Preserving Regularity and Related Properties of String-Rewriting Systems

Friedrich Otto

Fachbereich Mathematik/Informatik
Universität Kassel
D–34109 Kassel
Germany
Internet: otto@theory.informatik.uni-kassel.de

September 30, 1996

Abstract

A finite string-rewriting system R preserves regularity if and only if it preserves Σ-regularity, where Σ is the alphabet containing exactly those letters that have occurrences in the rules of R. This proves a conjecture of Gyenizse and Vágvölgyi (1995). In addition, some undecidability results are presented that generalize some results of Gilleron and Tison (1995) from term-rewriting systems to string-rewriting systems. In particular, it is shown that it is undecidable in general whether a finite, length-reducing, and confluent string-rewriting system yields a regular set of normal forms for each regular language.

1 Introduction

In the specification of abstract data types the use of rewriting techniques is by now well established. In this context the initial algebra defined by a given set of equations or rewrite rules is of particular interest. It is the set of ground terms in the signature considered modulo the congruence that is generated by the given equations or rules (see, e.g., [Wir90]).

A set of ground terms is a tree language. A class of tree languages that has been studied in great detail is the class of regular tree languages. This class is defined by tree automata, and it enjoys many closure properties as well as decidability results in common with the class of regular string languages [GS84].

If R is a left-linear term-rewriting system on a signature F, then the set of normal forms $\mathrm{IRR}(R)$ is a regular tree language. The system R is called *F-regularity preserving*, if, for each regular tree language $S \subseteq T(F)$, the set of descendants $\Delta_R^*(S)$ is again a regular tree language. Thus, if R is a left-linear and convergent term-rewriting system that is F-regularity preserving, then, for each regular tree language $S \subseteq T(F)$, the set of normal forms $\mathrm{NF}_R(S) = \Delta_R^*(S) \cap \mathrm{IRR}(R)$ of ground terms that are in normal form and that are congruent to some ground term in S is a regular tree language. Hence, for systems of this form, various decision problems can be solved efficiently by using tree automata (see, e.g., [GT95]).

It is known that F-regularity is preserved by term-rewriting systems that contain only ground rules [Bra69], by term-rewriting systems that are right-linear and monadic [Sal88], that are linear and semi-monadic [CDGV94], or that are linear and generalized semi-monadic [GV95].

However, the property of preserving F-regularity is undecidable in general. This already follows from the construction used in [Dau89] to prove that termination is undecidable for left-linear one-rule term-rewriting systems, but a less complicated direct proof is given in [GT95]. In addition, also the property that $\mathrm{NF}_R(S)$ is regular for each regular tree language $S \subseteq T(F)$ is undecidable in general [Gil91]. Actually, both these undecidability results remain valid even for the class of finite term-rewriting systems that are convergent [Gil91, GT95].

Interestingly, it has been observed by Gyenizse and Vágvölgyi that the property of preserving F-regularity does not only depend on the term-rewriting system R considered, but also on the actual signature F [GV95]. In fact, they present a signature F that contains a single constant plus some unary function symbols only, and a term-rewriting system R over F that consists of a single linear rule plus some ground rules such that R preserves F-regularity, but R does not preserve F_1-regularity, where the signature F_1 is obtained from F by introducing an additional unary function symbol. Accordingly, they call a system R on a signature F *regularity preserving*, if it is F_1-regularity preserving for each signature F_1 containing F. They show that the property of being regularity preserving is a modular property of linear term-rewriting systems, and they ask whether this property is undecidable in general.

Since each string-rewriting system can be seen as a linear term-rewriting system over a signature that contains unary function symbols and a single constant only, it follows that for string-rewriting systems the property of being regularity preserving is modular. Gyenizse and Vágvölgyi conjecture that for string-rewriting systems the properties of preserving F-regularity and of preserving regularity are equivalent. Here we prove this conjecture.

For string-rewriting systems the property of being a generalized semi-monadic system is just the property of being a left-basic string-rewriting system (see, e.g., [Sén90]), and it is known that a finite, length-reducing, confluent, and left-basic string-rewriting system R yields a regular set $\mathrm{NF}_R(S)$ for each regular language S [Sak79]. Thus, for string-rewriting systems the above-mentioned result that a linear, generalized semi-monadic system preserves F-regularity can be seen as a generalization of Sakarovitch's result. On the other hand, it is known that each monadic string-rewriting system preserves regularity [BJW82, Kre88, Sal88], while this is not true in general for finite, length-reducing, and confluent string-rewriting systems [Ó'D81, Ott84].

Here we give a simple proof for the fact that the property of preserving regularity is undecidable for finite string-rewriting systems in general. Further, we construct a finite, length-reducing, and confluent string-rewriting system R such that it is undecidable whether, for a regular language $S \subseteq \Sigma^*$, the set $\Delta_R^*(S)$ is a regular language. The same result is also established for the set of normal forms $\mathrm{NF}_R(S)$. In addition, a finite, length-reducing, and confluent string-rewriting system R and an infinite family of regular languages $(S_i)_{i \in \mathbb{N}}$ are constructed such that, for each $i \in \mathbb{N}$, the set of normal forms $\mathrm{NF}_R(S_i)$ is a singleton, but it is undecidable in general whether $\Delta_R^*(S_i)$ is a regular language.

Finally, using the undecidability of the *strong boundedness* of single-tape Turing machines we prove that it is undecidable in general whether a finite, length-reducing, and confluent string-rewriting system R yields a regular set of normal forms $\mathrm{NF}_R(S)$ for each regular language S.

This paper is organized as follows. In Section 2 we restate the basis definitions used in short in order to establish notation. In Section 3 we prove the conjecture of Gyenizse and Vágvölgyi stating that for string-rewriting systems the property of preserving regularity is independent of the alphabet actually considered, and in Section 4 we present the first of the undecidability results mentioned above. Then in Section 5 we consider the strong boundedness problem for single-tape Turing machines, and finally in Section 6 we reduce this problem to the problem of deciding whether a finite, length-reducing, and confluent string-rewriting system yields a regular set of normal forms for each regular language. The paper closes with a short discussion of some open problems.

2 String-rewriting systems and monoid presentations

After establishing notation we describe the problems considered in this paper in detail. For more information and a detailed discussion of the notions introduced the reader is refered to the literature, e.g., [BO93].

Let Σ be a finite alphabet. Then Σ^* denotes the set of all strings over Σ including the empty string λ. For $u, v \in \Sigma^*$, the concatenation of u and v is simply written as uv, and exponents are used to abbreviate strings, that is, $u^0 := \lambda$, $u^1 := u$, and $u^{n+1} := u^n u$ for all $n \geq 1$. For $u \in \Sigma^*$, the *length* of u is denoted by $|u|$.

A *string-rewriting system* R on Σ is a subset of $\Sigma^* \times \Sigma^*$, the elements of which are called (*rewrite*) *rules*. Often these rules will be written in the form $\ell \to r$ to improve readability. For a string-rewriting system R, $\mathrm{dom}(R) := \{\ell \mid \exists r \in \Sigma^* : (\ell \to r) \in R\}$ is the *domain* of R. The system R is called *length-reducing* if $|\ell| > |r|$ holds for each rule $(\ell \to r)$ of R, and it is called *monadic* if it is length-reducing, and $r \in \Sigma \cup \{\lambda\}$ for each rule $(\ell \to r)$ of R.

The *single-step reduction relation* induced by R is the following binary relation on Σ^* : $u \to_R v$ iff $\exists x, y \in \Sigma^* \exists (\ell \to r) \in R : u = x\ell y$ and $v = xry$. Its reflexive and transitive closure $\to_R^*$ is the *reduction relation* induced by R. The reflexive, symmetric, and transitive closure $\leftrightarrow_R^*$ of $\to_R$ is a congruence on Σ^*, the *Thue congruence* generated by R. For $u \in \Sigma^*$, $\Delta_R^*(u) := \{v \in \Sigma^* \mid u \to_R^* v\}$ is the *set of descendants* of $u \pmod R$, and $[u]_R := \{v \in \Sigma^* \mid u \leftrightarrow_R^* v\}$ is the *congruence class* of $u \pmod R$. For $L \subseteq \Sigma^*$, $\Delta_R^*(L) = \bigcup_{u \in L} \Delta_R^*(u)$ and $[L]_R := \bigcup_{u \in L} [u]_R$.

Let R be a string-rewriting system on Σ. We say that R *preserves* Σ-*regularity* if, for each regular language $L \subseteq \Sigma^*$, $\Delta_R^*(L)$ is again a regular language. We say that R *preserves regularity* if R preserves Γ-regularity for each finite alphabet Γ containing all the letters that have occurrences in the rules of R. In the next section we will investigate the relationship between the property of preserving regularity and the property of preserving Σ-regularity, where Σ is the smallest alphabet that contains all the letters with occurrences in R. After that we will address the following decision problems:

PROBLEM 1: Preserving regularity.
INSTANCE: A finite string-rewriting system R.
QUESTION: Does R preserve regularity?

PROBLEM 2: Regular descendants for a given language.
INSTANCE: A finite string-rewriting system R on Σ, and a regular language $L \subseteq \Sigma^*$.
QUESTION: Is $\Delta_R^*(L)$ a regular language?

A string $u \in \Sigma^*$ is called *reducible* $(\bmod R)$, if there exists a string $v \in \Sigma^*$ such that $u \rightarrow_R v$; otherwise, u is called *irreducible*. By $\mathrm{IRR}(R)$ we denote the set of all irreducible strings, and by $\mathrm{RED}(R)$ we denote the set of strings that are reducible $(\bmod R)$. Obviously, $\mathrm{RED}(R) = \Sigma^* \cdot \mathrm{dom}(R) \cdot \Sigma^*$, and $\mathrm{IRR}(R) = \Sigma^* \setminus \mathrm{RED}(R)$. Thus, if R is finite, then $\mathrm{RED}(R)$ and $\mathrm{IRR}(R)$ are both regular sets. Actually, from R deterministic finite-state acceptors (dfsa) can easily be constructed for $\mathrm{IRR}(R)$ and for $\mathrm{RED}(R)$ (cf., e.g., Lemma 2.1.3 of [BO93]).

If $u \in \Sigma^*$ and $v \in \mathrm{IRR}(R)$ such that $u \rightarrow_R^* v$, then v is called a *normal form* of u. Accordingly, $\Delta_R^*(u) \cap \mathrm{IRR}(R)$ is the set of all normal forms of u, and for $L \subseteq \Sigma^*$, $\Delta_R^*(L) \cap \mathrm{IRR}(R)$ is the set of all normal forms of L. If a finite string-rewriting system R preserves regularity, then the set $\mathrm{NF}_R(L) := \Delta_R^*(L) \cap \mathrm{IRR}(R)$ of normal forms of L is a regular language for each regular language $L \subseteq \Sigma^*$. On the other hand, $\mathrm{NF}_R(L)$ can be a regular language, even if $\Delta_R^*(L)$ is not. Thus, also the following decision problems are of interest:

PROBLEM 3: Regular sets of normal forms.

INSTANCE: A finite string-rewriting system R on Σ.

QUESTION: Is $\mathrm{NF}_R(L)$ a regular language for each regular language $L \subseteq \Sigma^*$?

PROBLEM 4: Regular set of normal forms for a given language.

INSTANCE: A finite string-rewriting system R on Σ, and a regular language $L \subseteq \Sigma^*$.

QUESTION: Is $\mathrm{NF}_R(L)$ a regular language?

We need a couple of more definitions. Since $\leftrightarrow_R^*$ is a congruence, the set $M_R := \{[u]_R \mid u \in \Sigma^*\}$ of congruence classes mod R is a monoid under the operation $[u]_R \circ [v]_R = [uv]_R$ with identity $[\lambda]_R$. It is just the factor monoid $\Sigma^*/\leftrightarrow_R^*$, and it is uniquely determined (up to isomorphism) by Σ and R. Hence, the ordered pair $(\Sigma; R)$ is called a *monoid-presentation* of M_R with *generators* Σ and *defining relations* R.

Some of the monoids that we will encounter are actually groups. Although groups can be described by monoid-presentations, they are usually defined through so-called group-presentations. A finite *group-presentation* $\langle \Sigma; L \rangle$ consists of a finite alphabet Σ and a finite set of defining relators $L \subseteq \underline{\Sigma}^*$, where $\overline{\Sigma}$ is an alphabet in one-to-one correspondence to Σ such that $\Sigma \cap \overline{\Sigma} = \emptyset$, and $\underline{\Sigma} := \Sigma \cup \overline{\Sigma}$. The group G_L defined by $\langle \Sigma; L \rangle$ coincides with the monoid M_{R_L} that is given through the monoid-presentation $(\underline{\Sigma}; R_L)$, where $R_L := \{a\bar{a} \rightarrow \lambda, \bar{a}a \rightarrow \lambda \mid a \in \Sigma\} \cup \{u \rightarrow \lambda \mid u \in L\}$. It is easily verified that the monoid M_{R_L} is indeed a group.

We are in particular interested in the decision problems above for those finite string-rewriting systems that are convergent. Here a string-rewriting system R on Σ is called

- *noetherian*, if there is no infinite sequence of reductions of the form $u_0 \rightarrow_R u_1 \rightarrow_R u_2 \rightarrow_R \ldots$;
- *confluent*, if, for all $u, v, w \in \Sigma^*$, $u \rightarrow_R^* v$ and $u \rightarrow_R^* w$ imply that $v \rightarrow_R^* z$ and $w \rightarrow_R^* z$ hold for some $z \in \Sigma^*$;
- *convergent*, if it is noetherian and confluent.

If R is convergent, then each congruence class $[u]_R$ contains a unique irreducible string u_0, and u_0 can be obtained from u by a finite sequence of reduction steps. Thus, in particular, the word problem is decidable for each finite convergent string-rewriting system.

Obviously, a length-reducing string-rewriting system is noetherian. The finite, length-reducing, and confluent string-rewriting systems are of particular interest, since their word problems are even solvable in linear time [BO93].

3 Independence of the alphabet considered

Gyenizse and Vágvögyi [GV95] present a linear term-rewriting system R that does not preserve regularity, although it preserves F-regularity, where F is the signature consisting of those function symbols and constants that do actually occur in the rules of R. Thus, for each regular tree language $L \subseteq T(F)$, $\Delta_R^*(L)$ is regular, while there is a regular tree language $L_1 \subseteq T(F_1)$, where F_1 is a signature that contains F properly, such that $\Delta_R^*(L_1)$ is not regular anymore. They conjecture that this phenomenon does not occur with string-rewriting systems. Here we provide a proof for this conjecture.

This result will be a consequence of the following technical result.

Lemma 3.1 *Let R be a string-rewriting system on some finite alphabet Σ, let a be an additional letter not in Σ, and let $\Gamma := \Sigma \cup \{a\}$. If R preserves Σ-regularity, then it also preserves Γ-regularity.*

Proof. Let $L \subseteq \Gamma^*$ be a regular language. We must verify that, under the hypothesis that R preserves Σ-regularity, $\Delta_R^*(L) := \{v \in \Gamma^* \mid \exists u \in L : u \to_R^* v\}$ is a regular language.

If $L \subseteq \Sigma^*$, then $\Delta_R^*(L) \subseteq \Sigma^*$, and hence, $\Delta_R^*(L)$ is a regular language by our assumption on R. So let us assume that $L \nsubseteq \Sigma^*$. Since $L \subseteq \Gamma^*$ is a regular language, there exists a nondeterministic finite state acceptor (nfsa) $A = (Q, \Gamma, q_0, F, \delta)$ such that $L(A) = L$. Without loss of generality we may assume that all the states of A are accessible as well as co-accessible, and that $|\delta(q, b)| \leq 1$ holds for each $q \in Q$ and $b \in \Gamma$. Let $q_1 \xrightarrow{a} p_1, \ldots, q_m \xrightarrow{a} p_m$ be the set of all a-transitions of A, which are numbered in an arbitrary, but fixed way. Based on A we define some auxiliary languages.

$$\begin{array}{lcl}
L_0 & := & \{w \in \Sigma^* \mid \delta(q_0, w) \in F\} = L \cap \Sigma^*, \\
P_i & := & \{w \in \Sigma^* \mid \delta(q_0, w) = q_i\}, \quad i = 1, \ldots, m, \\
S_j & := & \{w \in \Sigma^* \mid \delta(p_j, w) \in F\}, \quad j = 1, \ldots, m, \\
K_{i,j,k} & := & \{w \in \Gamma^* \mid \delta(p_i, w) = q_j \text{ and, for all } u, v \in \Gamma^*, \text{ if } w = uav, \text{ then} \\
& & \qquad \delta(p_i,\ u) = q_\ell \text{ for some } \ell \leq k\}, \quad i, j = 1, \ldots, m, \\
& & \qquad\qquad\qquad\qquad\qquad\qquad\qquad k\ = 0, 1, \ldots, m.
\end{array}$$

Thus, $w \in K_{i,j,k}$ if and only if $\delta(p_i, w) = q_j$, and the only a-transitions that are used by following the computation of $\delta(p_i, w)$ are those with index at most k.

Obviously, the languages L_0, P_i $(i = 1, \ldots, m)$, and S_j $(j = 1, \ldots, m)$ are regular.

Claim 1: The languages $K_{i,j,k}$ are regular $(i, j = 1, \ldots, m,\ k = 0, 1, \ldots, m)$.

Proof. We proceed by induction on k.

***k* = 0:** $K_{i,j,0} = \{w \in \Gamma^* \mid \delta(p_i, w) = q_j, \text{ and no } a\text{-transitions are used }\}$
$= \{w \in \Sigma^* \mid \delta(p_i, w) = q_j\}$.
These languages are certainly regular.

***k* = 1:** $K_{i,j,1} = \{w \in \Gamma^* \mid \delta(p_i, w) = q_j, \text{ and only the } a\text{-transition } q_1 \xrightarrow{a} p_1 \text{ may be used }\}$
$= K_{i,j,0} \cup K_{i,1,0} \cdot (\{a\} \cdot K_{1,1,0})^* \cdot \{a\} \cdot K_{1,j,0}$.
From the case $k = 0$ we conclude that these languages are regular.

***k* → *k*+1:** $K_{i,j,k+1} = \{w \in \Gamma^* \mid \delta(p_i, w) = q_j, \text{ and the } a\text{-transitions } q_1 \xrightarrow{a} p_1, \ldots, q_{k+1} \xrightarrow{a} p_{k+1} \text{ may be used }\}$
$= K_{i,j,k} \cup K_{i,k+1,k} \cdot (\{a\} \cdot K_{k+1,k+1,k})^* \cdot \{a\} \cdot K_{k+1,j,k}$.
From the induction hypothesis we see that these languages are regular.

This completes the proof of Claim 1. $\square$

In fact, it is easily seen that $K_{i,j,k}$ is accepted by the automaton $(Q, \Gamma, p_i, \{q_j\}, \delta_k)$, where δ_k is obtained from δ by removing the a-transitions with index larger than k. However, we will need the inductive representation of the sets $K_{i,j,k}$ developed above at a later stage of the proof of Lemma 3.1.

Now it is easily seen that

$$L = L_0 \cup \bigcup_{i=1}^{m} (P_i \cdot \{a\} \cdot S_i) \cup \bigcup_{i,j=1}^{m} (P_i \cdot \{a\} \cdot K_{i,j,m} \cdot \{a\} \cdot S_j).$$

Actually, since A is essentially deterministic, this is a disjoint partitioning of L.

The next claim states that the operations of union and of computing descendants mod R commute.

Claim 2: $\Delta_R^*(\bigcup_{i\in I} M_i) = \bigcup_{i\in I} \Delta_R^*(M_i)$ for all $M_i \subseteq \Gamma^*(i \in I)$.

Proof. Since $M_i \subseteq \bigcup_{i\in I} M_i$, $\Delta_R^*(M_i) \subseteq \Delta_R^*(\bigcup_{i\in I} M_i)$, and hence, $\bigcup_{i\in I} \Delta_R^*(M_i) \subseteq \Delta_R^*(\bigcup_{i\in I} M_i)$. Conversely, if $w \in \Delta_R^*(\bigcup_{i\in I} M_i)$, then there exists some $i \in I$ such that $w \in \Delta_R^*(M_i)$. Thus, $\Delta_R^*(\bigcup_{i\in I} M_i) = \bigcup_{i\in I} \Delta_R^*(M_i)$. $\square$

Hence, $\Delta_R^*(L) = \Delta_R^*(L_0) \cup \bigcup_{i=1}^{m} \Delta_R^*(P_i \cdot \{a\} \cdot S_i) \cup \bigcup_{i,j=1}^{m} \Delta_R^*(P_i \cdot \{a\} \cdot K_{i,j,m} \cdot \{a\} \cdot S_j)$.

The next two claims show that also the operations of concatenation and of computing descendants mod R commute in certain instances.

Claim 3: $\Delta_R^*(P_i \cdot \{a\} \cdot S_i) = \Delta_R^*(P_i) \cdot \{a\} \cdot \Delta_R^*(S_i)$.

Proof. Since no rule of R contains any occurrences of the letter a, this is obvious. $\square$

Claim 4: $\Delta_R^*(P_i \cdot \{a\} \cdot K_{i,j,m} \cdot \{a\} \cdot S_j) = \Delta_R^*(P_i) \cdot \{a\} \cdot \Delta_R^*(K_{i,j,m}) \cdot \{a\} \cdot \Delta_R^*(S_j)$.

Proof. Analogously. $\square$

$\Delta_R^*(L_0)$, $\Delta_R^*(P_i)$, and $\Delta_R^*(S_i)$ $(i = 1, \ldots, m)$ are regular by our assumption, since L_0, P_i, S_i are regular subsets of Σ^*. It remains to prove the following result.

Claim 5: $\Delta_R^*(K_{i,j,k})$ is a regular set for all $i, j = 1, \ldots, m$, and $k = 0, 1, \ldots, m$.

Proof. We proceed by induction on k, using the inductive representation of the set $K_{i,j,k}$ derived in the proof of Claim 1.

***k* = 0:** $K_{i,j,0} \subseteq \Sigma^*$.
Since $K_{i,j,0}$ is regular, $\Delta_R^*(K_{i,j,0})$ is regular by our assumption on R.

***k* = 1:** $K_{i,j,1} = K_{i,j,0} \cup K_{i,1,0} \cdot (\{a\} \cdot K_{1,1,0})^* \cdot \{a\} \cdot K_{1,j,0}$.
Hence, $\Delta_R^*(K_{i,j,1}) = \Delta_R^*(K_{i,j,0}) \cup \Delta_R^*(K_{i,1,0}) \cdot (\{a\} \cdot \Delta_R^*(K_{1,1,0}))^* \cdot \{a\} \cdot \Delta_R^*(K_{1,j,0})$. Thus, $\Delta_R^*(K_{i,j,1})$ is regular.

***k* → *k*+1:** analogously. $\square$

Claim 6: $\Delta_R^*(L)$ is regular.

Proof. $\Delta_R^*(L) = \Delta_R^*(L_0) \cup \bigcup_{i=1}^{m} (\Delta_R^*(P_i) \cdot \{a\} \cdot \Delta_R^*(S_i)) \cup \bigcup_{i,j=1}^{m} (\Delta_R^*(P_i) \cdot \{a\} \cdot \Delta_R^*(K_{i,j,m}) \cdot \{a\} \cdot \Delta_R^*(S_j))$. Hence, $\Delta_R^*(L)$ is indeed a regular set. □

This completes the proof of Lemma 3.1. □

Now we can prove Gynizse's and Vágvölgyi's conjecture.

Theorem 3.2 *Let R be a string-rewriting system, and let Σ be the alphabet consisting of all the letters that have occurrences in R. Assume that Σ is finite. Then R preserves Σ-regularity if and only if R preserves regularity.*

Proof. Let Δ be a finite alphabet containing Σ. If $\Delta = \Sigma$, then R preserves Δ-regularity. Otherwise, let $\Delta - \Sigma = \{a_1, a_2, \ldots, a_n\}$. Using Lemma 3.1 repeatedly, we see that R preserves Δ-regularity if it preserves Σ-regularity. □

Thus, when we talk about the property of preserving regularity, there is no need to specify the underlying alphabet in detail as long as it contains all the letters that have occurrences in the rules of the string-rewriting system considered. Observe that the proof of Lemma 3.1 is constructive in that, if R preserves Σ-regularity in an effective way, then it preserves Γ-regularity in an effective way, too. That is, if a regular language $L \subseteq \Gamma^*$ is given through some nfsa, then an nfsa for the language $\Delta_R^*(L)$ can be constructed effectively.

4 The property of preserving regularity is undecidable

Here we establish two undecidability results concerning the property of preserving regularity for string-rewriting systems. The first one says that it is undecidable in general whether a given string-rewriting system preserves regularity. This result is based on the following characterization.

Lemma 4.1 *Let R be a finite string-rewriting system on Σ such that the monoid M_R presented by $(\Sigma; R)$ is a group. Then the following two statements are equivalent:*

(a) the string-rewriting system $R \cup R^{-1}$ preserves regularity,

(b) the group M_R is finite.

Here R^{-1} denotes the string-rewriting system $R^{-1} := \{r \to \ell \mid (\ell \to r) \in R\}$.

Proof. (b) $\Rightarrow$ (a): Assume that M_R is a finite group. Let $L \subseteq \Sigma^*$ be a regular language. Then $\Delta_{R\cup R^{-1}}^*(L) = [L]_R$. Since M_R is finite, there are finitely many strings $w_1, \ldots, w_n \in L$ such that $[L]_R = \bigcup_{i=1,\ldots,n} [w_i]_R$. Since M_R is a finite group, $[w]_R$ is a regular language for each $w \in \Sigma^*$. Thus, $\Delta_{R\cup R^{-1}}^*(L) = \bigcup_{i=1,\ldots,n} [w_i]_R$ is a regular language, that is, $R \cup R^{-1}$ preserves regularity.

(a) $\Rightarrow$ (b): Assume that $R \cup R^{-1}$ preserves regularity. The singleton set $\{\lambda\} \subseteq \Sigma^*$ is a regular language, and $\Delta_{R\cup R^{-1}}^*(\lambda) = [\lambda]_R$. Since $R \cup R^{-1}$ preserves regularity, $[\lambda]_R$ is a regular language, and hence, M_R is a regular group. However, a finitely presented group is regular if and only if it is finite [Ani71]. Hence, M_R is a finite group. □

However, the property of being finite is a Markov property of finitely presented groups, and hence, the following decision problem cannot be solved algorithmically (see, e.g., [LS77]):

INSTANCE: A finite group presentation $\langle \Sigma; L \rangle$.

QUESTION: Is the group G_L presented by $\langle \Sigma; L \rangle$ finite?

Using Lemma 4.1 we now reduce this undecidable problem to the problem of deciding whether or not a finite string-rewriting system preserves regularity.

Theorem 4.2 *The following problem is undecidable in general:*

INSTANCE: *A finite string-rewriting system R on Σ.*

QUESTION: *Does R preserve regularity?*

Proof. Let $\langle \Gamma; u_1, \ldots, u_n \rangle$ be a finite group-presentation. Let $\overline{\Gamma}$ be an alphabet in one-to-one correspondence to Γ such that $\Gamma \cap \overline{\Gamma} = \emptyset$, let $\underline{\Gamma} := \Gamma \cup \overline{\Gamma}$, and let R be the string-rewriting system on $\underline{\Gamma}$ containing the following rules:

$$\begin{array}{ll} a\bar{a} \to \lambda, \quad \bar{a}a \to \lambda & (a \in \Gamma), \\ u_i \to \lambda \quad (i = 1, \ldots, n). & \end{array}$$

Then $(\underline{\Gamma}; R)$ is a finite monoid-presentation for the group G presented by $\langle \Gamma; u_1, \ldots, u_n \rangle$.

Now the string-rewriting system $R \cup R^{-1}$ preserves regularity if and only if the group G is finite (Lemma 4.1). Since $R \cup R^{-1}$ is easily obtained from the given group presentation $\langle \Gamma; u_1, \ldots, u_n \rangle$, the undecidability of finiteness for finitely presented groups implies Theorem 4.2. □

Actually, the proof of Lemma 4.1 shows that the group G presented by $\langle \Gamma; u_1, \ldots, u_n \rangle$ is finite if and only if the language $\Delta^*_{R \cup R^{-1}}(\lambda)$ is regular. Thus, we obtain the following stronger undecidability result.

Corollary 4.3 *The following problem is undecidable in general:*

INSTANCE: *A finite string-rewriting system R on Σ, and a regular language $L \subseteq \Sigma^*$.*

QUESTION: *Is the language $\Delta^*_R(L)$ regular?*

In fact, Corollary 4.3 remains valid even if the language L is fixed to the set $L := \{\lambda\}$. Theorem 4.2 and Corollary 4.3 improve upon Theorem 7 and Theorem 8 of [GT95], since here we are only dealing with strings, that is, with signatures containing only unary function symbols and possibly a single constant. In addition, our proof is much simpler than the one given in [GT95].

The second undecidability result improves upon Corollary 4.3. It states that the problem considered in Corollary 4.3 remains undecidable even if R is restricted to finite convergent string-rewriting systems. Actually, a fixed convergent system R can be chosen here, if it is constructed accordingly. Below such a construction is described in detail.

Let $M = (Q, \Sigma, \delta, q_0, q_a)$ be a deterministic single-tape Turing machine that accepts a language $L \subseteq \Sigma^*$, where $\Sigma_b := \Sigma \cup \{b\}$ is the tape alphabet, b denotes the blank symbol, Q is the set of states, $\delta : \Sigma_b \times (Q \smallsetminus \{q_a\}) \to Q \times (\Sigma \cup \{\ell, r\})$ is the transition function, $q_0 \in Q$ is the initial state, $q_a \in Q$ is the final state, and $\vdash^*_M$ denotes the reflexive and transitive closure of the single-step computation relation $\vdash_M$ of M. Observe that we assume without loss of generality that the Turing machine M cannot print the blank symbol b, that is, each tape square that has been visited by the head of M contains a non-blank symbol afterwards. Thus, for each $w \in \Sigma^*$, $q_0 w \vdash^*_M u q_a v$ for some $u, v \in \Sigma^*$ if and only if $w \in L$. Without loss of generality we may also assume that $\Sigma_b \cap Q = \emptyset$.

From M we obtain another single-tape Turing maschine $\overline{M} = (\overline{Q}, \Gamma, \bar{\delta}, q_0, q_a)$ as follows. Let $\Gamma := \Sigma \cup \{1, 2, \uparrow\}$, where 1, 2, and $\uparrow$ are three new symbols, and define $\overline{Q}$ and $\bar{\delta}$ in such a way that $\overline{M}$ simulates the Turing machine M as follows:

Whenever $q_0 w \vdash_M^n u_1 q_1 v_1 \vdash_M u_2 q_2 v_2$, where $w \in \Sigma^*$, $u_1, v_1, u_2, v_2 \in \Sigma^*$, $q_1, q_2 \in Q$, and $n \geq 0$, then $\overline{M}$ performs the following computation:

$$\begin{array}{llllll} q_0 w & \vdash_{\overline{M}}^* & 1^n 2^n u_1 q_1 v_1 & \vdash_{\overline{M}}^* & \tilde{q}_1 1^n 2^n u_1 \uparrow v_1 & \vdash_{\overline{M}}^* \; 1^{n+1} 2^{n+1} \hat{q}_1 u_1 \uparrow v_1 \\ & \vdash_{\overline{M}}^* & 1^{n+1} 2^{n+1} u_1 \bar{q}_1 v_1 & \vdash_{\overline{M}} & 1^{n+1} 2^{n+1} u_2 q_2 v_2. & \end{array}$$

Thus, if $q_0 w \vdash_M^n uqv$ for some $w \in \Sigma^*$, $u, v \in \Sigma^*$, $q \in Q$, and $n \in \mathbb{N}$, then $q_0 w \vdash_{\overline{M}}^* 1^n 2^n uqv$. In particular, $\overline{M}$ also accepts the language L.

For $w \in \Sigma^*$, let $\Delta_{\overline{M}}(w) := \{uqv \mid q_0 w \vdash_{\overline{M}}^* uqv \in \Gamma^* \cdot \overline{Q} \cdot \Gamma^*\}$.

Lemma 4.4 *For $w \in \Sigma^*$, the following two statements are equivalent:*

(a) $\Delta_{\overline{M}}(w)$ is a regular language;

(b) $w \in L$.

Proof. (b) $\Rightarrow$ (a): If $w \in L$, then $\overline{M}$ accepts on input w, that is, $\overline{M}$ halts on input w after performing a finite number of steps. Thus, the set $\Delta_{\overline{M}}(w)$ is finite.

(a) $\Rightarrow$ (b): If $w \notin L$, then $\overline{M}$ does not halt on input w, and the same is true for M. Thus, there is an infinite computation of the form $q_0 w \vdash_M u_1 q_1 v_1 \vdash_M u_2 q_2 v_2 \vdash_M \ldots$. Hence, $\overline{M}$ performs an infinite computation of the following form: $q_0 w \vdash_{\overline{M}}^* 12 u_1 q_1 v_1 \vdash_{\overline{M}}^* 1^2 2^2 u_2 q_2 v_2 \vdash_{\overline{M}}^* \ldots$. Assume that the set $\Delta_{\overline{M}}(w)$ were regular. Consider the set $\Delta'(w) := \Delta_{\overline{M}}(w) \cap 1^+ \cdot 2^+ \cdot ((\Gamma \smallsetminus \{1,2\}) \cup \overline{Q})^+$. With $\Delta_{\overline{M}}(w)$ also the set $\Delta'(w)$ is regular. However, for all $z \in \Delta'(w)$, there exists $i \in \mathbb{N}_+$ and $z' \in ((\Gamma \smallsetminus \{1,2\}) \cup \overline{Q})^+$ such that $z = 1^i 2^i z'$. Conversely, if $1^i 2^j z' \in \Delta'(w)$, where $z' \in ((\Gamma \smallsetminus \{1,2\}) \cup \overline{Q})^+$, then $i = j$, and for each $i \in \mathbb{N}_+$, $1^i 2^i z' \in \Delta'(w)$ for some $z' \in ((\Gamma \smallsetminus \{1,2\}) \cup \overline{Q})^+$. Thus, the set $\Delta'(w)$ does not satisfy the pumping lemma for regular languages, and hence, $\Delta'(w)$ is not regular. Accordingly, the set $\Delta_{\overline{M}}(w)$ is not regular. $\square$

From the Turing machine $\overline{M}$ we now construct a finite, length-reducing, and confluent string-rewriting system $R(\overline{M})$ that simulates the computations of $\overline{M}$. Let \$, ¢, and d be three additional symbols, and let $\Gamma_0 := \Gamma_b \cup \overline{Q} \cup \{\$, ¢, d\}$. The symbols \$ and ¢ will serve as left and right end markers, respectively, of configurations of $\overline{M}$, while the symbol d is being used to ensure that the rules of $R(\overline{M})$ are length-reducing. The system $R(\overline{M})$ consists of the following three groups of rules:

(1.) Rules to simulate the stepwise behaviour of $\overline{M}$:

$$\left.\begin{array}{rcll} q_i a_k dd & \to & q_j a_\ell & \text{if } \bar{\delta}(q_i, a_k) = (q_j, a_\ell) \\ q_i ¢ dd & \to & q_j a_\ell ¢ & \text{if } \bar{\delta}(q_i, b) = (q_j, a_\ell) \\ q_i a_k dd & \to & a_k q_j & \text{if } \bar{\delta}(q_i, a_k) = (q_j, r) \\ q_i ¢ dd & \to & b q_j ¢ & \text{if } \bar{\delta}(q_i, b) = (q_j, r) \\ a_\ell q_i a_k dd & \to & q_j a_\ell a_k & \text{if } \bar{\delta}(q_i, a_k) = (q_j, \ell) \\ a_\ell q_i ¢ dd & \to & q_j a_\ell ¢ & \text{if } \bar{\delta}(q_i, b) = (q_j, \ell) \\ \$ q_i a_k dd & \to & \$ q_j b a_k & \text{if } \bar{\delta}(q_i, a_k) = (q_j, \ell) \\ \$ q_i ¢ dd & \to & \$ q_j b ¢ & \text{if } \bar{\delta}(q_i, b) = (q_j, \ell) \end{array}\right\} \text{for all } a_\ell \in \Gamma_b$$

(2.) Rules to shift occurrences of the symbol d to the left:

$$\left.\begin{array}{rcl} a_i a_j dd & \to & a_i d a_j \\ a_i d a_j dd & \to & a_i dd a_j \end{array}\right\} \text{for all } a_i \in \Gamma_b, a_j \in \Gamma_b \cup \{¢\}$$

(3.) Rules to erase halting configurations:

$$\begin{array}{rcl} q_a a_i dd & \to & q_a \\ a_i q_a ¢ d & \to & q_a ¢ \\ \$ q_a ¢ d & \to & \$ q_a ¢ \end{array}$$

The system $R(\overline{M})$ has the following properties.

Proposition 4.5 (cf. [Ott91], Proposition 3.1)

(a) The string-rewriting system $R(\overline{M})$ is finite, length-reducing, and confluent.

(b) For $w \in \Sigma^*$, the following two statements are equivalent:

(1.) $w \in L$; and

(2.) $\exists m \in \mathbb{N} \forall n \geq m : \$q_0 w ¢ d^n \to^*_{R(\overline{M})} \$q_a ¢ d^{n-m} \to^*_{R(\overline{M})} \$q_a ¢$.

From Lemma 4.4 and Proposition 4.5(b) we now obtain the following characterization.

Lemma 4.6 *For $w \in \Sigma^*$, the following two statements are equivalent:*

(a) $\Delta^*_{R(\overline{M})}(\$q_0 w ¢ \cdot d^*)$ is a regular language;

(b) $w \in L$.

Proof. (b) $\Rightarrow$ (a): If $w \in L$, then $\$q_0 w ¢ d^n \to^*_{R(\overline{M})} \$q_a ¢ d^{n-m} \to^{n-m}_{R(\overline{M})} \$q_a ¢$ for all $n \geq m$ by Proposition 4.5(b). Thus, $\Delta^*_{R(\overline{M})}(\$q_0 w ¢ \cdot d^*) = \bigcup\limits_{i=0}^{m} \Delta^*_{R(\overline{M})}(\$q_0 w ¢ \cdot d^i) \cup \Delta^*_{R(\overline{M})}(\$q_0 w ¢ \cdot d^m) \cdot d^*$. Since $R(\overline{M})$ is length-reducing, each of the finitely many languages $\Delta^*_{R(\overline{M})}(\$q_0 w ¢ \cdot d^i)$, $i = 0, 1, \ldots, m$, is finite. Thus, $\Delta^*_{R(\overline{M})}(\$q_0 w ¢ \cdot d^*)$ is a regular language.

(a) $\Rightarrow$ (b): Assume that $w \notin L$. Let $\varphi : \Gamma_0 \to \Gamma_b \cup \overline{Q}$ denote the projection, that is, $\varphi(a) := a$ for all $a \in \Gamma_b \cup \overline{Q}$, and $\varphi(\$) = \varphi(¢) = \varphi(d) = \lambda$. Then $\varphi(\Delta^*_{R(\overline{M})}(\$q_0 w ¢ \cdot d^*)) = \Delta_{\overline{M}}(w)$, which is not regular by Lemma 4.4. Hence, the set $\Delta^*_{R(\overline{M})}(\$q_0 w ¢ \cdot d^*)$ is not regular, either. □

Now choose L to be a nonrecursive language. Then Lemma 4.6 yields the following undecidability result.

Theorem 4.7 *There exists a finite, length-reducing, and confluent string-rewriting system R such that the following problem is undecidable:*

INSTANCE: *A regular set $S \subseteq \Sigma^*$.*

QUESTION: *Is the set of descendants $\Delta^*_R(S)$ regular?*

If $w \in L$, then $\Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^n) \cap \mathrm{IRR}(R(\overline{M})) = \{\$q_a¢\}$ for all $n \geq m$. Thus, $\mathrm{NF}_{R(\overline{M})}(\$q_0w¢ \cdot d^*) = \Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^*) \cap \mathrm{IRR}(R(\overline{M}))$ is a finite set, and hence, it is regular. On the other hand, if $w \notin L$, then $\varphi(\Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^*) \cap \mathrm{IRR}(R(\overline{M}))) = \Delta_{\overline{M}}(w)$, which is not a regular set in this case by Lemma 4.4. Thus, $\mathrm{NF}_{R(\overline{M})}(\$q_0w¢ \cdot d^*) = \Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^*) \cap \mathrm{IRR}(R(\overline{M}))$ is not regular, if $w \notin L$, that is, the set of normal forms in the language $\Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^*)$ is regular if and only if $w \in L$. Thus, we obtain the following corollary, which improves upon Theorem 10 of [GT95].

Corollary 4.8 *There exists a finite, length-reducing, and confluent string-rewriting system R such that the following problem is undecidable:*

INSTANCE: *A regular set $S \subseteq \Sigma^*$.*
QUESTION: *Is the set $\mathrm{NF}_R(S)$ regular?*

For the string-rewriting system $R(\overline{M})$ and the languages of the form $\$q_0w¢ \cdot d^*$ ($w \in \Sigma^*$), the following properties hold:

If $w \in L$, then $\Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^*)$ is a regular language, and hence, also $\mathrm{NF}_{R(\overline{M})}(\$q_0w¢ \cdot d^*) = \Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^*) \cap \mathrm{IRR}(R(\overline{M}))$ is a regular language, but if $w \notin L$, then neither $\Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^*)$ nor $\mathrm{NF}_{R(\overline{M})}(\$q_0w¢ \cdot d^*)$ is a regular language. Thus, $\mathrm{NF}_{R(\overline{M})}(\$q_0w¢ \cdot d^*)$ is regular if and only if $\Delta^*_{R(\overline{M})}(\$q_0w¢ \cdot d^*)$ is regular. However, by adding some rules to the string-rewriting system $R(\overline{M})$ to obtain a string-rewriting system $R_0(\overline{M})$ on $\Gamma_0 \cup \{\#, z\}$, where $\#$ and z are two new symbols, we can avoid this equivalence, showing that $\mathrm{NF}_{R_0(\overline{M})}(S)$ can be regular even if $\Delta^*_{R_0(\overline{M})}(S)$ is not regular for some regular languages S.

Let $\Gamma_1 := \Gamma_0 \cup \{\#, z\}$, and let $R_0(\overline{M}) := R(\overline{M}) \cup R_0$, where R_0 contains the following rules:

$$¢\# \to z, \quad ¢d\# \to z, \quad za \to z, \quad az \to z \quad (a \in \Gamma_1).$$

Then $R_0(\overline{M})$ is a finite length-reducing string-rewriting system, and it can easily be verified that $R_0(\overline{M})$ is confluent.

For $w \in \Sigma^*$, consider the language $S(w) := \$q_0w¢ \cdot d^* \cdot \#$. If $w \in L$, and if n is sufficiently large, then $\$q_0w¢ \cdot d^n \cdot \# \to^*_{R(\overline{M})} \$q_a¢\# \to^*_{R(\overline{M})} z$. If n is small, then $\$q_0w¢ \cdot d^n \cdot \# \to^*_{R(\overline{M})} \$uqv¢d^\varepsilon \cdot \# \to^*_{R(\overline{M})} z$, where $\varepsilon \in \{0,1\}$. In this situation, $\Delta^*_{R_0(\overline{M})}(S(w))$ is a regular language, and $\mathrm{NF}_{R_0(\overline{M})}(S(w)) = \{z\}$. If $w \notin L$, then, for all $n \in \mathbb{N}$, $\$q_0w¢ \cdot d^n \cdot \# \to^*_{R(\overline{M})} \$uqv¢d^\varepsilon \cdot \# \to^*_{R(\overline{M})} z$, where $\varepsilon \in \{0,1\}$. The rules of R_0 cannot be used before all the d's (but one) to the right-hand side of the ¢-symbol have been used up. Thus, it follows as in the proof of Lemma 4.6 that $\Delta^*_{R_0(\overline{M})}(S(w))$ is not regular. However, $\mathrm{NF}_{R_0(\overline{M})}(S(w)) = \{z\}$ holds also in this situation. Thus, the following undecidability result follows.

Theorem 4.9 *There exists a finite, length-reducing, and confluent string-rewriting system R such that the following problem is undecidable:*

INSTANCE: *A regular set $S \subseteq \Sigma^*$ such that $\mathrm{NF}_R(S)$ is a singleton.*
QUESTION: *Is the set of descendants $\Delta^*_R(S)$ regular?*

So far we have seen that Problem 2 and Problem 4 are undecidable, even for a fixed finite string-rewriting system that is length-reducing and confluent (Theorem 4.7 and Corollary

4.8). In the remaining part of this paper we want to prove that also Problem 3 is undecidable in general, when it is restricted to finite string-rewriting systems that are length-reducing and confluent. For that, however, we need to consider the strong boundedness problem for single-tape Turing machines.

5 The strong boundedness problem for Turing machines

A possibly infinite configuration C of a single-tape Turing machine M is called *immortal* if M does never halt when starting with C. In [Hoo66] Hooper shows that it is undecidable whether a Turing machine has an immortal configuration.

We call a Turing machine M *strongly bounded* if there exists an integer k such that, for each finite configuration C, M halts after at most k steps when starting in configuration C. Here a configuration is called finite if almost all tape squares contain the blank symbol b. We are interested in the *strong boundedness problem* for Turing machines, which is the following decision problem:

INSTANCE: A single-tape Turing machine M.

QUESTION: Is M strongly bounded?

In [Hoo66] Hooper proceeds as follows. He first observes that the halting problem is undecidable for the class of two-counter Minsky machines that start with empty counters. Then he constructs a 2-symbol Turing machine $\overline{M}$ from a Minsky machine $\hat{M}$ such that $\overline{M}$ has an immortal configuration if and only if $\hat{M}$ does not halt from its initial configuration with empty counters. Since the configurations of $\hat{M}$ are encoded as certain finite configurations of $\overline{M}$, and since $\overline{M}$ simulates $\hat{M}$, though in a very involved way, this shows that $\overline{M}$ has an immortal finite configuration if and only if it has an immortal configuration. It follows that the immortality problem is undecidable for 2-symbol Turing machines, even when it is restricted to finite configurations.

Now assume that the Turing machine $\overline{M}$ has finite computations of arbitrary length. Then it must also have an infinite computation, though one that possibly starts with an infinite configuration (cf. the proof of Corollary 6 of [KTU93]). But then we see from the above discussion that $\overline{M}$ also has an infinite computation that starts with a finite configuration. Thus, if $\overline{M}$ has no immortal finite configuration, then it is strongly bounded. Since the converse is obvious, we obtain the following undecidability result.

Proposition 5.1 *The strong boundedness problem is undecidable for 2-symbol single-tape Turing machines.*

6 The reduction

We will prove that Problem 3 is undecidable for finite string-rewriting systems that are length-reducing and confluent by reducing the strong boundedness problem for Turing machines to it. For this, we need a simulation of Turing machines through finite, length-reducing, and confluent string-rewriting systems that is based on the simple simulation given in Section 4.

Let $M = (Q, \Sigma, q_0, q_a, \delta)$ be a deterministic single-tape Turing machine, where we assume that Σ consists of a symbol a and the blank symbol b only. From M we now construct a finite string-rewriting system R for simulating M.

Let $\overline{\Sigma} := \{\bar{a}, \bar{b}\}$, let $\overline{Q}$ be another new alphabet that is in 1-to-1 correspondence to Q, and let $\Gamma := Q \cup \overline{Q} \cup \Sigma \cup \overline{\Sigma} \cup \{1, 2, \$, ¢, d, \bar{d}, \hat{d}, d_0, \hat{c}, \bar{c}, 0\}$, where $1, 2, \$, ¢, d, \bar{d}, \hat{d}, d_0, \hat{c}, \bar{c}, 0$ are 11 additional new symbols.

The string-rewriting system R will consist of two main parts, that is, $R := R_1 \cup R_2$, where R_1 is a system that simulates the computations of the Turing machine M step by step, and R_2 is a system that destroys unwanted strings. We first define the system R_1. It consists of the following 5 groups of rules.

(1.) Rules to simulate the Turing machine M:

$$\begin{array}{lcll}
q_i a_k d d a_r & \to & \bar{q}_j a_\ell a_r & \text{for all } a_r \in \Sigma \cup \{¢\}, \text{if } \delta(q_i, a_k) = (q_j, a_\ell) \\
q_i ¢ d_0 d_0 & \to & \bar{q}_j a_\ell ¢ & \text{if } \delta(q_i, b) = (q_j, a_\ell) \\
q_i a_k d d a_r & \to & \bar{a}_k \bar{q}_j a_r & \text{for all } a_r \in \Sigma \cup \{¢\}, \text{if } \delta(q_i, a_k) = (q_j, r) \\
q_i ¢ d_0 d_0 & \to & \bar{b} \bar{q}_j ¢ & \text{if } \delta(q_i, b) = (q_j, r) \\
\bar{a}_\ell q_i a_k d d a_r & \to & \bar{q}_j a_\ell a_k a_r & \text{for all } a_r \in \Sigma \cup \{¢\}, \text{if } \delta(q_i, a_k) = (q_j, \ell) \quad \Big\} \text{ for all } \bar{a}_\ell \in \overline{\Sigma} \\
\bar{a}_\ell q_i ¢ d_0 d_0 & \to & \bar{q}_j a_\ell ¢ & \text{if } \delta(q_i, b) = (q_j, \ell) \\
\$ q_i a_k d d a_r & \to & \$ \bar{q}_j b a_k a_r & \text{for all } a_r \in \Sigma \cup \{¢\}, \text{if } \delta(q_i, a_k) = (q_j, \ell) \\
\$ q_i ¢ d_0 d_0 & \to & \$ \bar{q}_j b ¢ & \text{if } \delta(q_i, b) = (q_j, \ell).
\end{array}$$

(2.) Rules to shift d to the left:

$$\left.\begin{array}{lcl}
a_i a_j d d a_r & \to & a_i d a_j a_r \\
a_i d a_j d d a_r & \to & a_i d d a_j a_r
\end{array}\right\} \text{for all } a_i \in \Sigma \cup \overline{Q}, a_j \in \Sigma, \text{ and } a_r \in \Sigma \cup \{¢\}$$

$$\left.\begin{array}{lcl}
a_i ¢ d_0 d_0 & \to & a_i d ¢ \\
a_i d ¢ d_0 d_0 & \to & a_i d d ¢
\end{array}\right\} \text{for all } a_i \in \Sigma \cup \overline{Q}$$

$$\left.\begin{array}{lcl}
\bar{a}_i \bar{q}_j d d a_r & \to & \bar{a}_i \bar{d} \bar{q}_j a_r \\
\bar{a}_i \bar{d} \bar{q}_j d d a_r & \to & \bar{a}_i \bar{d} \bar{d} \bar{q}_j a_r
\end{array}\right\} \text{for all } \bar{a}_i \in \overline{\Sigma} \cup \{\$\}, \bar{q}_j \in \overline{Q}, \text{ and } a_r \in \Sigma \cup \{¢\}$$

(3.) Rules to shift $\bar{d}$ and $\hat{d}$ to the left:

$$\left.\begin{array}{lcl}
\bar{a}_i \bar{a}_j \bar{d} \bar{d} \bar{a}_r & \to & \bar{a}_i \bar{d} \bar{a}_j \bar{a}_r \\
\bar{a}_i \bar{d} \bar{a}_j \bar{d} \bar{d} \bar{a}_r & \to & \bar{a}_i \bar{d} \bar{d} \bar{a}_j \bar{a}_r
\end{array}\right\} \text{for all } \bar{a}_i \in \overline{\Sigma} \cup \{\$\}, \bar{a}_j \in \overline{\Sigma}, \text{ and } \bar{a}_r \in \overline{\Sigma} \cup \overline{Q} \cup Q$$

$$\left.\begin{array}{lcl}
2 \$ \bar{d} \bar{d} \bar{a}_r & \to & 2 \hat{d} \$ \bar{a}_r \\
2 \hat{d} \$ \bar{d} \bar{d} \bar{a}_r & \to & 2 \hat{d} \hat{d} \$ \bar{a}_r
\end{array}\right\} \text{for all } \bar{a}_r \in \overline{\Sigma} \cup \overline{Q} \cup Q$$

$$\begin{array}{lcl}
22 \hat{d} \hat{d} \$ & \to & 2 \hat{d} 2 \$ \\
22 \hat{d} \hat{d} 2 & \to & 2 \hat{d} 22 \\
2 \hat{d} 2 \hat{d} \hat{d} \$ & \to & 2 \hat{d} \hat{d} 2 \$ \\
2 \hat{d} 2 \hat{d} \hat{d} 2 & \to & 2 \hat{d} \hat{d} 22
\end{array}$$

(4.) Rules to increase the number of 1's and 2's:

$$\left.\begin{array}{lcl}
12 \hat{d} \hat{d} a & \to & 1 \hat{d} 2 a \\
1 \hat{d} 2 \hat{d} \hat{d} a & \to & 11 \hat{c} 2 a \\
1 \hat{c} 2 \hat{d} \hat{d} a & \to & 122 \hat{c} a
\end{array}\right\} \text{for all } a \in \{2, \$\}$$

(5.) Rules to shift $\hat{c}$ and $\bar{c}$ to the right:

$$
\begin{array}{lcll}
2\hat{c}2\hat{d}\hat{d}a & \rightarrow & 22\hat{c}a & \text{for all } a \in \{2,\$\} \\
2\hat{c}\$\bar{d}\bar{d}\bar{a} & \rightarrow & 2\$\bar{c}\bar{a} & \multirow{2}{*}{$\left.\right\}$ for all $\bar{a} \in \overline{\Sigma} \cup \overline{Q} \cup Q, \bar{a}_i \in \overline{\Sigma} \cup \{\$\}$, and $\bar{a}_j \in \overline{\Sigma}$} \\
\bar{a}_i\bar{c}\bar{a}_j\bar{d}\bar{d}\bar{a} & \rightarrow & \bar{a}_i\bar{a}_j\bar{c}\bar{a} & \\
\bar{a}_i\bar{c}\bar{q}_j dda & \rightarrow & \bar{a}_i q_j a & \text{for all } \bar{a}_i \in \overline{\Sigma} \cup \{\$\}, \bar{q}_j \in \overline{Q}, \text{ and } a \in \Sigma \cup \{¢\}.
\end{array}
$$

Obviously, R_1 is a finite and length-reducing system. Since the Turing machine M is deterministic, there are no overlaps between the rules of group (1.). There are overlaps between the rules of group (1.) and the rules of the groups (2.) to (5.), but they all resolve trivially. Also all the overlaps between the rules of groups (2.) to (5.) resolve trivially. Thus, R_1 is in addition confluent.

Lemma 6.1 *(i)* $\forall n \geq 2\ \forall a_1 \in \Sigma \cup \overline{Q}\ \forall a_2, \ldots, a_{n-1} \in \Sigma :$

$$a_1 a_2 \ldots a_{n-1} ¢ d_0^{2^n} \rightarrow^*_{R_1} a_1 d^2 a_2 \ldots a_{n-1} ¢.$$

(ii) $\forall m, n \geq 1\ \forall \bar{a}_1 \in \overline{\Sigma} \cup \{\$\}\ \forall \bar{a}_2, \ldots, \bar{a}_m \in \overline{\Sigma}\ \forall \bar{q} \in \overline{Q}\ \forall a'_1, \ldots, a'_{n-1} \in \Sigma :$

$$\bar{a}_1 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_{n-1} ¢ d_0^{2^{m+n+1}} \rightarrow^*_{R_1} \bar{a}_1 \bar{d}^2 \bar{a}_2 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_{n-1} ¢.$$

(iii) $\forall t, m, n \geq 0\ \forall \bar{a}_1, \ldots, \bar{a}_m \in \overline{\Sigma}\ \forall \bar{q} \in \overline{Q}\ \forall a'_1, \ldots, a'_n \in \Sigma :$

$$2^{t+1}\$\bar{a}_1 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_n ¢ d_0^{2^{t+m+n+4}} \rightarrow^*_{R_1} 2\hat{d}^2 2^t \$\bar{a}_1 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_n ¢.$$

(iv) $\forall t \geq 1\ \forall m, n \geq 0\ \forall \bar{a}_1, \ldots, \bar{a}_m \in \overline{\Sigma}\ \forall \bar{q} \in \overline{Q}\ \forall a'_1, \ldots, a'_n \in \Sigma :$

$$12^t\$\bar{a}_1 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_n ¢ d_0^{2^{t+m+n+3}} \rightarrow^*_{R_1} 1\hat{d}2^t\$\bar{a}_1 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_n ¢.$$

(v) $\forall s, t \geq 1\ \forall m, n \geq 0\ \forall \bar{a}_1, \ldots, \bar{a}_m \in \overline{\Sigma}\ \forall \bar{q} \in \overline{Q}\ \forall a'_1, \ldots, a'_n \in \Sigma :$

$$1^s 2^t \$\bar{a}_1 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_n ¢ d_0^{2^{t+m+n+5} - 2^{m+n+4}} \rightarrow^*_{R_1} 1^{s+1} 2^{t+1} \hat{c} \$\bar{a}_1 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_n ¢.$$

(vi) $\forall m, n \geq 0\ \forall \bar{a}_1, \ldots, \bar{a}_m \in \overline{\Sigma}\ \forall \bar{q} \in \overline{Q}\ \forall a'_1, \ldots, a'_n \in \Sigma :$

$$2\hat{c}\$\bar{a}_1 \ldots \bar{a}_m \bar{q} a'_1 \ldots a'_n ¢ d_0^{2^{m+n+4} - 2^{n+2}} \rightarrow^*_{R_1} 2\$\bar{a}_1 \ldots \bar{a}_m q a'_1 \ldots a'_n ¢.$$

Proof.

(i) We proceed by induction on n:

$\boldsymbol{n = 2}$**:** $a_1 ¢ d_0^4 \rightarrow_{R_1} a_1 d ¢ d_0^2 \rightarrow_{R_1} a_1 dd¢$.

$\boldsymbol{n \rightarrow n{+}1}$**:** $a_1 a_2 \ldots a_n ¢ d_0^{2^{n+1}} \rightarrow^*_{R_1} \underline{a_1 a_2 d^2 a_3} \ldots a_n ¢ d_0^{2^n}$ (by the induction hypothesis)

$\rightarrow_{R_1} a_1 d a_2 a_3 \ldots a_n ¢ d_0^{2^n} \rightarrow^*_{R_1} \underline{a_1 d a_2 d^2 a_3} \ldots a_n ¢$ (by the induction hypothesis)

$\rightarrow_{R_1} a_1 d^2 a_2 a_3 \ldots a_n ¢$.

(ii) We proceed by induction on m:

$\boldsymbol{m = 1}$**:** $\bar{a}_1 \bar{q} a'_1 \ldots a'_{n-1} ¢ d_0^{2^{n+2}} \rightarrow^*_{R_1} \underline{\bar{a}_1 \bar{q} d^2 a'_1} \ldots a'_{n-1} ¢ d_0^{2^{n+1}}$ (by (i))

$\rightarrow_{R_1} \bar{a}_1 \bar{d} \bar{q} a'_1 \ldots a'_{n-1} ¢ d_0^{2^{n+1}} \rightarrow^*_{R_1} \underline{\bar{a}_1 \bar{d} \bar{q} d^2 a'_1} \ldots a'_{n-1} ¢$ (by (i))

$\rightarrow_{R_1} \bar{a}_1 \bar{d}^2 \bar{q} a'_1 \ldots a'_{n-1} ¢$.

$\boldsymbol{m \to m{+}1}$: $\bar{a}_1\bar{a}_2\ldots\bar{a}_{m+1}\bar{q}a'_1\ldots a'_{n-1}¢\, d_0^{2^{m+n+2}}$
$\to^*_{R_1}$ $\underline{\bar{a}_1\bar{a}_2\bar{d}^2\bar{a}_3}\ldots\bar{a}_{m+1}\bar{q}a'_1\ldots a'_{n-1}¢\, d_0^{2^{m+n+1}}$ (by the induction hypothesis)

$\to_{R_1}$ $\bar{a}_1\bar{d}\bar{a}_2\bar{a}_3\ldots\bar{a}_{m+1}\bar{q}a'_1\ldots a'_{n-1}¢\, d_0^{2^{m+n+1}}$
$\to^*_{R_1}$ $\underline{\bar{a}_1\bar{d}\bar{a}_2\bar{d}^2\bar{a}_3}\ldots\bar{a}_{m+1}\bar{q}a'_1\ldots a'_{n-1}¢$ (by the induction hypothesis)
$\to_{R_1}$ $\bar{a}_1\bar{d}^2\bar{a}_2\bar{a}_3\ldots\bar{a}_{m+1}\bar{q}a'_1\ldots a'_{n-1}¢$.

(iii) We proceed by induction on t:

$\boldsymbol{t = 0}$: $2\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{m+n+4}} \to^*_{R_1} \underline{2\$\bar{d}^2\bar{a}_1}\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{m+n+3}}$ (by (ii))
$\to_{R_1}$ $2\hat{d}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{m+n+3}} \to^*_{R_1} \underline{2\hat{d}\$\bar{d}^2\bar{a}_1}\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢$ (by (ii))
$\to_{R_1}$ $2\hat{d}^2\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢$.

$\boldsymbol{t \to t{+}1}$: $2^{t+2}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{t+m+n+5}}$
$\to^*_{R_1}$ $\underline{22\hat{d}^2 2^t}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{t+m+n+4}}$ (by the induction hypothesis)
$\to_{R_1}$ $2\hat{d}2^{t+1}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{t+m+n+4}}$
$\to^*_{R_1}$ $\underline{2\hat{d}2\hat{d}^2 2^t}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢$ (by the induction hypothesis)
$\to_{R_1}$ $2\hat{d}^2 2^{t+1}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢$.

(iv) By (iii) we have $12^t\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{t+m+n+3}} \to^*_{R_1} \underline{12\hat{d}^2 2^{t-1}}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢$ $\to_{R_1}$ $1\hat{d}2^t\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢$.

(v) $1^s2^t\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{t+m+n+5}-2^{m+n+4}}$
$\to^*_{R_1}$ $\underline{1^s\hat{d}2\hat{d}^2 2^{t-1}}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{t+m+n+4}-2^{m+n+4}}$ (by (iii) and (iv))
$\to_{R_1}$ $1^{s+1}\hat{c}2^t\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{t+m+n+4}-2^{m+n+4}}$
$\to^*_{R_1}$ $1^{s+1}\hat{c}(2\hat{d}^2)^t\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢$ (by (iii))
$\to^*_{R_1}$ $1^{s+1}2^{t+1}\hat{c}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢$.

(vi) $2\hat{c}\$\bar{a}_1\ldots\bar{a}_m\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{m+n+4}-2^{n+2}} \to^*_{R_1} 2\hat{c}\$\bar{d}^2\bar{a}_1\ldots\bar{d}^2\bar{a}_m\bar{d}^2\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{n+2}}$ (by (ii))
$\to^*_{R_1}$ $2\$\bar{a}_1\ldots\bar{a}_m\bar{c}\bar{q}a'_1\ldots a'_n¢\, d_0^{2^{n+2}} \to^*_{R_1} 2\$\bar{a}_1\ldots\underline{\bar{a}_m\bar{c}\bar{q}d^2a'_1}\ldots a'_n¢$ (by (i))
$\to_{R_1}$ $2\$\bar{a}_1\ldots\bar{a}_m qa'_1\ldots a'_n¢$.

This completes the proof of Lemma 6.1. □

Lemma 6.2 *Let uqv be a configuration of the Turing machine M such that $uqv \vdash_M u_1q_1v_1$, and let $k \geq 1$. Then there exists a positive integer $p \in \mathbb{N}$ such that $1^\ell 2^k\$\bar{u}qv¢\, d_0^p \to^*_{R_1} 1^{\ell+1}2^{k+1}\$\bar{u}_1q_1v_1¢$ holds for all $\ell \geq 1$.*

Proof. Let $\ell, k \geq 1$, and let $uqv = a_1\ldots a_m qa'_1\ldots a'_n \vdash_M a_1\ldots a_m q_1\tilde{a}_1a'_2\ldots a'_n = u_1q_1v_1$, that is, $\delta(q, a'_1) = (q_1, \tilde{a}_1)$. The other cases can be dealt with analogously.

(1.) $1^\ell 2^k\$\bar{u}qv¢\, d_0^{2^{n+1}} \to^*_{R_1} 1^\ell 2^k\$\bar{u}\underline{qa'_1d^2a'_2}\ldots a'_n¢$ (by Lemma 6.1(i))

$\to_{R_1}$ $1^\ell 2^k\$\bar{u}\bar{q}_1\tilde{a}_1a'_2\ldots a'_n¢$.

(2.) $1^\ell 2^k\$\bar{u}\bar{q}_1\tilde{a}_1a'_2\ldots a'_n¢\, d_0^{2^{k+m+n+5}-2^{n+2}}$
$\to^*_{R_1}$ $1^{\ell+1}2^{k+1}\hat{c}\$\bar{u}\bar{q}_1\tilde{a}_1a'_2\ldots a'_n¢ \cdot d_0^{2^{m+n+4}-2^{n+2}}$ (by Lemma 6.1.(v))
$\to^*_{R_1}$ $1^{\ell+1}2^{k+1}\$\bar{u}q_1\tilde{a}_1a'_2\ldots a'_n¢$ (by Lemma 6.1(vi))
$= 1^{\ell+1}2^{k+1}\$\bar{u}q_1v_1¢$.

Hence, we have $p = 2^{k+m+n+5} - 2^{n+2} + 2^{n+1} = 2^{k+m+n+5} - 2^{n+1}$. □

Lemma 6.3 *If the Turing machine M has an immortal finite configuration, then R_1 does not preserve regularity.*

Proof. Assume that $u_0q_0v_0$ is an immortal finite configuration of M, that is, $u_0q_0v_0 \vdash_M u_1q_1v_1 \vdash_M u_2q_2v_2 \vdash_M \ldots$ is an infinite computation of M. Consider the regular language $S := \{12\$\bar{u}_0q_0v_0\text{¢} \cdot d_0^i \mid i \geq 0\}$. From Lemma 6.2 we see that, for each $k \geq 1$, there exists an integer $p_k \in \mathbb{N}_+$ such that $12\$\bar{u}_0q_0v_0\text{¢} \cdot d_0^{p_k} \to_{R_1}^* 1^{k+1}2^{k+1}\$\bar{u}_kq_kv_k\text{¢}$. Hence, we see from the form of the rules of R_1 that $\Delta_{R_1}^*(S) \cap 1^+ \cdot 2^+ \cdot \$ \cdot \overline{\Sigma}^* \cdot Q \cdot \Sigma^* \cdot \text{¢} = \{1^{k+1}2^{k+1}\$\bar{u}_kq_kv_k\text{¢} \mid k \geq 0\}$. Since this language does not satisfy the pumpig lemma for regular languages, it is not regular. Thus, the language $\Delta_{R_1}^*(S)$ is not regular. Hence, R_1 does not preserve regularity, if M has an immortal finite configuration. □

Observe that the strings in the set $\Delta_{R_1}^*(S) \cap 1^+ \cdot 2^+ \cdot \$ \cdot \overline{\Sigma}^* \cdot Q \cdot \Sigma^* \cdot \text{¢}$ are all irreducible mod R_1. Thus, $\mathrm{NF}_{R_1}(S) \cap 1^+ \cdot 2^+ \cdot \$ \cdot \overline{\Sigma}^* \cdot Q \cdot \Sigma^* \cdot \text{¢} = \Delta_{R_1}^*(S) \cap 1^+ \cdot 2^+ \cdot \$ \cdot \overline{\Sigma}^* \cdot Q \cdot \Sigma^* \cdot \text{¢}$, and hence, R_1 does not even give regular sets of normal forms for regular languages, if M has an immortal finite configuration.

Now we turn to the string-rewriting system R_2, which constitutes the second part of the system R. The system R_2 consists of the following 18 groups of monadic rules.

$$
\begin{array}{rlcll}
(1.) & s1 & \to & 0 & \text{for all } s \in \Gamma \setminus \{1\};\\
(2.) & s2 & \to & 0 & \text{for all } s \in \Gamma \setminus \{1, 2, \hat{c}, \hat{d}\};\\
(3.) & s\hat{d} & \to & 0 & \text{for all } s \in \Gamma \setminus \{1, 2, \hat{d}\};\\
(4.) & 1\hat{d}\hat{d} & \to & 0 & \\
 & 2\hat{d}\hat{d}\hat{d} & \to & 0 & \\
(5.) & s\hat{c} & \to & 0 & \text{for all } s \in \Gamma \setminus \{1, 2\};\\
(6.) & s\$ & \to & 0 & \text{for all } s \in \Gamma \setminus \{2, \hat{c}, \hat{d}\};\\
(7.) & s\bar{a} & \to & 0 & \text{for all } \bar{a} \in \overline{\Sigma} \text{ and } s \in \Gamma \setminus (\overline{\Sigma} \cup \{\$, \bar{c}, \bar{d}\});\\
(8.) & s\bar{q} & \to & 0 & \text{for all } \bar{q} \in \overline{Q} \text{ and } s \in \Gamma \setminus (\overline{\Sigma} \cup \{\$, \bar{c}, \bar{d}\});\\
(9.) & s\bar{d} & \to & 0 & \text{for all } s \in \Gamma \setminus (\overline{\Sigma} \cup \{\$, \bar{d}\};\\
(10.) & s\bar{d}\bar{d}\bar{d} & \to & 0 & \text{for all } s \in \overline{\Sigma} \cup \{\$\};\\
(11.) & s\bar{c} & \to & 0 & \text{for all } s \in \Gamma \setminus (\overline{\Sigma} \cup \{\$\});\\
(12.) & sq & \to & 0 & \text{for all } q \in Q \text{ and } s \in \Gamma \setminus (\overline{\Sigma} \cup \{\$, \bar{c}, \bar{d}\});\\
(13.) & sa & \to & 0 & \text{for all } a \in \Sigma \text{ and } s \in \Gamma \setminus (Q \cup \overline{Q} \cup \Sigma \cup \{d\});\\
(14.) & sd & \to & 0 & \text{for all } s \in \Gamma \setminus (\overline{Q} \cup \Sigma \cup \{d\});\\
(15.) & sddd & \to & 0 & \text{for all } s \in \overline{Q} \cup \Sigma;\\
(16.) & s\text{¢} & \to & 0 & \text{for all } s \in \Gamma \setminus (Q \cup \overline{Q} \cup \Sigma \cup \{d\});\\
(17.) & sd_0 & \to & 0 & \text{for all } s \in \Gamma \setminus \{\text{¢}, d_0\};\\
(18.) & \left.\begin{array}{l} s0 \\ 0s \end{array}\right. & \begin{array}{c} \to \\ \to \end{array} & \left.\begin{array}{l} 0 \\ 0 \end{array}\right\} & \text{for all } s \in \Gamma.
\end{array}
$$

Obviously, R_2 is a finite and monadic string-rewriting system on Γ. Since each rule of R_2 has right-hand side 0, and since 0 acts as a zero because of the rules of (18.), we can conclude that all the many critical pairs of R_2 resolve to 0. Thus, R_2 is also confluent.

Lemma 6.4 *(a)* $\forall s \in \Gamma \setminus \{d_0\} : d_0s \to 0$.

(b) The set $\mathrm{IRR}(R_2)$ *consists of all the factors of the strings in the following language* $\mathrm{CONF} := 1^* \cdot (\{\hat{c}, \hat{d}\} \cup (\{\lambda, \hat{c}, \hat{d}\} \cdot (2 \cdot \{\lambda, \hat{c}, \hat{d}, \hat{d}\hat{d}\})^+)) \cdot \$ \cdot (\{\lambda, \bar{c}, \bar{d}, \bar{d}\bar{d}\} \cdot \overline{\Sigma})^* \cdot \{\lambda, \bar{c}, \bar{d}, \bar{d}\bar{d}\} \cdot ((\overline{Q} \cdot \{\lambda, d, dd\}) \cup Q) \cdot (\Sigma \cdot \{\lambda, d, dd\})^* \cdot \text{¢} \cdot d_0^* \cup \{0\}$.

Proof.

(a) This is easily seen from the rules of R_2.

(b) First of all, it can be checked easily that $\mathrm{CONF} \subseteq \mathrm{IRR}(R_2)$. On the other hand, each string $w \in \mathrm{IRR}(R_2)$ is a factor of a string from CONF. $\square$

Observe that $1^+ \cdot 2^+ \cdot \$ \cdot \overline{\Sigma}^* \cdot Q \cdot \Sigma^* \cdot ¢ \subseteq \mathrm{CONF}$. Thus, all the strings in the language $\Delta^*_{R_1}(S) \cap 1^+ \cdot 2^+ \cdot \$ \cdot \overline{\Sigma}^* \cdot Q \cdot \Sigma^* \cdot ¢$ considered in the proof of Lemma 6.3 are irreducible mod R_2. Actually, if uqv is a configuration of the Turing machine M, then $\Delta^*_{R_1}(1^\ell 2^k \$ \bar{u} q v ¢ d_0^*) \subseteq \mathrm{CONF}$ for all $\ell, k \geq 0$. In fact, if $w \in \mathrm{IRR}(R_2)$, then $\Delta^*_{R_1}(w) \subseteq \mathrm{IRR}(R_2)$ as can be checked easily.

Finally, we consider the combined system $R := R_1 \cup R_2$.

Lemma 6.5 *R is a finite, length-reducing, and confluent system.*

Proof. It remains to verify that all overlaps between rules of R_1 and rules of R_2 resolve to 0. We look at each group of rules of R_2 in turn.

(1.) If $(1\ell_1 \to r) \in R_1$, then $r = 1 \cdot r_1$, and hence,

$$\begin{tikzcd}
s1\ell_1 \arrow[r] \arrow[d] & s1r_1 \arrow[r] & 0 \cdot r_1 \arrow[dl, "*"'] \\
0 \cdot \ell_1 \arrow[r, "*"] & 0 &
\end{tikzcd}$$

If $(\ell_1 s \to r) \in R_1$ for some $s \in \Gamma \setminus \{1\}$, then $r = r_1 t$ for some $t \in \Gamma \setminus \{1\}$.

Hence:

$$\begin{tikzcd}
\ell_1 s 1 \arrow[r] \arrow[d] & \ell_1 0 \arrow[dr, "*"] & \\
 & & 0 \\
r_1 t 1 \arrow[r] & r_1 0 \arrow[ur, "*"'] &
\end{tikzcd}$$

(2.) If $(2\ell_1 \to r) \in R_1$, then $r = 2r_1$, and if $(\ell_1 s \to r) \in R_1$ for some $s \in \Gamma \setminus \{1, 2, \hat{c}, \hat{d}\}$, then $r = r_1 t$ for some $t \in \Gamma \setminus \{1, 2, \hat{c}, \hat{d}\}$.

The remaining cases can be treated analogously. $\square$

From Lemma 6.3 and the remark following Lemma 6.4 we see the following.

Corollary 6.6 *If the Turing machine M has an immortal finite configuration, then R does not preserve regularity. In fact, in this situation there exists a regular language $S \subseteq \Gamma^*$ such that not even the set $\mathrm{NF}_R(S)$ is regular.*

It remains to prove the converse of this corollary. Actually, we will prove the following weaker statement only: if the Turing machine M is strongly bounded, then $\mathrm{NF}_R(S)$ is a regular set for each regular language $S \subseteq \Gamma^*$.

First we will show that it suffices to look at rather restricted regular languages $S \subseteq \Gamma^*$. Let $S \subseteq \Gamma^*$ be a regular language. Then $S = S_1 \cup S_2$, where $S_1 := S \cap \mathrm{IRR}(R_2)$ and

$S_2 := S \cap \mathrm{RED}(R_2)$. Since R_2 is a finite string-rewriting system, we see that S_1 and S_2 are both regular sets. Now $\mathrm{NF}_R(S) = \mathrm{NF}_{R_1}(S_1) \cup \mathrm{NF}_R(S_2)$, and $\mathrm{NF}_R(S_2) = \begin{cases} \emptyset & \text{if } S_2 = \emptyset, \\ \{0\} & \text{if } S_2 \neq \emptyset. \end{cases}$ Thus, $\mathrm{NF}_R(S)$ is regular if and only if $\mathrm{NF}_{R_1}(S_1)$ is regular. Hence, in the following we can restrict our attention to regular sets S that are contained in $\mathrm{IRR}(R_2)$. Thus, by Lemma 6.4(b) we only have to deal with regular sets of factors of the language CONF.

Let $S \subseteq \mathrm{IRR}(R_2)$ be a regular language. Again we partition S into two subsets $S_1 := S \cap \Gamma^* \cdot d_0^+$ and $S_2 := S \cap (\Gamma \smallsetminus \{d_0\})^*$. Then $S = S_1 \cup S_2$, and $\mathrm{NF}_R(S) = \mathrm{NF}_{R_1}(S_1) \cup \mathrm{NF}_{R_1}(S_2)$. With S also S_1 and S_2 are regular languages.

Lemma 6.7 *If $S_2 \subseteq \mathrm{IRR}(R_2) \cap (\Gamma \smallsetminus \{d_0\})^*$ is a regular language, then so is the language $\mathrm{NF}_{R_1}(S_2)$.*

Proof. Since a word $w \in S_2$ does not contain any occurrences of the symbol d_0, a reduction sequence $w = w_0 \to_{R_1} w_1 \to_{R_1} \ldots \to_{R_1} w_n \in \mathrm{IRR}(R)$ can contain at most a single application of a rule from group (1.) of R_1.

A sequence $w = w_0 \to_{R_1} w_1 \to_{R_1} \ldots \to_{R_1} w_m$ of reduction steps is called **connected** if, for $i = 1, \ldots, m-1$, $w_{i-1} = u_{i-1}\ell_{i-1}v_{i-1}$, $w_i = u_{i-1}r_{i-1}v_{i-1} = u_i\ell_i v_i$, and $w_{i+1} = u_i r_i v_i$, where $(\ell_{i-1} \to r_{i-1}), (\ell_i \to r_i) \in R_1$, imply that r_{i-1} and ℓ_i have a nonempty overlap in w_i, that is, either $|u_i| < |u_{i-1}r_{i-1}| \leq |u_i\ell_i|$ or $|u_{i-1}| < |u_i\ell_i| < |u_{i-1}r_{i-1}|$. A connected sequence of reduction steps shifts $\hat{c}$ and $\bar{c}$ symbols to the right or it shifts $\hat{d}$, $\bar{d}$ and d symbols to the left. A sequence of this form can be followed by a single reduction step of a different form, e.g., $q_i a_k d a_r d a_{r_1} d \ldots d a_{r_s} d d \cent \to_{R_1}^{s+1} q_i a_k d d a_r a_{r_1} \ldots a_{r_s} \cent \to_{R_1} \bar{q}_j a_\ell a_r a_{r_1} \ldots a_{r_s} \cent$, if $\delta(q_i, a_k) = (q_j, a_\ell)$.

However, since the words considered do not contain any occurrences of the symbol d_0, those occurrences of the symbols $\hat{d}$, $\bar{d}$, and d that are used up during such a reduction sequence cannot be restored. Thus, a generalized sequential machine (gsm) G can be constructed that works as follows: while processing an input string $w \in \mathrm{IRR}(R_2)$ from left to right, G guesses an output string y and checks whether $w \to_{R_1}^* y$ holds. This can be done because of the observation above. A computation of G is accepting if and only if $w \in S_2$, $y \in \mathrm{IRR}(R_1)$, and $w \to_{R_1}^* y$. Hence, $G(S_2) = \mathrm{NF}_{R_1}(S_2)$, and so with S_2 also $\mathrm{NF}_{R_1}(S_2)$ is a regular language [HU79]. □

Let $S_1 = S \cap \Gamma^* \cdot d_0^+$ be a regular language, and let ν denote the constant from the pumping lemma for regular languages that is associated with S_1. We partition S_1 even further as $S_1 = S_3 \cup S_4$, where $S_3 := S_1 \cap (\Gamma \smallsetminus \{d_0\})^* \cdot \{d_0, d_0^2, \ldots, d_0^{\nu-1}\}$ and $S_4 := S_1 \cap \Gamma^* \cdot d_0^\nu$.

Lemma 6.8 *If S_1 is a regular language, then so is the language $\mathrm{NF}_{R_1}(S_3)$.*

Proof. Obviously, with S_1 also S_3 is a regular language. Now $S_3 = \bigcup_{i=1}^{\nu-1} S_{3,i}$, where $S_{3,i} := S_1 \cap (\Gamma \smallsetminus \{d_0\})^* \cdot d_0^i$, that is, $S_3 = \bigcup_{i=1}^{\nu-1} S'_{3,i} \cdot d_0^i$ for some regular sets $S'_{3,i} \subseteq (\Gamma \smallsetminus \{d_0\})^*$, $i = 1, \ldots, \nu-1$. Hence, $\mathrm{NF}_{R_1}(S_3) = \bigcup_{i=1}^{\nu-1} \mathrm{NF}_{R_1}(S'_{3,i} \cdot d_0^i)$.

Let $i \in \{1, 2, \ldots, \nu-1\}$. Then $\mathrm{NF}_{R_1}(S'_{3,i} \cdot d_0^i) = \mathrm{NF}_{R_1}(\mathrm{NF}_{R_1}(S'_{3,i}) \cdot d_0^i)$. By Lemma 6.7 $\mathrm{NF}_{R_1}(S'_{3,i})$ is a regular language. Furthermore, by using the gsm G constructed in the proof of Lemma 6.7 it is easily seen that $\mathrm{NF}_{R_1}(S' \cdot d_0^2)$ is a regular language whenever S' is a regular language satisfying $S' \subseteq \mathrm{IRR}(R)$. It follows inductively that $\mathrm{NF}_{R_1}(S'_{3,i} \cdot d_0^i)$ is a regular language for all $i = 1, \ldots, \nu - 1$, and hence, $\mathrm{NF}_{R_1}(S_3) = \bigcup_{i=1}^{\nu-1} \mathrm{NF}_{R_1}(S'_{3,i} \cdot d_0^i)$ is a regular language. □

It remains to deal with the regular language $S_4 = S_1 \cap \Gamma^* \cdot d_0^\nu$. Since ν is the constant that the pumping lemma yields for the language S_1, we see that, for each word $w \in (\Gamma \smallsetminus \{d_0\})^*$ and each $m \geq \nu$, if $wd_0^m \in S_4$, then $wd_0^{m+\alpha\cdot r} \in S_4$ for some $\alpha \in \{1, \ldots, \nu\}$ and all $r \geq 0$.

Let $S_5 := \{w \in (\Gamma \smallsetminus \{d_0\})^* \mid \exists m \geq \nu : wd_0^m \in S_4\}$. Then S_5 is a regular language. For $m \in \{0, 1, \ldots, \nu - 1\}$ and $\alpha \in \{1, \ldots, \nu\}$, define the languages $S_{5,m,\alpha}$ and $S_{4,m,\alpha}$ as follows:

$$S_{5,m,\alpha} := \{w \in S_5 \mid w \cdot d_0^m \cdot (d_0^\alpha)^* \subseteq S_4\}, \text{ and } S_{4,m,\alpha} := S_{5,m,\alpha} \cdot d_0^m \cdot (d_0^\alpha)^*.$$

From the considerations above we see that $S_4 = \bigcup_{m=0}^{\nu-1} \bigcup_{\alpha=1}^{\nu} S_{4,m,\alpha}$. Obviously, each of the languages $S_{5,m,\alpha}$ and $S_{4,m,\alpha}$ is regular.

Since $\mathrm{NF}_{R_1}(S_4) = \bigcup_{m=0}^{\nu-1} \bigcup_{\alpha=1}^{\nu} \mathrm{NF}_{R_1}(S_{4,m,\alpha})$, it suffices to look at one of the languages $S_{4,m,\alpha}$. So let $m \in \{0, 1, \ldots, \nu - 1\}$ and $\alpha \in \{1, \ldots, \nu\}$.

Let $\Delta := Q \cup \Sigma \cup \overline{\Sigma} \cup \{\$, ¢\}$, and let $\varphi := \Gamma \to \Delta$ denote the morphism that is defined through

$$\begin{array}{lcll} a & \mapsto & a & (a \in \Delta) \\ \bar{q} & \mapsto & q & (\bar{q} \in \overline{Q}) \\ a & \mapsto & \lambda & (a \in \{1, 2, d, \bar{d}, \hat{d}, d_0, \hat{c}, \bar{c}, 0\}). \end{array}$$

We say that a string $w \in S_{5,m,\alpha}$ contains a configuration uqv of the Turing machine M, if $\varphi(w) = uqv$.

Assume that the Turing machine M is strongly bounded. Then there exits an integer k such that, when starting in an arbitrary configuration uqv, M halts after at most k steps. Thus, only the suffix u_2 of u of length k and the prefix v_1 of v of length k are affected by the resulting computation. Hence, all possible computations of M can be described through the finite table

$$T = \{(uqv, u'q'v', i) \mid q, q' \in Q, u, v, u', v' \in \Sigma^*, |u|, |v|, |u'|, |v'| \leq k, \\ \text{and } i \leq k \text{ such that } uqv \vdash_M^i u'q'v'\}.$$

Now let $w \in S_{5,m,\alpha}$. The case that w does not contain a configuration of M is easily dealt with. So let us assume that w contains a configuration $u_1u_2qv_1v_2$ of M, where $|u_2| = |v_1| = k$. If w does not begin with the symbol 1, or if $|w|_2 = 0$, then R_1 can simulate at most $\min\{|w|_{\hat{c}} + |w|_{\bar{c}}, k\}$ steps of M on wd_0^s, and in this case the numbers $|w|_1$ and $|w|_2$ are not changed at all (see the rules of the groups (3.) to (5.) of R_1). If, however, $|w|_1 = j_1$ and $|w|_2 = j_2$ for some $j_1, j_2 \geq 1$, then at most i additional occurrences of the symbols 1 and 2 can be created while reducing wd_0^s modulo R_1, where $i =$ number of simulated steps of $M - (|w|_{\hat{c}} + |w|_{\bar{c}})$, provided s is sufficiently large.

Hence, a gsm G can be constructed that works as follows: while processing an input string $x \in \mathrm{IRR}(R_2)$ from left to right, G guesses an output string y and checks the following three conditions:

1. is $x = wd_0^s$ for some $w \in S_{5,m,\alpha}$ and some integer $s = m + \alpha \cdot r$?
2. is $y = ud_0^t$ irreducible mod R_1, where $u \in (\Gamma \smallsetminus \{d_0\})^*$ and $t \geq 0$?
3. does $wd_0^\ell \to_{R_1}^* u$ hold for some $\ell \in \mathbb{N}$ satisfying the congruence $\ell + t \equiv m \bmod \alpha$, that is, $\ell + t = m + \alpha \cdot r$ for some $r \in \mathbb{N}$?

For the latter part of this test G uses the table T mentioned above for comparing the structure of the strings w and u. Obviously, G cannot compute the number ℓ of d_0-symbols that are used up in the reduction $wd_0^\ell \rightarrow_{R_1}^* u$, but it can determine whether the number $\ell + t$ satisfies the congruence $\ell + t = m + \alpha \cdot r$ for some $r \in \mathbb{N}$ by counting modulo α. For example, if G has already determined that ℓ_0 symbols d_0 are necessary to create the prefix v of u, and if $u = vau_1$ for some $a \in \Sigma$, then $2\ell_0$ symbols d_0 are needed to create the prefix va. However, if $\ell_0 \leq \alpha < 2\ell_0$, then $2\ell_0$ can be written as $\alpha + (2\ell_0 - \alpha)$, where $\ell_1 := 2\ell_0 - \alpha \leq \alpha$, and it suffices for G to remember the number ℓ_1. Thus, for all $w \in S_{5,m,\alpha}$ and $r \in \mathbb{N}$, $G(wd_0^{m+\alpha\cdot r}) = \Delta_{R_1}^*(w \cdot d_0^m \cdot (d_0^\alpha)^*) \cap \mathrm{IRR}(R_1)$. Hence, it can be shown that $G(S_{4,m,\alpha}) = \Delta_{R_1}^*(S_{4,m,\alpha}) \cap \mathrm{IRR}(R_1) = \mathrm{NF}_{R_1}(S_{4,m,\alpha})$, and thus, $\mathrm{NF}_{R_1}(S_{4,m,\alpha})$ is regular, too. Therewith we have proved the following lemma.

Lemma 6.9 *If the Turing machine M is strongly bounded, then $\mathrm{NF}_R(S)$ is a regular language for each regular language $S \subseteq \Gamma^*$.*

Given a single-tape Turing machine M that simulates a Minsky machine $\hat{M}$ as described in [Hoo66], we see from the discussion in Section 5 that M has an immortal finite configuration if and only if M is not strongly bounded if and only if $\hat{M}$ does not halt from its initial configuration. By Corollary 6.6 and Lemma 6.9 we see that, for each regular language $S \subseteq \Gamma^*$, $\mathrm{NF}_R(S)$ is regular if and only if M is strongly bounded. Thus, we have the following undecidability result.

Theorem 6.10 *The following problem is undecidable in general:*

INSTANCE: *A finite, length-reducing, and confluent string-rewriting system R on Γ.*

QUESTION: *Is $\mathrm{NF}_R(S)$ a regular language for each regular language $S \subseteq \Gamma^*$?*

This generalizes Theorem 9 of [GT95] to signatures containing unary function symbols only and possibly a single constant.

7 Conclusion

We have seen that for string-rewriting systems the property of preserving regularity is independent of the alphabet actually considered, that is, by adding some free symbols to an alphabet considered, the property of a string-rewriting system to preserve regularity is not affected. Further, we have seen that even for finite string-rewriting systems in general, the property of preserving regularity is undecidable in general. For finite, length-reducing, and confluent string-rewriting systems, it is undecidable in general whether the set of descendants or the set of normalforms of a given regular language is again regular. Also we have seen that for a string-rewriting system of this form it is undecidable whether or not each regular language has a regular set of normal forms.

However, it remains open whether the problem of deciding whether a string-rewriting system preserves regularity is undecidable for the class of finite string-rewriting systems that are length-reducing and confluent. In fact, we do not even know whether this is true for the class of all finite convergent string-rewriting systems, but we would certainly expect that. Also it remains the question whether finite, length-reducing, and confluent systems presenting groups preserve regularity.

However, in the latter case we do at least know the following. If R is a finite, length-reducing, and confluent string-rewriting system on Σ such that the monoid M_R presented by

$(\Sigma; R)$ is actually a group, then there exists a deterministic pushdown automaton P that, given a string $w \in \Sigma^*$ as input, computes the irreducible descendant w_0 of w mod R [MO87]. In fact, P can be realized in such a way that after processing the input w completely, it halts with w_0 in its pushdown store. For $L \subseteq \Sigma^*$, let $\mathrm{SC}_P(L)$ denote the language of final stack contents that P can generate given a string from L as input. Then $\mathrm{SC}_P(L) = \mathrm{NF}_R(L)$. If L is a regular language, then by a result of Greibach [Gre67] also $\mathrm{SC}_P(L)$ is a regular language. Thus, in this situation the set of normal forms of a regular language is itself always regular.

Acknowledgement. The author wants to thank Klaus Madlener and Paliath Narendran for very fruitful discussions regarding the result presented in Section 6 and for pointing out the papers [Hoo66] and [KTU93].

References

[Ani71] A.V. Anissimov. Group languages. *Cybernetics*, 7:594–601, 1971.

[BJW82] R.V. Book, M. Jantzen, and C. Wrathall. Monadic Thue systems. *Theoretical Computer Science*, 19:231–251, 1982.

[BO93] R.V. Book and F. Otto. *String-Rewriting Systems*. Springer-Verlag, New York, 1993.

[Bra69] W.J. Brainerd. Tree generating regular systems. *Information and Control*, 14:217–231, 1969.

[CDGV94] J.-L. Coquidé, M. Dauchet, R. Gilleron, and S. Vágvölgyi. Bottom-up tree pushdown automata: classification and connection with rewrite systems. *Theoretical Computer Science*, 127:69–98, 1994.

[Dau89] M. Dauchet. Simulation of Turing machines by a left-linear rewrite rule. In N. Dershowitz, editor, *Proc. of RTA'89*, Lecture Notes in Computer Science 355, pages 109–120. Springer-Verlag, Berlin, 1989.

[Gil91] R. Gilleron. Decision problems for term rewriting systems and recognizable tree languages. In C. Choffrut and M. Jantzen, editors, *Proc. of STACS'91*, Lecture Notes in Computer Science 480, pages 148–159. Springer-Verlag, 1991.

[Gre67] S. Greibach. A note on pushdown store automata and regular systems. *Proc. American Mathematical Society*, 18:263–268, 1967.

[GS84] F. Gécseg and M. Steinby. *Tree Automata*. Akadémiai Kiadó, Budapest, 1984.

[GT95] R. Gilleron and S. Tison. Regular tree languages and rewrite systems. *Fundamenta Informaticae*, 24:157–175, 1995.

[GV95] P. Gyenizse and S. Vágvölgyi. Linear generalized semi-monadic rewrite systems preserve recognizability, 1995. Technical Report.

[Hoo66] P.K. Hooper. The undecidability of the Turing machine immortality problem. *J. Symbolic Logic*, 31:219–234, 1966.

[HU79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, M.A., 1979.

[Kre88] T. Kretschmer. A closure property of regular languages. *Theoretical Computer Science*, 61:283–287, 1988.

[KTU93] A.J. Kfoury, J. Tiuryn, and P. Urzyczyn. The undecidability of the semi-unification problem. *Information and Computation*, 102:83–101, 1993.

[LS77] R.C. Lyndon and P.E. Schupp. *Combinatorial Group Theory*. Springer-Verlag, Berlin, 1977.

[MO87] K. Madlener and F. Otto. Groups presented by certain classes of finite length-reducing string-rewriting systems. In P. Lescanne, editor, *Rewriting Techniques and Applications*, Lecture Notes in Computer Science 256, pages 133–144. Springer-Verlag, Berlin, 1987.

[Ó'D81] C. Ó'Dúnlaing. *Finite and Infinite Regular Thue Systems*. PhD thesis, University of California, Santa Barbara, 1981.

[Ott84] F. Otto. Some undecidability results for non-monadic Church-Rosser Thue systems. *Theoretical Computer Science*, 33:261–278, 1984.

[Ott91] F. Otto. When is an extension of a specification consistent? Decidable and undecidable cases. *Journal of Symbolic Computation*, 12:255–273, 1991.

[Sak79] J. Sakarovitch. Syntaxe des langages de Chomsky, essai sur le déterminisme, 1979. Thèse de doctorat d'état de l'université Paris VII.

[Sal88] K. Salomaa. Deterministic tree pushdown automata and monadic tree rewriting systems. *J. Computer System Sciences*, 37:367–394, 1988.

[Sén90] G. Sénizergues. Some decision problems about controlled rewriting systems. *Theoretical Computer Science*, 71:281–346, 1990.

[Wir90] M. Wirsing. Algebraic specification. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Vol. B.: Formal Models and Semantics*, pages 675–788. Elsevier, Amsterdam, 1990.