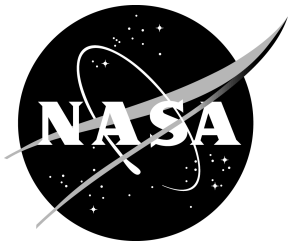
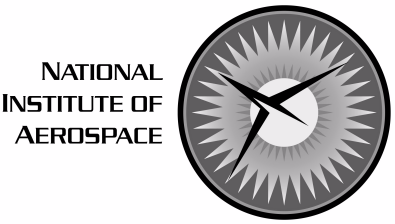
NASA/CR-2003-212685
NIA Report No. 2003-09





Match-bounded String Rewriting Systems

Alfons Geser
National Institute of Aerospace, Hampton, Virginia

Dieter Hofbauer
University of Kassel, Kassel, Germany

Johannes Waldmann
University of Leipzig, Leipzig, Germany

December 2003

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the lead center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

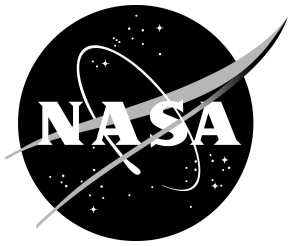
- Email your question via the Internet to help@sti.nasa.gov

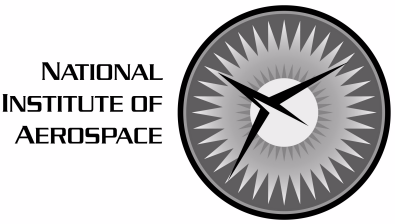
- Fax your question to the NASA STI Help Desk at (301) 621-0134

- Telephone the NASA STI Help Desk at (301) 621-0390

- Write to:
  NASA STI Help Desk
  NASA Center for AeroSpace Information
  7121 Standard Drive
  Hanover, MD 21076-1320

NASA/CR-2003-212685
NIA Report No. 2003-09





Match-bounded String Rewriting Systems

Alfons Geser
National Institute of Aerospace, Hampton, Virginia

Dieter Hofbauer
University of Kassel, Kassel, Germany

Johannes Waldmann
University of Leipzig, Leipzig, Germany

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

Prepared for Langley Research Center
under Contract NCC-1-02043

December 2003

Available from the following:

NASA Center for AeroSpace Information (CASI)
7121 Standard Drive
Hanover, MD 21076-1320
(301) 621-0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161-2171
(703) 487-4650

MATCH-BOUNDED STRING REWRITING SYSTEMS*

Alfons Geser†, Dieter Hofbauer‡, and Johannes Waldmann§

ABSTRACT

We introduce a new class of automated proof methods for the termination of rewriting systems on strings. The basis of all these methods is to show that rewriting preserves regular languages. To this end, letters are annotated with natural numbers, called *match heights*. If the minimal height of all positions in a redex is h then every position in the reduct will get height $h + 1$. In a *match-bounded* system, match heights are globally bounded. Using recent results on *deleting* systems, we prove that rewriting by a match-bounded system preserves regular languages. Hence it is decidable whether a given rewriting system has a given match bound. We also provide a sufficient criterion for the absence of a match-bound. The problem of existence of a match-bound is still open. Match-boundedness for all strings can be used as an automated criterion for termination, for match-bounded systems are terminating. This criterion can be strengthened by requiring match-boundedness only for a restricted set of strings, for instance the set of right hand sides of forward closures.

1 INTRODUCTION

Rewriting is a model of computation. It allows to handle questions like *termination* (there is no infinite computation), *normalization* (a final configuration is reachable) and *correctness* (no erroneous configuration is reachable). These questions can be stated in terms of sets of descendants: if R is a rewriting system, and L is a language, then $R^*(L) = \{y \mid x \in L, x \to_R^* y\}$. Now R is *correct* for L iff $R^*(L) \cap \mathrm{Err} = \emptyset$, and R is *normalizing* for L iff $L \subseteq R^{-*}(\mathrm{Final})$, with Err and Final denoting the set of erroneous and final configurations, respectively. Starting from classical program analysis, recent applications include verification of XML transformations [3] and cryptographic protocols [10].

From the point of view of these applications, the reachability relation R^* should effectively respect language classes with good decidability and closure properties—like the class of regular languages. Some of us recently showed [17] that *deleting* string rewriting systems respect regular languages. In the present paper, we transfer this result to *match-bounded* string rewriting.

Every match-bounded system terminates, and effectively preserves regularity of languages. Therefore it is decidable whether a given system has a given match-bound. This makes match-boundedness a new automatic criterion for termination. The criterion applies

*This work was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-97046.

†Senior Staff Scientist, National Institute of Aerospace (NIA), 144 Research Drive, Hampton, VA 23666. Email: geser@nianet.org, Web: http://research.nianet.org/~geser/.

‡Assistant Professor, Dept. of Mathematics/Informatics, University of Kassel, D-34109 Kassel, Germany. Email: dieter@theory.informatik.uni-kassel.de.

§Assistant Professor, Faculty of Mathematics/Informatics, University of Leipzig, D-04109 Leipzig, Germany. Email: joe@informatik.uni-leipzig.de.

for instance to Zantema's System $\{a^2b^2 \to b^3a^3\}$ (match-bound 4) for which hitherto all automated termination proof methods failed.

A string rewriting system R is called *deleting* if there exists a partial ordering on its alphabet such that each letter in the right hand side of a rule is less than some letter in the corresponding left hand side. Deleting systems can be understood as the inverses of *context limited* grammars as defined and investigated by Hibbard [16]. Deleting rewriting systems terminate and have linearly bounded derivational complexity.

To obtain automated termination proofs, we transform rewriting systems as follows: We annotate letters with numbers, which we call *match heights*. A position in a reduct will get height $h + 1$ if the minimal height of all positions in the redex is h. A rewriting system is *match-bounded* if match heights of derivations are globally bounded. In this case its annotated system is finite and deleting. Termination and regularity preservation carry over from the annotated to the original system. The recognizing automaton for the set of descendants modulo the annotated system is a certificate for match-boundedness.

We study also *RFC-match-boundedness*, a variant of the criterion, where a system has to be match-bounded only for the set of right hand sides of its forward closures. By a result of Dershowitz, termination there is sufficient for uniform termination.

Basic definitions, results and examples are given in Sections 3 and 4, while in Section 5 we discuss how to verify or refute match-boundedness. In Section 6 we introduce RFC-match-boundedness, and consider some variants of this notion in Section 7. All main criteria are implemented (Section 8). Section 9 contains a short comparison of our new termination criteria with Zantema's Termination Hierarchy. We conclude by discussing ramifications for further research in Section 10.

Some of the results reported here have been presented at the 28th International Symposium on Mathematical Foundations of Computer Science MFCS 2003 at Bratislava, Slovak Republic [12] and at the 6th International Workshop on Termination WST 2003 at Valencia, Spain [13].

2 PRELIMINARIES

We mostly stick to standard notations for strings and string rewriting, as e.g. in [2]. We use ϵ for the *empty string*, and $|x|$ is the *length* of a string x. Let REG denote the class of regular languages. Further, for a language $L \subseteq \Sigma^*$, let $\mathrm{factor}(L) = \{y \in \Sigma^* \mid \exists x, z \in \Sigma^* : xyz \in L\}$.

A *string rewriting system* over an alphabet Σ is a relation $R \subseteq \Sigma^* \times \Sigma^*$, inducing the *rewrite relation* $\to_R = \{(x\ell y, xry) \mid x, y \in \Sigma^*, (\ell, r) \in R\}$ on Σ^*. Unless indicated otherwise, all rewriting systems are finite. Pairs (ℓ, r) from R are frequently referred to as *rules* $\ell \to r$. By $\mathrm{lhs}(R)$ and $\mathrm{rhs}(R)$ we denote the sets of left (resp. right) hand sides of R. The reflexive and transitive closure of $\to_R$ is $\to_R^*$, often abbreviated as R^*, and $\to_R^+$ or R^+ denote the transitive closure. An *R-derivation* is a (finite or infinite) sequence $(x_0, x_1, \dots)$ with $x_i \to_R x_{i+1}$ for all i. We call R *terminating on* $L \subseteq \Sigma^*$ if there is no infinite derivation starting with some $x_0 \in L$. If $L = \Sigma^*$, we call R *terminating*. In order to classify lengths of derivations, define the *derivation height* function modulo R on Σ^* by $\mathrm{dh}_R(x) = \max\{n \in \mathbb{N} \mid \exists y \in \Sigma^* : x \to_R^n y\}$. The *derivational complexity* of R is defined as the function $n \mapsto \max\{\mathrm{dh}_R(x) \mid |x| \le n\}$ on $\mathbb{N}$.

A rewriting rule $\ell \to r$ is *context-free* if $|\ell| \le 1$, and a rewriting system is context-free if all its rules are.

For a relation $\rho \subseteq A \times B$ let $\rho(a) = \{b \in B \mid (a,b) \in \rho\}$ for $a \in A$ and $\rho(A') = \bigcup_{a \in A'} \rho(a)$ for $A' \subseteq A$. The inverse of ρ is $\rho^- = \{(b,a) \mid (a,b) \in \rho\} \subseteq B \times A$, and we say that ρ satisfies the property *inverse* P if ρ^- satisfies P. Thus, the set of *descendants* of a language $L \subseteq \Sigma^*$ modulo some rewriting system R is $R^*(L)$. The system R is said to *preserve regularity (context-freeness)* if $R^*(L)$ is a regular (context-free) language whenever L is.

For a relation $\rho \subseteq \Sigma^* \times \Sigma^*$ and a set $\Delta \subseteq \Sigma$, let $\rho|_\Delta$ denote $\rho \cap (\Delta^* \times \Delta^*)$. Note the difference between $R^*|_\Delta$ and $(R|_\Delta)^*$ for a string rewriting system R. E.g., for $R = \{a \to b, b \to c\}$ over $\Sigma = \{a,b,c\}$ and $\Delta = \{a,c\}$ we have $(a,c) \in R^*|_\Delta$, but $(a,c) \notin (R|_\Delta)^*$.

A relation $s \subseteq \Sigma^* \times \Gamma^*$ is a *substitution* if $s(\epsilon) = \{\epsilon\}$ and $s(xy) = s(x)s(y)$ for $x, y \in \Sigma^*$. So a substitution s is uniquely determined by the languages $s(a)$ for $a \in \Sigma$. If each language $s(a)$ for $a \in \Sigma$ is finite, then s is a *finite* substitution.

Now we recall definitions and results regarding *deleting* string rewriting systems [17], a topic that goes back to Hibbard [16]. A string rewriting system R over an alphabet Σ is $>$*-deleting* for an irreflexive partial ordering $>$ on Σ (a *precedence*) if $\epsilon \notin \mathrm{lhs}(R)$, and if for each rule $\ell \to r$ in R and for each letter a in r, there is some letter b in ℓ with $b > a$. The system R is *deleting* if it is $>$-deleting for some precedence $>$.

Proposition 1 ([17]). *Every deleting string rewriting system is terminating, and has linear derivational complexity.*

Furthermore, we have the following decomposition result.

Theorem 1 ([17]). *Let R be a deleting string rewriting system over Σ. Then there are an extended alphabet $\Gamma \supseteq \Sigma$, a finite substitution $s \subseteq \Sigma^* \times \Gamma^*$, and a context-free string rewriting system C over Γ such that $R^* = (s \circ C^{-*})|_\Sigma$.*

As a consequence, inverse deleting systems effectively preserve context-freeness, a result by Hibbard [16]. As another consequence we get:

Corollary 1 ([17]). *Every deleting string rewriting system effectively preserves regularity.*

3 MATCH-BOUNDED STRING REWRITING SYSTEMS

We will now apply the theory of deleting systems to obtain results for *match-bounded* rewriting. A derivation is match-bounded if dependencies between rule applications are limited. To make this precise, we will annotate positions in strings by natural numbers that indicate their *match height*. Positions in a reduct will get height $h+1$ if the minimal height of all positions in the corresponding redex was h.

Given an alphabet Σ, define the morphisms $\mathrm{lift}_c : \Sigma^* \to (\Sigma \times \mathbb{N})^*$ for $c \in \mathbb{N}$ by $\mathrm{lift}_c : a \mapsto (a,c)$, $\mathrm{base} : (\Sigma \times \mathbb{N})^* \to \Sigma^*$ by $\mathrm{base} : (a,c) \mapsto a$, and $\mathrm{height} : (\Sigma \times \mathbb{N})^* \to \mathbb{N}^*$ by $\mathrm{height} : (a,c) \mapsto c$. For a string rewriting system R over Σ such that $\epsilon \notin \mathrm{lhs}(R)$, we define the rewriting system

$$\mathrm{match}(R) = \{\ell' \to \mathrm{lift}_c(r) \mid (\ell \to r) \in R, \mathrm{base}(\ell') = \ell, c = 1 + \min(\mathrm{height}(\ell'))\}$$

over alphabet $\Sigma \times \mathbb{N}$. For instance, the system $\mathrm{match}(\{ab \to bc\})$ contains the rules $a_0b_0 \to b_1c_1$, $a_0b_1 \to b_1c_1$, $a_1b_0 \to b_1c_1$, $a_1b_1 \to b_2c_2$, $a_0b_2 \to b_1c_1$, ..., writing x_c as abbreviation for (x,c). For non-empty R, the system $\mathrm{match}(R)$ is always infinite. Note that systems

with $\epsilon \in \mathrm{lhs}(R)$ are trivially non-terminating, so the above restriction does not exclude any interesting cases.

Every derivation modulo $\mathrm{match}(R)$ corresponds to a derivation modulo R, (for $x, y \in (\Sigma \times \mathbb{N})^*$, if $x \to_{\mathrm{match}(R)} y$ then $\mathrm{base}(x) \to_R \mathrm{base}(y)$) and vice versa (for $v, w \in \Sigma^*$ and $x \in (\Sigma \times \mathbb{N})^*$, if $v \to_R w$ and $\mathrm{base}(x) = v$, then there is $y \in (\Sigma \times \mathbb{N})^*$ such that $\mathrm{base}(y) = w$ and $x \to_{\mathrm{match}(R)} y$). In particular, for $n \in \mathbb{N}$ we have $R^n = \mathrm{lift}_0 \circ \mathrm{match}(R)^n \circ \mathrm{base}$; thus

$$R^* = \mathrm{lift}_0 \circ \mathrm{match}(R)^* \circ \mathrm{base}\,.$$

Definition 1. A string rewriting system R over Σ is called *match-bounded* for $L \subseteq \Sigma^*$ by $c \in \mathbb{N}$ if $\epsilon \notin \mathrm{lhs}(R)$ and $\max(\mathrm{height}(x)) \leq c$ for every $x \in \mathrm{match}(R)^*(\mathrm{lift}_0(L))$. If we omit L, then it is understood that $L = \Sigma^*$.

Note that $\max(\mathrm{height}(x))$ (and $\min(\mathrm{height}(\ell'))$ in the definition of $\mathrm{match}(R)$) denotes the maximum (minimum, respectively) over the corresponding sequences of heights; we set $\max(\epsilon) = 0$, and we leave $\min(\epsilon)$ undefined as this case is excluded in the definition of $\mathrm{match}(R)$. Obviously, a system that is match-bounded for L is also match-bounded for any subset of L by the same bound. Further, if R is match-bounded for L then R is match-bounded for $R^*(L)$, again by the same bound.

For a match-bounded system R, the infinite system $\mathrm{match}(R)$ may be replaced by a finite restriction. Denote by $\mathrm{match}_c(R)$ the restriction of $\mathrm{match}(R)$ to the alphabet $\Sigma \times \{0, 1, \ldots, c\}$.

Lemma 1. *If R is match-bounded for L by c, then $R^n|_L = (\mathrm{lift}_0 \circ \mathrm{match}_c(R)^n \circ \mathrm{base})|_L$ for $n \in \mathbb{N}$, thus*

$$R^*|_L = (\mathrm{lift}_0 \circ \mathrm{match}_c(R)^* \circ \mathrm{base})|_L.$$

Lemma 2. *For all R with $\epsilon \notin \mathrm{lhs}(R)$ and all $c \in \mathbb{N}$, the system $\mathrm{match}_c(R)$ is deleting.*

Proof. Use the precedence $>$ on $\Sigma \times \{0, \ldots, c\}$ where $(a, m) > (b, n)$ iff $m < n$. (Letters of minimal match height are maximal in the precedence.) □

Theorem 2. *If R is match-bounded for L, then R is terminating on L.*

Proof. An infinite R-derivation starting from an element of L can be transformed into an infinite $\mathrm{match}(R)$-derivation from an element of $\mathrm{lift}_0(L)$. The latter, given that R is match-bounded by c, is a $\mathrm{match}_c(R)$-derivation. However, $\mathrm{match}_c(R)$ is deleting by Lemma 2 and hence terminating by Proposition 1. □

Likewise, Lemma 1 implies linearly bounded derivation lengths for match-bounded systems.

Proposition 2. *Every match-bounded string rewriting system has linear derivational complexity.*

We conclude this section with a few examples.

Example 1. The system $\{ab \to bc\}$ is match-bounded by 1, $\{aa \to aba\}$ is match-bounded by 2, $\{ab \to ac, ca \to bc\}$ is match-bounded by 2, and $\{ab \to ac, ca \to b\}$ is match-bounded by 3.

All these bounds can be verified automatically, as will be explained in Section 5. The next example illustrates that indeed any number can be a least match bound.

Example 2. The bubble sort system $B_2 = \{ab \to ba\}$ over the two-letter alphabet $\{a, b\}$ is match-bounded for a^*b^n by n, but not by $n-1$. The system $\{a_i \to a_{i+1} \mid 0 \le i < n\}$ over alphabet $\Sigma = \{a_i \mid 0 \le i \le n\}$ is match-bounded (for Σ^*) by n, but not by $n-1$. As a variant of the previous example, now over a fixed alphabet, consider the system $\{ab^ic \to ab^{i+1}c \mid 0 \le i < n\}$ over $\{a, b, c\}$; it is match-bounded by n, but not by $n-1$. The same holds true for the length-preserving variant $\{ab^ic^{n-i+1} \to ab^{i+1}c^{n-i} \mid 0 \le i < n\}$.

Example 3. System B_2 is not match-bounded (for $\{a, b\}^*$) since it has quadratic derivational complexity, contradicting the conclusion of Proposition 2.

Dually to Lemma 2, we have:

Proposition 3. *If R is deleting, then R is match-bounded.*

Proof. Assume R over Σ is deleting for the precedence $>$ on Σ. Then R is match-bounded by the maximal height (i.e., length of a descending chain) in $(\Sigma, >)$. □

Example 4. The system $\{ba \to cb, bd \to d, cd \to de\}$ is match-bounded by 2, since it is deleting for the precedence $a > b > d$, $a > c > e$, $c > d$.

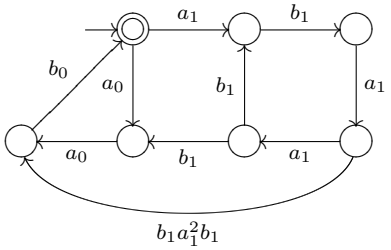
4 MATCH-BOUNDED SYSTEMS PRESERVE REGULARITY

Here, we elaborate on the fact that match-bounded string rewriting systems always preserve regularity. The section concludes on a short comparison of match-boundedness to the related concept of change-boundedness [25].

Theorem 3. *If R is match-bounded for $L \in$ REG, then $R^*(L) \in$ REG.*

Proof. By Lemma 1, $R^*(L) = \text{base}(\text{match}_c(R)^*(\text{lift}_0(L)))$ for some $c \in \mathbb{N}$. As $\text{match}_c(R)$ is deleting by Lemma 2, thus regularity preserving by Corollary 1, and since REG is closed under morphisms, we are done. □

Example 5. For $R = \{aaba \to abaab\}$ (cf. [19], p. 118) and $L = (aab)^*$, the language $\text{match}(R)^*(\text{lift}_0(L))$ is accepted by the following automaton. We use generalized automata where transitions are labelled by words instead of single letters.



By stripping heights from all letters, one obtains an automaton accepting $R^*(L)$.

Example 6. The bubble sort system $B_2 = \{ab \to ba\}$ is not regularity preserving, since $B_2^*((ab)^*) \cap a^*b^* = \{a^nb^n \mid n \ge 0\}$ is not regular. So Theorem 3 implies that B_2 is not match-bounded. (Cf. Example 3 for another indirect proof, and Example 10 for a direct proof of the same fact.)

However, not every regularity preserving string rewriting system is match-bounded. For instance, the system $\{aa \to a\}$ constitutes a counterexample. As a *monadic* system (i.e., $|\ell| > |r| \leq 1$ for $(\ell \to r) \in R$) it preserves regularity [1, 2], but it is not match-bounded as proven in Example 12.

Remark 1. There are terminating and regularity preserving systems with high derivational complexity as we are going to demonstrate.

For an alphabet Σ, define the string rewriting system $\mathrm{Embed}(\Sigma) = \{a \to \epsilon \mid a \in \Sigma\}$. By an application of Kruskal's Theorem, the *subword language* $\mathrm{Embed}(\Sigma)^*(L)$ is regular for *each* language L over Σ, cf. Theorem 7.3 in [4]. This implies that for any rewriting system R over Σ, the system $R \cup \mathrm{Embed}(\Sigma)$ preserves (in fact, *generates*) regularity.

Termination of $R \cup \mathrm{Embed}(\Sigma)$ is called *simple termination* of R. By the above, every simply terminating rewriting system R can be extended to a (simply) terminating and regularity preserving system while keeping or increasing its derivational complexity. E.g., $\{ab \to ba, a \to \epsilon, b \to \epsilon\}$ preserves regularity, and has quadratic complexity.

Example 7. Peg solitaire is a one-person game. The objective is to remove pegs from a board. A move consists of one peg X hopping over an adjacent peg Y, landing on the empty space on the opposite side of Y. After the hop, Y is removed. Peg solitaire on a one-dimensional board corresponds to the string rewriting system

$$P = \{■■□ \to □□■,\ □■■ \to ■□□\},$$

where ■ stands for "peg", and □ for "empty". One is interested in the language of all positions that can be reduced to one single peg, which is $P^{-*}(□^*■□^*)$. Regularity of $P^{-*}(□^*■□^*)$ is a "folklore theorem", see [24] for its history. The system P^- is match-bounded by 2, so we obtain yet another proof of that result.

Remark 2. Ravikumar [25] proves that P^- preserves regularity by considering the system's *change-bound* (which is 4). Change-boundedness is similar to match-boundedness. Given a length-preserving string rewriting system R (viz. $|\ell| = |r|$ for every rule $\ell \to r$), define the system

$$\mathrm{change}(R) = \{\ell \to r \mid (\mathrm{base}(\ell) \to \mathrm{base}(r)) \in R,\ \mathrm{height}(\mathrm{succ}(\ell)) = \mathrm{height}(r)\}$$

over alphabet $\Sigma \times \mathbb{N}$, where succ is the morphism $\mathrm{succ} : (\Sigma \times \mathbb{N})^* \to (\Sigma \times \mathbb{N})^*$ induced by $\mathrm{succ} : (a, h) \mapsto (a, h + 1)$. For instance, the system $\mathrm{change}(\{ab \to bc\})$ contains the rules $a_0b_0 \to b_1c_1$, $a_0b_1 \to b_1c_2$, $a_1b_0 \to b_2c_1$, $a_1b_1 \to b_2c_2$, $a_0b_2 \to b_1c_3$, ... Ravikumar proves that if $\mathrm{change}(R)^*(\mathrm{lift}_0(L))$ has bounded height, then R preserves regularity of L. In contrast to change-bounds, match-bounds are also applicable to non-length-preserving systems. For length-preserving systems, $\mathrm{match}(R)$ will always give lower or equal heights, so our result directly implies Ravikumar's. In fact, it can also be shown conversely that match-boundedness implies change-boundedness for length-preserving systems.

5 VERIFICATION AND REFUTATION OF MATCH-BOUNDS

In this section, we show that match-boundedness by a given bound is decidable. Further, we provide a sufficient condition for the absence of a match bound. We leave decidability of match-boundedness as an open problem.

Theorem 4. *The following problem is decidable:*
GIVEN: *A string rewriting system R, a regular language L, and $c \in \mathbb{N}$.*
QUESTION: *Is R match-bounded for L by c?*

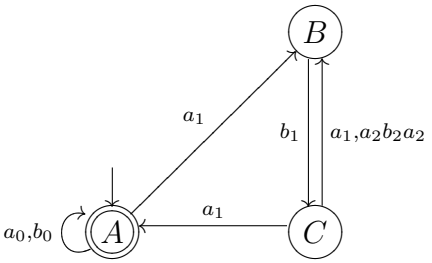
Proof. Construct a finite automaton for $L_{c+1} = \mathrm{match}_{c+1}(R)^*(\mathrm{lift}_0(L))$, using Theorem 3. Then R is match-bounded for L by c iff $\max(\mathrm{height}(L_{c+1})) \leq c$. □

Any given automaton over alphabet $\Sigma \times \mathbb{N}$ can be seen as a potential certificate of the fact that R is match-bounded for L by c, and hence of termination of R on L. The certificate is *valid* if the accepted language

1. includes $\mathrm{lift}_0(L)$,
2. is closed under rewriting modulo $\mathrm{match}_{c+1}(R)$, and
3. contains no letter of height $c+1$.

The first two items imply that $\mathrm{match}_{c+1}(R)^*(\mathrm{lift}_0(L))$ is included in the accepted language. Validity of such a certificate can be decided by standard algorithms for finite automata.

Example 8. For $R = \{aa \to aba\}$ and $L = \{a, b\}^*$, the set $\mathrm{match}(R)^*(\mathrm{lift}_0(L))$ is accepted by the following (non-deterministic) automaton. (Again, we use an obvious generalized notation where transitions are labelled by sets of words.)



redex path	reduct path
$A \xrightarrow{a_0} A \xrightarrow{a_0} A$	$A \xrightarrow{a_1} B \xrightarrow{b_1} C \xrightarrow{a_1} A$
$A \xrightarrow{a_0} A \xrightarrow{a_1} B$	$A \xrightarrow{a_1} B \xrightarrow{b_1} C \xrightarrow{a_1} B$
$C \xrightarrow{a_1} A \xrightarrow{a_0} A$	$C \xrightarrow{a_1} B \xrightarrow{b_1} C \xrightarrow{a_1} A$
$C \xrightarrow{a_1} A \xrightarrow{a_1} B$	$C \xrightarrow{a_2b_2a_2} B$

Closure under $\mathrm{match}(R)$ can be verified by checking off the table on the right. Since the highest label is 2, the automaton certifies that R is match-bounded by 2, as claimed in the introductory Example 1.

For an implementation, the growth of $|\mathrm{match}_c(R)|$ as a function of c is problematic. However, when computing $\mathrm{match}_c(R)^*(\mathrm{lift}_0(L))$, we may restrict attention to those rules of $\mathrm{match}_c(R)$ that are *accessible* in derivations starting from $\mathrm{lift}_0(L)$. For a language $L \subseteq \Sigma^*$, a system R over Σ, and a system $S \subseteq \mathrm{match}(R)$ define

$$\mathrm{accessible}(L, R, S) = \mathrm{match}(R) \cap (\mathrm{factor}(S^*(\mathrm{lift}_0(L))) \times (\Sigma \times \mathbb{N})^*).$$

Note that this construction is effective if a finite system S and a regular language L are effectively given. We construct a sequence of rewriting systems R_i by $R_0 = \emptyset$ and $R_{i+1} = \mathrm{accessible}(L, R, R_i)$. Induction on i shows $R_i \subseteq \mathrm{match}_i(R)$ for $i \geq 0$. In particular, every system R_i is finite. By induction on i, using that $S \subseteq S'$ implies $\mathrm{accessible}(L, R, S) \subseteq \mathrm{accessible}(L, R, S')$, one also proves that $R_i \subseteq R_{i+1}$. Define $R_\infty = \bigcup_{i \in \mathbb{N}} R_i$. Clearly, $R_\infty^*(\mathrm{lift}_0(L)) = \mathrm{match}(R)^*(\mathrm{lift}_0(L))$. If R is match-bounded for L by c, then R_∞ is a subset of $\mathrm{match}_c(R)$; so R_∞ is finite, and there is an index N such that $R_N = R_{N+1} = \cdots$. If R is

not match-bounded for L, then R_∞ contains for each c a rule with height c, and therefore is infinite. We remark that the enumeration of R_i up to $i = |\operatorname{match}_c(R)| + 1$ can be used as an alternative decision procedure for Theorem 4.

Example 9. Proving termination of the one-rule system $Z = \{a^2b^2 \to b^3a^3\}$ is known as *Zantema's Problem*. This is a "modern classic" in rewriting [5, 8, 19, 27, 28, 32], as it provides a test case where all previous automated methods for termination proofs fail. Our algorithm constructs in 6 iterations a deterministic automaton with 85 states. This automaton recognizes $\operatorname{match}(Z)^*(\operatorname{lift}_0(\Sigma^*))$ and certifies that Z is match-bounded for Σ^* by 4. This also proves that Z has only linear derivational complexity, a result by Tahhan-Bittar [28].

Sometimes we can also verify automatically that a given rewriting system R ($\epsilon \notin \operatorname{lhs}(R)$) is *not* match-bounded for a language L. For this purpose, we want a non-empty witnessing language $W \subseteq L$ such that every element in W can be reached from some element in W by an all-height increasing derivation. By chaining such derivations, strings of arbitrary height can be derived, disproving match-boundedness. In the remainder of this section we formalize this argument.

For $u, v \in (\Sigma \times \mathbb{N})^*$ we write $u \geq v$ if $\operatorname{base}(u) = \operatorname{base}(v)$ and $\operatorname{height}(u) \geq_n \operatorname{height}(v)$, where $\geq_n$ denotes the pointwise greater-or-equal ordering on $\mathbb{N}^n$. We assume $W \subseteq \Sigma^+$. A string $y \in W$ is *reached* from $x \in W$ if there is a derivation $\operatorname{lift}_0(x) \to^*_{\operatorname{match}(R)} py'q$ for some string $y' \geq \operatorname{lift}_1(y)$ and strings p, q. Now every element in W is reached from an element in W if $W \subseteq \operatorname{raised}(R, W)$, where the latter set of strings is defined by

$$\operatorname{raised}(R, W) = \operatorname{base}(\operatorname{factor}(\operatorname{match}(R)^*(\operatorname{lift}_0(W))) \cap (\Sigma \times (\mathbb{N} \setminus \{0\}))^+).$$

First we observe that a $\operatorname{match}(R)$-derivation can always be raised to greater heights since the two relations $\geq$ and $\to_{\operatorname{match}(R)}$ commute:

Lemma 3. $\geq \circ \to_{\operatorname{match}(R)} \subseteq \to_{\operatorname{match}(R)} \circ \geq$.

Proposition 4. *Let R be a string rewriting system such that $\epsilon \notin \operatorname{lhs}(R)$, and let W be a non-empty language, both over Σ. If $W \subseteq \operatorname{raised}(R, W)$, then R is not match-bounded for W.*

Proof. We prove a stronger claim: If $W \subseteq \operatorname{raised}(R, W)$ then, for every $c \geq 0$,

$$W \subseteq \operatorname{base}(\operatorname{factor}(\operatorname{match}(R)^*(\operatorname{lift}_0(W))) \cap (\Sigma \times \{c, c+1, \dots\})^+).$$

In other words, every element of W can receive unbounded match heights. We prove this claim by induction on c. Consider $y \in W$. For $c = 0$ we obtain $y \in \operatorname{base}(\operatorname{factor}(\operatorname{lift}_0(y)) \cap (\Sigma \times \mathbb{N})^+)$. So assume $c > 0$. By inductive hypothesis there is a string $u \in W$ and a derivation

$$\operatorname{lift}_0(u) \to^*_{\operatorname{match}(R)} py'q$$

with $y' \geq \operatorname{lift}_{c-1}(y)$. Since $u \neq \epsilon$, this derivation can be relabelled to a derivation

$$\operatorname{lift}_1(u) \to^*_{\operatorname{match}(R)} \operatorname{succ}(p) \operatorname{succ}(y') \operatorname{succ}(q)$$

with $\mathrm{succ}(y') \geq \mathrm{succ}(\mathrm{lift}_{c-1}(y)) = \mathrm{lift}_c(y)$, where succ is the morphism defined in Section 4, increasing the height of each position by 1. Since $u \in W \subseteq \mathrm{raised}(R, W)$, there is $v \in W$ and a derivation

$$\mathrm{lift}_0(v) \to^*_{\mathrm{match}(R)} p'u'q' \tag{1}$$

with $u' \geq \mathrm{lift}_1(u)$. By Lemma 3 we get a derivation

$$u' \to^*_{\mathrm{match}(R)} p''y''q'' \tag{2}$$

for some $y'' \geq \mathrm{succ}(y')$. We conclude by composing (1) and (2) into a derivation

$$\mathrm{lift}_0(v) \to^*_{\mathrm{match}(R)} p'u'q' \to^*_{\mathrm{match}(R)} p'p''y''q''q'$$

with $y'' \geq \mathrm{lift}_c(y)$. □

A slightly weaker version of Proposition 4 is obtained as follows: Define

$$\mathrm{raised}_c(R, W) = \mathrm{base}(\mathrm{factor}(\mathrm{match}_c(R)^*(\mathrm{lift}_0(W))) \cap (\Sigma \times (\mathbb{N} \setminus \{0\}))^+),$$

and replace $\mathrm{raised}(R, W)$ in Proposition 4 by $\mathrm{raised}_c(R, W)$:

Corollary 2. *Let R be a string rewriting system such that $\epsilon \notin \mathrm{lhs}(R)$, and let W be a non-empty language, both over Σ. If $W \subseteq \mathrm{raised}_c(R, W)$ for some $c \in \mathbb{N}$, then R is not match-bounded for W.*

This version can be effectively checked if a finite system R, a number $c \in \mathbb{N}$, and a regular language W are effectively given.

Example 10. The system $B_2 = \{ab \to ba\}$ (cf. Example 6) is not match-bounded for Σ^*. Take $W = (ab)^+$. Then $\mathrm{raised}_1(B_2, W) = \mathrm{factor}((ba)^+) \supseteq W$.

Example 11. Neither is $R = \{aabb \to ba\}$ match-bounded, as witnessed by $W = \{a, b\}^+ = \mathrm{raised}_1(R, W)$. This can be seen as follows. Define the two morphisms $\phi : a \mapsto a, b \mapsto abb$ and $\psi : a \mapsto aab, b \mapsto b$. Then, for each $y \in \Sigma^*$, there are derivations

$$a\ \phi(y) \to^*_R y\ a \quad \text{and} \quad \psi(y)\ b \to^*_R b\ y,$$

and these can be combined to a derivation

$$a\ \phi(\psi(y))\ abb = a\ \phi(\psi(y)\ b) \to^*_R \psi(y)\ b\ a \to^*_R b\ y\ a.$$

When lifting this to a match(R)-derivation, starting from heights 0, all final heights are 1. This proves that for each $y \in W = \Sigma^+$, there is $x = a\phi(\psi(y))abb \in \Sigma^+ = W$ with the required property. In contrast, the system $\{a^3b^3 \to b^2a^2\}$ is match-bounded by 2.

Example 12. The regularity preserving system $R = \{aa \to a\}$ is not match-bounded: check that $W = \{a^{2^n} \mid n \in \mathbb{N}\} \subseteq \mathrm{raised}_1(R, W)$. Alternatively, $W' = a^+ \subseteq \mathrm{raised}_1(R, W')$.

Example 13. The system $R = \{ab \to bba\}$ is not match-bounded for Σ^* because it admits derivations $a^n b \to_R^{2^n-1} b^{2^n} a^n$ of exponential lengths. Another proof can be given by Proposition 4. One shows by induction that, for $k \geq 0$,

1. $a_k b_k^i \to^*_{\mathrm{match}(R)} b_{k+1}^{2i} a_{k+1}$ for $i > 0$, and

2. $a_0 b_1 b_1 b_2^2 \dots b_k^{2^{k-1}} \to^*_{\mathrm{match}(R)} b_1 b_1 b_2^2 \dots b_{k+1}^{2^k} a_{k+1}$.

So $a_0^m b_0$, $m > 1$, rewrites to a string that contains the factor $a_{m-1} \dots a_1 b_1$:

$$a_0^m b_0 \to_{\mathrm{match}(R)} a_0^{m-1} b_1^2 a_1 \to^*_{\mathrm{match}(R)} b_1 b_1 b_2^2 \dots b_{m-1}^{2^{m-2}} a_{m-1} \dots a_1 b_1 a_1.$$

Hence $W \subseteq \mathrm{raised}(R, W)$ for $W = a^+ b$. Note that this set of witnesses is regular, but the given derivations (verifying the witnesses) are not globally match-bounded. On the other hand, we can have match-bounded verification for the non-regular set of witnesses $W' = \{ab^{2^n} ab^{2^{n-1}} \dots ab \mid n \in \mathbb{N}\}$, since $W' \subseteq \mathrm{raised}_1(R, W')$.

Looping string rewriting systems form a particular subclass of the class of all non-terminating systems. A *loop* is a derivation of the form $s \to_R^+ psq$ for strings $s, p, q \in \Sigma^*$. As it turns out, the existence of a loop can be characterized in terms of *finite* sets of witnesses, as follows.

Proposition 5. *A string rewriting system R admits a loop if and only if there is a finite, non-empty set W such that $W \subseteq \mathrm{raised}(R, W)$.*

Proof. If R admits a loop then it also admits a loop $s \to_R^+ psq$ during which every position between letters is touched [14]. So $\mathrm{lift}_0(s) \to^+_{\mathrm{match}(R)} p's'q'$ for some $s' \geq \mathrm{lift}_1(s)$. The claim holds with $W = \{s\}$. Conversely, let $W \subseteq \mathrm{raised}(R, W)$. Then for every $k > 0$ there is a sequence $w_0, w_1, \dots, w_k$ such that $w_{i+1} \in \mathrm{raised}(R, \{w_i\})$ for $0 \leq i < k$, thus $w_j \in \mathrm{raised}(R, \{w_i\})$ for $0 \leq i < j \leq k$. For $k = |W|$, by the pigeonhole principle, there are $i < j$ such that $w_i = w_j$. Hence $w_i = w_j \in \mathrm{factor}(R^+(\{w_i\}))$ forms the desired loop. □

The converse of Propositon 4 is open:

Problem 1. Does every string rewriting system R such that $\epsilon \notin \mathrm{lhs}(R)$ that is not match-bounded have a non-empty set $W \subseteq \mathrm{raised}(R, W)$?

If the stronger statement "…have some $c \in \mathbb{N}$ and a non-empty *regular* set $W \subseteq \mathrm{raised}_c(R, W)$" holds then match-boundedness is decidable: One can simultaneously enumerate these certificates (c, W) along with certificates for match-boundedness (according to Theorem 4). Example 13 seems to indicate that the stronger statement is false. So the following remains open:

Problem 2. Is match-boundedness decidable?

6 MATCH-BOUNDS FOR FORWARD CLOSURES

We have shown that match-boundedness for L is a criterion for termination on L. To prove termination on Σ^* however, the obvious choice $L = \Sigma^*$ may be too restrictive as it even entails linear derivational complexity. We are going to show that the set of right hand sides of forward closures [20, 7] is a better choice for L. For a string rewriting system R over Σ, the set of *forward closures* $\mathrm{FC}(R) \subseteq \Sigma^* \times \Sigma^*$ is defined as the least set containing R such that

- if $(u, v) \in \mathrm{FC}(R)$ and $v \to_R w$, then $(u, w) \in \mathrm{FC}(R)$ (*inside reduction*), and

- if $(u, v\ell_1) \in \mathrm{FC}(R)$ and $(\ell_1\ell_2 \to r) \in R$ for strings $\ell_1 \neq \epsilon$, $\ell_2 \neq \epsilon$, then $(u\ell_2, vr) \in \mathrm{FC}(R)$ (*right extension*).

Let $\mathrm{RFC}(R)$ denote the set of right hand sides of forward closures. Equivalently, $\mathrm{RFC}(R)$ is the least subset of Σ^* containing $\mathrm{rhs}(R)$ such that

- if $v \in \mathrm{RFC}(R)$ and $v \to_R w$, then $w \in \mathrm{RFC}(R)$, and
- if $v\ell_1 \in \mathrm{RFC}(R)$ and $(\ell_1\ell_2 \to r) \in R$ for $\ell_1 \neq \epsilon$, $\ell_2 \neq \epsilon$, then $vr \in \mathrm{RFC}(R)$.

Theorem 5 ([6]). *A string rewriting system R is terminating on Σ^* if and only if R is terminating on* $\mathrm{RFC}(R)$.

Theorem 2 for $L = \mathrm{RFC}(R)$ yields:

Corollary 3. *Every string rewriting system R that is match-bounded for* $\mathrm{RFC}(R)$ *is terminating.*

Example 14. The system $R = \{aa \to aba\}$ (cf. Example 8) is match-bounded for $\mathrm{RFC}(R)$ by 0 since the set $\mathrm{RFC}(R) = (ab)^+a$ consists of strings in normal form. Therefore, R is terminating.

We can obtain $\mathrm{RFC}(R)$ as a set of descendants modulo the rewriting system $R_\# = R \cup \{\ell_1\# \to r \mid (\ell_1\ell_2 \to r) \in R, \ell_1 \neq \epsilon, \ell_2 \neq \epsilon\}$ over alphabet $\Sigma \cup \{\#\}$, where right extension is simulated via the new end-marker $\# \notin \Sigma$. Indeed,

$$\mathrm{RFC}(R) = R_\#^*(\mathrm{rhs}(R) \cdot \#^*) \cap \Sigma^*$$

is an immediate consequence of the following equality.

Lemma 4. *Let R be a string rewriting system over Σ, where $\# \notin \Sigma$. Then* $\mathrm{RFC}(R) \cdot \#^* = R_\#^*(\mathrm{rhs}(R) \cdot \#^*)$.

Proof. Show the inclusion from left to right by induction over the definition of $\mathrm{RFC}(R)$. Conversely, $R_\#^n(\mathrm{rhs}(R) \cdot \#^*) \subseteq \mathrm{RFC}(R) \cdot \#^*$ is shown by induction over n. □

Definition 2. The string rewriting system R is *RFC-match-bounded* if $R_\#$ is match-bounded for $\mathrm{rhs}(R) \cdot \#^*$.

Recall that $R_\#$ is match-bounded for $\mathrm{rhs}(R) \cdot \#^*$ if and only if $R_\#$ is match-bounded for $R_\#^*(\mathrm{rhs}(R) \cdot \#^*)$.

Corollary 4. *If a string rewriting system R is RFC-match-bounded, then the language* $\mathrm{RFC}(R)$ *is regular.*

Lemma 5. *If a string rewriting system R is RFC-match-bounded, then R is match-bounded for* $\mathrm{RFC}(R)$.

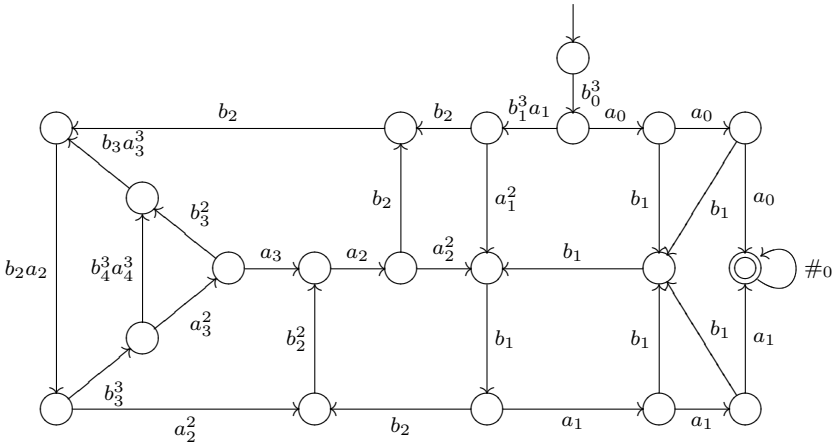
Proof. If $R_\#$ is match-bounded for $R_\#^*(\mathrm{rhs}(R) \cdot \#^*)$, then R is match-bounded for $\mathrm{RFC}(R)$ as $R \subseteq R_\#$ and, by Lemma 4, $\mathrm{RFC}(R) \subseteq R_\#^*(\mathrm{rhs}(R) \cdot \#^*)$. □

However, RFC-match-boundedness and match-boundedness for RFC(R) are not equivalent, see Example 20 for a counterexample.

Combining the previous lemma with Theorems 2 and 5, we obtain the following termination criterion.

Theorem 6. *Every RFC-match-bounded string rewriting system is terminating.*

Example 15. Zantema's system $Z = \{a^2b^2 \to b^3a^3\}$ from Example 9 is RFC-match-bounded by 4, as the following finite automaton accepts the language $\mathrm{match}_5(Z_\#)^*(\mathrm{lift}_0(\mathrm{rhs}(Z)\cdot\#^*))$.



This automaton is a certificate for termination (cf. the remark after Theorem 4).

Example 16. The system $B_2 = \{ab \to ba\}$ from Example 6 is RFC-match-bounded by 1 since $\mathrm{match}(B_{2\#})^*(\mathrm{lift}_0(\mathrm{rhs}(B_2)\cdot\#^*)) = b_0(a_0 \cup b_1^+a_1)\#_0^*$. It is not match-bounded, see Example 10.

Example 17. The system $R = \{ab \to bba\}$ is RFC-match-bounded by 1. Here, as can easily be seen, $\mathrm{match}(R_\#)^*(\mathrm{lift}_0(\mathrm{rhs}(R)\cdot\#^*)) = b_0^2(a_0 \cup (b_1^2)^+a_1)\#_0^*$. Again, this system is not match-bounded, see Example 13.

The Examples 16 and 17 show that RFC-match-bounded systems, unlike match-bounded systems, may have non-linear derivational complexities. We do not know of an RFC-match-bounded system with longer than exponential derivations.

Example 18. The bubble-sort system over a three-letter alphabet, $B_3 = \{ab \to ba, ac \to ca, bc \to cb\}$, is not match-bounded for RFC(B_3), and hence not RFC-match-bounded. To prove it, check $b^+c^+a \subseteq \mathrm{RFC}(B_3)$, and observe that $\{bc \to cb\} \subseteq B_3$ is not match-bounded for b^+c^+, cf. Example 6. In contrast, all proper subsystems of B_3 are RFC-match-bounded by 1.

Example 19. For $R = \{ab \to baa\}$, we have $\mathrm{RFC}(R) \cap b^*a^* = \{b^na^{2^n} | n \geq 1\} \notin \mathrm{REG}$. By Corollary 4, R is not RFC-match-bounded, in contrast to Example 17. This shows that the class of RFC-match-bounded systems is not closed under reversal, i.e., under the operation $R \mapsto \{\mathrm{rev}(\ell) \to \mathrm{rev}(r) \mid (\ell \to r) \in R\}$, where $\mathrm{rev}(a_1a_2\ldots a_n) = a_n\ldots a_2a_1$ for $a_i \in \Sigma$. (Note that the class of terminating systems is trivially closed under reversal.)

7 RFC-MATCH-BOUNDEDNESS AND RELATED CONDITIONS

As a sufficient condition for termination of a string rewriting system R, we introduced match-boundedness of R for $\mathrm{RFC}(R)$. In order to construct $\mathrm{RFC}(R)$, we used the enriched system $R_{\#}$. This system contains additional rules that subtly influence match heights, as indicated in this section.

Example 20. Here, we will present an example demonstrating that the inverse of Lemma 5 does not hold true. We claim that the string rewriting system R over alphabet $\{a, b, c, d, e\}$ with rules

$$\{a \to b,\ b \to cd,\ de \to a,\ cb \to a\}$$

is match-bounded for $\mathrm{RFC}(R)$, but not RFC-match-bounded.

Claim 1: R is match-bounded for $\mathrm{RFC}(R)$. Indeed, it is straightforward to verify that R is match-bounded by 3 for $\mathrm{RFC}(R) = c^*a \cup c^*b \cup c^+d$.

Claim 2: R is not RFC-match-bounded. This is a direct consequence of the fact that, for $z \in \{0, 1\}$ and for any $n \geq 1$,

$$a_z \#_0^{2^n-1} \to^*_{\mathrm{match}(R_{\#})} a_{2n+1}.$$

The proof is by induction on n. We have $a_z\#_0 \to b_{z+1}\#_0 \to c_{z+2}d_{z+2}\#_0 \to c_{z+2}a_1 \to c_{z+2}b_2 \to a_3$ for $n = 1$, and for $n > 1$ we obtain

$$a_z \#_0^{2^n-1} \to^* a_{2n-1}\#_0^{2^{n-1}} \to b_{2n}\#_0^{2^{n-1}} \to c_{2n+1}d_{2n+1}\#_0^{2^{n-1}} \to$$
$$c_{2n+1}a_1\#_0^{2^{n-1}-1} \to^* c_{2n+1}a_{2n-1} \to c_{2n+1}b_{2n} \to a_{2n+1},$$

the induction hypothesis being applied twice. Throughout, rewriting is modulo $\mathrm{match}(R_{\#})$.

Example 21. Even if a string rewriting system R is both match-bounded for $\mathrm{RFC}(R)$ and RFC-match-bounded, the corresponding least match-bounds may differ by any given number $k > 0$. This is shown for the system

$$R = \{a_{i-1} \to a_i, b_{i-1} \to b_i \mid 1 \leq i < k\} \cup \{a_{k-1} \to cd, de \to b_0, cb_{k-1} \to a_0\}$$

over alphabet $\{a_0, \ldots, a_{k-1}, b_0, \ldots, b_{k-1}, c, d, e\}$. As is easily seen, R is match-bounded for $\mathrm{RFC}(R)$ by $k+1$, whereas $R_{\#}$ is match-bounded for $\mathrm{rhs}(R) \cdot \#^*$ by $2k+1$. So the difference between these bounds is indeed k.

For completeness' sake we also mention a sufficient criterion for RFC-match-boundedness. We will use the set of left hand sides of forward closures of a rewriting system R, denoted by $\mathrm{LFC}(R)$.

We remark that computation of $\mathrm{LFC}(R)$ seems to require the construction of the full set $\mathrm{FC}(R)$, a step that could be avoided for $\mathrm{RFC}(R)$.

Proposition 6. *If a string rewriting system R is match-bounded for* $\mathrm{LFC}(R)$ *by c then R is RFC-match-bounded by c.*

Proof. For any step that uses a rule $\ell_1\# \to r$, it is possible to reconstruct some string ℓ_2 with $\ell_1\ell_2 \to r$ in R that $\#$ represents. This transformation preserves match heights. □

Example 22. The least RFC-match-bound of $R = \{aa \to aba\}$ from Example 8 is 1. The least match-bound of R for $\mathrm{LFC}(R) = aa^+$, however, is 2.

Example 23. The system $R = \{aba \to a, ab \to ba\}$ is RFC-match-bounded by 1, but R is not match-bounded for $a(ba)^+ \subseteq \mathrm{LFC}(R)$.

8 IMPLEMENTING MATCH-BOUNDS: MATCHBOX

We have implemented the algorithms presented in this paper (Theorems 4 and 6) in a program called `Matchbox`. It can be accessed via a CGI-interface at `http://theo1.informatik.uni-leipzig.de/matchbox/`, its Haskell source is available.

The program fared quite well in the recent "termination competition" held at the 6th International Workshop on Termination (WST 2003) at Valencia, Spain. Unlike its competitors, however, `Matchbox` only addresses string rewriting.

In particular, `Matchbox` is able to prove termination for a large number of one-rule string rewrite systems for which all standard automated methods (like path orderings and polynomial interpretations) fail, and for which only complicated ad-hoc proofs were known, if any. The list below contains those one-rule systems that are left from an attempt to classify termination of all (approx. $6.7 \cdot 10^9$) one-rule systems $\{\ell \to r\}$ where $|\ell| < |r| \leq 9$. They cannot be solved by any known method [11].

$$\begin{array}{rr} \{abaab \to baabbaa\}, & \{babbaa \to abbaabba\}, \\ \{aabaaab \to baaabbaaa\}, & \{ababaab \to baabbabaa\}, \\ \{baabba \to aabbaaabb\}, & \{caabca \to aabccaabc\}, \\ \{aabaaba \to abaabaaab\}, & \{abaab \to baabbaaba\}. \end{array}$$

`Matchbox` yields proofs that all these systems are RFC-match-bounded by 2.

9 A COMPARISON TO THE TERMINATION HIERARCHY

We have shown that for string rewriting systems R the following implications are valid:

$$\begin{array}{r} \text{match-bounded} \Rightarrow \\ \text{match-bounded for LFC}(R) \Rightarrow \\ \text{RFC-match-bounded} \Rightarrow \\ \text{match-bounded for RFC}(R) \Rightarrow \\ \text{terminating} \end{array}$$

None of these implications can be reversed: The system $\{ab \to ba\}$ from Example 10 is match-bounded for $\text{LFC}(R) = ab^+$, but not match-bounded. Example 23 is RFC-match-bounded but not match-bounded for $\text{LFC}(R)$. Example 20 contains a counterexample to the converse of the third implication. Finally, the system B_3 from Example 18 is terminating though not match-bounded for $\text{RFC}(B_3)$. It is interesting to compare these results with Zantema's *termination hierarchy* [30, 31]:

$$\begin{array}{c} \text{polynomially terminating} \Rightarrow \omega\text{-terminating} \qquad \Rightarrow \text{totally terminating} \Rightarrow \\ \text{simply terminating} \Rightarrow \text{non-self-embedding} \Rightarrow \text{terminating} \end{array}$$

As it turns out, this hierarchy is orthogonal to all four properties mentioned above. Indeed, $B_3 = \{ab \to ba, ac \to ca, bc \to cb\}$ is polynomially terminating (choose $n \mapsto 3n + 1$, $n \mapsto 2n + 1$ and $n \mapsto n + 1$ as interpretation for a, b and c respectively), but not match-bounded for $\text{RFC}(R)$. And $\{aa \to aba\}$ is a non-self-embedding system that is nevertheless match-bounded.

10 CONCLUSION

If the flow of information during rewriting is suitably restricted, some desirable properties hold: termination, bounded derivational complexity, or preservation of regular languages. For instance, McNaughton [21] and independently Ferreira and Zantema [9] use extra letters to indicate the absence of information flow through certain positions. Kobayashi et al. [18] restrict derivations by using markers for the start and the end of a redex. Sénizergues [27] constructs finite automata to solve the termination problem for certain one-rule string rewriting systems. Moczydłowski and Geser [22, 23] restrict the way the right hand side of a rule may be consumed in order to simulate the rewrite relation by the computation of a pushdown automaton.

With our concepts of deleting and match-bounded string rewriting, we aim at extending these approaches to a systematic theory of *termination by language properties*. Regularity preservation forms a basis for automated termination proofs. We present two variants to demonstrate some of the potential of this new approach. Match-boundedness on the set of all strings over the given alphabet is easiest to conceive. On the other hand, match-boundedness on more restricted sets, for instance the right hand sides of forward closures, may significantly enlarge the application domain. Each method can solve hard examples, like Zantema's system.

We expect these powerful criteria to enable some major progress in the decision problem of uniform termination of one-rule string rewriting systems, a problem open for 13 years [19] (see also [26, Problem 21]). Our hope is supported by the fact that some hard one-rule systems can now be proven terminating automatically.

Single-player games like Peg Solitaire can be analyzed through the construction of reachability sets. It is challenging to extend this approach to two-player rewriting games [29]. Interesting properties are termination, which is necessary for a well-defined game, or regularity of winning sets. Even the impartial case is hard; here the central question is whether Grundy values are bounded.

It seems natural to carry over the notion of match-boundedness to term rewriting, in order to obtain both closure properties and new automated termination proof methods.

Acknowledgements.

This research was supported in part by the National Aeronautics and Space Administration (NASA) while the last two authors were visiting scientists at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center (LaRC), Hampton, VA, in September 2002.

REFERENCES

[1] R. V. Book, M. Jantzen, and C. Wrathall. Monadic Thue systems. *Theoret. Comput. Sci.*, 19:231–251, 1982.

[2] R. V. Book and F. Otto. *String-Rewriting Systems*. Texts Monogr. Comput. Sci.. Springer-Verlag, New York, 1993.

[3] B. Chidlovskii. Using regular tree automata as XML schemas. In J. Hoppenbrouwers, T. de Souza Lima, M. Papazoglou, and A. Sheth (Eds.), *Proc. IEEE Advances in Digital Libraries 2000 ADL-00*, pp. 89–98. IEEE Comput. Society, 2000.

[4] C. Choffrut and J. Karhumäki. Combinatorics of words. In G. Rozenberg and A. Salomaa (Eds.), *Handbook of Formal Languages*, Vol. 1, pp. 329–438. Springer-Verlag, 1998.

[5] T. Coquand and H. Persson. A proof-theoretical investigation of Zantema's problem. In M. Nielsen and W. Thomas (Eds.), *Proc. 11th Annual Conf. of the EACSL CSL-97*, *Lecture Notes in Comput. Sci.* Vol. 1414, pp. 177–188. Springer-Verlag, 1998.

[6] N. Dershowitz. Termination of linear rewriting systems. In S. Even and O. Kariv (Eds.), *Proc. 8th Int. Coll. Automata, Languages and Programming ICALP-81*, *Lecture Notes in Comput. Sci.* Vol. 115, pp. 448–458. Springer-Verlag, 1981.

[7] N. Dershowitz. Termination of rewriting. *J. Symbolic Comput.*, 3(1–2):69–115, 1987.

[8] N. Dershowitz and C. Hoot. Topics in termination. In C. Kirchner (Ed.), *Proc. 5th Int. Conf. Rewriting Techniques and Applications RTA-93*, *Lecture Notes in Comput. Sci.* Vol. 690, pp. 198–212. Springer-Verlag, 1993.

[9] M. C. F. Ferreira and H. Zantema. Dummy elimination: Making termination easier. In H. Reichel (Ed.), *10th Int. Symp. Fundamentals of Computation Theory FCT-95*, *Lecture Notes in Comput. Sci.* Vol. 965, pp. 243–252. Springer-Verlag, 1995.

[10] T. Genet and F. Klay. Rewriting for Cryptographic Protocol Verification. In D. A. McAllester (Ed.), *17th Int. Conf. Automated Deduction CADE-17*, *Lecture Notes in Artificial Intelligence* Vol. 1831, pp. 271–290. Springer-Verlag, 2000.

[11] A. Geser. Is Termination Decidable for String Rewriting with only One Rule? Habilitationsschrift. Eberhard-Karls-Universität Tübingen, Germany, 2001.

[12] A. Geser, D. Hofbauer and J. Waldmann. Match-bounded string rewriting systems. In B. Rovan and P. Vojtas (Eds.), *Proc. 28th Int. Symp. Mathematical Foundations of Computer Science MFCS-03*, *Lecture Notes in Comput. Sci.* Vol. 2747, pp. 449-459. Springer-Verlag, 2003.

[13] A. Geser, D. Hofbauer, and J. Waldmann. Match-bounded string rewriting systems and automated termination proofs. In A. Rubio (Ed.), *Proc. 6th Int. Workshop on Termination WST-03*, Technical Report DSIC-II/15/03, Universidad Politécnica de Valencia, Spain, pp. 19–22, 2003.

[14] A. Geser and H. Zantema. Non-looping string rewriting. *RAIRO – Theoret. Inform. Appl.*, 33:279–302, 1999.

[15] S. Ginsburg and S. A. Greibach. Mappings which preserve context sensitive languages. *Inform. and Control*, 9(6):563–582, 1966.

[16] T. N. Hibbard. Context-limited grammars. *J. ACM*, 21(3):446–453, 1974.

[17] D. Hofbauer and J. Waldmann. Deleting string rewriting systems preserve regularity. In Z. Ésik and Z. Fülöp (Eds.), *Proc. 7th Int. Conf. Developments in Language Theory DLT-03*, *Lecture Notes in Comput. Sci.* Vol. 2710, pp. 337-348. Springer-Verlag, 2003.

[18] Y. Kobayashi, M. Katsura, and K. Shikishima-Tsuji. Termination and derivational complexity of confluent one-rule string-rewriting systems. *Theoret. Comput. Sci.*, 262(1-2):583–632, 2001.

[19] W. Kurth. Termination und Konfluenz von Semi-Thue-Systemen mit nur einer Regel. Dissertation, Technische Universität Clausthal, Germany, 1990.

[20] D. S. Lankford and D. R. Musser. A finite termination criterion. Technical Report, Information Sciences Institute, Univ. of Southern California, Marina-del-Rey, CA, 1978.

[21] R. McNaughton. Semi-Thue systems with an inhibitor. *J. Automat. Reason.*, 26:409–431, 2001.

[22] W. Moczydłowski Jr. Jednoregułowe systemy przepisywania słów. Masters thesis, Warsaw University, Poland, 2002.

[23] W. Moczydłowski Jr. and A. Geser. Termination of single-threaded one-rule Semi-Thue systems. Technical Report TR 02-08 (273), Warsaw University, Dec. 2002. Available at `http://research.nianet.org/~geser/papers/single.html`.

[24] C. Moore and D. Eppstein. One-dimensional peg solitaire, and duotaire. In R. J. Nowakowski (Ed.), *More Games of No Chance*, Cambridge Univ. Press, 2003.

[25] B. Ravikumar. Peg-solitaire, string rewriting systems and finite automata. In H.-W. Leong, H. Imai, and S. Jain (Eds.), *Proc. 8th Int. Symp. Algorithms and Computation ISAAC-97*, *Lecture Notes in Comput. Sci.* Vol. 1350, pp. 233–242. Springer-Verlag, 1997.

[26] The RTA list of open problems. `http://www.lsv.ens-cachan.fr/rtaloop/`.

[27] G. Sénizergues. On the termination problem for one-rule semi-Thue systems. In H. Ganzinger (Ed.), *Proc. 7th Int. Conf. Rewriting Techniques and Applications RTA-96*, *Lecture Notes in Comput. Sci.* Vol. 1103, pp. 302–316. Springer-Verlag, 1996.

[28] E. Tahhan Bittar. Complexité linéaire du problème de Zantema. *C. R. Acad. Sci. Paris Sér. I Inform. Théor.*, t. 323:1201–1206, 1996.

[29] J. Waldmann. Rewrite games. In S. Tison (Ed.), *Proc. 13th Int. Conf. Rewriting Techniques and Applications RTA-02*, *Lecture Notes in Comput. Sci.* Vol. 2378, pp. 144–158. Springer-Verlag, 2002.

[30] H. Zantema. Termination of term rewriting: interpretation and type elimination. *J. Symbolic Comput.*, 17(1):23–50, 1994.

[31] H. Zantema. The termination hierarchy for term rewriting. *Appl. Algebra Engrg. Comm. Comput.*, 12(1-2):3–19, 2001.

[32] H. Zantema and A. Geser. A complete characterization of termination of $0^p1^q \to 1^r0^s$. *Appl. Algebra Engrg. Comm. Comput.*, 11(1):1–25, 2000.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01/12/2003	Contractor Report	

4. TITLE AND SUBTITLE

Match-bounded String Rewriting Systems

5a. CONTRACT NUMBER

5b. GRANT NUMBER

NCC-1-02043

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Geser, Alfons, Hofbauer, Dieter, and Waldmann, Johannes

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

786-10-10-10-00

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NASA Langley Research Center
Hampton, VA 23681-2199

National Institute of Aerospace (NIA)
144 Research Drive
Hampton, VA 23666

8. PERFORMING ORGANIZATION REPORT NUMBER

NIA Report No. 2003-09

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITOR'S ACRONYM(S)

NASA

11. SPONSORING/MONITORING REPORT NUMBER

NASA/CR-2003-212685

12. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 60, 61
Availability: NASA CASI (301) 621-0390 Distribution: Nonstandard

13. SUPPLEMENTARY NOTES

Langley Technical Monitor: Ricky Butler

14. ABSTRACT

We introduce a new class of automated proof methods for the termination of rewriting systems on strings. The basis of all these methods is to show that rewriting preserves regular languages. To this end, letters are annotated with natural numbers, called match heights. If the minimal height of all positions in a redex is h then every position in the reduct will get height h+1. In a match-bounded system, match heights are globally bounded. Using recent results on deleting systems, we prove that rewriting by a match-bounded system preserves regular languages. Hence it is decidable whether a given rewriting system has a given match bound. We also provide a sufficient criterion for the abence of a match-bound. The problem of existence of a match-bound is still open. Match-boundedness for all strings can be used as an automated criterion for termination, for match-bounded systems are terminating. This criterion can be strengthened by requiring match-boundedness only for a restricted set of strings, for instance the set of right hand sides of forward closures.

15. SUBJECT TERMS

String Rewriting; Termination; Match-bounded; Deleting; Forward Closure

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19b. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			STI Help Desk (email: help@sti.nasa.gov)
U	U	U	UU	23	19b. TELEPHONE NUMBER *(Include area code)* (301) 621-0390