated as to how far the various decidability results for finite, monadic, and confluent string-rewriting systems can be carried over to the class of finite monadic string-rewriting systems that are only weakly confluent. Here a monadic string-rewriting system R on some alphabet Σ is called weakly confluent if it is confluent on all the congruence classes $[a]_R$, with $a \in \Sigma \cup \{e\}$. After establishing that the property of weak confluence is tractable for finite monadic string-rewriting systems, we prove that many decision problems that are tractable for finite, monadic, and confluent systems are, in fact, undecidable for finite monadic systems that are only weakly confluent. An example is the word problem. On the other hand, for finite, monadic, and weakly confluent systems that present groups, the validation problem for linear sentences is decidable. Many decision problems, among them the word problem and the generalized word problem, can be expressed through linear sentences and, hence, they all are decidable in this setting. The paper closes with

Correspondence to: F. Otto, Fachbereich Mathematik/Informatik, Gesamthochschule Kassel, Postfach 101380, 3500 Kassel, Germany. Email: otto@theory.informatik.uni-kassel.de.

* This paper combines and extends the results of three papers that have been presented at the 8th Annual Symposium on Theoretical Aspects of Computer Science (STACS '91) at Hamburg. February 1991, at the 18th International Colloquium on Automata, Languages, and Programming (ICALP '91) at Madrid, July 1991, and at the 9th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC-9) at New Orleans, October 1991, and that can be found in the corresponding proceedings. This paper has been prepared while the third author was visiting at the Fachbereich Informatik, Universität Kaiserslautern, during the winter semester 1991/92.

a specialized completion procedure for finite, monadic string-rewriting systems presenting groups. Given a system of this form, the completion procedure tries to construct an equivalent system of the same form that, in addition, is weakly confluent. The correctness and completeness of this procedure are shown, and some detailed examples are presented. This procedure, together with the decidability results mentioned before, presents an elegant and uniform way to perform computations in context-free groups effectively.

1. Introduction

Rewrite techniques have found many applications in computer science, ranging from automated theorem proving to rewrite-rule-based programming to the specification of abstract data types [2, 12]. For this reason, rewrite systems in their various forms are getting a lot of attention.

If the objects under consideration are strings or words over some finite alphabet Σ, then the appropriate notion of rewrite systems is that of string-rewriting systems. These systems, also known as semi-Thue systems, have been studied in computability theory, combinatorial (semi-) group theory, and formal language theory. Since a string-rewriting system on alphabet Σ can be interpreted as a term-rewriting system by simply regarding each letter $a \in \Sigma$ as a unary function symbol, results on string-rewriting systems can serve as valuable counterexamples for conjectures on term-rewriting systems, and, in many cases, they help to improve our intuition about the more general situation of term-rewriting systems.

A string-rewriting system R on alphabet Σ induces a congruence relation $\leftrightarrow_R^*$ on the set Σ^* of words over Σ. The set of congruence classes forms a monoid $\mathfrak{M}_R$, which is simply the factor monoid of the free monoid Σ^* modulo the congruence $\leftrightarrow_R^*$. Thus, string-rewriting systems can be used to present monoids and, therewith, they provide one way to perform computations in monoids effectively. In the present paper we are interested in these algorithmic aspects of string-rewriting systems.

For finite string-rewriting systems many decision problems have been investigated, among them the word problem, the group problem, and the generalized word problem, to name just a few (see Section 2 for the definitions). It is well known that all these problems are undecidable in general. On the other hand, if finite string-rewriting systems are considered that are noetherian and confluent, then some of these problems become decidable. For example, in this situation the word problem and the group problem [28] are decidable, while other problems like the generalized word problem still remain undecidable [27]. To overcome this difficulty, additional syntactical restrictions for string-rewriting systems have been studied. As it turned out, finite string-rewriting systems that are monadic and confluent have particularly nice algorithmic properties [5]. Here a string-rewriting system R on Σ is called monadic if $l >_{\ell\ell} r$ and $r \in \Sigma \cup \{e\}$ hold for each rule $(l \to r) \in R$, where e denotes the empty word, and $>_{\ell\ell}$ denotes the length-lexicographical ordering on Σ^* that is induced by a fixed linear ordering $>$ on Σ. This notion of monadic string-rewriting systems is slightly more general than the one used in the literature, where usually monadic systems are

required to be length-reducing (cf., e.g., [5]). Here we also allow length-preserving rules of the form $(b \to a)$, where a and b are letters such that $b > a$. Although this will make some of our arguments slightly more complicated, it is crucial for the specialized completion procedure for monadic systems that we present in Section 6. Fortunately, it will easily be verified that the results on monadic string-rewriting systems that we shall need from the literature extend to this more general notion.

Unfortunately, even if the word problem for a finite string-rewriting system S is easily decidable, there may not exist a finite, noetherian, and confluent system that generates the same congruence relation [10, 16, 18]. If we, nevertheless, want to use rewrite techniques to handle this congruence relation, we must relax the restriction placed on the systems. Since the property of being noetherian guarantees the termination of the process of rewriting, we want to keep this property. Thus, we relax the property of confluence, turning to those finite noetherian string-rewriting systems that are required to be confluent only on certain congruence classes.

Actually, the concept of confluence on some congruence class is not completely new. Let R be a finite length-reducing string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ presented by $(\Sigma; R)$ is a group, and let $^{-1}: \Sigma^* \to \Sigma^*$ be a mapping that associates with each word w a formal inverse w^{-1}. Then, for all $u, v \in \Sigma^*$, $u \leftrightarrow_R^* v$ if and only if $uv^{-1} \leftrightarrow_R^* e$. If R happens to be confluent on $[e]_R$, this holds if and only if $uv^{-1} \to_R^* e$, where $\to_R^*$ denotes the reduction relation induced by R. Thus, given $u, v \in \Sigma^*$, one simply computes an irreducible word $w \in \Sigma^*$ such that $uv^{-1} \to_R^* w$. Then $u \leftrightarrow_R^* v$ if and only if $w = e$. Dehn's algorithm for the word problem, which applies to certain small cancellation groups [21], can be expressed in this way. Bücken [7] and LeChenadec [20] have shown how certain small cancellation conditions yield a proof that the corresponding string-rewriting system R is confluent on $[e]_R$. In fact, LeChenadec presents a group symmetrization algorithm that, given as input a finite group presentation satisfying certain small cancellation conditions, computes a finite noetherian string-rewriting system R that is equivalent to the input system and is confluent on $[e]_R$. Finally, in [33], two of the authors investigate the algorithmic properties of finite special string-rewriting systems that are confluent on some congruence class.

In the present paper we focus on finite monadic string-rewriting systems that are weakly confluent, i.e. monadic systems R that are confluent on $[a]_R$ for all $a \in \Sigma \cup \{e\}$. The class of finite, monadic, and weakly confluent string-rewriting systems unifies the class of finite, monadic, and confluent systems and the class of finite, special, and e-confluent (i.e., confluent on $[e]$) systems. Since both these classes have nice algorithmic properties, it should be expected that this new class inherits some or all of these properties. Surprisingly, this is the case only if we look at systems of this form that present groups, but it is not true in general, as our results will show. Of course, it is not unexpected that the group property helps somewhat, since if $(\Sigma; R)$ presents a group, then this additional algebraic structure is mirrored by the properties of the congruence relation $\leftrightarrow_R^*$, but we did not expect that this would have that serious consequences already at the level of finite, monadic, and weakly confluent systems.

For finite length-reducing string-rewriting systems, confluence on a given congruence class is undecidable in general [29]. For finite monadic systems, this property is decidable; however, the algorithm given in [29] takes double exponential time. Here we do not directly improve upon this result, but at least we show that the property of weak confluence is tractable for finite monadic systems, i.e., it is decidable in polynomial time (Proposition 3.8). This result is obtained through a characterization of monadic and weakly confluent systems that extends a characterization of special and e-confluent systems given in [32].

Even though the property of weak confluence is tractable for finite monadic string-rewriting systems, the systems having this property do not, in general, have particularly nice algorithmic properties. Although it is decidable whether a system of this form presents a free monoid or whether it presents a group, there exists a system of this form that has an undecidable word problem (Theorem 4.9). Further, given a system R of this form on Σ and a proper subalphabet $\Sigma_1 \subsetneq \Sigma$, it is undecidable, in general, whether Σ_1 freely generates a submonoid of $\mathfrak{M}_R$ (Theorem 4.11). Since Σ_1 freely generates a submonoid of $\mathfrak{M}_R$ if and only if the system R (on Σ) is a consistent extension of the trivial system $R_1 := \emptyset$ (on Σ_1), this shows that in this setting it is undecidable whether a given system is a consistent extension of another given system, which nicely contrasts the decidability results for this problem presented in [30].

The situation improves considerably when we turn to those finite, monadic, and weakly confluent string-rewriting systems R that present groups, i.e., for which the monoid $\mathfrak{M}_R$ is a group. Autebert et al. [1] show that a group can be presented by a system of this form if and only if the group is context-free, which, by the results of Muller and Schupp [22] and Dunwoody [11], holds if and only if this group is virtually free. Thus, here we have the nice situation that the class of groups in question has three different characterizations: an algebraic one, a language-theoretic one, and one through a syntactical restriction for its presentations.

In [5], Book presents a class of logical formulae that he calls linear sentences. This class is defined syntactically. It consists of formulae of the form $\exists^p \forall^q F$ or $\forall^q \exists^p F$, where F is a positive combination of atomic formulae of the form $t_1 \equiv t_2$. Here t_1 and t_2 are terms (words) built from the letters of a fixed finite alphabet Σ and existential or universal variables (the exact definition can be found in Section 2). If φ is a linear sentence containing the variables $v_1, \ldots, v_p$, R is a string-rewriting system on Σ, and $S_1, \ldots, S_p$ are subsets of Σ^*, then R and $S_1, \ldots, S_p$ induce an interpretation of φ as follows: the symbol $\equiv$ is interpreted as the congruence $\leftrightarrow_R^*$ induced by R, and, for $i = 1, \ldots, p$, the set S_i is taken as the domain for the variable v_i. Thus, φ is either true or false as a statement on the congruence $\leftrightarrow_R^*$ and the sets $S_1, \ldots, S_p$. For example, the word problem and the generalized word problem can be expressed through linear sentences. Book proves that, if R is finite, monadic, and confluent, and if $S_1, \ldots, S_p$ are regular sets, then it is decidable whether the linear sentence φ is true under the induced interpretation. Otto and Zhang [33] extend this result to finite, special string-rewriting systems that are confluent on some congruence class, and here we extend it to those finite, monadic, and weakly confluent string-rewriting systems that present

groups (Theorem 5.8). Since the decidability of this problem is an invariant property of finitely generated monoids (Theorem 2.4), this shows that, for each finite presentation of a context-free group, it is decidable whether a linear sentence is true under the induced interpretation. Thus, finite, monadic, and weakly confluent string-rewriting systems are presentations of context-free groups that yield uniform algorithms for solving various decision problems. Therefore, given a context-free group $\mathfrak{G}$ through some finite presentation, it would be desirable to be able to construct a presentation of this particular form for $\mathfrak{G}$. This task, which is a variant of the problem of completion, which led Knuth and Bendix [19] to the development of their now famous completion procedure, is dealt with in Section 6.

Let R be a finite monadic string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. If R is e-confluent, then, for all $a \in \Sigma$ and all (irreducible) words w, if $aw \leftrightarrow_R^* e$, then $aw \rightarrow_R^* e$. In fact, this condition already suffices to guarantee that R is e-confluent. Unfortunately, the sets $\{w \in \mathrm{IRR}(R) \mid aw \leftrightarrow_R^* e\}$ $(a \in \Sigma)$ are, in general, not constructible in an effective way. Therefore, this ineffective characterization of e-confluence is replaced by two technically more involved, but decidable conditions (Theorem 6.4). These conditions characterize the property of e-confluence through the fact that certain regular sets must be empty. A nice aspect of this characterization is the fact that, from the given system R, nondeterministic finite-state acceptors for the regular sets in question can be constructed effectively, in fact, even in polynomial time.

If the system considered is already e-confluent, then weak confluence is again easily expressed (Theorem 6.6). Thus, these two results together give a characterization of weakly confluent monadic systems presenting groups that is much simpler than the corresponding characterization for monadic systems in general. Based on this characterization, a procedure WEAK_COMPLETION is presented that, given a finite monadic string-rewriting system R presenting a group as input, attempts to construct an equivalent monadic system that is weakly confluent. This procedure consists of two parts, labelled NORMALIZATION and CONTEXT_RESOLVING, where the former deletes superfluous rules in order to keep the system reduced, while the latter introduces new rules to make the system confluent on the relevant congruence classes. It is shown that this procedure either terminates with a finite monadic system R_i, or enumerates an infinite monadic system R_∞. In either case, the resulting system is reduced, weakly confluent, and equivalent to R. It is here that it becomes necessary to admit rules of the form $(b \rightarrow a)$ with $a, b \in \Sigma$, since, given a finite, monadic, and length-reducing string-rewriting system R on Σ such that $\mathfrak{M}_R$ is a (context-free) group, it can easily happen that there are letters $a, b \in \Sigma$ such that $a \leftrightarrow_R^* b$, but $a \not\rightarrow_R^* e$. Hence, there is no weakly confluent monadic system S that is strictly length-reducing and that is equivalent to R. Since there exists exactly one reduced, monadic, and weakly confluent system that is equivalent to R (Theorem 3.5), the procedure WEAK_COMPLETION terminates if and only if there exists a finite monadic system S that is weakly confluent and equivalent to R. Thus, this specialized completion procedure is correct and complete (Theorem 6.12). This corresponds to the situation of completing a finite special string-rewriting system R on $[e]_R$ considered by one of

the authors in [31]. However, there exists a finite monadic string-rewriting R on Σ such that the monoid $\mathfrak{M}_R$ is a context-free group, but there is no finite, monadic, and weakly confluent string-rewriting system S on Σ that is equivalent to R (Example 6.10). Hence, given R as input, the procedure WEAK_COMPLETION will not terminate, although the hypotheses on R imply that there exists a finite monadic system S' on some alphabet Σ' such that $\mathfrak{M}_{S'} \cong \mathfrak{M}_R$ and S' is weakly confluent. Thus, although we have extended the range where rewrite techniques apply, we are still faced with the same kind of problems. Hence, for future research, it seems worthwhile to try to develop completion procedures that not only introduce new rules in order to resolve critical pairs but also introduce new letters in a systematic way, if necessary. This would parallel certain developments in the theory of term-rewriting systems and their applications, but it would also take us to the problem of consistent extensions considered in [30].

2. Notation, definitions, and first results

Here we restate some of the basic definitions and results on string-rewriting systems that we will use throughout the paper. In addition, this section will lead to a new characterization of the property of confluence on a given congruence class for finite noetherian string-rewriting systems, improving upon the characterization given in [29]. In the next section we will exploit this characterization to develop a polynomial-time algorithm for deciding weak confluence of monadic string-rewriting systems. For additional information and comments regarding the various notions introduced, the reader is asked to consult the literature, e.g., Book's comprehensive overview article [6].

Let Σ be a finite alphabet. Then Σ^* denotes the *set of words* over Σ, including the empty word e, while Σ^+ denotes the set of all nonempty words. As usual, the *length* of a word $w \in \Sigma^*$ is denoted by $|w|$, and the *concatenation* of two words u and v is written as uv. To improve readability, words are often written using exponents, where $w^0 = e$ and $w^{n+1} = ww^n$ for all $w \in \Sigma^*$ and all $n \in \mathbb{N}$.

Let $>$ be a partial ordering on Σ^*. This partial ordering is called

- *well-founded* if there does not exist an infinite descending sequence $w_0 > w_1 > w_2 > \cdots$;
- *admissible* if it is compatible with the operation of concatenation, i.e., for all $u, v, x, y \in \Sigma^*$, if $u > v$, then also $xuy > xvy$;
- a *well-ordering* if it is well-founded and total.

The *length-ordering* $>$, defined by $u >_\ell v$ iff $|u| > |v|$, is an admissible well-founded partial ordering which is not total unless Σ contains a single letter only. In this paper we will mostly be concerned with another ordering: the *length-lexicographical ordering* $>_{\ell\ell}$. Let $>$ be a total ordering on the alphabet Σ. Then $u >_{\ell\ell} v$ iff $|u| > |v|$ or ($|u| = |v|$ and $u >_{\text{lex}} v$), where $>_{\text{lex}}$ denotes the pure lexicographical ordering on Σ^* induced by $>$. While the pure lexicographical ordering is not well-founded, the length-lexicographical ordering is an admissible well-ordering.

A *string-rewriting system* R on Σ is a subset of $\Sigma^* \times \Sigma^*$. The elements of R are called (*rewrite*) *rules*, and, to enhance readability, they are often written in the form $(l \to r)$. For a string-rewriting system R, $\mathrm{dom}(R) = \{l \mid \exists r \in \Sigma^*\colon (l \to r) \in R\}$, and $\mathrm{range}(R) = \{r \mid \exists l \in \Sigma^*\colon (l \to r) \in R\}$. The system R is called *length-reducing* if $|l| > |r|$ holds for each rule $(l \to r) \in R$, it is called *monadic* if $\mathrm{range}(R) \subseteq \Sigma \cup \{e\}$ and $l >_{\ell\ell} r$ for each rule $(l \to r) \in R$, and it is called *special* if it is length-reducing and $\mathrm{range}(R) = \{e\}$.

As explained in the introduction, our notion of monadic string-rewriting system is slightly more general than the one considered, e.g., in [6]. However, all the results on monadic string-rewriting systems that we will restate from the literature remain valid with this slightly more general notion.

A string-rewriting system R on Σ defines several binary relations on Σ^*, the most fundamental one of which is the *single-step reduction relation* $\to_R$:

$$u \to_R v \text{ iff } \exists x, y \in \Sigma^*\ \exists (l \to r) \in R\colon u = xly \text{ and } v = xry.$$

Its reflexive transitive closure $\to_R^*$ is the *reduction relation* induced by R, while its reflexive, symmetric, and transitive closure $\leftrightarrow_R^*$ is a congruence on Σ^*, the *Thue congruence* generated by R. For $w \in \Sigma^*$, $[w]_R$ denotes the *congruence class* $\{v \in \Sigma^* \mid w \leftrightarrow_R^* v\}$. The factor monoid $\Sigma^*/\leftrightarrow_R^*$ is denoted by $\mathfrak{M}_R$, and, whenever a monoid $\mathfrak{M}$ is isomorphic to $\mathfrak{M}_R$, we call the ordered pair $(\Sigma; R)$ a (*monoid-*) *presentation* of $\mathfrak{M}$ with *generators* Σ and *defining relations* R.

If $u, v \in \Sigma^*$ are such that $u \to_R^* v$, then u is an *ancestor* of v, and v is a *descendant* of u. If $u \to_R v$, then u is an *immediate ancestor* of v, and v is an *immediate descendant* of u. For $w \in \Sigma^*$, $\triangle_R(w) := \{v \in \Sigma^* \mid w \to_R v\}$ and $\triangle_R^*(w) := \{v \in \Sigma^* \mid w \to_R^* v\}$ are the set of immediate descendants and the set of all descendants of w, while $\nabla_R(w) := \{u \in \Sigma^* \mid u \to_R w\}$ and $\nabla_R^*(w) := \{u \in \Sigma^* \mid u \to_R^* w\}$ are the set of immediate ancestors and the set of all ancestors of w. For a language $L \subseteq \Sigma^*$, $[L]_R := \bigcup_{u \in L} [u]_R$, $\triangle_R^*(L) := \bigcup_{u \in L} \triangle_R^*(u)$, and $\nabla_R^*(L) := \bigcup_{u \in L} \nabla_R^*(u)$. Finally, a word $w \in \Sigma^*$ is called *irreducible* if it has no immediate descendant; otherwise, w is called *reducible* (mod R). $\mathrm{IRR}(R)$ denotes the set of all irreducible words (mod R).

The following algorithmic properties of finite string-rewriting systems will be used frequently in the paper.

Proposition 2.1. (a) *There is a polynomial-time algorithm that, given a finite string-rewriting system R as input, determines a deterministic finite-state acceptor (dfsa) for the set* $\mathrm{IRR}(R)$ [13].

(b) *Let R be a finite monadic string-rewriting system on Σ. Then there is a linear-time algorithm that, given a word $w \in \Sigma^*$ as input, computes an irreducible descendant of w* mod R [4].

(c) *There is a polynomial-time algorithm that, given a finite monadic string-rewriting system R on Σ and a regular set $L \subseteq \Sigma^*$ specified through some nondeterministic finite-state acceptor (nfsa) as input, constructs an nfsa for the set* $\triangle_R^*(L)$ [5].

Concerning (b), observe that in [4] Book deals only with finite length-reducing string-rewriting systems, but it is easily seen that, in the process of reducing a word w of length n modulo a finite monadic system R, rules of the form $(b \to a) \in R$, $a, b \in \Sigma$, $b > a$, can be applied less than $2 \cdot n \cdot |\Sigma|$ times. Thus, the algorithm given in [4] is linear-time even for monadic systems that are not length-reducing. Concerning (c), the algorithm described in [5] is easily extended to monadic systems that are not length-reducing.

A string-rewriting system R on Σ is called

- *noetherian* if there is no infinite sequence of reductions of the form $u_0 \to_R u_1 \to_R u_2 \to_R \cdots$,
- *locally confluent* if, for all $u, v \in \Sigma^*$, $\nabla_R(u) \cap \nabla_R(v) \neq \emptyset$ implies that $\triangle_R^*(u) \cap \triangle_R^*(v) \neq \emptyset$,
- *confluent* if, for all $u, v \in \Sigma^*$, $\nabla_R^*(u) \cap \nabla_R^*(v) \neq \emptyset$ implies that $\triangle_R^*(u) \cap \triangle_R^*(v) \neq \emptyset$.

If R is noetherian, then each word $w \in \Sigma^*$ has at least one irreducible descendant, and if R is both noetherian and confluent, then each congruence class $[w]_R$ contains a unique irreducible word w_0 that can serve as a normal form or representative of this class. Thus, if R is a finite noetherian and confluent system, then the *word problem for* R is effectively decidable:

Instance: Two words $u, v \in \Sigma^*$.
Question: Does $u \leftrightarrow_R^* v$ hold?

Now the string-rewriting system R on Σ is noetherian if and only if there exists an admissible well-founded partial ordering $>$ on Σ^* such that $l > r$ holds for each rule $(l \to r) \in R$, which in turn holds if and only if the transitive closure $\to_R^+$ of the single-step reduction relation $\to_R$ is a well-founded partial ordering. Unfortunately, this property is undecidable in general [15], but the above characterization gives at least a sufficient condition that can often be used to verify that a system is indeed noetherian.

In general, it is undecidable as well whether a finite string-rewriting system R is confluent, if, however, R is noetherian, then R is confluent if and only if it is locally confluent [24].

Let $(l_1 \to r_1)$ and $(l_2 \to r_2)$ be two (not necessarily distinct) rules of R. If there are words $x, y \in \Sigma^*$ such that $l_1 = x l_2 y$ or $l_1 x = y l_2$ and $0 < |y| < |l_1|$, then these rules are said to *overlap*. The pair of words $(r_1, x r_2 y)$ or $(r_1 x, y r_2)$, respectively, is then called a *critical pair* of R. By $\mathrm{CP}(R)$ we denote the *set of all critical pairs* of R. A critical pair (u, v) *resolves* if $\triangle_R^*(u) \cap \triangle_R^*(v) \neq \emptyset$, otherwise, it is *unresolvable*. $\mathrm{UCP}(R)$ denotes the *set of all unresolvable critical pairs* of R. Now R is locally confluent if and only if the set $\mathrm{UCP}(R)$ is empty, i.e., all critical pairs of R resolve [25]. For a finite and noetherian system R, this set can be computed effectively. Hence, in this situation it is decidable whether or not R is confluent.

In this paper we are concerned with a rather restricted notion of confluence that we choose to call *weak confluence*. Let R be a string-rewriting system on Σ, and let $w \in \Sigma^*$. The system R is said to be *confluent on* $[w]_R$ if, for all $u, v \in [w]_R$, $\nabla_R^*(u) \cap \nabla_R^*(v) \neq \emptyset$ implies that $\triangle_R^*(u) \cap \triangle_R^*(v) \neq \emptyset$. If R is noetherian, then this means that the

congruence class $[w]_R$ contains a unique irreducible word w_0, which can then serve as a normal form for $[w]_R$. Thus, if R is noetherian and $w \in \mathrm{IRR}(R)$, then R is confluent on $[w]_R$ if and only if $[w]_R = \nabla_R^*(w)$. In particular, if R is finite, this means that under these circumstances the *membership problem for* $[w]_R$ is effectively decidable. If R is confluent on $[e]_R$, then we say that R is *e-confluent*. Finally, R is called *weakly confluent* if R is confluent on $[w]_R$ for all $w \in \mathrm{range}(R)$. Observe that, for a monadic system R, weak confluence is equivalent to saying that R is confluent on $[a]_R$ for all $a \in \Sigma \cup \{e\}$. Thus, weak confluence can be seen as the generalization of e-confluence from special to monadic string-rewriting systems.

Our next goal is a characterization of the property of confluence on a given congruence class. To state this characterization, we need the following notions.

Let R be a string-rewriting system on Σ, let $u \in \Sigma^*$, and let $w \in \mathrm{IRR}(R)$. Then the *set* $\mathrm{Con}_u(w)$ *of contexts of* u *in* $\nabla_R^*(w)$ is defined by $\mathrm{Con}_u(w) := \{x \# y \mid x, y \in \mathrm{IRR}(R) \text{ and } xuy \to_R^* w\}$. Here $\#$ is an additional letter not in Σ. Further, we consider the subset $\mathrm{SCon}_u(w)$ of the set $\mathrm{Con}_u(w)$ which is defined through $\mathrm{SCon}_u(w) := \{x \# y \mid x, y \in \mathrm{IRR}(R) \text{ and } \triangle_R^*(xuy) \cap \mathrm{IRR}(R) = \{w\}\}$, i.e. $\mathrm{SCon}_u(w)$ contains those contexts of u in $\nabla_R^*(w)$ that cannot be reduced to any irreducible word other than w. Observe that $u \to_R^* v$ implies that $\mathrm{SCon}_u(w) \subseteq \mathrm{SCon}_v(w)$ and $\mathrm{Con}_v(w) \subseteq \mathrm{Con}_u(w)$.

Lemma 2.2. *Let R be a noetherian string-rewriting system on Σ, let $w \in \mathrm{IRR}(R)$, and let $u \in \Sigma^*$ satisfy $\triangle_R^*(u) \cap \mathrm{IRR}(R) \supsetneq \{w\}$. Then there are words $x, y \in \mathrm{IRR}(R)$ and a pair $(p, q) \in \mathrm{UCP}(R)$ such that*

(1) *$xpy, xqy \in \triangle_R^*(u)$, and*

(2) *either $x \# y \in \mathrm{SCon}_p(w)$ and $x \# y \notin \mathrm{Con}_q(w)$, or $x \# y \in \mathrm{SCon}_q(w)$ and $x \# y \notin \mathrm{Con}_p(w)$.*

Proof. Let S_u denote the set $S_u := \{v \in \triangle_R^*(u) \mid \triangle_R^*(v) \cap \mathrm{IRR}(R) \supsetneq \{w\}\}$. Since $u \in S_u$, this set is nonempty. Since R is noetherian, $\to_R^+$ is a well-founded partial ordering on Σ^*. So, we can choose a word $z \in S_u$ that is minimal with respect to this partial ordering. Thus, $z \in \triangle_R^*(u)$ and $\triangle_R^*(z) \cap \mathrm{IRR}(R) \supsetneq \{w\}$, but, for each word $z_1 \in \triangle_R^*(u)$, if $z \to_R^+ z_1$, then $z_1 \notin S_u$, i.e., either $w \notin \triangle_R^*(z_1)$ or $\triangle_R^*(z_1) \cap \mathrm{IRR}(R) = \{w\}$. Hence, z can be factored as $z = x_1 l_1 y_1 = x_2 l_2 y_2$ for some $x_1, x_2, y_1, y_2 \in \Sigma^*$ and $(l_1 \to r_1), (l_2 \to r_2) \in R$ such that $z = x_1 l_1 y_1 \to_R x_1 r_1 y_1 \to_R^* w$ and $z = x_2 l_2 y_2 \to_R x_2 r_2 y_2 \not\to_R^* w$. From the choice of z it follows easily that the displayed occurrences of l_1 and l_2 in z overlap, and that their overlap yields an unresolvable critical pair $(p, q) \in \mathrm{UCP}(R)$. Thus, $z = xsy \to_R xpy$ and $z = xsy \to_R xqy$ for some $x, s, y \in \Sigma^*$, and either $xpy = x_1 r_1 y_1$ and $xqy = x_2 r_2 y_2$, or $xpy = x_2 r_2 y_2$ and $xqy = x_1 r_1 y_1$. In the former case, $x \# y \in \mathrm{SCon}_p(w)$ and $x \# y \notin \mathrm{Con}_q(w)$, while in the latter $x \# y \in \mathrm{SCon}_q(w)$ and $x \# y \notin \mathrm{Con}_p(w)$. □

Next we introduce a restricted version of the reduction relation $\to_R$. A reduction step $u \to_R v$ is called *leftmost* if $u = xly$ and $v = xry$ for some rule $(l \to r) \in R$, and whenever $u = x_1 l_1 y_1$ for some rule $(l_1 \to r_1) \in R$, then xl is a proper prefix of $x_1 l_1$, or $xl = x_1 l_1$ and x is a proper prefix of x_1, or $x = x_1$ and $l = l_1$. We write $u \,{}_{\mathrm{L}}\!\to_R v$ if $u \to_R v$ is

leftmost, and by ${}_{L}\rightarrow_{R}^{*}$ we denote the reflexive transitive closure of ${}_{L}\rightarrow_{R}$. Further, a pair $(x, y)\in\Sigma^{*}\times\Sigma^{*}$ is called a *leftmost descendant pair* of a pair $(u, v)\in\Sigma^{*}\times\Sigma^{*}$ if $u\,{}_{L}\rightarrow_{R}^{*}x$ and $v\,{}_{L}\rightarrow_{R}^{*}y$. Let IDUCP($R$) be the set of irreducible leftmost descendant pairs of UCP(R), i.e., $\mathrm{IDUCP}(R):=\{(s, t)\in\mathrm{IRR}(R)\times\mathrm{IRR}(R)\mid\exists(p, q)\in\mathrm{UCP}(R)\colon p\,{}_{L}\rightarrow_{R}^{*}s$ and $q\,{}_{L}\rightarrow_{R}^{*}t\}$.

Now we are prepared to state the intended characterization theorem.

Theorem 2.3. *Let R be a noetherian string-rewriting system on Σ, and let $w\in\mathrm{IRR}(R)$. Then the following three statements are equivalent:*

(a) *The system R is confluent on $[w]_{R}$.*

(b) $\forall(u, v)\in\mathrm{UCP}(R)\colon \mathrm{SCon}_{u}(w)\subseteq\mathrm{Con}_{v}(w)$ *and* $\mathrm{SCon}_{v}(w)\subseteq\mathrm{Con}_{u}(w)$.

(c) $\forall(u, v)\in\mathrm{IDUCP}(R)\colon \mathrm{SCon}_{u}(w)\subseteq\mathrm{Con}_{v}(w)$ *and* $\mathrm{SCon}_{v}(w)\subseteq\mathrm{Con}_{u}(w)$.

Proof. (a) $\Rightarrow$ (c): Assume that R is confluent on $[w]_{R}$, and let $(u, v)\in\mathrm{IDUCP}(R)$. Then $u\leftrightarrow_{R}^{*}v$. Now, if $x\#y\in\mathrm{SCon}_{u}(w)$, then $\triangle_{R}^{*}(xuy)\cap\mathrm{IRR}(R)=\{w\}$, implying that $xvy\leftrightarrow_{R}^{*}xuy\leftrightarrow_{R}^{*}w$. Since R is confluent on $[w]_{R}$, we can conclude that $xvy\rightarrow_{R}^{*}w$, i.e., $x\#y\in\mathrm{Con}_{v}(w)$. Hence, $\mathrm{SCon}_{u}(w)\subseteq\mathrm{Con}_{v}(w)$, and $\mathrm{SCon}_{v}(w)\subseteq\mathrm{Con}_{u}(w)$ is shown analogously.

(c) $\Rightarrow$ (b): Let $(u, v)\in\mathrm{UCP}(R)$. Then there is an irreducible leftmost descendant pair $(r, s)\in\mathrm{IDUCP}(R)$ such that $u\,{}_{L}\rightarrow_{R}^{*}r$ and $v\,{}_{L}\rightarrow_{R}^{*}s$. Because of (c), we can deduce the following sequence of inclusions: $\mathrm{SCon}_{u}(w)\subseteq\mathrm{SCon}_{r}(w)\subseteq\mathrm{Con}_{s}(w)\subseteq\mathrm{Con}_{v}(w)$, and, analogously, $\mathrm{SCon}_{v}(w)\subseteq\mathrm{Con}_{u}(w)$ is obtained.

(b) $\Rightarrow$ (a): Assume to the contrary that R is not confluent on $[w]_{R}$. Then there is a word $v\in\mathrm{IRR}(R)$ such that $v\neq w$ but $v\leftrightarrow_{R}^{*}w$. Thus, there exists an integer $m\geqslant 1$ and words $w_{0}, w_{1}, \ldots, w_{m}\in\Sigma^{*}$ such that $w=w_{0}\leftrightarrow_{R}w_{1}\leftrightarrow_{R}\cdots\leftrightarrow_{R}w_{m}=v$. Since $w, v\in\mathrm{IRR}(R)$, we have $m\geqslant 2$, $w_{1}\rightarrow_{R}w_{0}$ and $w_{m-1}\rightarrow_{R}w_{m}$. Let $k:=\max\{i\mid w_{i}\rightarrow_{R}^{*}w\}$. Then $1\leqslant k\leqslant m-1$, and it is easily seen that $\triangle_{R}^{*}(w_{k})\cap\mathrm{IRR}(R)\supsetneq\{w\}$. Hence, by Lemma 2.2, there are words $x, y\in\mathrm{IRR}(R)$ and a pair $(u, v)\in\mathrm{UCP}(R)$ such that $x\#y\in\mathrm{SCon}_{u}(w)$ and $x\#y\notin\mathrm{Con}_{v}(w)$, or $x\#y\in\mathrm{SCon}_{v}(w)$ and $x\#y\notin\mathrm{Con}_{u}(w)$. In either case, this contradicts statement (b). This completes the proof of Theorem 2.3. □

Theorem 2.3 is an improvement of a characterization given in [29]. Recall that the sets UCP(R) and IDUCP(R) are effectively computable if R is finite and noetherian. Nevertheless, it is undecidable, in general, whether a finite noetherian (in fact, even length-reducing) string-rewriting system is confluent on a given congruence class [29].

Since there is nothing special about performing reductions from left to right, we can define the notion of *rightmost reduction* similarly by performing reductions from right to left. We write $u\,{}_{R}\rightarrow_{R}v$ if $u\rightarrow_{R}v$ is rightmost, and by ${}_{R}\rightarrow_{R}^{*}$ we denote the corresponding reflexive transitive closure.

A string-rewriting system R on Σ is called *reduced* if the following holds for each rule $(l\rightarrow r)\in R$: $l\in\mathrm{IRR}(R\setminus\{l\rightarrow r\})$, i.e., no left-hand side contains another left-hand side as a factor, and $r\in\mathrm{IRR}(R)$. For a reduced system, the process of leftmost reduction is

deterministic, i.e., for each $w \notin \mathrm{IRR}(R)$, there exists a unique word z such that $w_{\mathrm{L}} \to_R z$. Of course, the same is true for rightmost reductions.

Finally, two string-rewriting systems R and S on the same alphabet are called *equivalent* if their induced Thue congruences coincide, i.e., $\leftrightarrow_R^* = \leftrightarrow_S^*$. Obviously, R and S are equivalent if and only if $l \leftrightarrow_S^* r$ holds for each rule $(l \to r) \in R$, and $p \leftrightarrow_R^* q$ holds for each rule $(p \to q) \in S$.

In addition to the word problem, there are many other important decision problems for finite string-rewriting systems. In this paper we will be concerned with the following ones:

- *The finiteness problem*
  Instance: A finite string-rewriting system R on Σ.
  Question: Is the monoid $\mathfrak{M}_R$ presented by $(\Sigma; R)$ finite?

- *The group problem*
  Instance: A finite string-rewriting system R on Σ.
  Question: Is the monoid $\mathfrak{M}_R$ a group?

- *The free monoid problem*
  Instance: A finite string-rewriting system R on Σ.
  Question: Is the monoid $\mathfrak{M}_R$ a free monoid?

- *The generalized word problem*
  Instance: A finite string-rewriting system R on Σ, a regular subset $U \subseteq \Sigma^*$, and a word $w \in \Sigma^*$.
  Question: Does $w \in \mathrm{SUBM}(U)$ hold?

Here $\mathrm{SUBM}(U) = \{u \in \Sigma^* \mid \exists m \geqslant 0\ \exists u_1, \ldots, u_m \in U: u \leftrightarrow_R^* u_1 \ldots u_m\}$ is the submonoid of $\mathfrak{M}_R$ that is generated by the set U. If the system R is not considered a part of the problem instance, we talk about the *generalized word problem for R*.

- *The inclusion problem*
  Instance: A finite string-rewriting system R on Σ, and two regular subsets $U, V \subseteq \Sigma^*$.
  Question: Is the submonoid $\mathrm{SUBM}(U)$ contained in the submonoid $\mathrm{SUBM}(V)$?

Again, if the system R is not a part of the problem instance, we talk about the *inclusion problem for R*.

- *The free submonoid problem*
  Instance: A finite string-rewriting system R on Σ, and a subalphabet $\Sigma_1 \subsetneq \Sigma$.
  Question: Is the submonoid $\mathrm{SUBM}(\Sigma_1)$ freely generated by Σ_1?

The submonoid SUBM(Σ_1) is freely generated by Σ_1 if and only if $\leftrightarrow_R^*|_{\Sigma_1^* \times \Sigma_1^*} = \mathrm{id}_{\Sigma_1^*}$, i.e., for all $x, y \in \Sigma_1^*$, $x \leftrightarrow_R^* y$ if and only if $x = y$. This fact is equivalent to saying that the string-rewriting system R is a *consistent extension* of the trivial system $R_1 = \emptyset$ on Σ_1 [30].

Thus, the first three problems ask particular questions about the algebraic structure of the monoid $\mathfrak{M}_R$ presented by a given presentation $(\Sigma; R)$, while the last three deal with certain questions concerning submonoids of $\mathfrak{M}_R$. They all are undecidable in general. On the other hand, they have all been solved for certain restricted classes of finite string-rewriting systems. If R is a finite, noetherian and confluent system, then there is a one-to-one correspondence between the monoid $\mathfrak{M}_R$ and the set IRR(R). From R, a dfsa for the set IRR(R) can be computed in polynomial time [Proposition 2.1(a)] and, hence, we can decide in polynomial time whether this set is finite. Thus, the finiteness problem is decidable in polynomial time for these systems. The same result holds for finite special systems that are e-confluent [33]. The group problem and the free monoid problem are decidable for each class $\mathfrak{C}$ of finite string-rewriting systems for which the following *restricted version of the uniform word problem* can be solved [28]:

Instance: A finite string-rewriting system $R \in \mathfrak{C}$, and a word $w \in \Sigma^*$.
Question: Does $w \leftrightarrow_R^* e$ hold?

Thus, these two problems are also decidable for the two classes of systems mentioned above. The free submonoid problem is undecidable, in general, even if it is restricted to finite, length-reducing, and confluent string-rewriting systems, but it is decidable for finite, monadic, and confluent systems [30], and the same is true for the generalized word problem and the inclusion problem [5, 27]. In [5], Book presents a class of logical formulae that he calls *linear sentences*. These sentences, which are defined syntactically, can be used to express properties of Thue congruences. We restate the definition of these sentences in short.

Let Σ be a finite alphabet, and let V_E and V_U be two disjoint countable alphabets such that $(V_E \cup V_U) \cap \Sigma = \emptyset$. The elements of V_E are called *existential variables*, those of V_U are called *universal variables*. Further, let $\equiv$ be a binary predicate symbol.

A *term* is a word $t \in (\Sigma \cup V_E \cup V_U)^*$ that does not contain both existential and universal variables. An *atomic formula* is an expression of the form $s \equiv t$, where s and t are terms. It is called *constant* if neither s nor t contains a variable, *existential* if neither s nor t contains a universal variable, *universal* if neither s nor t contains an existential variable, and *mixed* if one of s and t contains existential variables, while the other contains universal variables. A *linear formula* F is a combination of finitely many atomic formulae by the operations of conjunction ($\wedge$) and disjunction ($\vee$) such that no variable appears more than once in F. Finally, if F is a linear formula containing the existential variables $v_1, \ldots, v_p \in V_E$ and the universal variables $u_1, \ldots, u_q \in V_U$, then

$$\forall u_1 \forall u_2 \ldots \forall u_q \exists v_1 \exists v_2 \ldots \exists v_p F$$

and

$$\exists v_1 \exists v_2 \dots \exists v_p \forall u_1 \forall u_2 \dots \forall u_q F$$

are *linear sentences*. By LINSEN(Σ), we denote the set of all linear sentences over Σ.

Let R be a string-rewriting system on Σ. If φ is a linear sentence over Σ containing the variables $v_1, \dots, v_p \in V_E \cup V_U$, and if $S_1, \dots, S_p$ are subsets of Σ^*, then by interpreting the predicate symbol $\equiv$ as the Thue congruence $\leftrightarrow_R^*$, and by letting the variable v_i range over the set S_i ($i = 1, \dots, p$), we obtain an *interpretation* of the linear sentence φ. Thus, φ can be seen as a statement on the Thue congruence $\leftrightarrow_R^*$ and the sets $S_1, \dots, S_p \subseteq \Sigma^*$ which is either true or false. For example, the property "$w \in$ SUBM(U)", where $w \in \Sigma^*$ and $U \subseteq \Sigma^*$, can be expressed through the linear sentence $\exists v$: $w \equiv v$ if the set U^* is taken as the domain of v. Analogously, the property "SUBM(U) $\subseteq$ SUBM(V)", where $U, V \subseteq \Sigma^*$, is expressed through the linear sentence $\forall u\ \exists v$: $u \equiv v$ if the sets U^* and V^* are taken as the domains of the variables u and v, respectively.

For a string-rewriting system R on Σ, the *validation problem for linear sentences* is the following decision problem:

Instance: A linear sentence $\varphi \in$ LINSEN(Σ) containing the variables $v_1, \dots, v_p \in V_E \cup V_U$, and regular sets $S_1, S_2, \dots, S_p \subseteq \Sigma^*$ specified through regular expressions.

Question: Is φ true under the interpretation induced by R and $S_1, \dots, S_p$?

Book [5] shows that this problem is decidable for finite, monadic, and confluent string-rewriting systems, and Otto and Zhang [33] show that it is decidable for finite special string-rewriting systems that are e-confluent. Actually, since a linear sentence is a statement on the Thue congruence $\leftrightarrow_R^*$ rather than the reduction relation $\rightarrow_R^*$, this problem is, thus, decidable for each string-rewriting system that is equivalent to a system of one of these two types. In fact, we can generalize this result somewhat.

Let R_1 be a string-rewriting system on some alphabet Σ_1 and let R_2 be a string-rewriting system on some alphabet Σ_2 such that the monoids $\mathfrak{M}_1$ and $\mathfrak{M}_2$ presented by $(\Sigma_1; R_1)$ and $(\Sigma_2; R_2)$, respectively, are isomorphic. We claim that, if the validation problem for linear sentences is decidable for R_2, then so is the validation problem for linear sentences for R_1, i.e., the decidability of this problem is an invariant of finitely generated monoids.

Indeed, let $h: \mathfrak{M}_1 \rightarrow \mathfrak{M}_2$ be an isomorphism, and, for each letter $a \in \Sigma_1$, let u_a be a word from Σ_2^* such that $u_a \in h([a]_{R_1})$. We define a homomorphism $f: \Sigma_1^* \rightarrow \Sigma_2^*$ through $a \mapsto u_a$ ($a \in \Sigma_1$). Then it is easily seen that, for all $x, y \in \Sigma_1^*$, $x \leftrightarrow_{R_1}^* y$ if and only if $f(x) \leftrightarrow_{R_2}^* f(y)$. We extend this homomorphism $f: \Sigma_1^* \rightarrow \Sigma_2^*$ to a homomorphism $f: (\Sigma_1 \cup V_E \cup V_U)^* \rightarrow (\Sigma_2 \cup V_E \cup V_U)^*$ by taking $f(v) := v$ for all variables $v \in V_E \cup V_U$. If ψ is a linear sentence over Σ_1 containing the variables $v_1, \dots, v_p \in V_E \cup V_U$, then we obtain a linear sentence $\varphi \in$ LINSEN(Σ_2) from ψ by replacing each term w in ψ by the

term $f(w)$. Further, if $T_1, \ldots, T_p \subseteq \Sigma_1^*$ are regular sets specified through regular expressions, then the sets $S_i := f(T_i) = \{f(x) \mid x \in T_i\}$, $i = 1, \ldots, p$, are regular subsets of Σ_2^*, and regular expressions specifying these sets can be constructed effectively. Now it is straightforward to verify that the linear sentence ψ is true under the interpretation induced by R_1 and $T_1, \ldots, T_p$ if and only if the linear sentence φ is true under the interpretation induced by R_2 and $S_1, \ldots, S_p$. This gives the following result.

Theorem 2.4. *If R_1 and R_2 are string-rewriting systems that present isomorphic monoids, then the validation problem for linear sentences is decidable for R_1 if and only if it is decidable for R_2, i.e., the decidability of the validation problem for linear sentences is an invariant property of finitely generated monoids.*

Because of the results of Book [5] and Otto and Zhang [33], this means that, if Σ is a finite alphabet, and R is a string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ can also be presented by a finite, monadic, and confluent string-rewriting system or by a finite, special, and e-confluent string-rewriting system, then the validation problem for linear sentences is decidable for R. In fact, the complexity bounds of Book [5] and Otto and Zhang [33] carry over to R. Thus, for R, the validation problem for linear sentences is decidable in polynomial space, and, for linear sentences that do not contain mixed atomic formulae or that have quantifier prefix $\exists^s \forall^r$, the validation problem is decidable in polynomial time.

3. Deciding weak confluence

Here we establish two results. First we show that, without loss of generality, we can restrict our attention to finite monadic string-rewriting systems that are reduced. Actually, given a finite monadic system R, we construct an equivalent finite monadic system R_1 that is reduced. Observe that we do not require that the system R be confluent, which means that the process for constructing an equivalent reduced system as, e.g., presented in [17] does not suffice. In fact, our construction can be seen as a restricted form of completion, which, however, is guaranteed to terminate. In addition, if R is weakly confluent, then so is R_1. The second result then states that weak confluence is a tractable property for finite monadic string-rewriting systems. We begin with a useful general lemma.

Lemma 3.1. *Let R_1 and R_2 be two string-rewriting systems on Σ such that*

(i) *R_1 is noetherian,*

(ii) *$\rightarrow_{R_2} \subseteq \rightarrow_{R_1}^+$, and*

(iii) *$\mathrm{IRR}(R_1) = \mathrm{IRR}(R_2)$.*

Then, for every word $w \in \Sigma^*$, if R_1 is confluent on $[w]_{R_1}$, then $[w]_{R_1} = [w]_{R_2}$, and R_2 is confluent on $[w]_{R_1}$, too.

Proof. Let $w \in \Sigma^*$ be such that R_1 is confluent on $[w]_{R_1}$. Then there exists a unique word $w_1 \in \mathrm{IRR}(R_1)$ such that $w \leftrightarrow_{R_1}^* w_1$. From (i) and (ii) we see that $[w]_{R_2} \subseteq [w]_{R_1}$, and that R_2 is noetherian, too. Hence, for each word $u \in [w]_{R_1}$, there is some $v \in \mathrm{IRR}(R_2)$ such that $u \to_{R_2}^* v$. By (ii) and (iii) $v = w_1$, implying that $[w]_{R_1} = [w]_{R_2}$, and that R_2 is confluent on $[w]_{R_1}$. □

Let $>$ be an admissible well-founded partial ordering on Σ^*, and let R be a finite string-rewriting system on Σ that is compatible with this ordering. In general, there will not exist a reduced system R_1 that is equivalent to R, and that is compatible with the given ordering. For example, let $>$ be the ordering by length, i.e., $u > v$ iff $|u| > |v|$, and let $R := \{a^2 \to b, a^2 \to c\}$. Obviously, no length-reducing system can both be reduced and equivalent to R. The situation is different when we deal with an admissible well-ordering $>$ on Σ^*. Given a finite string-rewriting system R compatible with $>$, we will construct a finite reduced system R_1 that is also compatible with $>$, and that is equivalent to R. For the first stage of this construction, we need the following lemma, which is easily derived from Lemma 3.1 by standard techniques.

Lemma 3.2. *Let $>$ be an admissible well-ordering on Σ^*, and let R be a string-rewriting system on Σ that is compatible with $>$. For each $w \in \Sigma^*$, let $\hat{w}$ denote some irreducible descendant of w. Then the string-rewriting system $R_0 := \{l \to \hat{r} \mid (l \to r) \in R\}$ has the following properties:*

(i) *R_0 is compatible with $>$,*
(ii) $\to_{R_0} \subseteq \to_R^+$,
(iii) $\mathrm{range}(R_0) \subseteq \mathrm{IRR}(R_0) = \mathrm{IRR}(R)$,
(iv) *R_0 is equivalent to R, and*
(v) *for each word $w \in \Sigma^*$, R is confluent on $[w]_R$ if and only if R_0 is.*

Actually, Lemma 3.2 remains valid even if $>$ is just an admissible well-founded partial ordering. Although $\mathrm{range}(R_0) \subseteq \mathrm{IRR}(R_0)$, the system R_0 will, in general, not be reduced since a left-hand side of a rule of R_0 may contain another left-hand side as a factor. The second stage of our construction now takes care of this situation. We present the full construction in the form of an algorithm.

Algorithm 3.3. Construction of an equivalent reduced system
REDUCE_SYSTEM:
INPUT: An admissible well-ordering $>$ on Σ^*, and a finite string-rewriting system R on Σ compatible with $>$;
begin $R_1 := R$;
reduce the right-hand sides of the rules of R_1 using the first rules applicable;
{* By Lemma 3.2, the resulting system R_1 is equivalent to R, it is compatible with $>$, and range $(R_1) \subseteq \mathrm{IRR}(R_1)$ *}
while $\exists (l_1, r_1), (l_2, r_2) \in R_1 \; \exists x, y \in \Sigma^*$: $l_2 = x l_1 y$ **and** ($xy \neq \mathrm{e}$ **or** $r_2 > r_1$) **do**

begin $R_1 := R_1 \setminus \{l_2 \to r_2\}$;
  if $r_2 \notin \triangle^*_{R_1}(xr_1y)$ **then**
  begin if $xr_1y > r_2$ **then** $R_1 := R_1 \cup \{xr_1y \to r_2\}$
    else $R_1 := R_1 \cup \{r_2 \to xr_1y\}$;
    reduce the right-hand sides of the rules of R_1 using the first rules applicable
  end
end;
OUTPUT: R_1
end.

Based on Lemma 3.2, it is easily seen that, whenever the above algorithm terminates, it correctly computes a finite reduced string-rewriting system R_1 that is equivalent to the input system R, and that is compatible with the given ordering $>$. Thus, it remains to prove the termination of this algorithm.

Let $>$ be the admissible well-ordering on Σ^* that is used in the above algorithm. We extend this ordering to an ordering of finite sets of pairs of words from Σ^* as follows.

Let $x_1, x_2, y_1, y_2 \in \Sigma^*$. We define an ordering $>_2$ on $\Sigma^* \times \Sigma^*$ by setting $(x_1, x_2) >_2 (y_1, y_2)$ if and only if $x_1 > y_1$ or ($x_1 = y_1$ and $x_2 > y_2$). Obviously, $>_2$ is a well-ordering.

Now, for finite subsets $S_1, S_2 \subseteq \Sigma^* \times \Sigma^*$, we use the multiset ordering induced by $>_2$, i.e.,

> $S_1 \gg_2 S_2$ if and only if there exists a nonempty subset $T_1 \subseteq S_1$ and a subset $T_2 \subseteq S_2$ such that $S_2 = (S_1 \setminus T_1) \cup T_2$, and, for each pair $(y_1, y_2) \in T_2$, there is a pair $(x_1, x_2) \in T_1$ satisfying $(x_1, x_2) >_2 (y_1, y_2)$.

Then $\gg_2$ is a well-ordering on the finite subsets of $\Sigma^* \times \Sigma^*$ [9]. Now, for all $i \geqslant 0$, if R_i denotes the string-rewriting system with which the $(i+1)$st execution of the **while**-loop is entered, then $R_i \gg_2 R_{i+1}$. Hence, this **while**-loop is executed only a finite number of times, i.e., algorithm REDUCE_SYSTEM terminates. This proves the following result.

Theorem 3.4. *When given as input an admissible well-ordering $>$ on Σ^*, and a finite string-rewriting system R on Σ compatible with $>$, algorithm* REDUCE_SYSTEM *computes a finite reduced string-rewriting system R_1 that is equivalent to R, and that is also compatible with $>$.*

If the given system R is monadic, and if the length-lexicographical ordering $>_{ll}$ is used, then the reduced system R_1 is also monadic. In addition, if R is weakly confluent, then R_1 is obtained from the system $R_0 = \{l \to \hat{r} \mid (l \to r) \in R\}$ of Lemma 3.2 by simply deleting those rules for which the left-hand side properly contains the left-hand side of another rule as a factor; hence, R_1 is weakly confluent, too. Thus, for each finite,

monadic, and weakly confluent string-rewriting system R, there is a finite, monadic, and weakly confluent system R_1 that is equivalent to R, and that is reduced. In fact, R_1 is uniquely determined by R as shown by the following result, which can be proved along the lines of the corresponding result for systems that are confluent (everywhere) [17].

Theorem 3.5. *Let R_1 and R_2 be two reduced monadic and weakly confluent string-rewriting systems on Σ. If R_1 and R_2 are equivalent, then they are, in fact, identical.*

Thus, for monadic and weakly confluent string-rewriting systems, reduced systems are normal forms.

Theorem 3.5 rests on the fact that we associate a fixed linear ordering $>$ with the alphabet Σ. If we change this linear ordering, we get reduced monadic and weakly confluent string-rewriting systems R_1 and R_2 that are equivalent, and where the one is obtained from the other by a permutation of the alphabet Σ, i.e., there is a bijection $\sigma: \Sigma \to \Sigma$ such that $R_2 = \{\sigma(l) \to \sigma(r) \mid (l \to r) \in R_1\}$.

It is easily seen that for finite monadic systems algorithm REDUCE_SYSTEM runs in polynomial time. In fact, it even gives a partial test for weak confluence. Indeed, if R_0 denotes the system $R_0 = \{l \to \hat{r} \mid (l \to r) \in R\}$, then R_0 (and, therewith, the initial system R) is not weakly confluent if, for some rules $(l_1, r_1), (l_2, r_2) \in R_0$ satisfying $l_2 = x l_1 y$ and ($xy \neq \mathrm{e}$ or $r_2 > r_1$), $x r_1 y$ does not reduce to r_2 modulo R_0. Thus, the reduced system R_1 is simply the subsystem of R_0 obtained by deleting all rules for which the left-hand side properly contains the left-hand side of another rule, or the initial system R is not weakly confluent. In the first case, if R is weakly confluent, then so is R_1, but also the converse implication holds, i.e., R is weakly confluent if and only if R_1 is. Thus, we can combine these observations as follows.

Corollary 3.6. *There is a polynomial-time algorithm that solves the following task:*

Input: A finite monadic string-rewriting system R on Σ.

Task: Either recognize correctly that R is not weakly confluent or determine a finite reduced monadic system R_1 on Σ such that R and R_1 are equivalent, and R is weakly confluent if and only if R_1 is.

In the rest of this section we present a test for deciding whether a finite monadic string-rewriting system is weakly confluent. By Corollary 3.6, we can restrict our attention to finite monadic systems that are reduced.

For the following considerations let R be a fixed system of this form. Recall from Section 2 that the leftmost reduction ${}_{\mathrm{L}}\!\to_R$ induced by R is a deterministic process, and the same holds for the rightmost reduction ${}_{\mathrm{R}}\!\to_R$. For a word $x \in \mathrm{IRR}(R) \setminus \{\mathrm{e}\}$ and a symbol $a \in \Sigma \cap \mathrm{IRR}(R)$ or $a = \mathrm{e}$, we define the *set* $\mathrm{RF}_R(x, a)$ *of right factors of a with*

respect to x and the *set* $\mathrm{LF}_R(x, a)$ *of left factors of* a *with respect to* x as follows:

$$\mathrm{RF}_R(x, a) := \{ y \in \mathrm{IRR}(R) \mid \exists (l \to a) \in R\colon xy \,{}_{\mathrm{L}}\!\to_R^* l \},\ \text{and}$$

$$\mathrm{LF}_R(x, a) := \{ y \in \mathrm{IRR}(R) \mid \exists (l \to a) \in R\colon yx \,{}_{\mathrm{R}}\!\to_R^* l \}.$$

In addition, we take $\mathrm{RF}_R(\mathrm{e}, a) := \{a\}$ and $\mathrm{LF}_R(\mathrm{e}, a) := \{a\}$.

The following lemma shows that these sets are regular, and that we can easily obtain specifications through nondeterministic finite-state acceptors for them.

Lemma 3.7. *The following task can be solved in polynomial time:*

Input: *A finite reduced monadic string-rewriting system* R *on* Σ, *and an irreducible word* $x \in \Sigma^*$

Output: *A collection* $\{\mathfrak{A}_a \mid a \in (\Sigma \cup \{\mathrm{e}\}) \cap \mathrm{IRR}(R)\}$ *of nfsa's* $\mathfrak{A}_a = (Q, \Sigma, q_0, \delta, q_a)$ *that differ only in their accepting states such that, for each* $a \in (\Sigma \cup \{\mathrm{e}\}) \cap \mathrm{IRR}(R)$, $\mathfrak{A}_a$ *accepts the language* $\mathrm{RF}_R(x, a)$.

Proof. Let $a \in (\Sigma \cup \{\mathrm{e}\}) \cap \mathrm{IRR}(R)$. Then $\mathfrak{A}_a$ will be the product of a dfsa $\mathfrak{B}$ accepting the set $\mathrm{IRR}(R)$ and the nfsa $\mathfrak{C}_a := (Q_1, \Sigma, q_1, \delta_1, q_a)$, which is defined as follows:

Let $\mathrm{Pre}(x) := \{x_1 \mid x_1$ is a proper prefix of $x\}$, and let $\mathrm{Sub}(R) := \{z \mid \exists u, v \in \Sigma^*\colon uzv \in \mathrm{dom}(R)\}$. Then the set Q_1 of states of $\mathfrak{C}_a$ is taken to be

$$Q_1 := (\mathrm{Pre}(x) \times (\Sigma \cup \{\mathrm{e}\}) \times \mathrm{Sub}(R)) \cup \{q_b \mid b \in (\Sigma \cup \{\mathrm{e}\}) \cap \mathrm{IRR}(R)\}.$$

Let $x = \tilde{x}_1 c$ for some symbol $c \in \Sigma$. Then the initial state q_1 is $q_1 := [\tilde{x}_1, c, \mathrm{e}]$. Finally, the transition function δ_1 is given through the following equations:

(1) $\delta_1([x_1, b, l_1], d) = \{[x_1, b, l_1 d]\}$ if $x_1 b l_1 d \in \mathrm{IRR}(R)$ and $l_1 d \in \mathrm{Sub}(R)$,

(2) $\delta_1([x_1, b, l_1], d) \ni [x_2, g, \mathrm{e}]$ if $x_1 b l_1 d \,{}_{\mathrm{L}}\!\to_R^+ x_2 g$ and $x_2 g \in \mathrm{IRR}(R)$,

(3) $\delta_1([x_1, b, l_1], d) \ni q_g$ if $(x_1 b l_1 d \to g) \in R$ or $x_1 b l_1 d \,{}_{\mathrm{L}}\!\to_R^+ l$ for some rule $(l \to g) \in R$.

Obviously, $\mathfrak{C}_a$ can be constructed in polynomial time from R and x. Thus, according to the results presented in Section 2, the nfsa $\mathfrak{A}_a = \mathfrak{B} \times \mathfrak{C}_a$ can be obtained in polynomial time. It remains to prove the following claim.

Claim. *Let* $y \in \mathrm{IRR}(R)$. *Then* $q_a \in \delta_1([\tilde{x}_1, c, \mathrm{e}], y)$ *if and only if there is a rule* $(l \to a) \in R$ *such that* $xy \,{}_{\mathrm{L}}\!\to_R^* l$.

Proof. Assume first that $xy \,{}_{\mathrm{L}}\!\to_R^* l$ for some rule $(l \to a) \in R$. Then there exist words $w_0, w_1, \ldots, w_k \in \Sigma^*$ and rules $(l_0 \to r_0), \ldots, (l_{k-1} \to r_{k-1}) \in R$ such that the following conditions are satisfied:

(i) $xy = w_0 \,{}_{\mathrm{L}}\!\to_R w_1 \,{}_{\mathrm{L}}\!\to_R \cdots \,{}_{\mathrm{L}}\!\to_R w_k \equiv l$,

(ii) $w_i = x_i l_i y_i$ and $w_{i+1} = x_i r_i y_i$ for some words $x_0, \ldots, x_{k-1}, y_0, \ldots, y_{k-1} \in \Sigma^*$,

where x_0 is a proper prefix of x, x_i is a prefix of x_{i-1} for $i>0$, y_0 is a proper suffix of y, and y_i is a suffix of y_{i-1} for $i>0$.

From the definition of the transition relation δ_1, it can now be easily verified that, on input y, $\mathfrak{C}_a$ can simulate the above sequence of leftmost reductions, i.e., $q_a \in \delta_1([\tilde{x}_1, c, \mathrm{e}], y)$.

Conversely, if $q_a \in \delta_1([\tilde{x}_1, c, \mathrm{e}], y)$, then $\mathfrak{C}_a$ necessarily mimics a reduction sequence of the above form, i.e., $xy_{\mathrm{L}} \to_R^* l$ for some rule $(l \to a) \in R$. This completes the proof of Lemma 3.7. □

Proof of Lemma 3.7 (*conclusion*). This concludes the proof of Lemma 3.7. □

Analogously, the sets $\mathrm{LF}_R(x, a)$ can be shown to be regular. Using the regular sets of the form $\mathrm{RF}_R(x, a)$ and $\mathrm{LF}_R(x, a)$, a syntactical characterization can be derived for those reduced monadic string-rewriting systems that are weakly confluent. By Theorem 2.3, a reduced monadic system R on Σ is weakly confluent if and only if, for each $a \in \mathrm{range}(R)$ and each pair $(p, q) \in \mathrm{IDUCP}(R)$, the set $\mathrm{SCon}_p(a)$ is contained in $\mathrm{Con}_q(a)$, and $\mathrm{SCon}_q(a)$ is contained in $\mathrm{Con}_p(a)$. For these inclusions syntactical conditions can be established. For example, let $a \in \mathrm{range}(R)$, let $(p, q) \in \mathrm{IDUCP}(R)$, and let $x \# y \in \mathrm{SCon}_p(\mathrm{a})$. Then x and y are irreducible, and $xpq \to_R^* a$. By analyzing this reduction sequence in detail, we see that it must be of one of the following two forms:

- either the factor p is completely used up in the reduction process, i.e., $x = x_1x_2x_3$, $p = p_1p_2p_3$, and $y = y_3y_2y_1$ such that $x_3p_{1\,\mathrm{R}} \to_R^* b$, $p_3y_{3\,\mathrm{L}} \to_R^* c$, $(x_2bp_2cy_2 \to d) \in R$, and $x_1dy_1 \to_R^* a$, or
- a is already a factor of p which is not touched in the reduction process, i.e., $p = p_1ap_2$ such that $xp_{1\,\mathrm{R}} \to_R^* \mathrm{e}$ and $p_2y_{\mathrm{L}} \to_R^* \mathrm{e}$.

In the former case $x_3 \in \mathrm{LF}_R(p_1, b)$ and $y_3 \in \mathrm{RF}_R(p_3, c)$, and, in order for R to be weakly confluent, $\triangle_R^*(x_2x_3qy_3y_2) \cap \mathrm{IRR}(R)$ must be $\{d\}$ since $x_2x_3qy_3y_2 \leftrightarrow_R^* x_2x_3py_3y_2 = x_2x_3p_1p_2p_3y_3y_2 \to_R^* x_2bp_2cy_2 \to_R^* d$. Thus, we obtain the condition that $\triangle_R^*(x_2 \cdot \mathrm{LF}_R(p_1, b) \cdot q \cdot \mathrm{RF}_R(p_3, c) \cdot y_2) \cap \mathrm{IRR}(R) = \{d\}$. In the second case we have $x \in \mathrm{LF}_R(p_1, \mathrm{e})$ and $y \in \mathrm{RF}_R(p_2, \mathrm{e})$ and, hence, we obtain the condition that $\triangle_R^*(\mathrm{LF}_R(p_1, \mathrm{e}) \cdot q \cdot \mathrm{RF}(p_2, \mathrm{e})) \cap \mathrm{IRR}(R) = \{a\}$.

In this way, we get a collection of syntactical conditions stating that certain regular sets must contain only particular irreducible words. For obtaining this collection of conditions, all pairs $(p, q) \in \mathrm{IDUCP}(R)$ and all possible factorizations $p = p_1p_2p_3$ or $p = p_1ap_2$ must be considered. The syntactical characterization thus obtained is a direct, though technically involved, generalization of a characterization of e-confluence for special systems [32].

Each of the above syntactical conditions can be checked in polynomial time by Lemma 3.7. If the system R is finite, then only polynomially many conditions are involved, which yields the following decidability result.

Proposition 3.8. *It is decidable in polynomial time whether a finite monadic string-rewriting system is weakly confluent.*

This extends a result of Otto [32], where it is shown that it is decidable in polynomial time whether a finite special string-rewriting system R is confluent on $[e]_R$.

4. Some undecidability results for weakly confluent monadic string-rewriting systems

For finite special string-rewriting systems that are e-confluent, all the decision problems listed in Section 2 are decidable [33]. Thus, for finite special systems, the property of e-confluence is very useful. Is the property of weak confluence as useful for finite monadic systems? Here we shall answer this question in the negative by presenting some undecidability results for finite monadic string-rewriting systems that are weakly confluent. We begin with a simple example that already indicates some of the technical problems.

If R is a finite, special, and e-confluent system, then $[w]_R \cap \mathrm{IRR}(R)$ is a finite set for each word $w \in \Sigma^*$, and if $L \subseteq \Sigma^*$ is a regular set, then so is the set $[L]_R \cap \mathrm{IRR}(R)$ [33]. For finite, monadic, and weakly confluent string-rewriting systems, the situation is quite different, as shown by the following example.

Example 4.1. Let $\Sigma = \{a, b\}$ and $R = \{aba \to b\}$. Then $[e]_R = \{e\}$, $[a]_R = \{a\}$, and $[b]_R = \{a^n b a^n \mid n \geqslant 0\}$, implying that R is indeed weakly confluent. However, $[b^3]_R = \{a^i b a^j b a^k b a^n \mid i, j, k, n \geqslant 0 \text{ such that } i - j = n - k\}$ and, hence, $[b^3]_R \cap \mathrm{IRR}(R) = \{a^n b^3 a^n \mid n \geqslant 0\}$, which is not only infinite, but even nonregular.

This shows that finite, monadic, and weakly confluent systems will confront us with problems not encountered with finite, special, and e-confluent systems. In fact, many problems that are decidable for the latter kind of systems will turn out to be undecidable for the former. These undecidability results will be proved by a construction that allows one to simulate the computations of a Turing machine through a finite, monadic, and weakly confluent string-rewriting system. We now present this construction.

Let $M = (Q, \Sigma, \mathrm{b}, \Gamma, q_0, q_h, \delta)$ be a single-tape Turing machine (TM), where Q denotes the finite set of states, Σ is the input alphabet, $\mathrm{b} \notin \Sigma$ is the distinguished blank symbol, $\Gamma \supseteq \Sigma \cup \{\mathrm{b}\}$ is the tape alphabet, $q_0 \in Q$ is the initial state, $q_h \in Q$ is the halting state, and δ is the transition function.

A configuration of M is a word of the form $uqav$, where $u, v \in \Gamma^*$, $a \in \Gamma$, and $q \in Q$. By $\models_M$ we denote the single-step computation relation that M induces on the set of configurations, and by $\models_M^*$ we denote its reflexive and transitive closure. Without loss of generality, we may assume that M satisfies the following technical restrictions:

(1) M halts if and only if it enters state q_h, i.e., the transition function δ has the following form:

$$\delta : ((Q \setminus \{q_h\}) \times \Gamma) \to Q \times \Gamma \times \{\lambda, \rho\}.$$

Here λ and ρ denote the operations of moving M's read/write head one cell to the left or to the right, respectively.

(2) When it is started with an initial configuration q_0w, $w\in\Sigma^*$, then M will not move its head to the left of the initial head position, and if M halts eventually, then its tape is empty, and its head has returned to its initial position.

Thus, $L(M)$, the *language accepted* by M, can be described as $L(M)=\{w\in\Sigma^* \mid q_0w \models_M^* q_h\mathrm{b}\}$.

As a first step in our intended construction, we now define a finite string-rewriting system $T:=T_M$ that will simulate the TM M. This system will not be monadic; however, we shall construct a finite monadic system $S:=S_M$ from T in a second step.

Let Γ' be a new set of letters that is in one-to-one correspondence with Γ, and let $':\Gamma^*\to\Gamma'^*$ denote the obvious isomorphism. Further, let $\mathrm{¢}$, \$be two additional letters, and let $Q':=\{[q, a] \mid q\in Q,\ a\in\Gamma\}$ be another new alphabet. We assume that all the various alphabets considered are pairwise-disjoint. Finally, let $\Omega:=Q'\cup\Gamma'\cup\Gamma\cup\{\mathrm{¢}, \$\}$. Then we will encode a configuration $uqav$ of M by all the words in the set $\{\mathrm{¢}u'[q, a]v\mathrm{b}^k\$ \mid k\in\mathbb{N}\}$.

Now $T:=T_M$ is the finite string-rewriting system on Ω that consists of the following two groups of rules:

(1) $[p, a]d\leftrightarrow c'[q, d]$ for all $d\in\Gamma$, if $\delta(p, a)=(q, c, \rho)$;

(2) $d'[p, a]\leftrightarrow[q, d]c$ for all $d'\in\Gamma'$, if $\delta(p, a)=(q, c, \lambda)$.

This construction is similar to the standard way of simulating a TM through a string-rewriting system [8]. The following lemma states that T does indeed simulate the TM M. Its proof is left to the reader.

Lemma 4.2. *For each $a\in\Sigma$ and $w\in\Sigma^*$, the following two statements are equivalent:*

(1) *$aw\in L(M)$, i.e., M halts on input aw eventually.*

(2) $\exists k\geqslant 0$: $\mathrm{¢}[q_0, a]w\mathrm{b}^k\$ \leftrightarrow_T^* \mathrm{¢}[q_h, \mathrm{b}]\mathrm{b}^{|w|+k}\$$.

Now we define a finite monadic string-rewriting system $S:=S_M$. For this we need some more additional letters. So, we take

$$\Pi:=\Omega\cup\{R(p, a, d) \mid p\in Q,\ a\in\Gamma \text{ such that } \delta(p, a)=(q, c, \rho),\ d\in\Gamma\}$$
$$\cup\{L(d, p, a) \mid p\in Q,\ a\in\Gamma \text{ such that } \delta(p, a)=(q, c, \lambda),\ d\in\Gamma\},$$

and define $S:=S_M$ on Π as follows:

(1) $c'R(p, a, d)\to[p, a]$ for all $d\in\Gamma$;
$R(p, a, d)d\to[q, d]$ for all $d\in\Gamma$ — if $\delta(p, a)=(q, c, \rho)$,

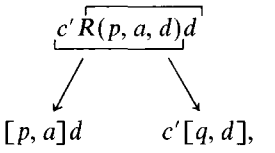
(2) $d'L(d, p, a)\to[q, d]$ for all $d\in\Gamma$;
$L(d, p, a)c\to[p, a]$ for all $d\in\Gamma$ — if $\delta(p, a)=(q, c, \lambda)$.

Since S is defined on a larger alphabet than T, these two systems cannot be equivalent. Nevertheless, they are closely related, as we shall see in the following.

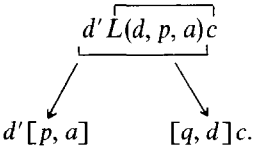
Lemma 4.3. *$T=\mathrm{CP}(S)$, i.e., T is the set of critical pairs of S.*

Proof. From the form of the rules of S, we see that S only admits the following overlaps between its rules:

(1) if $\delta(p, a)=(q, c, \rho)$, and $d\in\Gamma$:



(2) if $\delta(p, a)=(q, c, \lambda)$, and $d\in\Gamma$:



Thus, $T=\mathrm{CP}(S)$. □

Hence, $\leftrightarrow_T^* \subseteq \leftrightarrow_S^*$, i.e., S can simulate T. In fact, we have the following correspondence between $\leftrightarrow_T^*$ and $\leftrightarrow_S^*$.

Lemma 4.4. *$\leftrightarrow_T^* = \leftrightarrow_S^*|_{\Omega^*\times\Omega^*}$, i.e., for all $x, y\in\Omega^*$, $x\leftrightarrow_T^* y$ if and only if $x\leftrightarrow_S^* y$.*

Proof. It remains to prove that, for all $x, y\in\Omega^*$, if $x\leftrightarrow_S^* y$, then $x\leftrightarrow_T^* y$. Following Huet [14], we call the string-rewriting system S *confluent modulo* T, if, for all $x, y\in\Pi^*$, $x\leftrightarrow_{S\cup T}^* y$ implies that there exist $u, v\in\Pi^*$ such that $x\to_S^* u\leftrightarrow_T^* v\,{}_S^*\!\leftarrow y$. Now S has this property if and only if the following two conditions are satisfied [14, Lemma 2.8]:

(A) For all $x, y, z\in\Pi^*$, if $x\to_S y$ and $x\to_S z$, then there are $u, v\in\Pi^*$ such that $y\to_S^* u\leftrightarrow_T^* v\,{}_S^*\!\leftarrow z$, and

(B) for all $x, y, z\in\Pi^*$, if $x\to_S y$ and $x\leftrightarrow_T z$, then there are $u, v\in\Pi^*$ such that $y\to_S^* u\leftrightarrow_T^* v\,{}_S^*\!\leftarrow z$.

By Lemma 4.3, $T=\mathrm{CP}(S)$. Thus, if $x\to_S y$ and $x\to_S z$, then either $y=spt$ and $z=sqt$ for some critical pair (p, q) of S, implying that $y\leftrightarrow_T z$, or $x=rl_1sl_2t$, $y=rr_1sl_2t$ and $z=rl_1sr_2t$ for some rules $(l_1\to r_1), (l_2\to r_2)\in S$. In this situation, $y\to_S rr_1sr_2t\,{}_S\!\leftarrow z$. Thus, S satisfies condition (A).

Concerning condition (B), it can easily be checked that the left-hand side of no rule of S overlaps with any side of any rule of T. Thus, whenever $y\,{}_S\!\leftarrow x\leftrightarrow_T z$, $x=plsut$ or $x=puslt$ for some rules $(l\to r)\in S$ and $(u\leftrightarrow v)\in T$. In the former case, $y=prsut$ and $z=plsvt$ and, so, $y\leftrightarrow_T prsvt\,{}_S\!\leftarrow z$; in the latter case, $y=pusrt$ and $z=pvslt$ and, so, $y\leftrightarrow_T pvsrt\,{}_S\!\leftarrow z$. Thus, S satisfies condition (B) as well, i.e., S is indeed confluent modulo T.

Now, let $x, y\in\Omega^*$, satisfying $x\leftrightarrow_S^* y$. Since S is confluent modulo T, there exist $u, v\in\Pi^*$ such that $x\to_S^* u\leftrightarrow_T^* v\,{}_S^*\!\leftarrow y$. However, $\Omega^*\subseteq \mathrm{IRR}(S)$ and, hence, $x=u$ and $y=v$. Thus, $x\leftrightarrow_T^* y$, i.e., $\leftrightarrow_S^*|_{\Omega^*\times\Omega^*}=\leftrightarrow_T^*$. □

Since the system T contains length-preserving rules only, Lemmas 4.2 and 4.4 yield the following result.

Corollary 4.5. *For each $a\in\Sigma$ and $w\in\Sigma^*$, the following two statements are equivalent:*
(1) $aw\in L(M)$.
(2) $\exists k, l\geqslant 0$: $\mathrm{¢}[q_0, a]w\mathrm{b}^k\$ \leftrightarrow_S^* \mathrm{¢}[q_h, \mathrm{b}]\mathrm{b}^l\$$.

Thus, the finite monadic string-rewriting system S does actually simulate the TM M. To complete the construction, it remains to show the following.

Lemma 4.6. *The finite monadic string-rewriting system S is weakly confluent.*

Proof. Since S is reduced, and since $\mathrm{dom}(S)\subseteq(\Pi\setminus\mathrm{range}(S))^*$, i.e., no letter occurs both in the left-hand side of a rule of S and as the right-hand side of a rule of S, we have $[a]_S=\{a\}\cup\{l\mid(l\to a)\in S\}$ for all $a\in\Pi\cup\{\mathrm{e}\}$. Hence, S is indeed weakly confluent. □

Let $a\in\Sigma$ and $w\in\Sigma^*$. With the word aw we associate the following existential linear sentence over Π:

$$\exists u\ \exists v\colon \mathrm{¢}[q_0, a]wu\$ \equiv \mathrm{¢}[q_h, \mathrm{b}]v\$.$$

If we fix the domains for the existential variables u and v to be the regular set $\{\mathrm{b}\}^*$, then this linear sentence is true as a statement on $\leftrightarrow_S^*$ if and only if $aw\in L(M)$ (Corollary 4.5). Thus, if M is a Turing machine with an undecidable halting problem, it is undecidable, in general, whether a linear sentence is true as a statement on $\leftrightarrow_S^*$. Hence, we have derived the following undecidability result.

Theorem 4.7. *There exists a finite monadic string-rewriting system S such that S is weakly confluent, but the validation problem for (existential) linear sentences is undecidable for S.*

In fact, we can derive an even stronger undecidability result. To this end, we define another finite monadic string-rewriting system $R:=R_M$ as follows: $R:=S_M\cup\{\mathrm{b}\$\to\$\}$. Then R has the following properties.

Lemma 4.8. (a) *R is weakly confluent.*
(b) *For each $a\in\Sigma$ and $w\in\Sigma^*$, $aw\in L(M)$ if and only if $\mathrm{¢}[q_0, a]w\$ \leftrightarrow_R^* \mathrm{¢}[q_h, \mathrm{b}]\$$.*

Proof. (a) It is easily seen that $[p]_R=[p]_S$ for all $p\in(\Pi-\{\$\})\cup\{\mathrm{e}\}$. Further, $[\$]_R=\{\mathrm{b}^m\$\mid m\geqslant 0\}$, and $\mathrm{b}^m\$\to_R^*\$$ for all $m\geqslant 0$. Thus, R is, in fact, weakly confluent. To prove (b), we prove the following claims.

Claim 1. *For each $a\in\Sigma$ and $w\in\Sigma^*$, if $aw\in L(M)$, then $\mathrm{¢}[q_0, a]w\$ \leftrightarrow_R^* \mathrm{¢}[q_h, \mathrm{b}]\$$.*

Proof of Claim 1. By Corollary 4.5, there exist integers $k, l \geqslant 0$ such that $¢[q_0, a]wb^k\$ \leftrightarrow_S^* ¢[q_h, b]b^l\$$. Hence, $¢[q_0, a]w\$ \rightarrow_R^* ¢[q_0, a]wb^k\$ \leftrightarrow_R^* ¢[q_h, b]b^l\$ \rightarrow_R^* ¢[q_h, b]\$$. □

Claim 2. *For each $a \in \Sigma$ and $w \in \Sigma^*$, if $¢[q_0, a]w\$ \leftrightarrow_R^* ¢[q_h, b]\$$, then $aw \in L(M)$.*

Proof of Claim 2. Let $¢[q_0, a]w\$ = ¢u_0'[q_0, a]v_0b^{i_0}\$ \leftrightarrow_R ¢u_1'[q_1, a_1]v_1b^{i_1}\$ \leftrightarrow_R \cdots \leftrightarrow_R ¢u_n'[q_n, a_n]v_nb^{i_n}\$ = ¢[q_h, b]\$$ be an R-derivation. Then there exists an index $\mu \in \{0, 1, \ldots, n\}$ such that $|u_\mu'| + |v_\mu| + i_\mu$ is maximal. For $j = 0, 1, \ldots, n$, let $k_j := (|u_\mu'| + |v_\mu| + i_\mu) - (|u_j'| + |v_j| + i_j)$. Then $¢[q_0, a]wb^{k_0}\$ = ¢u_0'[q_0, a]v_0b^{i_0 + k_0}\$ \leftrightarrow_S^* ¢u_1'[q_1, a_1]v_1b^{i_1 + k_1}\$ \leftrightarrow_S^* \cdots \leftrightarrow_S^* ¢u_n'[q_n, a_n]v_nb^{i_n + k_n}\$ = ¢[q_h, b]b^{k_n}\$$, where each part of this S-derivation consists of 0 or 1 S-step. Hence, by Corollary 4.5, $aw \in L(M)$. □

Proof of Lemma 4.8 (*conclusion*). This concludes the proof of Lemma 4.8. □

Thus, we have the following result.

Theorem 4.9. *There exists a finite monadic string-rewriting system R that is weakly confluent, but that has an undecidable word problem. In fact, there is a word u such that the membership problem for the congruence class $[u]_R$ is undecidable.*

Finally, let us consider the free submonoid problem (cf. Section 2). This problem is a restriction of the following problem, called the *problem of consistency*:

Instance: A finite string-rewriting system R_1 on an alphabet Σ_1, and a finite string-rewriting system R_2 on an alphabet $\Sigma_2 \supseteq \Sigma_1$.

Question: Is R_2 a consistent extension of R_1, i.e., does $\leftrightarrow_{R_2}^*|_{\Sigma_1^* \times \Sigma_1^*} = \leftrightarrow_{R_1}^*$ hold?

Lemma 4.4 shows that the string-rewriting system $S = S_M$ is a consistent extension of the system $T = T_M$. As shown in [30], the problem of consistency is decidable if it is restricted to string-rewriting systems R_1 and R_2 such that R_1 has a decidable word problem, and R_2 is finite, monadic, and confluent, or R_1 presents a group with a decidable word problem, and R_2 is finite, monadic, and e-confluent. On the other hand, the free submonoid problem is undecidable, in general, even for finite monadic systems. The construction which led to Theorem 4.9 can now be used to extend this undecidability result to finite monadic systems that are weakly confluent.

Let $R := R_M$ be the string-rewriting system on Π constructed before Lemma 4.8. Obviously, each word $w \in \Pi^*$ has a unique factorization of the following form:

$$w = u_0v_1u_1 \ldots v_mu_m,$$

where the factors $v_1, \ldots, v_m$ are maximal factors of w that are from the set $\{e, ¢\} \cdot \Gamma'^* \cdot Q'' \cdot \Gamma^* \cdot \{e, \$\}$, and $u_0, u_1, \ldots, u_m \in (\Pi - Q'')^*$. Here $Q'' := Q' \cup \{R(p, a, d) \mid p \in Q, a \in \Gamma$

such that $\delta(p, a)=(q, c, \rho)$, $d\in\Gamma\}\cup\{L(d, p, a)\mid p\in Q,\ a\in\Gamma$ such that $\delta(p, a)=(q, c, \lambda)$, $d\in\Gamma\}$. Further, if $w\leftrightarrow_R^* w'$, then w' has the corresponding factorization

$$w'=u_0'v_1'u_1'\dots v_m'u_m',$$

where $v\leftrightarrow_R^* v_i'$ for $i=1, \dots, m$, and $u_j\leftrightarrow_{R-S}^* u_j'$ for $j=0, 1, \dots, m$. Based on this observation, we can prove the following characterization.

Lemma 4.10. *Let* $\Sigma_1:=\{\mathrm{¢}, \$, [q_0, a], [q_h, \mathrm{b}]\}$, *where* a *is a fixed letter chosen from* Σ. *Then* $\leftrightarrow_R^*|_{\Sigma_1^*\times\Sigma_1^*}=\mathrm{id}_{\Sigma_1^*}$ *if and only if* $a\notin L(M)$.

Proof. If $a\in L(M)$, then $\mathrm{¢}[q_0, a]\$\leftrightarrow_R^*\mathrm{¢}[q_h, \mathrm{b}]\$$ by Lemma 4.8 (b); hence, $\leftrightarrow_R^*|_{\Sigma_1^*\times\Sigma_1^*}\neq\mathrm{id}_{\Sigma_1^*}$. To prove the converse implication, assume that $\leftrightarrow_R^*|_{\Sigma_1^*\times\Sigma_1^*}\neq\mathrm{id}_{\Sigma_1^*}$, and let $w, w'\in\Sigma_1^*$ such that $w\neq w'$ but $w\leftrightarrow_R^* w'$. According to the observation above, w and w' can be factored as $w=u_0v_1u_1\dots v_mu_m$ and $w'=u_0'v_1'u_1'\dots v_m'u_m'$, where $v_1, \dots, v_m, v_1', \dots, v_m'\in\{\mathrm{e}, \mathrm{¢}\}\cdot\{[q_0, a], [q_a, \mathrm{b}]\}\cdot\{\mathrm{e}, \$\}$ and $u_0, u_1, \dots, u_m, u_0', u_1', \dots, u_m'\in\{\mathrm{¢}, \$\}^*$. Further, $v_i\leftrightarrow_R^* v_i'$ for $i=1, \dots, m$ and $u_j\leftrightarrow_{R-S}^* u_j'$ for $j=0, 1, \dots, m$. Since $R-S=\{\mathrm{b}\$\to\$\}$, we see that $u_j=u_j'$, $j=0, 1, \dots, m$. Hence, $w\neq w'$ implies that there exists an index $i\in\{1, \dots, m\}$ such that $v_i\neq v_i'$. Since ¢- and \$-symbols can neither be generated nor deleted, this means that v_i begins with a ¢-symbol if and only if v_i' does, and v_i ends with a \$-symbol if and only if v_i' does. Thus, $v_i=\alpha\cdot[q_0, a]\cdot\beta$ and $v_i'=\alpha\cdot[q_h, \mathrm{b}]\cdot\beta$, where $\alpha\in\{\mathrm{e}, \mathrm{¢}\}$ and $\beta\in\{\mathrm{e}, \$\}$, or vice versa. Since $v_i\leftrightarrow_R^* v_i'$, we also have $\mathrm{¢}[q_0, a]\$\leftrightarrow_R^*\mathrm{¢}[q_h, \mathrm{b}]\$$, which implies that $a\in L(M)$ by Lemma 4.8(b). □

Fix a letter a. Given a single-tape TM $M=(Q, \Sigma, \mathrm{b}, \Gamma, q_0, q_h, \delta)$ such that $a\in\Sigma$, the string-rewriting system $R=R_M$ can be constructed effectively. Since it is undecidable, in general, whether a TM M accepts on input a, Lemma 4.10 yields the following undecidability result.

Theorem 4.11. *The free submonoid problem is undecidable for finite, monadic, and weakly confluent string-rewriting systems.*

Comparing this undecidability result with the decidability results of [30] mentioned above, we observe that the trivial system $R_1=\emptyset$ on Σ_1 certainly has a decidable word problem, but it does not present a group unless $\Sigma_1=\emptyset$. Thus, this additional hypothesis is crucial for the decidability of the problem of consistency restricted to finite, monadic, and e-confluent systems R_2.

5. Some decidability results for context-free groups

In the previous section we have seen that many decision problems are undecidable for finite, monadic, and weakly confluent string-rewriting systems. On the other hand, since the free monoid problem reduces to the task of determining all letters that are

congruent to the empty word [28], this problem is decidable for these systems, in fact, in linear space. Further, the monoid $\mathfrak{M}_R$ presented by a finite, monadic, and c-confluent string-rewriting system R on Σ is a group if and only if $\mathrm{RF}_R(a, \mathrm{e}) \neq \emptyset$ for each letter $a \in \Sigma \cap \mathrm{IRR}(R)$. By Lemma 3.7, these sets are regular, and nfsas for them can be constructed in polynomial time. Thus, the group problem is tractable for the class of finite, monadic, and e-confluent string-rewriting systems. In what follows, we will be concerned with some decision problems for the class of finite, monadic, and e-confluent string-rewriting systems that present groups. The class of groups thus presented has been characterized algebraically as well as through language-theoretic means.

Let R be a finite string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. This group is called *context-free* if the congruence class $[\mathrm{e}]_R \subseteq \Sigma^*$ is a context-free language. The class of context-free groups, which is defined through a language-theoretic property, has been characterized algebraically by Muller and Schupp [22] using a result of Dunwoody [11].

Theorem 5.1. *A finitely generated group is context-free if and only if it is virtually free.*

A group $\mathfrak{G}$ is called *virtually free* if it contains a free subgroup of finite index. If R is a finite, monadic, and e-confluent string-rewriting system on Σ, then $[\mathrm{e}]_R = \nabla_R^*(\mathrm{e})$ and, hence, $[\mathrm{e}]_R$ is a context-free language [4]. Thus, if a group $\mathfrak{M}_R$ is presented by a finite, monadic, and e-confluent string-rewriting system R, then $\mathfrak{M}_R$ is a context-free group. In fact, the following characterization holds.

Theorem 5.2 (Autebert et al. [1]). *A group $\mathfrak{G}$ has a presentation of the form $(\Sigma; R)$, where R is a finite, monadic, and e-confluent string-rewriting system on Σ, if and only if $\mathfrak{G}$ is a finitely generated context-free group.*

In what follows, we shall show that all the decision problems listed in Section 2 become decidable when they are restricted to finite, monadic, and e-confluent string-rewriting systems presenting groups. Thus, all these problems are decidable for context-free groups. In fact, we shall see that the presentations involving finite, monadic, and e-confluent string-rewriting systems provide a uniform approach for solving them.

In Section 3 we have seen that the process of reducing a finite, monadic, and weakly confluent string-rewriting system R involves the following two simple steps only:

(1) Replace each right-hand side by some irreducible descendant of it.

(2) Delete each rule the left-hand side of which properly contains the left-hand side of another rule.

Now let R be a finite monadic string-rewriting system on Σ that is e-confluent, and let the finite monadic system R_0 be obtained from R through the above two steps. Then R_0 is reduced, $[\mathrm{e}]_R = [\mathrm{e}]_{R_0}$, and R_0 is e-confluent by Lemma 3.1. In general, R and R_0 will not be equivalent. For example, consider the system $R = \{abc \to b, ab \to a\}$.

The two steps above result in the system $R_0=\{ab\to a\}$, which obviously is not equivalent to R. However, if the monoid $\mathfrak{M}_R$ presented by $(\Sigma; R)$ is a group, the situation is different, since in this case $[e]_R=[e]_{R_0}$ already implies that $\leftrightarrow_R^*=\leftrightarrow_{R_0}^*$, i.e., R_0 is equivalent to R. Thus, when we consider a finite, monadic, and e-confluent string-rewriting system presenting a group, the two steps (1) and (2) above still suffice to reduce this system. In particular, when talking about finite, monadic, and e-confluent string-rewriting systems presenting groups, we may always assume that these systems are reduced.

Lemma 3.7 shows that, for each word $x\in\mathrm{IRR}(R)\setminus\{e\}$, the set $\mathrm{RF}_R(x, e)=\{y\in\mathrm{IRR}(R)\mid\exists(l\to e)\in R: xy_L\to_R^* l\}$ is regular if R is a finite reduced monadic string-rewriting system. In fact, from R and x, an nfsa for this set can be constructed in polynomial time. If, in addition, R is e-confluent, then $\mathrm{RF}_R(x, e)=\{y\in\mathrm{IRR}(R)\mid xy\leftrightarrow_R^* e\}$, i.e., $\mathrm{RF}_R(x, e)$ simply consists of all irreducible right inverses of x.

Consider, e.g., the finite monadic string-rewriting system $R=\{ab\to a, ac\to e\}$ on $\Sigma=\{a, b, c\}$. This system is confluent and, therewith, it is, in particular, e-confluent. Further, $\mathrm{IRR}(R)=\{b, c\}^*\cdot\{a\}^*$, while $\mathrm{RF}_R(a, e)=\{b\}^*\cdot\{c\}$. Thus, in general, the sets of the form $\mathrm{RF}_R(x, e)$ are infinite. However, this changes when we restrict our attention to finite, monadic, and e-confluent string-rewriting systems that present groups.

Lemma 5.3. *Let R be a finite, monadic, and e-confluent string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. Then, for each word $x\in\Sigma^+$, the set $\{y\in\mathrm{IRR}(R)\mid xy\leftrightarrow_R^* e\}$ is finite.*

Proof. Let x_0 be an irreducible word such that $x_0\leftrightarrow_R^* x$. Then $\{y\in\mathrm{IRR}(R)\mid xy\leftrightarrow_R^* e\}=\{y\in\mathrm{IRR}(R)\mid x_0y\leftrightarrow_R^* e\}$. If $x_0=e$, this set contains only the word e, otherwise, it is the set $\mathrm{RF}_R(x_0, e)$, which is regular by Lemma 3.7. Assume that $\mathrm{RF}_R(x_0, e)$ is infinite. Then, by the pumping lemma for regular sets, $\mathrm{RF}_R(x_0, e)$ contains a subset of the form $\{uv^iw\mid i\geqslant 0\}$, where $v\in\Sigma^+$. Thus, $x_0uvw\leftrightarrow_R^* e\leftrightarrow_R^* x_0uw$, which implies that $v\leftrightarrow_R^* e$, since $\mathfrak{M}_R$ is a group. Hence, $v\to_R^* e$, contradicting the fact that $\mathrm{RF}_R(x_0, e)$ contains irreducible words only. Thus, the set $\{y\in\mathrm{IRR}(R)\mid xy\leftrightarrow_R^* e\}$ is finite for each word $x\in\Sigma^+$. □

Let R be a finite, monadic, and e-confluent string-rewriting system on Σ, let $a\in\Sigma$ be irreducible, and let $y\in\mathrm{IRR}(R)$ be such that $ay\to_R^* e$, where $|y|$ is minimal with this property. Then $ay=au_1\dots u_r\to_R b_1u_2\dots u_r\to_R\cdots\to_R b_{r-1}u_r\to_R e$ for some $r\geqslant 1$, where $b_1, \dots, b_{r-1}\in\Sigma$, and $(au_1\to b_1), (b_1u_2\to b_2), \dots, (b_{r-2}u_{r-1}\to b_{r-1})$, $(b_{r-1}u_r\to e)\in R$. From the minimality of y, we see that all the letters $a, b_1, \dots, b_{r-1}$ are pairwise-distinct and, so, $r\leqslant|\Sigma|$. Thus, $|y|<|\Sigma|\cdot\lambda$, where $\lambda:=\max\{|l|\mid l\in\mathrm{dom}(R)\}$. Hence, if $\mathfrak{M}_R$ is a group, then, for each letter $a\in\Sigma$, there exists an irreducible word $u_a\in\Sigma^*$ such that $|u_a|<|\Sigma|\cdot\lambda$, and $au_a\to_R^* e$. Define a function $^{-1}:\Sigma^*\to\Sigma^*$ through $e^{-1}:=e$, and $(wa)^{-1}:=u_aw^{-1}$ for all $w\in\Sigma^*$ and $a\in\Sigma$. Then $ww^{-1}\to_R^* e\,{}_R^*\!\leftarrow w^{-1}w$ for all $w\in\Sigma^*$, i.e., w^{-1} is a *formal inverse* of w. From R and Σ this function can be constructed in polynomial time. Since for all $u, v\in\Sigma^*$, $u\leftrightarrow_R^* v$ if and only if $uv^{-1}\to_R^* e$, we thus have the following result.

Theorem 5.4. *The following problem is decidable in polynomial time:*

Instance: *A finite, monadic, and* e*-confluent string-rewriting system R on Σ such that the monoid $\mathfrak{M}_R$ is a group, and two words $u, v \in \Sigma^*$.*
Question: *Does $u \leftrightarrow_R^* v$ hold?*

Theorem 5.4 shows that the uniform word problem for context-free groups is decidable in polynomial time if these groups are presented through finite, monadic, and e-confluent string-rewriting systems. In fact, a careful analysis reveals that the word problem for a fixed system R of this form is decidable in time $O(|u|+|v|)$, while the uniform word problem is decidable in time $O((|u|+|v|)\cdot|R|^2)$, where $|R| = \sum_{l \in \mathrm{dom}(R)} |l|$.

Now, for a word $w \in \Sigma^*$, let $I_R(w)$ denote the set of all irreducible words that are congruent to w, i.e., $I_R(w) = [w]_R \cap \mathrm{IRR}(R)$. For finite, monadic, and weakly confluent string-rewriting systems, in general, sets of this form can be nonregular, as we saw in Example 4.1. However, for finite, monadic, and e-confluent systems presenting groups, the situation is different.

Lemma 5.5. *Let R be a finite, monadic, and* e*-confluent string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. Then the set $I_R(w)$ is finite for each word $w \in \Sigma^*$.*

Proof. Let $w \in \Sigma^+$. Since $\mathfrak{M}_R$ is a group, there exists an irreducible word v such that $vw \leftrightarrow_R^* e$. Now let $x \in I_R(w)$, i.e., $x \in \mathrm{IRR}(R)$ and $x \leftrightarrow_R^* w$. Then $vx \leftrightarrow_R^* vw \leftrightarrow_R^* e$, implying that $I_R(w) \subseteq \{y \in \mathrm{IRR}(R) \mid vy \leftrightarrow_R^* e\}$. By Lemma 5.3, this latter set is finite, and, thus, so is the set $I_R(w)$. □

Since each congruence class contains finitely many irreducible words only, the monoid $\mathfrak{M}_R$ is finite if and only if the set $\mathrm{IRR}(R)$ is finite. This gives the following decidability result.

Corollary 5.6. *For finite, monadic, and* e*-confluent string-rewriting systems presenting groups, the finiteness problem is decidable in polynomial time.*

One of the main technical results of this section is now obtained as a generalization of Lemma 5.5. For a regular set $S \subseteq \Sigma^*$, let $I_R(S)$ denote the set $I_R(S) := [S]_R \cap \mathrm{IRR}(R)$. Since $I_R(S) = I_R(\triangle_R^*(S) \cap \mathrm{IRR}(R))$, and since the set $\triangle_R^*(S) \cap \mathrm{IRR}(R)$ is regular, if the set S is regular, we can restrict our attention in the sequel to regular sets S of irreducible words.

Lemma 5.7. *Let R be a finite, monadic, and* e*-confluent string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. Then, for each regular set $S \subseteq \Sigma^*$, the set $I_R(S)$ is regular. In addition, from R and an nfsa for the set S, an nfsa for $I_R(S)$ can be constructed in polynomial time.*

Proof. Let $S \subseteq \mathrm{IRR}(R)$ be a regular set that is specified through some nfsa $\mathfrak{A}_1$. Since the monoid $\mathfrak{M}_R$ is a group, we can determine a homomorphism $^{-1}: \Sigma^* \to \Sigma^*$ such that $ww^{-1} \leftrightarrow_R^* \mathrm{e} \leftrightarrow_R^* w^{-1}w$ holds for each word $w \in \Sigma^*$. Let $S^{-1} := \{w^{-1} \mid w \in S\}$, and let $\mathrm{INV}(S) \subseteq \Sigma^*$ be defined through $\mathrm{INV}(S) := \wedge_R^*(S^{-1}) \cap \mathrm{IRR}(R)$. Then this set $\mathrm{INV}(S)$ satisfies the following properties:

(i) $\forall u \in S\ \exists v \in \mathrm{INV}(S)$: $uv \leftrightarrow_R^* \mathrm{e} \leftrightarrow_R^* vu$.

(ii) $\forall v \in \mathrm{INV}(S)\ \exists u \in S$: $uv \leftrightarrow_R^* \mathrm{e} \leftrightarrow_R^* vu$.

(iii) The set $\mathrm{INV}(S)$ is regular, and from $\mathfrak{A}_1$ and R, an nfsa $\mathfrak{A}_2$ for this set can be constructed in polynomial time.

For each nonempty word $w \in \mathrm{IRR}(R)$, the following statements are equivalent:

$$
\begin{aligned}
w \in I_R(S) \quad & \text{iff } \exists u \in S\colon u \leftrightarrow_R^* w, \\
& \text{iff } \exists v \in \mathrm{INV}(S)\colon wv \leftrightarrow_R^* \mathrm{e} \text{ (by properties (i) and (ii) above)}, \\
& \text{iff } \exists v \in \mathrm{INV}(S)\colon wv \to_R^* \mathrm{e} \text{ (since } R \text{ is e-confluent)}.
\end{aligned}
$$

For each pair of letters $a, b \in \Sigma$, there are only finitely many irreducible words z satisfying $bz \to_R^* a$. In fact, if $z \in \mathrm{IRR}(R)$ satisfies $bz \to_R^* a$, then $|z| \leqslant \lambda \cdot |\Sigma|$, where $\lambda := \max\{|l| \mid l \in \mathrm{dom}(R)\}$. To see this, assume that $|z| > \lambda \cdot |\Sigma|$. Since z is irreducible, $bz \to_R^* a$ implies that there exist $c \in \Sigma$ and $z_1, z_2, z_3 \in \Sigma^*$, $z_2 \neq \mathrm{e}$, such that $z = z_1 z_2 z_3$, $bz_1 \to_R^* c$, $cz_2 \to_R^* c$, and $cz_3 \to_R^* a$. Since $\mathfrak{M}_R$ is a group, $cz_2 \to_R^* c$ implies that $z_2 \leftrightarrow_R^* \mathrm{e}$, and since R is e-confluent, this yields that $z_2 \to_R^* \mathrm{e}$, contradicting the fact that z is irreducible.

If $w, v \in \mathrm{IRR}(R)$ satisfy $wv \to_R^* \mathrm{e}$, then w and v can be factored as follows:

- $w = y_m \dots y_1 y_0$, and
- $v = x_0 z_0 x_1 z_1 \dots x_m z_m$

such that $(y_0 x_0 \to b_1) \in R$, $b_1 z_0 \to_R^* a_1 \in \Sigma \cup \{\mathrm{e}\}$, $(y_1 a_1 x_1 \to b_2) \in R$, $b_2 z_1 \to_R^* a_2 \in \Sigma \cup \{\mathrm{e}\}, \dots,$ $(y_{m-1} a_{m-1} x_{m-1} \to b_m) \in R$, $b_m z_{m-1} \to_R^* a_m \in \Sigma \cup \{\mathrm{e}\}$, $(y_m a_m x_m \to b_{m+1}) \in R$, and $b_{m+1} z_m \to_R^* \mathrm{e}$. If $b_i = \mathrm{e}$ for some i, then $z_{i-1} = \mathrm{e}$ and $a_i = \mathrm{e}$, too. Using the above observation that there are only finitely many irreducible words z satisfying $bz \to_R^* a$ for any pair $a, b \in \Sigma$, we can construct a generalized sequential machine (gsm) $\mathfrak{G}$ such that, for each word $w \in \Sigma^*$, the set $\mathfrak{G}(w)$ of possible outputs of $\mathfrak{G}$ on input w is the following:

$$
\mathfrak{G}(w) = \begin{cases}
\emptyset & \text{if } w \text{ is reducible mod } R, \\
\{\mathrm{e}\} & \text{if } w = \mathrm{e}, \\
\{v \mid \exists m \in \mathbb{N}\ \exists y_0, y_1, \dots, y_m \in \Sigma^+\ \exists x_0, \dots, x_m, z_0, \dots, z_m \in \Sigma^* \\
\quad \exists b_1, \dots, b_{m+1}, a_1, \dots, a_m \in \Sigma \cup \{\mathrm{e}\}\colon w = y_m \dots y_1 y_0, \\
\quad v = \rho(z_m)\rho(x_m) \dots \rho(z_0)\rho(x_0), (y_0 x_0 \to b_1) \in R, (y_1 a_1 x_1 \to b_2) \in \\
\quad R, \dots, (y_m a_m x_m \to b_{m+1}) \in R, b_1 z_0 \to_R^* a_1, \dots, b_m z_{m-1} \to_R^* a_m, \\
\quad \text{and } b_{m+1} z_m \to_R^* \mathrm{e}\} & \text{if } w \in \mathrm{IRR}(R) \setminus \{\mathrm{e}\}.
\end{cases}
$$

Here $\rho: \Sigma^* \to \Sigma^*$ denotes the function *reversal*.

Now $I_R(S) \cap \Sigma^+ = \{w \mid \rho(\mathfrak{G}(w)) \cap \mathrm{INV}(S) \neq \emptyset\} \cap \Sigma^+$. Since $\rho(\mathfrak{G}(\mathrm{e})) = \{\mathrm{e}\}$, and since $\mathrm{e} \in \mathrm{INV}(S)$ if and only if $\mathrm{e} \in I_R(S)$, we see that $I_R(S) = \{w \mid \rho(\mathfrak{G}(w)) \cap \mathrm{INV}(S) \neq \emptyset\} =$

$\{w \mid \exists v \in \rho(\mathrm{INV}(S)): v \in \mathfrak{G}(\omega)\} = \mathfrak{G}^{-1}(\rho(\mathrm{INV}(S)))$. Thus, the set $I_R(S)$ is indeed regular, and an nfsa $\mathfrak{U}_3$ for this set can be constructed in polynomial time. □

Using Lemma 5.7, we can now carry over Book's [5] original proof of the decidability of the validation problem for linear sentences from finite, monadic, and confluent string-rewriting systems to finite, monadic, and e-confluent systems presenting groups. In fact, let φ be a linear sentence over Σ containing the variables $v_1, \ldots, v_p \in V_E \cup V_U$, and let $S_1, \ldots, S_p \subseteq \Sigma^*$ be regular sets. Assume that φ contains only the single atomic formula $t_1 \equiv t_2$. With t_1 and t_2 we can associate regular sets T_1 and T_2, respectively, by replacing each variable occurrence of a variable v_j in t_i by the corresponding regular set S_j. Then the validity of φ under the interpretation induced by R and $S_1, \ldots, S_p$ can be expressed in terms of the regular sets $I_R(T_1)$ and $I_R(T_2)$. For example, if the atomic formula $t_1 \equiv t_2$ is existential, then φ is true under the interpretation induced by R and $S_1, \ldots, S_p$ if and only if $I_R(T_1) \cap I_R(T_2) \neq \emptyset$, and the similar for the other cases. This gives the following result.

Theorem 5.8. *The validation problem for linear sentences with respect to a finite, monadic, and* e*-confluent string-rewriting system presenting a group is decidable.*

It should be noted that the complexity results of Book [5] remain valid in this setting (cf. the remark following Theorem 2.4).

Corollary 5.9. *The validation problem for linear sentences is decidable for finitely generated context-free groups.*

Thus, if $(\Sigma; R)$ is a finitely generated presentation of a context-free group, then there is an algorithm that solves uniformly all those decision problems for $(\Sigma; R)$ that can be expressed through linear sentences. The generalized word problem and the inclusion problem are examples of decision problems that can be expressed in this way (cf. Section 2).

There is one other decidability result that we want to discuss in short, and for which we need the following notions.

Let R be a string-rewriting system on Σ. A word $w \in \Sigma^*$ presents a *nontrivial idempotent* of the monoid $\mathfrak{M}_R$ if it satisfies $\mathrm{e} \not\leftrightarrow_R^* w \leftrightarrow_R^* w^2$. More generally, w presents a *nontrivial element of finite order* of $\mathfrak{M}_R$ if $w \not\leftrightarrow_R^* \mathrm{e}$, but there are integers $k \geqslant 0$ and $n \geqslant 1$ such that $w^{k+n} \leftrightarrow_R^* w^k$ holds. If $\mathfrak{M}_R$ does not contain any nontrivial elements of finite order, then $\mathfrak{M}_R$ is said to be *torsion-free*.

In general, it is undecidable whether a monoid $\mathfrak{M}_R$ that is given through a finite presentation $(\Sigma; R)$ contains a nontrivial idempotent, or whether it contains a nontrivial element of finite order. On the other hand, Narendran and Otto [23] have shown that both these problems are decidable in polynomial time when they are restricted to presentations that involve finite, length-reducing, and confluent string-rewriting systems. While it is open whether the former problem is also decidable for

the class of presentations that involve finite, monadic, and weakly confluent string-rewriting systems (in fact, we conjecture that it is undecidable in this setting), the arguments of [23, Section 4] easily carry over to finite, monadic, and e-confluent systems that present groups. In particular, this yields the following decidability result.

Theorem 5.10. *The following problem is decidable in polynomial time:*

Instance: *A finite, monadic, and* e-*confluent string-rewriting system* R *on* Σ *such that* $\mathfrak{M}_R$ *is a group, and a word* $u \in \Sigma^*$.
Question: *Is* u *an element of finite order of* R?

Further, it can be shown that, if R is a finite, monadic, and e-confluent system on Σ such that the monoid $\mathfrak{M}_R$ is a group, then this monoid contains an element of finite order if and only if there is a prefix u of the left-hand side of a rule of R such that u presents an element of finite order. Hence, we get the following from Theorem 5.10.

Theorem 5.11. *It is decidable in polynomial time whether a group* $\mathfrak{M}_R$ *that is given through a finite, monadic, and* e-*confluent string-rewriting system is torsion-free.*

Since a virtually free group is a free group if and only if it is torsion-free, we can state this result also in the following form.

Corollary 5.12. *The following problem is decidable in polynomial time:*

Instance: *A finite, monadic, and* e-*confluent string-rewriting system* R *on* Σ *such that* $\mathfrak{M}_R$ *is a group.*
Question: *Is* $\mathfrak{M}_R$ *a free group*?

In this section we have so far dealt only with finite monadic string-rewriting systems that present groups, and that are e-confluent. How do these systems relate to finite, monadic, and weakly confluent systems presenting groups? Are there Thue congruences that can be presented by the former but not by the latter kind of systems? And, on the other hand, are there decision problems that can easily be solved for the latter but not for the former kind of systems? Based on Lemma 5.5, we now show that these two kinds of systems are, in fact, very closely related.

Theorem 5.13. *Let* R *be a finite, monadic, and* e-*confluent string-rewriting system on* Σ *such that the monoid* $\mathfrak{M}_R$ *is a group. Then there exists a finite monadic string-rewriting system* R_0 *on* Σ *that is weakly confluent and equivalent to* R. *In fact,* R_0 *can be determined effectively from* R.

Proof. Let R be a finite, monadic, and e-confluent string-rewriting system on Σ presenting a group, and let $a \in \Sigma$ such that R is not confluent on $[a]_R$. By Lemma 5.5, the

set $I_R(a)$ is finite, say $I_R(a)=\{a, u_1, \ldots, u_k\}$. Without loss of generality, we may assume that $u_i \gg_{\ell\ell} a$ for $i=1, \ldots, k$, where $>_{\ell\ell}$ denotes the length-lexicographical ordering on Σ^*. Consider the string-rewriting system $R_1 := R \cup \{u_1 \to a, \ldots, u_k \to a\}$. Then R_1 is finite and monadic and, obviously, $\leftrightarrow_{R_1}^* = \leftrightarrow_R^*$, i.e., R and R_1 are equivalent. For each word $w \in [a]_{R_1}$, we have $w \to_R^* a$, or $w \to_R^* u_i \to_{R_1} a$ for some $i \in \{1, \ldots, k\}$. Thus, R_1 is confluent on $[a]_{R_1}$. Finally, let $b \in \Sigma \cup \{e\}$ such that R is confluent on $[b]_R$. Since $R \subseteq R_1$, and since R and R_1 are equivalent, we can conclude that R_1 is confluent on $[b]_{R_1}$, too. Thus, by iterating this process, we eventually get a finite monadic string-rewriting system R_0 on Σ such that R_0 is equivalent to R, and R_0 is confluent on $[a]_{R_0}$ for all $a \in \Sigma \cup \{e\}$. □

Hence, we can conclude from Theorem 5.2 that a group $\mathfrak{G}$ has a presentation of the form $(\Sigma; R)$, where R is a finite, monadic, and weakly confluent string-rewriting system on Σ if and only if $\mathfrak{G}$ is a finitely generated context-free group [1].

6. A specialized completion procedure for monadic string-rewriting systems presenting groups

Each finitely presented group $\mathfrak{G}$ has a presentation through some finite special string-rewriting system. If $\mathfrak{G}$ is context-free, then it even has a presentation through some finite monadic string-rewriting system that is weakly confluent. As we just saw, many decision problems can be solved in a uniform way for context-free groups that are given through presentations of this form. Thus, given a context-free group $\mathfrak{G}$ through some finite presentation, it would be desirable to be able to construct a presentation of this particular form for $\mathfrak{G}$. Obviously, this task is similar to the one that led to the development of the Knuth–Bendix completion procedure [19]. Accordingly, the algorithm developed in this section, which attempts to solve this task, can be seen as a specialized completion procedure.

In Section 3 we have described a test for weak confluence for finite monadic string-rewriting systems. Although being polynomial-time, this test is technically rather involved. Therefore, we first develop a simpler test that exploits the fact that here we are dealing only with finite monadic systems presenting groups. This test then will serve as a basis for the specialized completion procedure.

Let R be a finite, reduced, monadic string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. For each $a \in \Sigma$, let $u_a \in \Sigma^*$ be such that $au_a \leftrightarrow_R^* e$; in fact, by the results of [28], these words can be found effectively.

Lemma 6.1. *R is e-confluent if and only if, for all $a \in \Sigma$ and all $w \in I_R(u_a)$, $aw \to_R^* e$.*

Proof. Let $a \in \Sigma$ and $w \in I_R(u_a)$, i.e., $w \in \mathrm{IRR}(R)$ and $w \leftrightarrow_R^* u_a$. Hence, $aw \leftrightarrow_R^* au_a \leftrightarrow_R^* e$ and, thus, $aw \to_R^* e$ if R is e-confluent. Conversely, assume that $aw \to_R^* e$ for all $a \in \Sigma$ and all $w \in I_R(u_a)$, and let $v \in [e]_R$. If $v \neq e$, then $v = av_1$ for some $a \in \Sigma$ and, hence, $v \to_R^* av_2$

for some $v_2 \in \mathrm{IRR}(R)$. Now, $au_a \leftrightarrow_R^* \mathrm{e} \leftrightarrow_R^* v = av_1 \leftrightarrow_R^* av_2$, and, since $\mathfrak{M}_R$ is a group, this means that $u_a \leftrightarrow_R^* v_2$. Thus, $v_2 \in I_R(u_a)$ and, hence, $v = av_1 \rightarrow_R^* av_2 \rightarrow_R^* \mathrm{e}$ by the hypothesis. □

Unfortunately, the sets $I_R(u_a)$ $(a \in \Sigma)$ might, in general, not be easy to construct. Therefore, we shall use some approximations for them.

Definition 6.2. For $u \in \Sigma^*$, the *set of simple right inverses of* u, $\mathrm{RI}_R(u)$, consists of all words $v \in \Sigma^*$ such that there is a reduction sequence $uv = u_k \ldots u_1 v_1 \ldots v_k \rightarrow_R u_k \ldots u_2 a_1 v_2 \ldots v_k \rightarrow_R \cdots \rightarrow_R u_k a_{k-1} v_k \rightarrow_R \mathrm{e}$, no step of which is performed entirely within u or within v, i.e., for $u \in \Sigma^+$,

$$\begin{aligned}\mathrm{RI}_R(u) := \{ v \in \Sigma^* \mid \exists k \geqslant 1 \ \exists u_1, \ldots, u_k, v_1, \ldots, v_k \in \Sigma^* \ \exists a_1, \ldots, a_{k-1} \in \Sigma \cup \{\mathrm{e}\}: \\ u = u_k \ldots u_1, \\ v = v_1 \ldots v_k, u_1 \neq \mathrm{e} \neq v_1, (u_1 v_1 \rightarrow a_1) \in R, (u_2 a_1 v_2 \rightarrow a_2) \in R, \ldots, \\ (u_k a_{k-1} v_k \rightarrow \mathrm{e}) \in R, \text{ and, whenever } a_{i-1} = \mathrm{e}, u_i \neq \mathrm{e} \neq v_i \},\end{aligned}$$

and

$$\mathrm{RI}_R(\mathrm{e}) := \{\mathrm{e}\}.$$

For an irreducible word u, the set $\mathrm{RI}_R(u)$ can be seen as an extension of the set $\mathrm{RF}_R(u, \mathrm{e})$ defined in Section 3. Accordingly, the proof of the following lemma is similar to the proof of Lemma 3.7.

Lemma 6.3. *Let R be a finite, reduced, monadic string-rewriting system on Σ, and let $u \in \Sigma^*$. Then $\mathrm{RI}_R(u)$ is a regular set, and, from R and u, an nfsa for $\mathrm{RI}_R(u)$ can be constructed in polynomial time.*

Proof. Let $\mathrm{Pre}(u) := \{u_1 \mid u_1 \text{ is a proper prefix of } u\}$, and let $\mathrm{Sub}(R) := \{z \mid \exists x, y \in \Sigma^+ : xzy \in \mathrm{dom}(R)\}$. We define an nfsa $\mathfrak{U}(u) = (Q, \Sigma, q_0, q_a, \delta)$ as follows:

- $Q := \mathrm{Pre}(u) \times (\Sigma \cup \{\mathrm{e}\}) \times \mathrm{Sub}(R)$ is the finite set of states,
- Σ is taken as the input alphabet,
- $q_0 := (u_1, a, \mathrm{e})$ is the initial state, where $u = u_1 a$, $a \in \Sigma$,
- $q_a := (\mathrm{e}, \mathrm{e}, \mathrm{e})$ is the accepting state, and
- δ is the transition relation, which is defined through
  $(u_2, b, v_2) \in \delta((u_1, a, v_1), c)$ if and only if
  (α) $(u_1 = u_2$ and $b = a$ and $v_2 = v_1 c \in \mathrm{Sub}(R))$ or
  (β) $(\exists k \geqslant 1 \ \exists x_0, x_k \in \Sigma^* \ \exists x_2, \ldots, x_{k-1} \in \Sigma^+ \ \exists a_1, \ldots, a_{k-1} \in \Sigma: u_1 = x_k x_{k-1} \ldots x_1 x_0$, $x_0 a \neq \mathrm{e}, (x_0 a v_1 c \rightarrow a_1) \in R, (x_1 a_1 \rightarrow a_2) \in R, \ldots, (x_{k-1} a_{k-1} \rightarrow b) \in R, u_2 = x_k$, and $v_2 = \mathrm{e})$.

Case (α) corresponds to the situation that no reduction is performed on the word $u_1 a v_1 c$ $(= u_2 b v_2)$, while case (β) corresponds to the situation that $u_1 a v_1 c =$

$x_k x_{k-1} \dots x_1 x_0 a v_1 c$ and reductions are performed, none of which takes place entirely within $u_1 a$ or within $v_1 c$, i.e., $u_1 a v_1 c = x_k x_{k-1} \dots x_1 x_0 a v_1 c \to_R x_k x_{k-1} \dots x_1 a_1 \to_R \cdots \to_R x_k x_{k-1} a_{k-1} \to_R x_k b = u_2 b v_2$. It can easily be verified that $q_a = (e, e, e) \in \delta((u_1, a, e), v) = \delta(q_0, v)$ if and only if $v \in \mathrm{RI}_R(u)$. Thus, $L(\mathfrak{U}(u)) = \mathrm{RI}_R(u)$ and, obviously, $\mathfrak{U}(u)$ is obtained in polynomial time from R and u. □

Using various sets of the form $\mathrm{RI}_R(u)$, we now formulate another characterization of the property of e-confluence for monadic string-rewriting systems presenting groups.

Theorem 6.4. *Let R be a monadic string-rewriting system on Σ such that $\mathfrak{M}_R$ is a group. Then R is e-confluent if and only if the following two conditions are satisfied:*

(1) $\forall a \in \Sigma: (\triangle_R^*(\mathrm{RI}_R(a) \cdot a) \cap \mathrm{IRR}(R)) \setminus \{e\} = \emptyset$, *and*

(2) $\forall (p, q) \in \mathrm{IDUCP}(R): (\triangle_R^*(q \cdot \mathrm{RI}_R(p)) \cap \mathrm{IRR}(R)) \setminus \{e\} = \emptyset = (\triangle_R^*(p \cdot \mathrm{RI}_R(q)) \cap \mathrm{IRR}(R)) \setminus \{e\}$.

Proof. First let us show that conditions (1) and (2) are necessary for R to be e-confluent. So, assume that R is e-confluent. If $a \in \Sigma$ and $v \in \mathrm{RI}_R(a)$, then $av \to_R^* e$ by definition of $\mathrm{RI}_R(a)$. Since $\mathfrak{M}_R$ is a group, $va \leftrightarrow_R^* e$ and, so, $\triangle_R^*(va) \subseteq [e]_R$. Since R is e-confluent, this means that $\triangle_R^*(va) \cap \mathrm{IRR}(R) = \{e\}$. Thus, condition (1) holds. If $(p, q) \in \mathrm{IDUCP}(R)$, and $v \in \mathrm{RI}_R(p)$, then $pv \to_R^* e$. Hence, $qv \leftrightarrow_R^* pv \leftrightarrow_R^* e$, which yields $\triangle_R^*(qv) \cap \mathrm{IRR}(R) = \{e\}$ since R is e-confluent. It follows that condition (2) also holds.

To prove that conditions (1) and (2) are also sufficient to guarantee that R is e-confluent, we now assume that they hold for the system R under consideration.

Claim 1. $\nabla_R^*(e)$ *is closed under cyclic permutation.*

Proof of Claim 1. Assume to the contrary that this is not the case, and let $x \in \Sigma^*$ be a word that is minimal with respect to the length-lexicographical ordering $>_{\ell\ell}$, and that satisfies the following condition:

- there is a cyclic permutation x_1 of x such that $x_1 \to_R^* e$, and there is a cyclic permutation x_2 of x such that $x_2 \not\to_R^* e$.

Then there is a cyclic permutation $y = az$ of x, where $a \in \Sigma$ and $z \in \Sigma^*$, such that $y = az \to_R^* e$ while $za \not\to_R^* e$. In the reduction sequence $az \to_R^* e$, no step is entirely performed within the factor a or within the factor z since, otherwise, x would not be minimal. Hence, $z \in \mathrm{RI}_R(a)$. Condition (1), thus, implies that $za \to_R^* e$, contradicting our choice of x. Hence, $\nabla_R^*(e)$ is closed under cyclic permutation. □

Proof of Theorem 6.4 (*continued*). Using the fact that $\nabla_R^*(e)$ is closed under cyclic permutation, we now complete the proof of the theorem by establishing the following claim.

Claim 2. *R is* e*-confluent.*

Proof. Let $(p, q) \in \mathrm{IDUCP}(R)$. By Theorem 2.3, it suffices to show that $\mathrm{SCon}_p(\mathrm{e}) \subseteq \mathrm{Con}_q(\mathrm{e})$ and $\mathrm{SCon}_q(\mathrm{e}) \subseteq \mathrm{Con}_p(\mathrm{e})$. If $x \# y \in \mathrm{SCon}_p(\mathrm{e})$, then $xpy \to_R^* \mathrm{e}$. By Claim 1, $\nabla_R^*(\mathrm{e})$ is closed under cyclic permutation and, so, $pyx \to_R^* \mathrm{e}$. Since p is irreducible, and since R is monadic, this means that there exists a word $w \in \triangle_R^*(yx)$ such that $pw \to_R^* \mathrm{e}$, and no step in this reduction sequence is performed entirely within p or within w. Thus, $w \in \mathrm{RI}_R(p)$. By condition (2), this means that $qw \to_R^* \mathrm{e}$ and, therewith, $qyx \to_R^* qw \to_R^* \mathrm{e}$. Again, by Claim 1, this means that $xqy \to_R^* \mathrm{e}$, implying that $x \# y \in \mathrm{Con}_q(\mathrm{e})$. Hence, $\mathrm{SCon}_p(\mathrm{e}) \subseteq \mathrm{Con}_q(\mathrm{e})$ and, by symmetry, $\mathrm{SCon}_q(\mathrm{e}) \subseteq \mathrm{Con}_p(\mathrm{e})$. Thus, R is indeed e-confluent. □

Proof of Theorem 6.4 (*conclusion*). This concludes the proof of Theorem 6.4. □

For finite R the set $\mathrm{IDUCP}(R)$ of irreducible leftmost descendants of the unresolvable critical pairs of R can be determined in polynomial time. Hence, Proposition 2.1(c) and Lemma 6.3 yield the following decidability result.

Corollary 6.5. *The following problem is decidable in polynomial time:*

Instance: A finite monadic string-rewriting system R on Σ such that $\mathfrak{M}_R$ is a group.
Question: Is R e*-confluent?*

If R is not e-confluent, then certainly R is not weakly confluent. If, however, R is e-confluent, then the property of weak confluence can easily be characterized for R.

Theorem 6.6. *Let R be a monadic and* e*-confluent string-rewriting system on Σ such that $\mathfrak{M}_R$ is a group, and, for each letter $a \in \Sigma$, let a^{-1} denote an irreducible word such that $aa^{-1} \leftrightarrow_R^* \mathrm{e}$. Then R is weakly confluent if and only if, for each letter $a \in \Sigma \cap \mathrm{IRR}(R)$, $\mathrm{RI}_R(a^{-1}) \cap \mathrm{IRR}(R) = \{a\}$.*

Proof. Let $a \in \Sigma \cap \mathrm{IRR}(R)$. If $w \in \mathrm{RI}_R(a^{-1})$, then $a^{-1}w \to_R^* \mathrm{e} \leftrightarrow_R^* a^{-1}a$, implying that $\mathrm{RI}_R(a^{-1}) \subseteq [a]_R$. On the other hand, if $w \in [a]_R \cap \mathrm{IRR}(R)$, then $a^{-1}w \leftrightarrow_R^* a^{-1}a \leftrightarrow_R^* \mathrm{e}$ and, so, $a^{-1}w \to_R^* \mathrm{e}$ since R is e-confluent. However, a^{-1} and w are both irreducible and, hence, $w \in \mathrm{RI}_R(a^{-1})$. Thus, $\mathrm{RI}_R(a^{-1}) \cap \mathrm{IRR}(R) = [a]_R \cap \mathrm{IRR}(R)$, i.e., R is confluent on $[a]_R$ if and only if $\mathrm{RI}_R(a^{-1}) \cap \mathrm{IRR}(R) = \{a\}$. □

Theorems 6.4 and 6.6 give a fairly simple polynomial-time algorithm for deciding whether or not a finite monadic string-rewriting system presenting a group is weakly confluent. In general, this test is substantially simpler than the one for finite monadic systems described in Section 3.

Using these theorems, we now present a procedure for *weak completion* of finite monadic string-rewriting systems presenting groups, i.e., if R is a system of this form, and R is not weakly confluent, then this procedure attempts to construct a weakly confluent monadic system R_0 that is equivalent to R when R is given as the input.

So, let R be a finite monadic string-rewriting system on Σ such that $\mathfrak{M}_R$ is a group. We may assume that R is reduced. If condition (1) of Theorem 6.4 is not satisfied, there is a letter $a \in \Sigma$ such that the set $E_a := (\triangle_R^*(\mathrm{RI}_R(a) \cdot a) \cap \mathrm{IRR}(R)) \setminus \{\mathrm{e}\}$ is nonempty. Since $E_a \subseteq [\mathrm{e}]_R$, we need to add rules such that, with the help of these additional rules, each word $w \in E_a$ reduces to e. In general, the set E_a will be infinite. Does this mean that we may have to add infinitely many rules to achieve this goal? Fortunately, this is not the case. The set E_a is regular, and, using Proposition 2.1(c) and Lemma 6.3, we can construct an nfsa $\mathfrak{U}(E_a)$ for this set. Consider some word $w \in E_a$. If there is an accepting computation of $\mathfrak{U}(E_a)$ on input w such that during this computation no state of $\mathfrak{U}(E_a)$ is visited more than once, then w is the label of a simple accepting path in the state graph of $\mathfrak{U}(E_a)$. In particular, $|w|$ is bounded from above by the number of states of $\mathfrak{U}(E_a)$. Hence, there are only finitely many words $w \in E_a$ of this form. By GENSPATH, we denote a procedure that from $\mathfrak{U}(E_a)$ extracts all the words of this form. If, on input w, there is no accepting computation of $\mathfrak{U}(E_a)$ of this form, then w can be factored as $w = xyz$, $y \neq \mathrm{e}$, such that there is an accepting computation of $\mathfrak{U}(E_a)$ on input w such that the processing of y corresponds to the traversal of a simple loop in the state graph of $\mathfrak{U}(E_a)$. Hence, $xz \in E_a$ as well and, thus, $xyz \leftrightarrow_R^* \mathrm{e} \leftrightarrow_R^* xz$, implying that $y \leftrightarrow_R^* \mathrm{e}$. By GENSLOOP, we denote a procedure that from $\mathfrak{U}(E_a)$ extracts all the words that correspond to simple loops within accepting computations of $\mathfrak{U}(E_a)$. By adding all the rules $\{w \to \mathrm{e} \mid w \in \mathrm{GENSPATH}(\mathfrak{U}(E_a)) \cup \mathrm{GENSLOOP}(\mathfrak{U}(E_a))\}$ to R, we obtain a finite monadic system that is equivalent to R such that all the words in E_a are reducible with respect to this system.

If condition (2) of Theorem 6.4 is not satisfied, then, for some pair $(p, q) \in \mathrm{IDUCP}(R)$, the set $S(p) := (\triangle_R^*(q \cdot \mathrm{RI}_R(p)) \cap \mathrm{IRR}(R)) \setminus \{\mathrm{e}\}$ or the set $S(q) := (\triangle_R^*(p \cdot \mathrm{RI}_R(q)) \cap \mathrm{IRR}(R)) \setminus \{\mathrm{e}\}$ is nonempty. Since $S(p), S(q) \subseteq [\mathrm{e}]_R$, and since these sets are again regular, we would add the rules $\{w \to \mathrm{e} \mid w \in \mathrm{GENSPATH}(\mathfrak{U}(S(p))) \cup \mathrm{GENSLOOP}(\mathfrak{U}(\mathrm{S}(p)))\}$ and $\{w \to \mathrm{e} \mid w \in \mathrm{GENSPATH}(\mathfrak{U}(S(q))) \cup \mathrm{GENSLOOP}(\mathfrak{U}(S(q)))\}$, respectively, to R.

Observe that so far we have introduced only special rules. This, of course, is a consequence of the fact that Theorem 6.4 gives conditions for testing e-confluence for R. Finally, let $a^{-1} \in \mathrm{IRR}(R)$ denote an irreducible inverse of the letter a, i.e., $aa^{-1} \leftrightarrow_R^* \mathrm{e} \leftrightarrow_R^* a^{-1}a$. If the set $L_a := (\mathrm{RI}_R(a^{-1}) \cap \mathrm{IRR}(R)) \setminus \{a\}$ is nonempty, then the above considerations show that we would add the rules $\{w \to a \mid w \in \mathrm{GENSPATH}(\mathfrak{U}(L_a))\}$, since $L_a \subseteq [a]_R$, and the rules $\{y \to \mathrm{e} \mid y \in \mathrm{GENSLOOP}(\mathfrak{U}(L_a))\}$, since $xyz \leftrightarrow_R^* a \leftrightarrow_R^* xz$ implies that $y \leftrightarrow_R^* \mathrm{e}$.

Now we are prepared to present the announced procedure. It will contain two main subroutines: NORMALIZATION and CONTEXT_RESOLVING. The first one reduces the actual system by applying algorithm REDUCE_SYSTEM (3.3), while the second one introduces new rules as explained above.

Procedure 6.7.
WEAK_COMPLETION:
INPUT: A finite monadic string-rewriting system R on Σ such that the monoid $\mathfrak{M}_R$ is a group;
begin $i:=0$; $R_i:=R$;
for each $a\in\Sigma\cap\mathrm{IRR}(R_i)$ **do** compute an irreducible inverse a^{-1} of a;
NORMALIZATION: $R_i:=\mathrm{REDUCE\_SYSTEM}(R_i)$;
CONTEXT_RESOLVING:
(0) **for** each $a\in\Sigma\cap\mathrm{IRR}(R_i)$ **do** $a^{-1}:=$ some irreducible descendant of $a^{-1} \bmod R_i$;
compute $\mathrm{IDUCP}(R_i)$;
$R_i':=\emptyset$;
(1) **for** all $a\in\Sigma$ **do**
begin compute an nfsa $\mathfrak{U}(E_a)$ for the set $E_a:=(\triangle_{R_i}^*(\mathrm{RI}_{R_i}(a)\cdot a)\cap\mathrm{IRR}(R_i))\setminus\{\mathrm{e}\}$;
$R_i':=R_i'\cup\{l\to\mathrm{e}\mid l\in\mathrm{GENSPATH}(\mathfrak{U}(E_a))\cup\mathrm{GENSLOOP}(\mathfrak{U}(E_a))\}$
end;
(2) **for** all $(p,q)\in\mathrm{IDUCP}(R_i)$ **do**
begin compute an nfsa $\mathfrak{U}(S(p))$ for the set $S(p):=(\triangle_{R_i}^*(q\cdot\mathrm{RI}_{R_i}(p))\cap\mathrm{IRR}(R_i))\setminus\{\mathrm{e}\}$;
compute an nfsa $\mathfrak{U}(S(q))$ for the set $S(q):=(\triangle_{R_i}^*(p\cdot\mathrm{RI}_{R_i}(q))\cap\mathrm{IRR}(R_i))\setminus\{\mathrm{e}\}$;
$\hat{S}(p):=\mathrm{GENSPATH}(\mathfrak{U}(S(p)))\cup\mathrm{GENSLOOP}(\mathfrak{U}(S(p)))$;
$\hat{S}(q):=\mathrm{GENSPATH}(\mathfrak{U}(S(q)))\cup\mathrm{GENSLOOP}(\mathfrak{U}(S(q)))$;
$R_i':=R_i'\cup\{l\to\mathrm{e}\mid l\in\hat{S}(p)\cup\hat{S}(q)\}$
end;
(3) **for** all $a\in\Sigma\cap\mathrm{IRR}(R_i)$ **do**
begin compute an nfsa $\mathfrak{U}(L_a)$ for the set $L_a:=(\mathrm{RI}_{R_i}(a^{-1})\cap\mathrm{IRR}(R_i))\setminus\{a\}$;
$R_i':=R_i'\cup\{l\to a\mid l\in\mathrm{GENSPATH}(\mathfrak{U}(L_a))\}\cup\{l\to\mathrm{e}\mid l\in\mathrm{GENSLOOP}(\mathfrak{U}(L_a))\}$
end;
comment: The new rules are now collected in R_i'. The left-hand side as well as the right-hand side of each rule in R_i' is R_i-irreducible;
(4) **if** $R_i'\neq\emptyset$ **then**
begin $R_{i+1}:=R_i\cup R_i'$;
$i:=i+1$;
goto NORMALIZATION
end;
comment: When we get past the test in (4), then R_i is weakly confluent and reduced;
OUTPUT: R_i
end.

When given input a finite monadic string-rewriting system R on Σ such that $\mathfrak{M}_R$ is a group, procedure WEAK_COMPLETION computes a sequence of finite monadic systems $R_0, R_1, R_2, \ldots$, where R_i denotes the system obtained immediately after executing algorithm REDUCE_SYSTEM with index i. Then the following properties hold for all $i\geqslant 0$:
– R_i is equivalent to R,

- $\mathrm{IRR}(R_{i+1}) \subsetneq \mathrm{IRR}(R_i)$, and
- R_i is reduced.

From this observation, we can easily derive the following correctness result.

Lemma 6.8. *Let R be a finite monadic string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. If procedure* WEAK_COMPLETION *terminates on input R, then it generates a finite monadic system R_i on Σ that is equivalent to R, weakly confluent, and reduced.*

Proof. If procedure WEAK_COMPLETION terminates on input R, then for some $i \geqslant 0$ the set R_i' is empty. Thus, neither in (1), nor in (2), nor in (3) is a new rule generated, i.e., R_i satisfies the conditions of Theorems 6.4 and 6.6. Hence, R_i is weakly confluent. □

Thus, whenever procedure WEAK_COMPLETION terminates, the system constructed has indeed all the nice properties we are looking for. It remains to verify that this procedure is also complete in the sense that it succeeds on input R whenever a finite monadic system S exists that is weakly confluent and equivalent to R. However, let us first go through a detailed example to illustrate how procedure WEAK_COMPLETION works.

Example 6.9. Let $\Sigma = \{a, b, c\}$ and $R = \{ab \to \mathrm{e},\ ba \to \mathrm{e},\ c^2 \to \mathrm{e},\ cac \to b\}$. Then $\mathfrak{M}_R$ is a group, and the system R is monadic and reduced. Hence, when running procedure WEAK_COMPLETION on input R, $R_0 = R$ is the actual system when the subroutine CONTEXT_RESOLVING is executed for the first time. For the irreducible inverses of the letters in Σ, we take $a^{-1} := b$, $b^{-1} := a$, and $c^{-1} := c$. Further, $\mathrm{IDUCP}(R_0) = \{(ac, cb), (bc, ca)\}$.

Next the sets E_a, E_b, and E_c are computed:

$$\mathrm{RI}_{R_0}(a) = \{b\} \text{ and, so, } E_a = (\triangle^*_{R_0}(\mathrm{RI}_{R_0}(a) \cdot a) \cap \mathrm{IRR}(R_i)) \setminus \{\mathrm{e}\} = \emptyset,$$

$$\mathrm{RI}_{R_0}(b) = \{a\} \text{ and, so, } E_b = \emptyset, \text{ and}$$

$$\mathrm{RI}_{R_0}(c) = \{c, aca\} \text{ and, so, } E_c = \emptyset, \text{ too.}$$

Thus, no rule is generated in step (1).

Then the sets $S(p)$ and $S(q)$, $(p, q) \in \mathrm{IDUCP}(R_0)$, are constructed:

$$\mathrm{RI}_{R_0}(ac) = \{cb, ac, acab\} \text{ and, so, } S(ac) = (\triangle^*_{R_0}(cb \cdot \mathrm{RI}_{R_0}(ac)) \cap \mathrm{IRR}(R_0)) \setminus \{\mathrm{e}\} = \{cbcb\},$$

$$\mathrm{RI}_{R_0}(cb) = \{ac, aaca\} \text{ and, so, } S(cb) = \{acaaca\},$$

$$\mathrm{RI}_{R_0}(bc) = \{ca, acaa\} \text{ and, so, } S(bc) = \{caacaa\}, \text{ and}$$

$$\mathrm{RI}_{R_0}(ca) = \{ca, bc, baca\} \text{ and, so, } S(ca) = \{bcbc\}.$$

Thus, in step (2) we get the system $R_0' = \{cbcb \to e, bcbc \to e, acaaca \to e, caacaa \to e\}$.

Finally, the sets L_a, L_b, and L_c are constructed:

$L_a = (\mathrm{RI}_{R_0}(a^{-1}) \cap \mathrm{IRR}(R_0)) \setminus \{a\} = \emptyset$, $L_b = \emptyset$, and $L_c = \{aca\}$.

Hence, the rule $aca \to c$ is added to R_0'.

Since $R_0' \neq \emptyset$, $R_1 := R_0 \cup R_0' = \{ab \to e, ba \to e, c^2 \to e, cac \to b, aca \to c, cbcb \to e, bcbc \to e, acaaca \to e, caacaa \to e\}$ is formed, and we return to the subroutine NORMALIZATION. On input R_1, the algorithm REDUCE_SYSTEM deletes the last two rules from R_1, i.e., $R_1 = \{ab \to e, ba \to e, c^2 \to e, cac \to b, aca \to c, cbcb \to e, bcbc \to e\}$ is the actual system when we enter the subroutine CONTEXT_RESOLVING for the second time.

For a^{-1}, b^{-1}, and c^{-1}, we get the same words as before, and $\mathrm{IDUCP}(R_1) = \{(ac, cb), (bc, ca), (cbc, a), (bcb, c), (bbcb, ca), (bcbb, ac)\}$.

Next the sets E_a, E_b, and E_c are recomputed:

$$\mathrm{RI}_{R_1}(a) = \{b, cac, caaca, caaccbc, cabcb\} \text{ and, so, } E_a = \emptyset,$$

$$\mathrm{RI}_{R_1}(b) = \{a, cbc\} \text{ and, so, } E_b = \emptyset, \text{ and}$$

$$\mathrm{RI}_{R_1}(c) = \{c, aca, accbc, bcb\} \text{ and, so, } E_c = \emptyset.$$

Thus, no rules are generated in step (1).

Next the sets $S(p)$ and $S(q)$, $(p, q) \in \mathrm{IDUCP}(R_1)$, should be computed. However, these sets are too large to be listed here and, therefore, we simply skip this step. We shall later argue why this does not change the resulting system. Instead, we proceed to reconstruct the sets L_a, L_b, and L_c:

$$L_a = (\mathrm{RI}_{R_1}(a^{-1}) \cap \mathrm{IRR}(R_1)) \setminus \{a\} = \{cbc\},\ L_b = \emptyset, \text{ and } L_c = \{bcb\}.$$

Hence, $R_1' = \{cbc \to a, bcb \to c\}$ and, so, $R_2 = R_1 \cup R_1' = \{ab \to e, ba \to e, c^2 \to e, cac \to b, aca \to c, cbcb \to e, bcbc \to e, cbc \to a, bcb \to c\}$.

Algorithm REDUCE_SYSTEM applied to R_2 yields the new reduced system $R_2 = \{ab \to e, ba \to e, c^2 \to e, cac \to b, aca \to c, cbc \to a, bcb \to c\}$. As it turns out, this system satisfies all the conditions of Theorems 6.4 and 6.6. Thus, R_2 is a finite monadic string-rewriting system that is equivalent to the input system R, that is reduced, and that is weakly confluent. By Theorem 3.5, R_2 is uniquely determined by R. Thus, any additional rules that procedure WEAK_COMPLETION might have introduced based on the sets $S(p)$ and $S(q)$, $(p, q) \in \mathrm{IDUCP}(R_1)$, would have been deleted again when reducing the system R_2. Hence, procedure WEAK_COMPLETION terminates on input R with the system R_2 displayed above.

Does procedure WEAK_COMPLETION terminate successfully for each finite monadic string-rewriting system that presents a context-free group? Unfortunately, we must answer this question in the negative, as the following example shows.

Example 6.10. Let $\Sigma = \{a, b, c\}$, and $R = \{ab \to e, ba \to e, c^3 \to e, c^2ac \to a, c^2bc \to b\}$. Then the monoid $\mathfrak{M}_R$ is isomorphic to the group $\mathbb{Z} \times \mathbb{Z}_3$, the direct product of the free group of rank 1, $\mathbb{Z}$, with the cyclic group of order 3, $\mathbb{Z}_3$, which is context-free. For all

$n \geq 1$, $ca^n c^2 b^n \leftrightarrow_R^* e \leftrightarrow_R^* cb^n c^2 a^n$. Since $bc^2 a \leftrightarrow_R^* c^2 \leftrightarrow_R^* ac^2 b$, no factor u of $ca^n c^2 b^n$ or $cb^n c^2 a^n$ satisfying $1 < |u| \leq n$ is congruent to any word $d \in \Sigma \cup \{e\}$. Thus, there is no finite monadic system S that is both equivalent to R and e-confluent. Hence, on input R, procedure WEAK_COMPLETION will not terminate because of Lemma 6.8, although $\mathfrak{M}_R$ is a context-free group. By Theorems 5.2 and 5.13, there must exist a finite, monadic, and weakly confluent string-rewriting system S on some alphabet $\triangle$ such that $(\triangle; S)$ is a presentation of $\mathfrak{M}_R$. In fact, by introducing a new letter d and the rules $c^2 \to d$, $cd \to e$, $dc \to e$, $d^2 \to c$ and the rules $axb \to x$, $bxa \to x$ $(x \in \{c, d\})$, such a presentation is obtained.

Since procedure WEAK_COMPLETION generates only string-rewriting systems that are equivalent to the given input system, it cannot terminate successfully if this system is not equivalent to any finite, monadic, and weakly confluent system. The above example shows that this happens even for presentations of context-free groups. However, we claim that procedure WEAK_COMPLETION succeeds whenever a finite, monadic, and weakly confluent system exists that is equivalent to the given input system. The proof of this completeness result will be based on the following technical result.

Lemma 6.11. *Let R be a finite monadic string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. If procedure* WEAK_COMPLETION *does not terminate on input R, then it enumerates an infinite monadic system R_∞ that is reduced, equivalent to R, and weakly confluent.*

Proof. Assume that procedure WEAK_COMPLETION does not terminate on input R. Then it enumerates an infinite sequence $R_0, R_1, R_2, \ldots$ of finite monadic string-rewriting systems, where, for each $i \geq 0$, R_i denotes the system obtained immediately after executing algorithm REDUCE_SYSTEM with index i. As observed before, these systems have the following properties:

(a) R_i is equivalent to R,
(b) $\mathrm{IRR}(R_{i+1}) \subsetneq \mathrm{IRR}(R_i)$, and
(c) R_i is reduced.

The lemma will now be proved by establishing a sequence of claims.

Claim 1. *If a rule $(l \to r)$ is deleted from R_i for some $i \geq 0$ during the subroutine* NORMALIZATION, *then no rule with left-hand side l is ever introduced again.*

Proof of Claim 1. It is only in the subroutine NORMALIZATION that rules are deleted. Assume the rule $(l \to r)$ is deleted while the system R_i is being reduced. Then $l \notin \mathrm{IRR}(R_i)$; so, by (b), $l \notin \mathrm{IRR}(R_j)$ for all $j \geq i$. However, if $(l' \to r') \in R'_j$ for some $j \geq i$, then, by construction, $l' \in \mathrm{IRR}(R_j)$. Thus, $l' \neq l$, i.e., no rule with left-hand side l is ever introduced again. □

Proof of Lemma 6.11 (*continued*). We now define the following system:

$$R_\infty := \{l \to r \mid \exists j \geqslant 0\ \forall i \geqslant j\colon (l \to r) \in R_i\},$$

i.e., R_∞ is the set of persistent rules. Because of Claim 1, R_∞ is an infinite monadic string-rewriting system on Σ. Procedure WEAK_COMPLETION can be interpreted as enumerating this system. However, this is not an effective enumeration since the procedure does not identify the persistent rules. We shall prove that the system R_∞ has all the properties required. As a first step towards proving this, we establish an important property of the algorithm REDUCE_SYSTEM.

Claim 2. *Let S be a finite monadic string-rewriting system on Σ, let T be the reduced system obtained from S through algorithm* REDUCE_SYSTEM, *and let $u, v \in \Sigma^*$ such that $u \to_S v$. Then there is a sequence $u = u_0 \leftrightarrow_T u_1 \leftrightarrow_T u_2 \leftrightarrow_T \cdots \leftrightarrow_T u_n = v$ such that, for each $i = 0, 1, \ldots, n-1$, one of the following conditions holds:*

(i) $\exists x, y \in \Sigma^*\ \exists (l \to r) \in T\colon u_i = xly \to_T xry = u_{i+1}$, *or*

(ii) $\exists x, y \in \Sigma^*\ \exists (b \to a) \in T\colon b \in \Sigma$, $u_i = xay$, and $u_{i+1} = xby$,

i.e., to transform u into v mod *T, only the rules of the form $(b \to a) \in T$, $b \in \mathrm{dom}(T) \cap \Sigma$, need to be used from right to left.*

Proof of Claim 2. The system S is transformed into the system T by a sequence of steps of the following forms:

(α) A rule $(l \to r)$ is replaced by a rule $(l \to \hat{r})$, where $r \to_{S \setminus \{l \to r\}} \hat{r}$; or

(β) a rule $(l_2 \to r_2)$ is deleted, if there is another rule $(l_1 \to r_1)$ such that $l_2 = xl_1y$ and $xr_1y \to^*_{S \setminus \{l_2 \to r_2\}} r_2$; or

(γ) a rule $(l_2 \to r_2)$ is replaced by the rule $(xr_1y \to r_2)$, where $(l_1 \to r_1)$ is another rule such that $l_2 = xl_1y$ and $xr_1y >_{\ell\ell} r_2$; or

(δ) a rule $(l_2 \to r_2)$ is replaced by the rule $(r_2 \to xr_1y)$, where $(l_1 \to r_1)$ is another rule such that $l_2 = xl_1y$ and $r_2 >_{\ell\ell} xr_1y$.

Let $S = S_0, S_1, S_2, \ldots, S_m = T$ denote the sequence of finite monadic string-rewriting systems that the algorithm REDUCE_SYSTEM generates on input S, where, for $j = 0, 1, \ldots, m-1$, S_{j+1} is obtained from S_j through a single step of the form (α), (β), (γ), or (δ).

Let $u, v \in \Sigma^*$ such that $u \to_S v$. We claim that, for each $j \in \{1, \ldots, m\}$, there is a sequence $u \leftrightarrow_{S_j} \cdots \leftrightarrow_{S_j} v$ satisfying conditions (i) and (ii) of Claim 2. To verify this claim, let $j \in \{1, \ldots, m\}$, and let $u, v \in \Sigma^*$. It suffices to prove the following claim.

Claim 3. *If $u \to_{S_j} v$, or if $u = xay \leftarrow_{S_j} xby = v$ for some rule $(b \to a) \in S_j \cap (\Sigma \times (\Sigma \cup \{e\}))$, then there is a sequence $u = u_0 \leftrightarrow_{S_{j+1}} u_1 \leftrightarrow_{S_{j+1}} \cdots \leftrightarrow_{S_{j+1}} u_n = v$ that satisfies conditions* (i) *and* (ii) *above.*

Proof of Claim 3. S_j is transformed into S_{j+1} through a single step of the form (α), (β), (γ), or (δ). By checking each of these four cases in turn, it is easily established that the reduction step $u \to_{S_j} v$ or the reverse reduction step $u = xay \leftarrow_{S_j} xby = v$ can indeed be

simulated through a finite sequence of steps mod S_{j+1} that satisfies conditions (i) and (ii). □

Proof of Claim 2 (*conclusion*). Now Claim 2 follows immediately by induction on j. □

Using Claim 2, we can now establish the following claim.

Claim 4. *For all* $i \geqslant 0$ *and all* $a \in \Sigma \cup \{e\}$, *if* $a \in \mathrm{IRR}(R_{i+1})$, *then* $\nabla^*_{R_i}(a) \subseteq \nabla^*_{R_{i+1}}(a)$.

Proof of Claim 4. Let $i \geqslant 0$, let $a \in (\Sigma \cup \{e\}) \cap \mathrm{IRR}(R_{i+1})$, and let $w \in \Sigma^*$ such that $w \to^*_{R_i} a$ and, so, $w \to^*_{R_i \cup R_i'} a$. The system R_{i+1} is obtained from $R_i \cup R_i'$ through the algorithm REDUCE_SYSTEM. Hence, by Claim 2, there is a sequence $w = w_0 \leftrightarrow_{R_{i+1}} w_1 \leftrightarrow_{R_{i+1}} \cdots \leftrightarrow_{R_{i+1}} w_n = a$ such that, for each $j = 0, 1, \ldots, n-1$, either $w_j \to_{R_{i+1}} w_{j+1}$ or $w_j = xcy \leftarrow_{R_{i+1}} xby = w_{j+1}$ for some rule $(b \to c) \in R_{i+1} \cap (\Sigma \times (\Sigma \cup \{e\}))$. If the latter case does not occur, then $w \to^*_{R_{i+1}} a$ and, so, $w \in \nabla^*_{R_{i+1}}(a)$. Otherwise, let k be the largest index such that $w_k = xcy \leftarrow_{R_{i+1}} xby = w_{k+1} \to^*_{R_{i+1}} a$. Since $a \in \mathrm{IRR}(R_{i+1})$, and since $b \in \mathrm{dom}(R_{i+1})$, $k < n-1$, and there is an index $p > k$ such that $w_p \to_{R_{i+1}} w_{p+1}$ by applying a rule $(l \to r) \in R_{i+1}$ such that $|l|_b > 0$. However, the system R_{i+1} is reduced, i.e., $(b \to c) \in R_{i+1}$ is the only rule containing an occurrence of the letter b. Thus, we have the following situation: $w_k = xcy \leftarrow_{R_{i+1}} xby = w_{k+1} \to^*_{R_{i+1}} w_p = x'by' \to_{R_{i+1}} x'cy' \to^*_{R_{i+1}} a$, where $x \to^*_{R_{i+1}} x'$ and $y \to^*_{R_{i+1}} y'$. Thus, $w = w_0 \leftrightarrow_{R_{i+1}} \cdots \leftrightarrow_{R_{i+1}} w_k = xcy \leftrightarrow^*_{R_{i+1}} x'cy' \to^*_{R_{i+1}} a$ is a shorter R_{i+1}-derivation of a from w. Continuing in this way, we can remove all the reverse reduction steps from this derivation, which shows that $w \to^*_{R_{i+1}} a$. Thus, we have $\nabla^*_{R_i}(a) \subseteq \nabla^*_{R_{i+1}}(a)$. □

Further, it is easily seen that $\mathrm{IRR}(R_\infty) \subseteq \bigcap_{i \geqslant 0} \mathrm{IRR}(R_i)$. Now we are ready to verify that the system R_∞ has the intended properties.

Claim 5. R_∞ *is equivalent to* R.

Proof of Claim 5. For all $i \geqslant 0$, $\leftrightarrow^*_{R_i} = \leftrightarrow^*_R$. Thus, since $R_\infty \subseteq \bigcup_{i \geqslant 0} R_i$, we see that $\leftrightarrow^*_{R_\infty} \subseteq \leftrightarrow^*_R$. Conversely, let $(l \to a) \in R$. Because of Claim 2, l can be transformed into a in R_i, for all $i \geqslant 0$, such that only rules of the form $(b \to c)$, $b \in \Sigma$, are used in the reverse direction. Thus, whenever $(l' \to r') \in \bigcup_{i \geqslant 0} R_i$ is a rule that is used in these derivations, $|l'| \leqslant |l|$. There is an index j such that no rule $l' \to r'$ satisfying $|l'| \leqslant |l|$ is generated after stage j, i.e. $R_j \cap \{(w, v) \mid |w| \leqslant |l|\} = R_{j+k} \cap \{(w, v) \mid |w| \leqslant |l|\}$ for all $k \geqslant 0$. Thus, R_∞ contains all these rules and, so, $l \leftrightarrow^*_{R_j} a$ implies that $l \leftrightarrow^*_{R_\infty} a$. Hence, R_∞ is equivalent to R. □

Claim 6. R_∞ *is reduced.*

Proof of Claim 6. There are only finitely many rules of the form $(b \to a) \in R_\infty$ with $b \in \Sigma$ and $a \in \Sigma \cup \{e\}$. Hence, there is an index k such that all these rules are in R_k. Since R_k is reduced, there is at most one rule for which the left-hand side is some fixed letter, and this letter does not occur in any other rule. So, the right-hand side of each rule of R_∞ is irreducible. If rules $(l_1 \to b_1)$ and $(xl_1y \to b_2)$ were in R_∞, then these rules would belong to R_j for some index $j \geqslant k$. However, this would contradict the fact that R_j is reduced. Thus, R_∞ is reduced. □

The next claim completes the proof of Lemma 6.11.

Claim 7. *R_∞ is weakly confluent.*

Proof of Claim 7. Let $e \neq w \in \triangle^*_{R_\infty}(\mathrm{RI}_{R_\infty}(a) \cdot a) \cap \mathrm{IRR}(R_\infty)$ for some letter $a \in \Sigma$, i.e., $w \in \mathrm{IRR}(R_\infty)$, and there is a word $u \in \mathrm{RI}_{R_\infty}(a)$ such that $ua \to^*_{R_\infty} w$. Thus, we have $au \to^*_{R_\infty} e$ and $ua \to^*_{R_\infty} w \in \mathrm{IRR}(R_\infty)$. Hence, we see from the definition of R_∞ that there is an index $k \geqslant 0$ such that $au \to^*_{R_k} e$ and $ua \to^*_{R_k} w \in \mathrm{IRR}(R_k)$, i.e., $w \in \triangle^*_{R_k}(\mathrm{RI}_{R_k}(a) \cdot a) \cap \mathrm{IRR}(R_k)$. However, this implies that R_{k+1} will contain a rule that is applicable to w, i.e., $w \notin \mathrm{IRR}(R_{k+1}) \supseteq \mathrm{IRR}(R_\infty)$, contradicting the choice of w. Thus, R_∞ satisfies condition (1) of Theorem 6.4. The condition of Theorem 6.6 is verified in the same way.

Finally, let $(p, q) \in \mathrm{IDUCP}(R_\infty)$. Then there are rules $(l_1 \to b_1), (l_2 \to b_2) \in R_\infty$ such that $l_1 x = y l_2$ for some $x, y \in \Sigma^*$, $0 < |y| < |l_1|$, $l_1 x \to b_1 x \to^*_{R_\infty} p \in \mathrm{IRR}(R_\infty)$ and $yl_2 \to yb_2 \to^*_{R_\infty} q \in \mathrm{IRR}(R_\infty)$. Hence, there is some index $i \geqslant 0$ such that $(p, q) \in \mathrm{IDUCP}(R_{i+j})$ for all $j \geqslant 0$. Now, let $x \in \mathrm{RI}_{R_\infty}(p)$. Then $x \in \mathrm{RI}_{R_{k+j}}(p)$ for some $k \geqslant i$ and all $j \geqslant 0$. By Theorem 6.4, we need to verify that $\triangle^*_{R_\infty}(qx) \cap \mathrm{IRR}(R_\infty) = \{e\}$, i.e., that e is the only irreducible descendant of $qx \bmod R_\infty$. Assume to the contrary that $qx \to^*_{R_\infty} y \in \mathrm{IRR}(R_\infty) \setminus \{e\}$. Then $qx \to^*_{R_l} y \in \mathrm{IRR}(R_l) \setminus \{e\}$ for some index $l \geqslant k$, which implies that $y \in S(p)$ at stage l. Hence, y is reducible mod R_{l+1}, which contradicts the fact that y is irreducible mod R_∞. By symmetry, also the other part of condition (2) of Theorem 6.4 holds. Thus, R_∞ is indeed weakly confluent by Theorems 6.4 and 6.6. □

Proof of Lemma 6.11 (*conclusion*). This concludes the proof of Lemma 6.11. □

Thus, given a finite monadic string-rewriting system R presenting a group as input, procedure WEAK_COMPLETION always "computes" a monadic system R_∞ that is reduced, equivalent to R, and weakly confluent. Further, this procedure terminates if and only if the system R_∞ is finite. However, by Theorem 3.5, R_∞ is uniquely determined by R, i.e., if there does at all exist a finite, monadic, and weakly confluent system S that is equivalent to R, then R_∞ is the reduced form of S and, hence, R_∞ is finite. Thus, we have the following completeness result.

Theorem 6.12. *Let R be a finite monadic string-rewriting system on Σ such that the monoid $\mathfrak{M}_R$ is a group. Given R as input, procedure* WEAK_COMPLETION *terminates if and only if there exists a finite monadic system S on Σ that is equivalent to R and* e-*confluent.*

Note that the existence of a finite, monadic, and weakly confluent string-rewriting system S that is equivalent to a given system R does not depend on the ordering on Σ that we fixed in the beginning. Using a different ordering just means a renaming of the letters as mentioned in Section 3 after Theorem 3.5. Thus, whether or not the procedure WEAK_COMPLETION terminates on input R is a property of the Thue congruence $\leftrightarrow_R^*$ only. However, this property is undecidable in general, since the following undecidability result is an immediate consequence of [26, Theorem 5.1.3].

Theorem 6.13. *The following problem is undecidable in general:*

Instance: A finite monadic string-rewriting system R on Σ presenting a group.
Question: Does there exist a finite, monadic, and weakly confluent system S on Σ such that S is equivalent to R?

Since all finitely presented groups can be presented by finite special string-rewriting systems, this result is not surprising. As one of the referees pointed out, it would be more interesting to find out whether the above problem remains undecidable even when it is restricted to finite monadic string-rewriting systems that present *context-free* groups; however, this question has to remain open at this time.

7. Conclusion

In this paper we have investigated various aspects of those finite string-rewriting systems that are monadic and weakly confluent. The class of finite, monadic, and weakly confluent string-rewriting systems contains properly both the class of finite, monadic, and confluent systems and the class of finite, special, and e-confluent systems. These latter two classes have received a lot of attention in the literature (cf., e.g., [6, 33]), and it has been shown that they both have nice algorithmic properties. So, one of the main objectives of this paper was to investigate as to how far these properties carry over to finite, monadic, and weakly confluent string-rewriting systems.

As a first step, we proved in Section 3 that when dealing with finite, monadic, and weakly confluent systems it suffices to look at systems that are reduced. In fact, the reduced system equivalent to a given system is uniquely determined (Theorem 3.5), and it can be obtained in polynomial time (Corollary 3.6). Exploiting this fact, a polynomial-time algorithm for deciding whether a finite monadic system is weakly confluent can be developed (Proposition 3.8). The characterization leading to this

algorithm is the direct generalization of the characterization of e-confluence for special string-rewriting systems given in [32].

Unfortunately, only very few decision problems, that are undecidable in general, become decidable in the setting of finite, monadic, and weakly confluent string-rewriting systems, an example being the group problem, while many others, among them the word problem (Theorem 4.9), remain undecidable. The finiteness problem and the problem of deciding whether a string-rewriting system presents a cancellative monoid are still open in this setting.

The situation improved dramatically when we turned to those finite, monadic, and weakly confluent string-rewriting systems that present groups. For these systems many decision problems can be solved efficiently; in fact, their uniform versions, where the presentation is also a part of the problem instance, are decidable. So, these string-rewriting systems, which are known to present the class of context-free groups, form an algorithmically nicely behaved class of presentations. All these results already hold for finite, monadic, and e-confluent string-rewriting systems presenting groups, but, for each system of this form, there exists an equivalent one that is finite, monadic, and weakly confluent; in fact, the latter system can effectively be constructed from the former (Theorem 5.13).

Instead of presenting a context-free group through a finite, monadic, and weakly confluent string-rewriting system, we could simply describe it through a context-free grammar for the language [e]. However, no characterization is known for those context-free grammars that generate languages of this form. Further, exploiting the algebraic characterization of context-free groups given by Muller and Schupp (Theorem 5.1), we could also present context-free groups explicitly as finite extensions of free groups. One could then expect that a decision problem for virtually free groups, which is independent of the chosen presentation, can be reduced effectively to some (not necessarily the same) decision problem for free groups. For example, this is the case for the word problem. However, this reduction might not be uniform for the class of all decision problems that can be expressed through linear sentences. Therefore, we feel that the presentation through finite, monadic, and weakly confluent string-rewriting systems is more useful. Furthermore, it has the additional advantage that the restriction placed on the presentations considered is purely syntactic, and that it is decidable in polynomial time (Proposition 3.8).

Contrasting the results of Section 4 with those of Section 5, we see that the algebraic restriction that the monoids presented are groups helps a lot. Can we weaken this additional restriction without losing the decidability results? In particular, which of these decidability results carry over to finite, monadic and weakly confluent string-rewriting systems that present cancellative monoids? For example, the finiteness problem remains decidable in this less restricted situation, but what can be said about the other decision problems?

Finally, in Section 6 we presented a specialized completion procedure for finite monadic string-rewriting systems presenting groups. Given such a system as input, this procedure, called WEAK_COMPLETION, attempts to construct an equivalent

system that is weakly confluent. It is based on a simplified test for weak confluence, and we could establish that it is correct and complete. The procedure WEAK_COMPLETION can be seen as a kind of unfailing completion procedure [3], where the unresolvable critical pairs take the role of the nonorientable equations, and the property of ground confluence is replaced by the property of weak confluence. In this way, the nontermination of the regular completion procedure can be avoided in some cases. A generalization to finite monadic systems, in general, is possible, but this does not seem to be very useful because of the undecidability results of Section 4. A generalization to other classes of systems, e.g., length-reducing ones, seems to be very hard, since for these classes no decidable criteria for characterizing the property of confluence on a single congruence class are known at this time.

In the subroutine CONTEXT_RESOLVING, the procedure WEAK_COMPLETION adds special rules to the actual system R when $ax \to_R^* e$, but $xa \not\to_R^* e$ for some letter $a \in \Sigma$ and some word $x \in \Sigma^*$. In this way, one tries to enforce that the set $\nabla^*(e)$ is closed under cyclic permutation. A possible improvement of the procedure would be to add special or monadic rules that guarantee that w and *all* its cyclic permutations reduce to e whenever an irreducible word $w \in [e]_R$ is found. This idea is similar to the one used in the completion procedures of [7, 20], and is based on the notion of *symmetrized group presentation* [21]. If we start with a special string-rewriting system R such that $\mathrm{dom}(R)$ forms a symmetrized set, i.e., every element is cyclically reduced, and $\mathrm{dom}(R)$ is closed under the operations of cyclic permutation and of taking inverses, then $\nabla_R^*(e)$ is closed under cyclic permutation. LeChenadec [20] presents a process he calls the *group symmetrization algorithm* that, when given as input a finite symmetrized group presentation $\langle \Sigma; L \rangle$ satisfying certain small cancellation conditions, generates the finite length-reducing system S that is used in Dehn's algorithm to solve the word problem for groups of this form. In the group symmetrization algorithm, a rule of the form $w \to e$ is split into a new rule $u \to v^{-1}$, where $w = uv$, and u is the minimal prefix of w satisfying $u > v^{-1}$. In fact, we are doing the same in case v^{-1} is a single letter.

There are examples in which the sets $\mathrm{RI}_R(a)$ and $\mathrm{RI}_R(p)$ are indeed infinite. It is an interesting question whether the e-confluence criteria of Theorem 6.4 can be specialized in such a way that only finite test sets have to be considered, as is the case for special systems [32]. Actually, Theorem 6.4 remains valid if the sets $\mathrm{RF}_R(a)$ and $\mathrm{RF}_R(p)$ are restricted to contain only irreducible words, but in the monadic case even these restricted sets can be infinite. Thus, the investigation of specialized completion procedures will certainly continue. Currently, projects are being carried out at Kaiserslautern and Kassel to get various implementations of the procedure WEAK_COMPLETION. The objectives are to gain further insights into how this procedure behaves in practice, and to experiment with various improvements.

References

[1] J. Autebert, L. Boasson and G. Senizergues, Groups and NTS languages, *J. Comput. System Sci.* **35** (1987) 243–267.

[2] J. Avenhaus and K. Madlener, Term rewriting and equational reasoning, in: R.B. Banerji, ed., *Formal Techniques in Artificial Intelligence – A Sourcebook* (Elsevier, Amsterdam, 1990).
[3] L. Bachmair, *Canonical Equational Proofs* (Birkhäuser, Boston, 1991).
[4] R.V. Book, Confluent and other types of Thue systems, *J. ACM* **29** (1982) 171–182.
[5] R.V. Book, Decidable sentences of Church–Rosser congruences, *Theoret. Comput. Sci.* **23** (1983) 301–312.
[6] R.V. Book, Thue systems as rewriting systems, *J. Symbolic Comput.* **3** (1987) 39–68.
[7] H. Bücken, Reduction systems and small cancellation theory, in: *Proc. 4th Workshop on Automated Deduction* (1979) 53–59.
[8] M. Davis, *Computability and Unsolvability* (McGraw-Hill, New York, 1958).
[9] N. Dershowitz and Z. Manna, Proving termination with multiset orderings, *Comm. ACM* **22** (1979) 465–476.
[10] V. Diekert, Complete semi-Thue systems for abelian groups, *Theoret. Comput. Sci.* **44** (1986) 199–208.
[11] M.J. Dunwoody, The accessibility of finitely presented groups, *Invent. Math.* **81** (1985) 449–457.
[12] H. Ehrig and B. Mahr, *Fundamentals of Algebraic Specification* 1, *EATCS Monograph Vol. 6* (Springer, Berlin, 1985)
[13] R.H. Gilman, Presentations of groups and monoids, *J. Algebra* **57** (1979) 544–554.
[14] G. Huet, Confluent reductions: abstract properties and applications to term rewriting systems, *J. ACM* **27** (1980) 797–821.
[15] G. Huet and D.S. Lankford, On the uniform halting problem for term rewriting systems, Lab. Rep. 283, INRIA, Le Chesnay, France, 1978.
[16] M. Jantzen, *Confluent String Rewriting*, *EATCS Monograph Vol. 14* (Springer, Berlin, 1988).
[17] D. Kapur and P. Narendran, The Knuth–Bendix completion procedure and Thue systems, *SIAM J. Comput.* **14** (1985) 1052–1072.
[18] D. Kapur and P. Narendran, A finite Thue system with decidable word problem and without equivalent finite canonical system, *Theoret. Comput. Sci.* **35** (1985) 337–344.
[19] D. Knuth and P. Bendix, Simple word problems in universal algebras, in: J. Leech, ed., *Computational Problems in Abstract Algebra* (Pergamon, New York, 1970) 263–297.
[20] Ph. LeChenadec, *Canonical Forms in Finitely Presented Algebras* (Pitman, London, Wiley, New York, 1986).
[21] R.C. Lyndon and P.E. Schupp, *Combinatorial Group Theory* (Springer, Berlin, 1977).
[22] D.E. Muller and P.E. Schupp, Groups, the theory of ends, and context-free languages, *J. Comput. System Sci.* **26** (1983) 295–310.
[23] P. Narendran and F. Otto, Elements of finite order for finite weight-reducing and confluent Thue systems, *Acta Inform.* **25** (1988) 573–591.
[24] M.H.A. Newman, On theories with a combinatorial definition of equivalence, *Ann. of Math.* **43** (1943) 223–243.
[25] M. Nivat (with M. Benois), Congruences parfaites et quasi-parfaites, in: *Seminaire Dubreil*, 25^{e} Année, 1971–72, 7-01-09.
[26] C. O'Dunlaing, Finite and infinite regular Thue systems, Ph.D. Dissertation, Department of Mathematics, University of California at Santa Barbara, 1981.
[27] F. Otto, Some undecidability results for nonmonadic Church–Rosser Thue systems, *Theoret. Comput. Sci.* **33** (1984) 261–278.
[28] F. Otto, On deciding whether a monoid is a free monoid or is a group, *Acta Inform.* **23** (1986) 99–110.
[29] F. Otto, On deciding the confluence of a finite string-rewriting system on a given congruence class, *J. Comput. System Sci.* **35** (1987) 285–310.
[30] F. Otto, When is an extension of a specification consistent? Decidable and undecidable cases, *J. Symbolic Comput.* **12** (1991) 255–273.
[31] F. Otto, Completing a finite special string-rewriting system on the congruence class of the empty word, *Appl. Algebra Engrg. Comm. Comput.* **2** (1992) 257–274.
[32] F. Otto, The problem of deciding confluence on a given congruence class is tractable for finite special string-rewriting systems, *Math. Systems Theory* **25** (1992) 241–251.
[33] F. Otto and L. Zhang, Decision problems for finite special string-rewriting systems that are confluent on some congruence class, *Acta Inform.* **28** (1991) 477–510.