On the Connections Between Rewriting and Formal Language Theory

Friedrich Otto

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

Abstract. Formal language theory, and in particular the theory of automata, has provided many tools that have been found extremely useful in rewriting theory, since automata can be used for deciding certain properties of rewriting systems as well as for constructing (weakly) confluent rewriting systems. On the other hand, rewriting theory has had some influence on the development of formal language theory, since based on certain rewriting systems some interesting classes of formal languages have been defined. Here a survey on some connections between rewriting and formal language theory is given, starting from the classical string languages and string-rewriting systems and continuing with tree automata and term-rewriting systems.

1 Introduction

String-rewriting systems (or *semi-Thue systems*) are intimately connected with formal language theory, since under the name of *sets of productions* they form an essential part of Chomsky's phrase-structure grammars (see, for example, [17]). In particular, the various classes of the Chomsky hierarchy are defined by placing certain restrictions on the form of the productions (that is, the rewrite rules) that are admitted in a grammar. Hence, it is not surprising that techniques and results that have been developed in formal language theory are often very helpful in investigating certain properties of string-rewriting systems.

In fact, many sets associated with string-rewriting systems are context-free or even regular languages, and in fact corresponding descriptions, for example accepting automata, can often be constructed effectively from the string-rewriting system under consideration. This is the case for the sets of reducible and irreducible strings with respect to a finite (or left-regular) system, and the same is true for certain sets of descendants and unions of congruence classes with respect to some restricted systems. Based on these language-theoretical properties, some decision problems can be solved effectively, in some cases even efficiently. An example is Book's reduction algorithm for deciding the word problem for finite convergent string-rewriting systems that is based on a realization of a left-most reduction through a two-pushdown automaton [1]. Another example is Book's decision algorithm for linear sentences that express properties of Thue congruences generated by finite monadic and confluent string-rewriting systems [2,3].

P. Narendran and M. Rusinowitch (Eds.): RTA'99, LNCS 1631, pp. 332–355, 1999.

The test for confluence of a finite noetherian system reduces to checking emptiness of finitely many intersections of finite sets [5]. However, if we want to verify that a finite noetherian system is confluent on a certain congruence class only, then this test is much more complicated. In fact, this task turns out to be undecidable in general, and even for finite monadic systems it reduces to checking equality for finitely many pairs of one-turn languages [51]. String-rewriting systems will be addressed in Section 2.

Prefix-rewriting systems can be used to describe left-congruences in monoids. Hence, in the case of groups they yield descriptions of subgroups [41]. Under certain restrictions a prefix-rewriting system can be completed using a Knuth-Bendix-style completion procedure [40], thus giving a rewrite-based algorithm for deciding membership in the subgroup considered. However, there is a simpler method for completing prefix-rewriting systems that is based on finite-state acceptors, and that is applicable to certain classes of finite convergent presentations of groups [43]. Prefix-rewriting systems will be discussed in Section 3.

The finite, length-reducing, and confluent string-rewriting systems have been used to define the class *CRL* of Church-Rosser languages in [45]. From the definition it follows immediately that the membership problem for each Church-Rosser language is decidable in linear time. Hence, *CRL* is contained in the class *CSL* of context-sensitive languages, and it is shown in the original paper that *CRL* contains the class *DCFL* of deterministic context-free languages. However, only very recently the exact relationship between the class *CRL* and the class *CFL* of context-free languages could be settled [7,8,48]. The Church-Rosser languages will be the contents of Section 4.

The concepts of formal language and automata theory have been generalized to first-order terms and term languages (see, for example, [22,23]), while on the other hand term-rewriting systems can be seen as a corresponding generalization of string-rewriting systems. Accordingly, automata-theoretical notions and techniques have been applied successfully to describe certain sets of terms that are associated with term-rewriting systems, and to solve certain decision problems.

Here, however, some technical complications arise that have no counterparts in the string case. A string-rewriting system S on an alphabet $\Sigma = \{a_1, \ldots, a_m\}$ can be interpreted as a term-rewriting system $R_S = \{\ell(x) \to r(x) \mid (\ell \to r) \in S\}$ on the signature $F_\Sigma = \{a_1, \ldots, a_m, \phi\}$, where each letter a_i is considered as a function symbol of arity one, and ϕ is a constant. Thus, *linear* term-rewriting systems form a generalization of string-rewriting systems in that function symbols of arity larger than one are admitted. However, general term-rewriting systems embody a further generalization in that they may contain *non-linear* terms. Hence, the problem of linearity versus non-linearity plays an important role in the study of term-rewriting systems.

On the other hand, *ground term-rewriting systems* have particularly nice properties due to the serious restriction to the applicability of their rules. These systems will be discussed in Section 5, while Section 6 is devoted to the various generalizations of techniques and results of automata theory to term languages and term-rewriting systems. Here we will in particular address the question of

presenting the set of irreducible (ground) terms of a finite term-rewriting system through a finite tree-automaton, and the property of preserving regularity.

Due to space limitations only some fundamental definitions will be given in the paper. For further information regarding the notions introduced and for proofs of the results presented, the interested reader is asked to consult the literature, where [5] serves as our main reference on string-rewriting systems, [18] is our main reference on term-rewriting systems, and [23] is our main reference on tree automata.

Being a contribution to the proceedings of the 10th International Conference on Rewriting Techniques and Applications (RTA'99) this survey paper cannot possibly cover all the various aspects of the many connections between rewriting and formal language theory. Therefore, this article only presents some of the more fundamental connections that I have chosen based on my personal taste and experience. Others may feel that some important connections have been neglected. I apologize to all of them.

2 String-Rewriting Systems

Let Σ be a finite alphabet. Then Σ^* denotes the set of strings over Σ including the empty string λ. As usual the concatenation of two strings u and v will be denoted as uv, and numerical exponents will be used to abbreviate strings.

A *string-rewriting system* S on Σ is a subset of $\Sigma^* \times \Sigma^*$, the elements of which are called (*rewrite*) *rules*. By $\mathrm{dom}(S)$ we denote the set of all left-hand sides of rules of S, and by $\mathrm{range}(S)$ we denote the set of all right-hand sides. The *reduction relation* $\to_S^*$ defined by S is the reflexive and transitive closure of the *single-step reduction relation* $\to_S := \{(u\ell v, urv) \mid u, v \in \Sigma^*, (\ell, r) \in S\}$. A string $w \in \Sigma^*$ is called *reducible* if $w \to_S z$ holds for some string $z \in \Sigma^*$, otherwise w is called *irreducible*. By $\mathrm{RED}(S)$ ($\mathrm{IRR}(S)$) we denote the set of all strings that are reducible (irreducible) modulo S. Obviously, $\mathrm{RED}(S) = \Sigma^* \cdot \mathrm{dom}(S) \cdot \Sigma^*$ and $\mathrm{IRR}(S) = \Sigma^* \setminus \mathrm{RED}(S)$. Thus, if S is a finite system, then $\mathrm{RED}(S)$ and $\mathrm{IRR}(S)$ are regular languages. Actually, we have the following result.

Proposition 1. [27]
Given a finite string-rewriting system S, deterministic finite-state acceptors for the sets $\mathrm{RED}(S)$ and $\mathrm{IRR}(S)$ can be constructed in polynomial time.

For $w \in \Sigma^*$, $\Delta_S^*(w) := \{z \in \Sigma^* \mid w \to_S^* z\}$ is the set of *descendants* of w, $\nabla_S^*(w) := \{z \in \Sigma^* \mid z \to_S^* w\}$ is the set of *ancestors* of w, and $[w]_S := \{z \in \Sigma^* \mid w \leftrightarrow_S^* z\}$ is the *congruence class* of w. Here $\leftrightarrow_S^*$ denotes the *Thue congruence* generated by S, which is simply the reflexive, symmetric, and transitive closure of the relation $\to_S$. For a language $L \subseteq \Sigma^*$, the sets $\Delta_S^*(L)$, $\nabla_S^*(L)$, and $[L]_S$ are defined accordingly.

For a finite system S the sets of the form $\Delta_S^*(w)$, $\nabla_S^*(w)$, and $[w]_S$ are clearly recursively enumerable, but in general they are not even recursive. For certain restricted classes of string-rewriting systems however, we obtain much stronger results.

A string-rewriting system S is called

- *length-reducing* if $|\ell| > |r|$ holds for each rule (ℓ, r) of S, where $|w|$ denotes the *length* of the string w,
- *monadic* if it is length-reducing, and $\text{range}(S) \subseteq \Sigma \cup \{\lambda\}$,
- *special* if it is length-reducing, and $\text{range}(S) = \{\lambda\}$.

Obviously, a length-reducing system is *noetherian*. In fact, a system of this form has a linear upper bound on the length of reduction sequences. Although there are much more general classes of noetherian string-rewriting systems, we will not consider them in this paper.

If S is a length-reducing system, then $\Delta_S^*(w)$ is a finite set for each string w. However, already for finite confluent systems of this form we obtain very general languages once we consider sets of the form $\Delta_S^*(L)$, where $L \subseteq \Sigma^*$ is a regular language.

Proposition 2. [50]
Let $E \subseteq \Sigma^*$ be a recursively enumerable language. Then there exist a finite, length-reducing, and confluent string-rewriting system S on some alphabet Γ properly containing Σ and two regular languages $L_1, L_2 \subseteq \Gamma^*$ such that

$$\pi_\Sigma(\Delta_S^*(L_1) \cap L_2) = E = \pi_\Sigma([L_1]_S \cap L_2),$$

where π_Σ denotes the projection from Γ^* onto Σ^*.

On the other hand we have the following positive result.

Proposition 3. [4,5,37,63]
Let S be a monadic string-rewriting system on Σ, and let $L \subseteq \Sigma^*$ be a regular language. Then the set $\Delta_S^*(L)$ is again a regular language. If S is finite, then a finite-state acceptor for this language can be constructed in polynomial time from a finite-state acceptor for L.

The acceptor for $\Delta_S^*(L)$ is simply obtained from the one for L by adding transitions. Accordingly the polynomial time-bound carries over even to certain classes of infinite monadic systems.

If S is a finite monadic system, then $S^{-1} := \{(r, \ell) \mid (\ell, r) \in S\}$ can be interpreted as the set of productions of a context-free grammar. Hence, it is easily seen that the set $\nabla_S^*(L)$ is a context-free language for each finite monadic string-rewriting system S and each context-free language L. If S is confluent, then $[w]_S = \nabla_S^*(w)$ for each irreducible string w. Hence, we obtain the following result.

Proposition 4. [4]
Let S be a finite string-rewriting system on Σ that is monadic and confluent.

(a) For each context-free set $L \subseteq \text{IRR}(S)$ of irreducible strings, $[L]_S$ is a context-free language.
(b) For each regular set $L \subseteq \Sigma^*$, $[L]_S$ is a deterministic context-free language.

Underlying part (b) of Proposition 4 is the following general result.

Proposition 5. [1,5]
Let S be a finite and noetherian string-rewriting system on Σ. Then there exists a deterministic automaton with two-pushdown stores that, given a string $w \in \Sigma^*$ as input, computes the irreducible descendant of w modulo S with respect to left-most reductions. If S is monadic, then this computation can be performed by a standard deterministic pushdown automaton.

Based on the positive results for monadic systems above Book has developed a decision procedure for a restricted class of sentences of first-order predicate calculus without equality, where the set of nonlogical symbols consists of a binary predicate symbol $\equiv$, a binary function symbol $\cdot$, a constant symbol a for each letter a from a fixed finite alphabet Σ, and a constant symbol 1.

Let S be a string-rewriting system on Σ. By interpreting the function symbol $\cdot$ as the multiplication in the monoid $M_S := \Sigma^*/\leftrightarrow_S^*$, by interpreting each constant a as the monoid element $[a]_S$ and the constant 1 as the identity $[\lambda]_S$ of the monoid M_S, and by interpreting the predicate symbol $\equiv$ as the congruence $\leftrightarrow_S^*$, we obtain an interpretation for these sentences expressing some properties of M_S.

Let Σ be a finite alphabet, and let V_E and V_U be two disjoint countable sets of symbols such that $(V_E \cup V_U) \cap \Sigma = \emptyset$. The symbols of V_E are *existential variables*, while those of V_U are *universal variables*. A string in $(\Sigma \cup V_U)^*$ is a *universal term*, and a string in $(\Sigma \cup V_E)^*$ is an *existential term*.

If x and y are two existential terms, then $x \equiv y$ is an *existential atomic formula*. If x and y are two universal terms, then $x \equiv y$ is a *universal atomic formula*. Finally, if one of x and y is an existential term and the other is a universal term, then $x \equiv y$ is a *mixed atomic formula*.

An atomic formula is a *formula*. If F_1 and F_2 are formulas, then $(F_1 \wedge F_2)$ and $(F_1 \vee F_2)$ are *formulas*. A formula is called *linear* if no variable occurs twice in it.

If F is a formula with existential variables $v_1, \ldots, v_q$ and universal variables $u_1, \ldots, u_p$, then

$$\forall u_1 \forall u_2 \ldots \forall u_p \exists v_1 \exists v_2 \ldots \exists v_q F \text{ and } \exists v_1 \exists v_2 \ldots \exists v_q \forall u_1 \forall u_2 \ldots \forall u_p F$$

are sentences. By LINSEN(Σ) we denote the set of all sentences over Σ that contain only linear formulas.

Let S be a string-rewriting system on Σ. If φ is a sentence over Σ containing the variables $v_1, \ldots, v_p \in (V_E \cup V_U)$, and if $L_1, \ldots, L_p$ are subsets of Σ^*, then we obtain the following interpretation of φ:

(i) for each i, $1 \leq i \leq p$, the variable v_i takes values in the set L_i;
(ii) the symbol $\equiv$ is interpreted as the congruence $\leftrightarrow_S^*$;
(iii) the symbol $\wedge$ is interpreted as conjunction and the symbol $\vee$ is interpreted as disjunction.

Under this interpretation the sentence φ is either true or false as a statement about the congruence $\leftrightarrow_S^*$ and the sets $L_1, \ldots, L_p \subseteq \Sigma^*$, and hence about the monoid M_S.

For example, a string w is *left-divisible* by z if and only if w is congruent to a string with prefix z, that is, if the linear sentence $\exists v : w \equiv z \cdot v$ is true under the interpretation induced by S and the set Σ^*.

If S is a finite, monadic, and confluent system, then with each term x of a linear sentence we can associate a regular set $L(x)$ of irreducible strings based on the structure of the term and the regular sets serving as domains for the variables occurring in x. In this way the question of whether or not the linear sentence is true under the given interpretation is reduced to a question about regular languages. This yields the following decidability result.

Proposition 6. [3]
Let S be a string-rewriting system on Σ that is finite, monadic, and confluent. Then the following validity problem for linear sentences is decidable in polynomial space:

INSTANCE : *A sentence $\varphi \in \mathrm{LINSEN}(\Sigma)$ containing variables $v_1, v_2, \ldots, v_m$, and regular sets $L_1, \ldots, L_m \subseteq \Sigma^*$ that are specified by finite-state acceptors.*
QUESTION : *Is φ true under the interpretation induced by S and $L_1, \ldots, L_m$?*

Actually, if the linear sentences φ considered do not contain mixed atomic formulas or if their quantifier prefixes are of the form $\exists^i \forall^j$, then the validity of these sentences is even decidable in polynomial time.

In fact, also some other decision problems, for which there does not seem to be a way of expressing them by linear sentences, can be solved for finite, monadic, and confluent string-rewriting systems in a similar way. An example for this is the property of *left-cancellativity*. Here the monoid M_S is called *left-cancellative* if, for all $u, v, w \in \Sigma^*$, $uv \leftrightarrow_S^* uw$ implies that $v \leftrightarrow_S^* w$ holds.

Proposition 7. [46]
Let S be a string-rewriting system that is length-reducing, interreduced, and confluent. Then the monoid M_S is not left-cancellative if and only if there exists a rule $(au, v) \in S$, where $a \in \Sigma$ and $u, v \in \mathrm{IRR}(S)$, such that $\Delta_S^*(L_1) \cap \Delta_S^*(L_2) \neq \emptyset$, where $L_1 = \{auw \mid w \in \Sigma^* \text{ such that } uw \in \mathrm{IRR}(S)\}$ and $L_2 = \{ax \mid x \in \mathrm{IRR}(S), u \text{ is not a prefix of } x\}$.

If S is finite, then L_1 and L_2 are regular languages, finite-state acceptors for which can be constructed in polynomial time. Thus, if additionally S is monadic, then the condition stated in the proposition above can be verified in polynomial time. By considering various other regular languages associated with monadic string-rewriting systems, it can be shown that the (*left-, right-*) *conjugacy problem* and the *common left-* (*right-*) *multiplier problem* are decidable in polynomial time for each finite, monadic, and confluent string-rewriting system [47].

For a finite noetherian string-rewriting system S on Σ the test for confluence of S reduces to checking whether the intersection $\Delta_S^*(u) \cap \Delta_S^*(v)$ is non-empty

for each of the finitely many critical pairs (u, v) of S. However, it is much more difficult in general to decide whether the system S is confluent on a certain congruence class $[w]_S$. Here S is called *confluent on* $[w]_S$ for some string $w \in \Sigma^*$ if, for all $u, v, x \in [w]_S$, $u \to_S^* v$ and $u \to_S^* x$ imply that $\Delta_S^*(v) \cap \Delta_S^*(x)$ is non-empty.

For $u \in \Sigma^*$ and $w \in \mathrm{IRR}(S)$, let $\mathrm{Con}_u(w) := \{x\#y \mid x, y \in \mathrm{IRR}(S) \text{ and } xuy \to_{\ell,S}^* w\}$ be the *set of contexts of* u *for* w modulo S, where $\to_{\ell,S}^*$ denotes the *left-most reduction* modulo S, and $\#$ is a new letter. Further, let $\mathrm{UCP}(S)$ denote the set of those critical pairs (y, z) of S for which the intersection $\Delta_S^*(y) \cap \Delta_S^*(z)$ is empty. Then we have the following characterization.

Proposition 8. [51]
Let S be a finite noetherian string-rewriting system on Σ, and let $w \in \mathrm{IRR}(S)$. Then S is confluent on $[w]_S$ if and only if $\mathrm{Con}_y(w) = \mathrm{Con}_z(w)$ holds for each pair $(y, z) \in \mathrm{UCP}(S)$.

Even for finite length-reducing systems confluence on a given congruence class is undecidable in general [51]. If, however, S is a finite monadic system, then each language of the form $\mathrm{Con}_u(w)$ is a deterministic one-turn language, and in fact, from S and the strings u and w, a deterministic one-turn pushdown automaton for $\mathrm{Con}_u(w)$ can be constructed effectively. Thus, we obtain the following decidability result due to the solvability of the equivalence problem for deterministic one-turn pushdown automata [70].

Corollary 1. [51]
For finite monadic string-rewriting systems confluence on a given congruence class is decidable in doubly exponential time.

However, for special systems this result can be improved considerably by analyzing the form of the generated reduction sequences in more detail.

Corollary 2. [53]
For finite special string-rewriting systems confluence on a given congruence class is decidable in polynomial time.

This result even extends to testing whether a finite monadic system S is *weakly confluent*, that is, whether S is confluent on $[a]_S$ for each $a \in \mathrm{range}(S)$ [44]. I would like to mention in passing that based on these confluence tests Knuth-Bendix like procedures for weak completion have been developed

1. for finite special systems [52], and
2. for finite monadic systems presenting groups [44].

Actually, for these two classes of string-rewriting systems further interesting results have been obtained that are based on language properties of certain associated sets.

For a string-rewriting system S on Σ and a language $L \subseteq \Sigma^*$, we denote by $I_S(L)$ the set $I_S(L) := [L]_S \cap \mathrm{IRR}(S)$ of irreducible strings that are congruent to some string from L.

Proposition 9. [60]
Let S be a finite special string-rewriting system that is confluent on $[\lambda]_S$, and let $L \subseteq \Sigma^*$ be a regular language. Then the set $I_S(L)$ is also regular, and a finite-state acceptor for $I_S(L)$ can be constructed in polynomial time from a finite-state acceptor for L.

Proposition 9 also holds for finite, monadic, and weakly confluent systems that present groups [44]. In particular, this implies that the result on linear sentences (Proposition 6) carries over to finite, special systems that are weakly confluent and to finite, monadic, and weakly confluent systems presenting groups.

3 Prefix-Rewriting Systems

In this section we take a look at prefix-rewriting systems and relate them to the subgroup problem of finitely presented groups.

A *prefix-rewriting system* on Σ is a subset of $\Sigma^* \times \Sigma^*$. Its elements are called *prefix-rules*. If P is a prefix-rewriting system, then $\mathrm{dom}(P)$ and $\mathrm{range}(P)$ are defined as for string-rewriting systems.

The *prefix-reduction relation* $\Rightarrow_P^*$ defined by P is the reflexive transitive closure of the *single-step prefix-reduction relation* $\Rightarrow_P := \{(\ell w, rw) \mid (\ell, r) \in P, w \in \Sigma^*\}$, and by $\Leftrightarrow_P^*$ we denote the reflexive, symmetric, and transitive closure of $\Rightarrow_P$. Obviously $\Leftrightarrow_P^*$ is a left-congruence on Σ^*. By $\mathrm{RED}(P)$ we denote the set of all reducible strings, and $\mathrm{IRR}(P)$ denotes the set of irreducible strings. Obviously, $\mathrm{RED}(P) = \mathrm{dom}(P) \cdot \Sigma^*$ and $\mathrm{IRR}(P) = \Sigma^* \setminus \mathrm{RED}(P)$. Hence, if $\mathrm{dom}(P)$ is a regular language, then $\mathrm{RED}(P)$ and $\mathrm{IRR}(P)$ are regular languages as well. In this situation the prefix-rewriting system P is called *left-regular*.

The prefix-rewriting system P is called *noetherian*, *confluent*, *convergent*, *λ-confluent*, *λ-convergent*, *interreduced*, or *canonical* if the corresponding condition is satisfied by $\Rightarrow_P$. It is interesting to observe that a prefix-rewriting system is convergent whenever it is interreduced, that is, it is canonical if and only if it is interreduced. This is an immediate consequence of the corresponding result for ground-term rewriting systems (Proposition 17), since a prefix-rewriting system P on Σ can be interpreted as a ground-term rewriting system on the signature F_Σ.

Next we will show how prefix-rewriting systems are related to the *subgroup problem*. Let G be a group that is given through a finite presentation $(\Sigma; S)$, and let $^{-1} : \Sigma^* \to \Sigma^*$ denote a function realizing the inverse function of G. Further, let $U \subseteq \Sigma^*$ be a finite set, where we assume without loss of generality that U is *closed under inverses*, that is, for each $u \in U$, there exists an element $v \in U$ such that $v \leftrightarrow_S^* u^{-1}$. Then a string $w \in \Sigma^*$ presents an element of the subgroup $\langle U \rangle$ of G that is generated by U if and only if there exist $u_1, \ldots, u_k \in U$ such that $w \leftrightarrow_S^* u_1 u_2 \cdots u_k$. The *subgroup problem* for G is the problem of deciding, given a finite set $U \subset \Sigma^*$ and a string $w \in \Sigma^*$, whether or not w belongs to the subgroup $\langle U \rangle$ of G.

With U we associate a binary relation $\sim_U$ on Σ^* as follows:

$$x \sim_U y \text{ iff } \exists u \in \langle U \rangle : x \leftrightarrow_S^* uy.$$

Then $w \in \langle U \rangle$ if and only if $w \sim_U \lambda$.

With $(\Sigma; S)$ and U we now associate a prefix-rewriting system $P := P_U \cup P_S$, where

$$P_U := \{(u, \lambda) \mid u \in U\}$$

and

$$P_S := \{(x\ell, xr) \mid x \in \Sigma^* \text{ and } (\ell \to r) \in S\}.$$

Then P is a left-regular system, and the following property is easily verified.

Proposition 10. [41] *The left-congruences $\sim_U$ and $\Leftrightarrow_P^*$ coincide.*

Hence, if P is λ-confluent, then a string $w \in \Sigma^*$ belongs to $\langle U \rangle$ if and only if $w \Rightarrow_P^* \lambda$, and if P is convergent, then $\mathrm{IRR}(P)$ is a complete set of coset representatives for $\langle U \rangle$ in G.

If S is noetherian, then P is noetherian, but in general P will not be convergent even in case S is. However, as for string-rewriting systems confluence of the prefix-rewriting system P can be characterized through the convergence of finitely many critical pairs. Based on this confluence test a Knuth-Bendix style completion procedure for prefix-rewriting systems has been developed in [41] that applies to groups G that are given through finite convergent presentations.

Also confluence on $[\lambda]_{\sim_U}$ can be characterized as for string-rewriting systems (Proposition 8) [42]. However, there is another criterion for deciding this property that exploits automata-theoretical arguments.

Let $(\Sigma; S)$ be a finite convergent presentation of a group, let P_U be a set of prefix-rules on Σ, and let $P := P_U \cup P_S$, where we assume that the set $U := \{uv^{-1} \mid (u, v) \in P_U\}$ is closed under taking inverses and that P is noetherian. Then $[\lambda]_P = \langle U \rangle$, and hence, $w \in [\lambda]_P$ if and only if $w \Leftrightarrow_P^* z$ for some $z \in U^*$. Now P is confluent on $[\lambda]_{\sim_U}$ if and only if each string $w \in \langle U \rangle \setminus \{\lambda\}$ is reducible by P, that is, if and only if $[\lambda]_P \cap \mathrm{IRR}(P) = \{\lambda\}$. However, since S is convergent, the latter equality is equivalent to the equality $(\Delta_S^*(U^*) \cap \mathrm{IRR}(S)) \cap \mathrm{IRR}(P) = \{\lambda\}$.

If S and P_U are both finite, then the sets $\mathrm{IRR}(S)$ and $\mathrm{IRR}(P)$ are both regular, and finite-state acceptors for them can be constructed effectively. Also U^* is a regular set in this situation. Hence, this criterion becomes decidable whenever the set $\Delta_S^*(U^*)$ (or the set $\Delta_S^*(U^*) \cap \mathrm{IRR}(S)$) allows an effective specification for which the intersection with the regular set $\mathrm{IRR}(P)$ can be determined effectively.

If $(\Sigma; S)$ is a finite, weight-reducing, and confluent presentation of a group and $U \subseteq \Sigma^*$ is a finite set, then it is still an open problem whether or not the set $\Delta_S^*(U^*)$ is necessarily regular. However, if we restrict the set $\Delta_S^*(U^*)$ to only those strings that are obtained by left-most reductions, then this subset $\Delta_{L,S}^*(U^*)$ of $\Delta_S^*(U^*)$ can be shown to always be regular [40]. In fact, a finite-state acceptor for this language can be constructed effectively. Since $\Delta_{L,S}^*(U^*) \cap \mathrm{IRR}(S) = \Delta_S^*(U^*) \cap \mathrm{IRR}(S)$, we obtain a finite-state acceptor for the set $\Delta_S^*(U^*) \cap \mathrm{IRR}(P)$. This gives the following decidability result.

Proposition 11. [40]
Let $(\Sigma; S)$ be a finite, weight-reducing, and confluent presentation of a group,

and let P_U be a finite set of prefix-rules on Σ such that the set $U := \{uv^{-1} \mid (u,v) \in P_U\}$ is closed under taking inverses, and $P := P_U \cup P_S$ is noetherian. Then it is decidable whether the prefix-rewriting system P is λ-confluent.

Now assume that $(\Sigma; S)$ is a finite, weight-reducing, and confluent presentation of a group G, let $U \subseteq \Sigma^+$ be a finite set that is closed under taking inverses, and let $P_U := \{(u, \lambda) \mid u \in U\}$. From $(\Sigma; S)$ and U we can construct a finite-state acceptor $A = (Q, \Sigma, q_0, F, \delta)$ for the language $\Delta_S^*(U^*) \cap \mathrm{IRR}(S)$. From A we extract a finite set of prefix-rules P'_U as follows, where we identify A with its state graph in order to simplify the notation:

(i) For every simple path in A leading from the initial state q_0 to a final state $q_f \in F$, which does not pass through any final state, we put the rule (x, λ) into P'_U, where x is the label along the path considered.

(ii) For every path p in A from q_0 to a final state $q_f \in F$, which does not pass through any final state, and which can be partitioned into three parts $p = p_1, p_2, p_3$ such that p_1 is a simple path, and p_2 is a simple loop, we put the rule (x_1x_2, x_1) into P'_U, where x_i is the label along the subpath p_i, $i = 1, 2$.

Obviously, P'_U is a finite set of prefix-rules that can effectively be obtained from A. For $w \in \langle U \rangle$ there exists a unique string $w_0 \in \mathrm{IRR}(S)$ such that $w \to_S^* w_0$. Since $w \in \langle U \rangle$, $w_0 \in \Delta_S^*(U^*) \cap \mathrm{IRR}(S)$, and hence, w_0 is accepted by A. From the construction of P'_U it follows that $w \Rightarrow_{P'}^* \lambda$ holds, where $P' := P'_U \cup P_S$. Since $u \sim_U v$ holds for each rule $(u, v) \in P'_U$, it follows that $\Leftrightarrow_{P'}^* = \sim_U$, and P' is confluent on $[\lambda]_{\sim_U}$.

Proposition 12. [40]
Let $(\Sigma; S)$ be a finite, weight-reducing, and confluent presentation of a group G, and let $U \subseteq \Sigma^+$ be a finite set. Then a finite set of length-reducing prefix-rules P'_U can be determined effectively such that the prefix-rewriting system $P := P'_U \cup P_S$ is confluent on $[\lambda]_P$ and $\Leftrightarrow_P^* = \sim_U$.

Actually, this construction carries over to the case of groups that are presented through finite and monadic string-rewriting systems that are only confluent on the congruence class of the empty string [43].

Finally, we want to address *automatic structures* for monoids, which is a fairly recent development. An automatic structure for a monoid-presentation $(\Sigma; S)$ can be interpreted as a finite description of the multiplication table of the monoid M_S. Originally automatic structures were developed for groups (see [19] for a detailed presentation), but recently automatic structures have also been considered for semigroups and monoids [9].

In order to define automatic structures we need the following definition as we will be dealing with infinite sets of pairs of strings that are to be recognized by finite-state acceptors.

Let Σ be a finite alphabet, and let $\# \notin \Sigma$ be an additional "padding" symbol. Then by $\Sigma_\#$ we denote the following finite alphabet:

$$\Sigma_\# := ((\Sigma \cup \{\#\}) \times (\Sigma \cup \{\#\})) \setminus \{(\#, \#)\}.$$

This alphabet is called the *padded extension* of Σ. An encoding $\nu : \Sigma^* \times \Sigma^* \to \Sigma_\#^*$ is now defined as follows:
if $u := a_1 a_2 \cdots a_n$ and $v := b_1 b_2 \cdots b_m$, where $a_1, \ldots, a_n, b_1, \ldots, b_m \in \Sigma$, then

$$\nu(u,v) := \begin{cases} (a_1,b_1)(a_2,b_2)\cdots(a_m,b_m)(a_{m+1},\#)\cdots(a_n,\#), & \text{if } m < n, \\ (a_1,b_1)(a_2,b_2)\cdots(a_m,b_m), & \text{if } m = n, \\ (a_1,b_1)(a_2,b_2)\cdots(a_n,b_n)(\#,b_{n+1})\cdots(\#,b_m) & \text{if } m > n. \end{cases}$$

A *prefix-rewriting system* P on Σ is called *synchronously regular*, *s-regular* for short, if $\nu(P)$ is accepted by some finite-state acceptor over $\Sigma_\#$. Obviously, if P is s-regular, then $\mathrm{dom}(P)$ and $\mathrm{range}(P)$, and therewith also $\mathrm{RED}(P)$ and $\mathrm{IRR}(P)$, are regular languages.

An *automatic structure* for a finitely generated monoid-presentation $(\Sigma; S)$ consists of finite-state acceptors W over Σ and $M_=$ and M_a $(a \in \Sigma)$ over $\Sigma_\#$ satisfying the following conditions:

(0.) $L(W) \subseteq \Sigma^*$ is a complete set of (not necessarily unique) representatives for the monoid M_S, that is, $L(W) \cap [u]_S \neq \emptyset$ holds for each $u \in \Sigma^*$,
(1.) $L(M_=) = \{\nu(u,v) \mid u, v \in L(W) \text{ and } u \leftrightarrow_S^* v\}$, and
(2.) for all $a \in \Sigma$, $L(M_a) = \{\nu(u,v) \mid u, v \in L(W) \text{ and } ua \leftrightarrow_S^* v\}$.

Actually, one may require that the set $L(W)$ is a cross-section for M_S, in which case we say that we have an *automatic structure with uniqueness* [19]. In this situation the finite-state acceptor $M_=$ is trivial, and hence, it will not be mentioned explicitly.

A monoid-presentation is called *automatic* if it has an automatic structure, and a monoid is called *automatic* if it has an automatic presentation. Automatic monoids have word problems that are decidable in quadratic time based on the automatic structure. For automatic groups many additional nice properties have been obtained, while for automatic monoids in general the situation is not quite as nice [9,54,59]. Here we are interested in automatic structures with uniqueness, for which the set of representatives considered is in addition prefix-closed. It is an open problem whether or not every automatic group does have an automatic structure with this additional property. But at least the following characterization can be obtained.

Proposition 13. [55]
Let $(\Sigma; S)$ be a finitely generated monoid-presentation. Then the following two statements are equivalent:

(a) There exists an automatic structure $(W, A_a(a \in \Sigma))$ with uniqueness for $(\Sigma; S)$ such that the set $L(W)$ is prefix-closed.
(b) There exists an s-regular canonical prefix-rewriting system P on Σ that is equivalent to S, that is, the left-congruence $\Leftrightarrow_P^*$ coincides with the Thue congruence $\leftrightarrow_S^*$.

There exists a group with a finite convergent presentation, which does not admit an automatic structure [24]. Hence, no finitely generated presentation of

this group has an s-regular canonical prefix-rewriting system that defines the corresponding Thue congruence.

The monoid N of [59] has an automatic structure that is based on a regular cross-section that is the set of irreducible strings modulo some infinite left-regular convergent string-rewriting system. Hence, this set is certainly prefix-closed and so Proposition 13 shows that this presentation of N admits an s-regular canonical prefix-rewriting system. However, N does not admit any finite convergent presentation. These observations yield the following result.

Corollary 3. *The class of finitely presented monoids that admit a finite convergent presentation and the class of finitely presented monoids that admit an s-regular canonical prefix-rewriting system are incomparable under set inclusion.*

4 Church-Rosser Languages

In the previous sections we have seen how techniques from automata theory have been used to establish properties for string-rewriting systems. Here we show that also rewriting theory has had some influence on formal language theory.

Definition 1. [45]

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

(b) A language $L \subseteq \Sigma^*$ is a Church-Rosser decidable language (*CRDL*) *if there exist an alphabet $\Gamma \supsetneq \Sigma$, a finite, length-reducing, confluent string-rewriting system R on Γ, two strings $t_1, t_2 \in (\Gamma \setminus \Sigma)^* \cap \mathrm{IRR}(R)$, and two distinct letters $Y, N \in (\Gamma \setminus \Sigma) \cap \mathrm{IRR}(R)$ such that, for all $w \in \Sigma^*$, the following statements hold:*

- *$t_1 w t_2 \to_R^* Y$ if and only if $w \in L$, and*
- *$t_1 w t_2 \to_R^* N$ if and only if $w \notin L$.*

By admitting weight-reducing instead of length-reducing string-rewriting systems in the definition, we obtain the class *GCRL* of *generalized Church-Rosser languages* [8]. Obviously, the membership problem for a *GCRL* is decidable in linear time, and so *GCRL* is contained in the class *CSL* of context-sensitive languages. Further, it is shown in [45] that each deterministic context-free language is a Church-Rosser decidable language, while there exist languages in *CRDL* that are not context-free. Hence, we have the following sequence of inclusions:

$$DCFL \subset CRDL \subseteq CRL \subseteq GCRL \subset CSL.$$

However, while it was conjectured in [45] that the class *CFL* of context-free languages is not contained in *CRL*, this remained open at the time.

Another subclass of *CSL* that received quite some attention in the literature is the class *GCSL* of growing context-sensitive languages. Here a language is

called *growing context-sensitive* if it is generated by a *growing context-sensitive grammar*, that is, a grammar $G = (N, \Sigma, S, P)$ satisfying the following conditions:

1. the start symbol S does not occur on the right-hand side of any production, and
2. for each production $(\ell, r) \in P$, $|\ell| < |r|$ or $\ell = S$.

In [12] Dahlhaus and Warmuth proved that the membership problem for a growing context-sensitive language can be solved in polynomial time. In [7] Buntrock and Otto introduced the following type of automaton in order to characterize the class *GCSL* of growing context-sensitive languages.

Definition 2.

(a) A two-pushdown automaton (*TPDA*) *is a nondeterministic automaton with two pushdown stores. Formally, it is a 7-tuple* $M = (Q, \Sigma, \Gamma, \delta, q_0, \bot, F)$*, where*
- Q *is the finite set of states,*
- Σ *is the finite input alphabet,*
- Γ *is the finite tape alphabet with* $\Gamma \supsetneq \Sigma$ *and* $\Gamma \cap Q = \emptyset$*,*
- $q_0 \in Q$ *is the initial state,*
- $\bot \in \Gamma \setminus \Sigma$ *is the bottom marker of the pushdown stores,*
- $F \subseteq Q$ *is the set of final (or accepting) states, and*
- $\delta : Q \times \Gamma \times \Gamma \to 2^{Q \times \Gamma^* \times \Gamma^*}$ *is the transition relation, where* $\delta(q, a, b)$ *is a finite set for each triple* $(q, a, b) \in Q \times \Gamma \times \Gamma$*.*

M is a deterministic two-pushdown automaton (*DTPDA*)*, if* δ *is a (partial) function from* $Q \times \Gamma \times \Gamma$ *into* $Q \times \Gamma^* \times \Gamma^*$*.*

(b) A (DTPDA) TPDA M is called shrinking *if there exists a weight function* $\varphi : Q \cup \Gamma \to \mathbb{N}_+$ *such that, for all* $q \in Q$ *and* $a, b \in \Gamma$*, if* $(p, u, v) \in \delta(q, a, b)$*, then* $\varphi(puv) < \varphi(qab)$*. By sTPDA and sDTPDA we denote the corresponding classes of shrinking automata.*

A *configuration* of a (*DTPDA*) *TPDA M* can be described as uqv with $q \in Q$ and $u, v \in \Gamma^*$, where u is the contents of the first pushdown store with the first letter of u at the bottom and the last letter of u at the top, q is the current state, and v is the contents of the second pushdown store with the last letter of v at the bottom and the first letter of v at the top. M induces a *computation relation* $\vdash_M^*$ on the set of configurations, which is the reflexive, transitive closure of the *single-step computation relation* $\vdash_M$ (see, e.g., [31]). For an input string $w \in \Sigma^*$, the corresponding *initial configuration* is $\bot q_0 w \bot$. M accepts by empty pushdown stores:

$$L(M) := \{w \in \Sigma^* \mid \exists q \in F : \bot q_0 w \bot \vdash_M^* q\}.$$

Buntrock and Otto established the following characterization for the classes of languages that are accepted by nondeterministic or deterministic shrinking *TPDA*s, respectively.

Proposition 14. [7,8]

(a) A language is accepted by some shrinking TPDA if and only if it is growing context-sensitive.
(b) A language is accepted by some shrinking DTPDA if and only if it is a generalized Church-Rosser language.

Thus, the generalized Church-Rosser languages can be viewed as the deterministic variants of the growing context-sensitive languages. Further, it is observed in [8] that the language $L = \{ww \mid w \in \{a,b\}^+\}$ does not belong to the class *GCSL*. Since the class *GCRL* is clearly closed under complement due to Proposition 14(b), it follows that the language $L^c = \{a,b\}^* \setminus L$ is a context-free language that is not generalized Church-Rosser. This finally settled the conjecture of [45] mentioned above.

Finally, Niemann and Otto showed that each *sDTPDA* can be simulated by some finite length-reducing and confluent string-rewriting system [48], thus establishing the following equalities.

Proposition 15. [48] *The classes CRDL, CRL, and GCRL coincide.*

Thus, *CRL* is incomparable with the class *CFL* under set inclusion, it is closed under complement and under left and right quotient with a single string [45]. However, it is not closed under union or intersection [48], and it is not closed under homomorphisms, since *CRL* is a basis for the recursively enumerable languages [57,58]. Since *CFL* is a full abstract family of languages [31], this indicates a certain duality between *CRL* and *CFL*. Based on a generalization of the so-called *restarting automata with rewriting* [35] this duality is further explored in [49].

5 Ground Term-Rewriting Systems

Finally we turn to rewriting systems over terms. For the following considerations let F denote a finite *signature*, that is, F is a finite set of *function symbols*, each of which is associated with a fixed arity. For each $n \geq 0$, F_n is the subset of F consisting of the function symbols of arity n. The elements of F_0 are called *constants*. To avoid degenerate cases we will always assume that the set of constants is non-empty.

The set of *terms* $T(F)$ is defined inductively as follows:

(1.) Each constant is a term.
(2.) If $f \in F_n$ for some $n > 0$ and $t_1, \ldots, t_n \in T(F)$, then $f(t_1, \cdots, t_n)$ is a term.

Actually, since they do not contain any variables, the terms considered here are usually called *ground terms*. However, we call them simply terms here, as we will not consider terms with variables until the next section.

A term $t \in T(F)$ can be seen as a finite ordered tree, the leaves of which are labeled with constants and the internal nodes of which are labeled with function

symbols of positive arity such that the outdegree of an internal node equals the arity of its label. Thus, a *position* within a term can be represented – in Dewey decimal notation – as the sequence of positive integers which describes the path from the root to that position. Accordingly, the set $O(t)$ of *occurrences* of the term t is the set of sequences of positive integers describing the positions in t. The length of the longest of these sequences is called the *depth* of the term t, which is denoted as $\mathrm{depth}(t)$, and the number of sequences in $O(t)$ is the *size* of t, denoted as $\mathrm{size}(t)$. For $p \in O(t)$, t/p denotes the subterm of t at occurrence p. If s is another term, then $t[p \leftarrow s]$ denotes the term that is obtained by replacing the subterm of t at occurrence p by the term s.

A *ground term-rewriting system* R is a subset of $T(F) \times T(F)$, the elements of which are called (*rewrite*) *rules*. The *reduction relation* associated with a ground term-rewriting system R is the reflexive and transitive closure $\to_R^*$ of the following *single-step reduction relation*: $s \to_R t$ if and only if there exist an occurrence $p \in O(s)$ and a rule $(\ell \to r) \in R$ such that $s/p = \ell$ and $t = s[p \leftarrow r]$. A term t is said to be in *normal form* or *irreducible* modulo the ground term-rewriting system R if no reduction can be applied to t. By $\mathrm{IRR}(R)$ we denote the set of all these irreducible terms, and $\mathrm{RED}(R) = T(F) \setminus \mathrm{IRR}(R)$ is the set of reducible terms.

The equational theory that is associated with a ground term-rewriting system R is the congruence $=_R$ that is generated by the reduction relation $\to_R$, that is, it is the congruence $(\to_R \cup \leftarrow_R)^*$.

A ground term-rewriting system R is called *noetherian*, (*locally*) *confluent*, or *convergent* if the reduction relation $\to_R$ has the corresponding property. It is *depth-reducing* if $\mathrm{depth}(\ell) > \mathrm{depth}(r)$ holds for each rule $\ell \to r$ of R. Finally, R is called *interreduced* if $\mathrm{range}(R) \subseteq \mathrm{IRR}(R)$ and $\ell \in \mathrm{IRR}(R \setminus \{\ell \to r\})$ for each rule $(\ell \to r) \in R$. If R is convergent and interreduced, then it is called *canonical*.

A *term language* over F is a subset of $T(F)$. As for strings term languages can be defined by formal grammars and by various types of automata. Here we are mainly interested in the class of *regular term languages* which can be defined as follows.

A *non-deterministic bottom-up tree automaton* (*NBUTA*) is given through a 4-tuple $A = (Q, F, R_A, Q_a)$, where Q is a finite set of states, F is a finite signature, $Q_a \subseteq Q$ is the set of accepting states, and R_A is a ground term-rewriting system on the signature $F \cup Q$, where each state symbol from Q is considered as a new constant. The rules of R_A are of the form

(i) $c \to q$, where $c \in F_0$ and $q \in Q$, and
(ii) $f(q_1, \ldots, q_n) \to q$, where $f \in F_n$ for some $n > 0$ and $q_1, \ldots, q_n, q \in Q$.

A is a *deterministic bottom-up tree automaton* (*BUTA*), if R_A does not contain two rules with the same left-hand side. The language $L(A)$ accepted by A is defined as $L(A) = \{t \in T(F) \mid t \to_{R_A}^* q \text{ for some } q \in Q_a\}$. A language $L \subseteq T(F)$ is *regular* if and only if it is accepted by some *NBUTA*, and this is the case if and only if it is accepted by some *BUTA* [22].

For a finite ground term-rewriting system R, a *BUTA* A can easily be constructed such that $L(A) = \mathrm{RED}(R)$. Since the class of regular term languages is

effectively closed under complement, we also obtain a *BUTA* for the set of irreducible terms $\mathrm{IRR}(R)$. Thus, $\mathrm{RED}(R)$ and $\mathrm{IRR}(R)$ are regular term languages.

In contrast to the situation for string-rewriting systems (or for that matter general term-rewriting systems) it is decidable whether or not a finite ground term-rewriting system is noetherian [32]. Further, even confluence is decidable for these systems [14,15,61]. Oyamaguchi's proof, which is combinatorically quite involved, reduces the confluence property of finite ground-term rewriting systems to the equivalence problem for non-deterministic top-down tree automata [61], while Dauchet and his co-authors invented a new kind of transducer to describe the confluence property [14,15].

A *ground tree transducer* (*GTT*) consists of a pair (G, D) of *NBUTA*s $G = (Q_G, F, R_G, Q_G)$ and $D = (Q_D, F, R_D, Q_D)$ such that $Q_G \cap Q_D$ is non-empty. The relation $\rightarrow^{(G,D)}\leftarrow$ on $T(F)$ that is induced by (G, D) is defined as follows:

$$t \rightarrow^{(G,D)}\leftarrow t' \text{ iff } \exists s \in T(F \cup (Q_G \cap Q_D)) : t \rightarrow_G^* s \leftarrow_D^* t'.$$

If a binary relation $\sim$ on $T(F)$ coincides with the relation $\rightarrow^{(G,D)}\leftarrow$ induced by a *GTT*, then $\sim$ is called a *GTT-relation*.

Proposition 16. [14,15]

(1.) The inverse of a GTT-relation is a GTT-relation.
(2.) The semi-congruence closure of a GTT-relation is a GTT-relation.
(3.) The composition of two GTT-relations is a GTT-relation.

In fact, these closure properties are effective in that, given a *GTT* for a relation $\sim$, a *GTT* for the inverse relation $\sim^{-1}$ can be constructed effectively, and similar for the other two operations.

Now from a finite ground term-rewriting system R a *GTT* A_R can be constructed for the reduction relation $\rightarrow_R^*$ on $T(F)$. From A_R we obtain *GTT*s $A_{diverge}$ and $A_{converge}$, where $A_{diverge}$ realizes the relation $\leftarrow_R^* \circ \rightarrow_R^*$ and $A_{converge}$ realizes the relation $\rightarrow_R^* \circ \leftarrow_R^*$. Observe that R is confluent if and only if $\leftarrow_R^* \circ \rightarrow_R^* \subseteq \rightarrow_R^* \circ \leftarrow_R^*$. Hence, the test for confluence of R is reduced to the inclusion problem for two *GTT*-relations. Since the inclusion of *GTT*-relations is decidable [14,15], this immediately yields the announced decidability result.

Corollary 4. [14,15,61]
The confluence property is decidable for finite ground term-rewriting systems.

Based on the same technique Dauchet and Tison even show that the first-order theory of a ground term-rewriting system is decidable [16].

We close this section with a remarkable observation concerning interreduced ground term-rewriting systems. Note that the following considerations also apply to prefix-rewriting systems, as a prefix-rewriting system on some alphabet Σ can be interpreted as a ground term-rewriting system on the signature F_Σ.

Let R be a ground term-rewriting system that is interreduced. Then $\mathrm{range}(R) \subseteq \mathrm{IRR}(R)$, and hence, it is easily seen that R is noetherian. Further, the left-hand side of no rule of R contains the left-hand side of another rule as a subterm.

Therefore, R has no critical pairs at all, and hence, it is also confluent. Hence, we have the following characterization.

Proposition 17. [66]
A ground term-rewriting system is canonical if and only if it is interreduced.

Thus, by interreduction a finite ground-term rewriting system R that is noetherian can be transformed into an equivalent finite system R_0 that is canonical. By reorienting some of its rules if necessary, R can always be turned into an equivalent system that is noetherian. This yields the following result.

Corollary 5. *For each finite ground term-rewriting system an equivalent finite ground term-rewriting system can effectively be determined that is canonical.*

In fact this process can be performed in time $O(n \log n)$ [66,67] exploiting Shostak's congruence closure algorithm [65]. Also see [36] for a discussion of this algorithm and its relation to completion of ground term-rewriting systems.

6 Term-Rewriting Systems

In this section we will consider terms with variables, which we will again simply call terms. Accordingly, the terms without variables considered in the previous section will be called *ground terms* in the following.

Let F be a finite signature, and let V be a countable set of variables. Then $T(F, V)$ denotes the set of *terms* generated by F and V. As before $T(F)$ denotes the subset of ground terms of $T(F, V)$. For a term $t \in T(F, V)$, $\mathrm{Var}(t)$ denotes the set of variables that have occurrences in t. If no variable occurs more than once in t, then t is called a *linear* term.

A *substitution* is a mapping $\sigma : V \to T(F, V)$ such that $\sigma(v) = v$ holds for almost all variables v. It can uniquely be extended to a morphism $\sigma : T(F, V) \to T(F, V)$.

A *term-rewriting system* R is a (finite) set of *rules* $R = \{\ell_i \to r_i \mid i \in I\}$, where ℓ_i and r_i are terms from $T(F, V)$. While ground term-rewriting systems can be seen as a generalization of the prefix-rewriting systems considered in Section 3, term-rewriting systems are the corresponding generalization of string-rewriting systems to general finite signatures.

A term t is *reducible* modulo R if there is a rule $\ell \to r$ in R, an occurrence $p \in O(t)$, and a substitution σ such that $\sigma(\ell) = t/p$. The term $t[p \leftarrow \sigma(r)]$ is the result of *reducing* t by $\ell \to r$ at p, and this reduction is written as $t \to_R t[p \leftarrow \sigma(r)]$. The *reduction relation* associated with the term-rewriting system R is the reflexive and transitive closure $\to_R^*$ of this *single-step reduction relation* $\to_R$. A term t is said to be in *normal form* or *irreducible* modulo R if no reduction can be applied to t. By $\mathrm{IRR}(R)$ we denote the set of all those ground terms that are irreducible, and $\mathrm{RED}(R)$ is the set $T(F) \setminus \mathrm{IRR}(R)$ of reducible ground terms.

As for ground term-rewriting systems the equational theory that is associated with a term-rewriting system R is the congruence $=_R$ that is generated by the

reduction relation $\rightarrow_R$, that is, it is the congruence $(\rightarrow_R \cup \leftarrow_R)^*$. Usually we are only interested in the restriction of this congruence to ground terms.

A term-rewriting system is called *noetherian*, (*locally*) *confluent*, *convergent* or *canonical* if the induced reduction relation has the corresponding property. It is called *left-linear* if the left-hand side of each rule of R is a linear term.

If R is a finite term-rewriting system that is left-linear, then a regular tree grammar can easily be constructed from R that generates the set $\mathrm{RED}(R)$ of reducible ground terms. Hence, we have the following result.

Proposition 18. [**21**]
For a finite term-rewriting system that is left-linear the set of irreducible ground terms as well as the set of reducible ground terms is a regular term language.

The left-linearity of the term-rewriting system considered is a crucial hypothesis for Proposition 18, as a finite non-left-linear system can easily be constructed for which the set of irreducible ground terms is not regular. However, some finite systems yield regular sets of irreducible ground terms although they are not left-linear. An example in kind is the following system which is essentially taken from [38]:

$$\begin{array}{llll} eq(x,x) & \rightarrow s(0), & eq(0,s(x)) & \rightarrow 0, \\ eq(s(x),0) & \rightarrow 0, & eq(s(x),s(y)) & \rightarrow eq(x,y), \\ eq(eq(x,y),z) & \rightarrow 0, & eq(x,eq(y,z)) & \rightarrow 0, \\ s(eq(x,y)) & \rightarrow 0. & & \end{array}$$

Thus, the question arises whether there are regular tree languages that occur as sets of irreducible ground terms for some finite term-rewriting systems that are not left-linear, but that do not occur as sets of irreducible ground terms for any finite left-linear systems. Surprisingly this is not the case.

Proposition 19. [**38**]
For a finite term-rewriting system R, if $\mathrm{IRR}(R)$ *is a regular term language, then there exists a finite left-linear system R_{lin} such that* $\mathrm{IRR}(R_{lin}) = \mathrm{IRR}(R)$.

In fact, R_{lin} consists of linear instantiations of rules of R. By associating with each transition rule of a top-down tree automaton a regular set of ground terms governing the applicability of that rule, the class of *deterministic top-down tree automata with prefix look-ahead* is defined in [20]. It yields the following characterization.

Proposition 20. [**20**]
A term language $L \subseteq T(F)$ is recognized by a one-state deterministic top-down tree automata with prefix look-ahead if and only if there exists a finite term-rewriting system R satisfying $\mathrm{IRR}(R) = L$.

Thus, the one-state deterministic top-down tree automata with prefix look-ahead, the finite left-linear term-rewriting systems, and the finite term-rewriting systems that are not left-linear all define the same subclass of the class of all regular term languages. In addition, the following decidability result holds.

Proposition 21. [29,39,69]
Given a finite term-rewriting system R, it is decidable whether or not $\mathrm{IRR}(R)$ *is a regular term language. If* $\mathrm{IRR}(R)$ *is indeed a regular term language, then a linear instantiation R_{lin} of R can be constructed such that* $\mathrm{IRR}(R_{lin}) = \mathrm{IRR}(R)$ *holds.*

A term-rewriting system R on a signature F is called *F-regularity preserving* if, for each regular term language $L \subseteq T(F)$, the set $\Delta_R^*(L)$ of all descendants is again regular. It is called *regularity preserving* if it is F-regularity preserving for each signature F containing all the function symbols that actually occur in the rules of R.

If $F := \{f, g, a\}$, where f and g are unary symbols and a is a symbol of arity 0 (a constant), then for $R := \{f(g(x)) \to f(f(g(g(x)))), f(a) \to a, g(a) \to a, a \to f(a), a \to g(a)\}$ it is easily seen that $\Delta_R^*(t) = T(F)$ holds for all ground terms $t \in T(F)$. However, if $F_1 := F \cup \{h\}$, where h is another unary function symbol, then $\Delta_R^*(f(g(h(a)))) = \{f^n(g^n(h(t))) \mid t \in T(F)\}$, which is not regular. Thus, R does preserve F-regularity, but not F_1-regularity [28]. Obviously the ground rules contained in R are responsible for this, since the subsystem $R' := \{f(g(x)) \to f(f(g(g(x))))\}$ of R does not even preserve F-regularity.

It is well-known that the property of being F-regularity preserving is undecidable in general [25,26]. In fact, this property is even undecidable for finite string-rewriting systems [56]. On the other hand, while for term-rewriting systems the property of regularity preservation depends on the actually chosen signature as indicated by the example above, this is not true for string-rewriting systems [56]. Actually, we have the following result.

Proposition 22. [56]
Let S be a string-rewriting system on Σ, let $F_\Sigma = \Sigma \cup \{\not\in\}$, where $\not\in$ is a constant and each letter from Σ is interpreted as a unary function symbol, and let R_S be the term-rewriting system $R_S = \{\ell(x) \to r(x) \mid (\ell, r) \in S\}$ on F_Σ. Then R_S is regularity preserving if and only if it preserves F_Σ-regularity.

On the other hand it is known that certain restricted classes of term-rewriting systems are regularity preserving. This applies to those systems that contain only ground rules [6], to term-rewriting systems that are right-linear and monadic [63], that are linear and semi-monadic [11], or that are linear and generalized semi-monadic [28].

While we refer to the literature for the other notions mentioned above, we recall the definition of monadic term-rewriting systems. These systems were introduced by Book and Gallier as a direct generalization of the monadic string-rewriting systems [21]. A term-rewriting system R is called *monadic* if it is left-linear and if $\mathrm{depth}(r) \leq 1$ holds for each rule $\ell \to r$ of R.

The process of reduction with respect to a finite monadic term-rewriting system that is noetherian can be realized by a *tree pushdown automaton* (*TreePDA*) [21]. For a *TreePDA* A, $L(A, B)$ denotes the set of all ground terms t for which there exists an accepting computation of A that, while processing t, produces a term from B.

Proposition 23. [21]
Let R be a finite monadic term-rewriting system that is convergent. Then for every regular tree language B, there exists a deterministic TreePDA A such that $L(A, B)$ coincides with the set of terms $[B]_R = \bigcup\{[t]_R \mid t \in B \cap \mathrm{IRR}(R)\}$.

As shown by K. Salomaa [63] the deterministic tree pushdown automata of Gallier and Book are more powerful than the corresponding automata of Schimpf [64]. An investigation of various classes of tree pushdown automata and a generalization of the results on monadic term-rewriting systems to semi-monadic systems can be found in [11].

The technique for deciding linear sentences (see Proposition 6) can obviously be lifted to those finite convergent term-rewriting systems which are effectively regularity preserving and for which the set of irreducible ground terms is regular. In particular, this has the following consequence.

Corollary 6. [25,26]
The validity of linear sentences is decidable for finite convergent term-rewriting systems that are (1.) linear and monadic, or (2.) linear and semi-monadic, or (3.) linear generalized semi-monadic.

There are many more applications of tree automata to rewriting systems. For example, Comon shows that *strong sequentiality* [33] of left-linear rewriting systems and *NV-sequentiality* [62] of linear rewriting systems are definable in WSkS, the weak second-order monadic logic of k successor functions [10], by exploiting the correspondence between this logic and tree automata [68]. Following Comon's approach Jacquemard shows that *sequentiality* is decidable for each linear rewriting system that is *growing* [34].

Further, finite *test sets* have been found to be a useful tool for deciding the membership problem for the universal closure of a given tree language, that is, for deciding whether all the ground instances of a given term belong to the language considered. By relating test sets to tree automata and to appropriate congruences Hofbauer and Huber [30] obtain characterizations of ground and non-ground test sets, and they show how to compute and to minimize these test sets.

Finally, by introducing a class of more powerful bottom-up tree automata, called *reduction automata*, Dauchet et al prove that the *first-order theory of reduction* is decidable [13].

7 Conclusion

As we have seen automata theory provides essential tools for the study of rewriting systems and their properties. On the other hand, rewriting theory has influenced the theory of automata considerably in that motivated by problems encountered in rewriting theory new classes of automata have been developed. In fact, rewriting theory with its many applications to such diverse fields as automated theorem proving, functional and logic programming, and semigroup

and group theory to mention just a few, can be seen as one of the main users of and contributors to automata theory.

Acknowledgment. I would like to thank the program committee of RTA'99 for inviting me to present this survey. Further, I want to thank Dieter Hofbauer and Klaus Madlener for their comments on a preliminary version of this paper.

References

1. R.V. Book. Confluent and other types of Thue systems. *Journal Association Computing Machinery*, 29:171–182, 1982.
2. R.V. Book. The power of the Church-Rosser property in string-rewriting systems. In D.W. Loveland, editor, *6th Conference on Automated Deduction*, Lecture Notes in Computer Science 138, pages 360–368. Springer-Verlag, Berlin, 1982.
3. R.V. Book. Decidable sentences of Church-Rosser congruences. *Theoretical Computer Science*, 24:301–312, 1983.
4. R.V. Book, M. Jantzen, and C. Wrathall. Monadic Thue systems. *Theoretical Computer Science*, 19:231–251, 1982.
5. R.V. Book and F. Otto. *String-Rewriting Systems.* Springer-Verlag, New York, 1993.
6. W.J. Brainerd. Tree generating regular systems. *Information and Control*, 14:217–231, 1969.
7. G. Buntrock and F. Otto. Growing context-sensitive languages and Church-Rosser languages. In E.W. Mayr and C. Puech, editors, *Proc. of STACS 95*, Lecture Notes in Computer Science 900, pages 313–324. Springer-Verlag, Berlin, 1995.
8. G. Buntrock and F. Otto. Growing context-sensitive languages and Church-Rosser languages. *Information and Computation*, 141:1–36, 1998.
9. C.M. Campbell, E.F. Robertson, N. Ruškuc, and R.M. Thomas. Automatic semigroups. Technical Report No. 1997/29, Dep. of Mathematics and Computer Science, University of Leicester, 1997.
10. H. Comon. Sequentiality, second order monadic logic and tree automata. In *Proceedings 10th Symposium on Logic in Computer Science*, pages 508–517. IEEE Computer Society Press, San Diego, 1995.
11. 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.
12. E. Dahlhaus and M. Warmuth. Membership for growing context-sensitive grammars is polynomial. *Journal Computer System Sciences*, 33:456–472, 1986.
13. M. Dauchet, A.-C. Caron, and J.-L. Coquidé. Automata for reduction properties solving. *Journal of Symbolic Computation*, 20:215–233, 1995.
14. M. Dauchet, T. Heuillard, P. Lescanne, and S. Tison. Decidability of the confluence of finite ground term rewrite systems and of other related term rewrite systems. *Information and Computation*, 88:187–201, 1990.
15. M. Dauchet and S. Tison. Decidability of confluence for ground term rewriting systems. In *Fundamentals of Comp. Theory, Cottbus 1985*, Lecture Notes in Computer Science 199, pages 80–89. Springer-Verlag, Berlin, 1985.
16. M. Dauchet and S. Tison. The theory of ground rewriting systems is decidable. In J.C. Mitchell, editor, *Proc. of 5th LICS*, pages 242–248. IEEE Computer Society Press, Los Alamitos, CA, 1990.

17. M.D. Davis and E.J. Weyuker. *Computability, Complexity, and Languages.* Academic Press, New York, 1983.
18. N. Dershowitz and J.P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Vol. B.: Formal Models and Semantics*, pages 243–320. Elsevier, Amsterdam, 1990.
19. D.B.A. Epstein, J.W. Cannon, D.F. Holt, S.V.F. Levy, M.S. Paterson, and W.P. Thurston. *Word Processing In Groups.* Jones and Bartlett Publishers, Boston, 1992.
20. Z. Fülöp and S. Vágvölgyi. A characterization of irreducible sets modulo left-linear term rewriting systems by tree automata. *Fundamenta Informaticae*, 13:211–226, 1990.
21. J. Gallier and R.V. Book. Reductions in tree replacement systems. *Theoretical Computer Science*, 37:123–150, 1985.
22. F. Gécseg and M. Steinby. *Tree Automata.* Akadémiai Kiadó, Budapest, 1984.
23. F. Gécseg and M. Steinby. Tree languages. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages, Vol. III*, pages 1–68. Springer-Verlag, Berlin, 1997.
24. S.M. Gersten. Dehn functions and l1-norms of finite presentations. In G. Baumslag and C.F. Miller III, editors, *Algorithms and Classification in Combinatorial Group Theory*, Math. Sciences Research Institute Publ. 23, pages 195–224. Springer-Verlag, New York, 1992.
25. 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.
26. R. Gilleron and S. Tison. Regular tree languages and rewrite systems. *Fundamenta Informaticae*, 24:157–175, 1995.
27. R. Gilman. Presentations of groups and monoids. *Journal of Algebra*, 57:544–554, 1979.
28. P. Gyenizse and S. Vágvölgyi. Linear generalized semi-monadic rewrite systems effectively preserve recognizability. *Theoretical Computer Science*, 194:87–122, 1998.
29. D. Hofbauer and M. Huber. Linearizing term rewriting systems using test sets. *Journal of Symbolic Computation*, 17:91–129, 1994.
30. D. Hofbauer and M. Huber. Test sets for the universal and existential closure of regular tree languages. *This volume.*
31. J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, M.A., 1979.
32. G. Huet and D. Lankford. *On the uniform halting problem for term rewriting systems.* Lab. Report No. 283, INRIA, Le Chesnay, France, March 1978.
33. G. Huet and J.J. Lévy. Computations in orthogonal rewriting systems I and II. In J.L. Lassez and G. Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson*, pages 395–443. MIT Press, 1991. This paper was written in 1979.
34. F. Jacquemard. Decidable approximations of term rewriting systems. In H. Ganzinger, editor, *Proc. of RTA'96*, Lecture Notes in Computer Science 1103, pages 362–376. Springer-Verlag, Berlin, 1996.
35. P. Jančar, F. Mráz, M. Plátek, and J. Vogel. On restarting automata with rewriting. In G. Păun and A. Salomaa, editors, *New Trends in Formal Languages*, Lecture Notes in Computer Science 1218, pages 119–136. Springer-Verlag, Berlin, 1997.
36. D. Kapur. Shostak's congruence closure as completion. In H. Comon, editor, *Rewriting Techniques and Applications, Proc. of RTA'97*, Lecture Notes in Computer Science 1232, pages 23–37. Springer-Verlag, Berlin, 1997.

37. T. Kretschmer. A closure property of regular languages. *Theoretical Computer Science*, 61:283–287, 1988.
38. G. Kucherov. On relationship between term rewriting systems and regular tree languages. In R.V. Book, editor, *Rewriting Techniques and Applications, Proceedings RTA'91*, Lecture Notes in Computer Science 488, pages 299–311. Springer-Verlag, Berlin, 1991.
39. G. Kucherov and M. Tajine. Decidability of regularity and related properties of ground normal form languages. *Information and Computation*, 118:91–100, 1995.
40. N. Kuhn. *Zur Entscheidbarkeit des Untergruppenproblems für Gruppen mit kanonischen Darstellungen.* Dissertation, Universität Kaiserslautern, Fachbereich Informatik, 1991.
41. N. Kuhn and K. Madlener. A method for enumerating cosets of a group presented by a canonical system. In *Proc. ISSAC'89*, pages 338–350. ACM Press, New York, 1989.
42. N. Kuhn, K. Madlener, and F. Otto. A test for λ-confluence for certain prefix rewriting systems with applications to the generalized word problem. In S. Watanabe and M. Nagata, editors, *Proceedings ISSAC'90*, pages 8–15. ACM, New York, 1990.
43. N. Kuhn, K. Madlener, and F. Otto. Computing presentations for subgroups of polycyclic groups and of context-free groups. *Applicable Algebra in Engineering, Communication and Computing*, 5:287–316, 1994.
44. K. Madlener, P. Narendran, F. Otto, and L. Zhang. On weakly confluent monadic string-rewriting systems. *Theoretical Computer Science*, 113:119–165, 1993.
45. R. McNaughton, P. Narendran, and F. Otto. Church-Rosser Thue systems and formal languages. *Journal Association Computing Machinery*, 35:324–344, 1988.
46. P. Narendran and C. Ó'Dúnlaing. Cancellativity in finitely presented semigroups. *Journal of Symbolic Computation*, 7:457–472, 1989.
47. P. Narendran and F. Otto. The problems of cyclic equality and conjugacy for finite complete rewriting systems. *Theoretical Computer Science*, 47:27–38, 1986.
48. G. Niemann and F. Otto. The Church-Rosser languages are the deterministic variants of the growing context-sensitive languages. In M. Nivat, editor, *Foundations of Software Science and Computation Structures, Proceedings FoSSaCS'98*, Lecture Notes in Computer Science 1378, pages 243–257. Springer-Verlag, Berlin, 1998.
49. G. Niemann and F. Otto. *Restarting automata, Church-Rosser languages, and confluent internal contextual languages.* Mathematische Schriften Kassel 4/99, Universität Kassel, March 1999.
50. F. Otto. Some undecidability results for non-monadic Church-Rosser Thue systems. *Theoretical Computer Science*, 33:261–278, 1984.
51. F. Otto. On deciding the confluence of a finite string-rewriting system on a given congruence class. *Journal Computer System Sciences*, 35:285–310, 1987.
52. F. Otto. Completing a finite special string-rewriting system on the congruence class of the empty word. *Applicable Algebra in Engineering, Communication and Computing*, 2:257–274, 1992.
53. F. Otto. The problem of deciding confluence on a given congruence class is tractable for finite special string-rewriting systems. *Mathematical Systems Theory*, 25:241–251, 1992.
54. F. Otto. *On Dehn functions of finitely presented bi-automatic monoids.* Mathematische Schriften Kassel 8/98, Universität Kassel, July 1998.
55. F. Otto. *On s-regular prefix-rewriting systems and automatic structures.* Mathematische Schriften Kassel 9/98, Universität Kassel, September 1998.

56. F. Otto. Some undecidability results concerning the property of preserving regularity. *Theoretical Computer Science*, 207:43–72, 1998.
57. F. Otto, M. Katsura, and Y. Kobayashi. Cross-sections for finitely presented monoids with decidable word problems. In H. Comon, editor, *Rewriting Techniques and Applications, Proceedings RTA '97*, Lecture Notes in Computer Science 1232, pages 53–67. Springer-Verlag, Berlin, 1997.
58. F. Otto, M. Katsura, and Y. Kobayashi. Infinite convergent string-rewriting systems and cross-sections for finitely presented monoids. *Journal of Symbolic Computation*, 26:621–648, 1998.
59. F. Otto, A. Sattler-Klein, and K. Madlener. Automatic monoids versus monoids with finite convergent presentations. In T. Nipkow, editor, *Rewriting Techniques and Applications, Proceedings RTA '98*, Lecture Notes in Computer Science 1379, pages 32–46. Springer-Verlag, Berlin, 1998.
60. F. Otto and L. Zhang. Decision problems for finite special string-rewriting systems that are confluent on some congruence class. *Acta Informatica*, 28:477–510, 1991.
61. M. Oyamaguchi. The Church-Rosser property for ground term-rewriting systems is decidable. *Theoretical Computer Science*, 49:43–79, 1987.
62. M. Oyamaguchi. NV-sequentiality: a decidable condition for call-by-need computations in term-rewriting systems. *SIAM Journal on Computing*, 22:114–135, 1993.
63. K. Salomaa. Deterministic tree pushdown automata and monadic tree rewriting systems. *Journal Computer System Sciences*, 37:367–394, 1988.
64. K.M. Schimpf and J.H. Gallier. Tree pushdown automata. *Journal Computer System Sciences*, 30:25–40, 1985.
65. R.E. Shostak. An algorithm for reasoning about equality. *Communications of the Association for Computing Machinery*, 21:583–585, 1978.
66. W. Snyder. Efficient ground completion: an $O(n \log n)$ algorithm for generating reduced sets of ground rewrite rules equivalent to a set of ground equations E. In N. Deshowitz, editor, *Rewriting Techniques and Applications, Proceedings RTA '89*, Lecture Notes in Computer Science 355, pages 419–433. Springer-Verlag, Berlin, 1989.
67. W. Snyder. A fast algorithm for generating reduced ground rewriting systems from a set of ground equations. *Journal of Symbolic Computation*, 15:415–450, 1993.
68. J. Thatcher and J. Wright. Generalized finite automata with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, 2:57–82, 1968.
69. S. Vágvölgyi and R. Gilleron. For a rewrite system it is decidable whether the set of irreducible ground terms is recognizable. *Bulletin of the EATCS*, 48:197–209, 1992.
70. L.G. Valiant. The equivalence problem for deterministic finite-turn pushdown automata. *Information and Control*, 25:123–133, 1974.