# Regularity Preservation by String-Rewriting Systems Based on Periods

Peter Leupold*

Department of Mathematics, Faculty of Science
Kyoto Sangyo University
Kyoto 603-8555, Japan
eMail:leupold@cc.kyoto-su.ac.jp

When using string-rewriting systems in the context of formal languages, one of the most common questions is whether they preserve regularity. A class of string-rewriting systems that has received attention lately are idempotency relations. They were mainly used to generate languages starting from a single word.

Here we apply these relations to entire languages and investigate whether they preserve regularity. For this, it turns out to be convenient to define two more general classes of string-rewriting systems, the $k$-period expanding and the $k$-period reducing ones. We show that both preserve regularity. This implies regularity preservation for many classes of idempotency relations.

## 0 Introduction

The root of our investigations lies in the operation called duplication and introduced by Dassow et al. [3], who rediscovered a result shown earlier by Bovet and Varricchio [2] for so-called copy systems introduced by Ehrenfeucht and Rozenberg [4]. Mainly, a string-rewriting system that duplicates factors via rules $u \to u^2$ is applied iteratively to a word; then the question is whether the resulting language is regular or context-free. Later, this operation was also applied to entire languages rather than single words [7]. On

*This work was done, while the author was funded by the Japanese Society for the Promotion of Science under the postdoctoral grant P07810.

the other hand, the duplication of words was also generalized to so-called idempotency languages [8]. These are generated by rules $u^m \to u^n$ for any fixed $m$ and $n$ rather than only by rules $u^1 \to u^2$.

Another line of research has dealt with classes of string-rewriting system like monadic or prefix rewriting ones. It was investigated whether the result is regular/ context-free if they are iteratively applied to regular/context-free languages. An example is the work of Hofbauer and Waldmann on deleting string-rewriting systems [5] which provides many references to earlier work. Also the book by Book and Otto contains a few results in this direction [1].

So far, investigations on idempotency relations have been focused on whether they produce regular or context-free languages when applied to singleton languages. In the context of the work on regularity preservation of string-rewriting systems it seems even more interesting to look at their behaviour when applied to entire languages. This is the object of this article, which therefore in some sense brings the two lines of research described above together. We consider only the length-bounded variants of idempotency relations; only here the underlying rewriting-systems are finite and therefore they are more tractable in this context.

In the section about uniformly length-bounded systems we actually treat a more general class of systems. Namely, we abandon the restriction that all rules must be of the form $u^m \to u^n$ for fixed $m$ and $n$. First we define $k$-period-expanding string-rewriting systems, where only $m \leq n$ is required. Then we also consider the somewhat inverse class of $k$-period-reducing string-rewriting systems, which are characterized by the condition $m \geq n$. Both of these classes are shown to preserve regularity. Finally, we show that also finite unions of $k$-expanding and $k$-reducing systems, so-called $k$-periodic systems preserve regularity. It is a direct consequence of these results that all relations ${}^{=k}{\bowtie}^{n}_{m}$ preserve regularity except for $k \geq 2$, $m = 0$, and $n \geq 2$. For the latter cases it is already known that they generate non-regular languages from single words [8].

There are less results about idempotency relations with just an upper bound on the left side of rules. Mainly a result on $k$-period reducing systems can be adapted and yields that length-decreasing bounded idempotency relations preserve regularity. The proof also shows that their inverses preserve context-freeness. We also solve a problem left open in earlier work [7]: duplications with length three preserve regularity.

## 1 String-Rewriting Systems

Terms and notation from general formal language theory, logic and set theory are assumed to be known by the reader. Let $w[i]$ denote the $i$-th letter of a word $w$ for $1 \leq i \leq |w|$, where $|w|$ is $w$'s length. By $w[i \dots j]$ we denote the factor of a word $w$, which begins in position $i$ and ends in $j$. A word $w$ has a positive integer $k$ as a *period*, iff for all $i, j$ such that $i \equiv j (mod\ k)$ we have $w[i] = w[j]$, if both $w[i]$ and $w[j]$ are defined. $u \subset_{\text{pref}} v$ means that $u$ is a prefix of $v$, $\subset_{\text{suff}}$ is our symbol for suffix. Two words $u$ and $v$ are *conjugates* iff there exists a factorization $u = rs$ such that $v = sr$. If not specified otherwise, the alphabet we use will be denoted by $\Sigma$.

In our notation on string-rewriting systems we mostly follow Book and Otto [1] and define a *string-rewriting system (SRS) $R$* on $\Sigma$ to be a subset of $\Sigma^* \times \Sigma^*$. Its single-step reduction relation is defined as $u \to_R v$ iff there exists $(\ell, r) \in R$ such that for some $u_1, u_2$ we have $u = u_1 \ell u_2$ and $v = u_1 r u_2$. We also write simpler just $\to$, if it is clear which is the underlying rewriting system. By $\xrightarrow{*}$ we denote the relation's reflexive and transitive closure, which is called the *reduction relation* or *rewrite relation*. The inverse of a single-step reduction relation $\to$ is $\to^{-1} := \{(r, \ell) : (\ell, r) \in \to\}$. Note that we also use the notation $u \to v$ for rewrite rules, mainly when speaking about rules in a natural language sentence to make it graphically clear that we are speaking about a rewrite rule and not some other ordered pair. All the SRSs in this article will be finite.

An SRS is said to be *confluent*, iff for all $w, w_1, w_2 \in \Sigma^*$ always $w_1 \xleftarrow{*} w \xrightarrow{*} w_2$ implies the existence of some $w'$ such that $w_1 \xrightarrow{*} w' \xleftarrow{*} w_2$. Here we use $w_1 \leftarrow w$ as a sometimes convenient way of writing $w \to w_1$.

By imposing restrictions on the format of the rewriting rules, many special classes of rewriting systems can be defined. Following Hofbauer and Waldmann [5], we will call a rule $(\ell, r)$ *context-free* (*inverse context-free*), if $|\ell| \leq 1$ ($|r| \leq 1$). A system is *monadic*, if it is inverse context-free and for all its rewrite rules $(\ell, r)$ we have $|\ell| > |r|$. Finally, we define *deleting* SRSs again following Hofbauer and Waldmann [5]. For these, we need a precedence, i.e. a irreflexive partial ordering $<$ on the alphabet. This is extended to words by defining that $u < v$ holds iff $u$ and $v$ do not use the same set of letters, and for every letter $x$ which occurs in $u$ there exists a letter $y$ which occurs in $v$ such that $x < y$. Now a SRS over the alphabet $\Sigma$ is called *$<$-deleting*, iff it is a subset of $<^{-1}$; this means every right side of a rule is smaller than the corresponding left side with respect to $<$. More general, a SRS is called *deleting* iff it is deleting for some precedence. Hofbauer and Waldmann have shown that all deleting SRSs preserve regularity.

The *bounded idempotency* relations, which are one of the origins of the work here were first defined in [8]. For fixed parameters $m$, $n$, and $k$ they are the rewrite relations

$$u \,{}^{\leq k}\!\!\bowtie^{n}_{m} v :\Longleftrightarrow \exists z[z \in \Sigma^+ \wedge u = u_1 z^m u_2 \wedge v = u_1 z^n u_2 \wedge |z| \leq k]$$

and the corresponding SRSs are $\{(z^m, z^n) : |z| \leq k\}$. We will denote it by the same symbol ${}^{\leq k}\!\!\bowtie^{n}_{m}$ ; no confusion should arise. A restricted version are the *uniformly bounded idempotency* relations

$$u \,{}^{= k}\!\!\bowtie^{n}_{m} v :\Longleftrightarrow \exists z[z \in \Sigma^+ \wedge u = u_1 z^m u_2 \wedge v = u_1 z^n u_2 \wedge |z| = k]$$

and the corresponding SRSs are $\{(z^m, z^n) : |z| = k\}$. We denote the languages generated by these relations from a word $w$ by

$w^{{}^{\leq k}\bowtie^n_m} := \{u : w({}^{\leq k}\bowtie^n_m)^* u\}$ and $w^{{}^{=k}\bowtie^n_m} := \{u : w({}^{=k}\bowtie^n_m)^* u\}$.

For a string-rewriting system $R$ and a language $L$ we denote the set of all descendants of words from $L$ modulo $R$ by $R^*(L)$ following Hofbauer and Waldmann [5]. In the case of idempotency relations, however, we will still use the established notation $L^{\bowtie^n_m}$ meaning the same as $(\bowtie^n_m)^*(L)$. A class of languages $\mathcal{C}$ is said to be closed under (rewriting by) a class $\mathcal{S}$ of SRSs, iff the following holds: $\forall L, R[L \in \mathcal{C} \wedge R \in \mathcal{S} \Rightarrow R^*(L) \in \mathcal{C}]$.

## 2 Uniformly Length-Bounded Systems

Idempotency relations without restrictions on their rules' lengths often generate very complicated structures. The relations with length bound are in general much more accessible, especially the ones with uniform length bound. The main reasons for this are that on the one hand they are closely related to periodicity and thus tools from that field can be used; on the other hand, the underlying SRSs are finite and thus more tractable. Here we will not only use periodicity as a tool, but we will define a new class of SRSs based on periodicity in their rules. These will include almost the entire class of uniformly length bounded idempotency relations. Thus regularity preservation of the latter will be implied by our results.

But first we recall a single non-closure result that follows directly from prior work on the idempotency closure of words [8].

**Proposition 2.1.** *String-rewriting systems* ${}^{=k}\bowtie^n_m$ *do not preserve regularity for* $k \geq 2$, $m = 0$, *and* $n \geq 2$.

In the course of this section, we will see that these are actually the only combinations of parameters, for which regularity is not preserved. Now we define a class of SRSs all of whose rules increase the length of factors with period $k$.

**Definition 2.2.** An SRS is called *k-period-expanding*, if for all of its rules $(\ell, r)$

(i) $\ell$ is non-empty,

(ii) $\ell, r \in w^*$ for a word $w$ of length $k$, and

(iii) $\ell \subset_{\text{pref}} r$.

Thus the left sides of all rules of a $k$-period-expanding SRS have period $k$ and the corresponding right sides add repetitions of that period — therefore the name. Now we establish an interesting property of this type of SRS.

**Lemma 2.3.** *k-period-expanding SRS are confluent.*

*Proof.* It is known that the diamond property implies confluence [1]. Therefore it suffices to show for $k$-period-expanding SRS that for every pair of derivation steps $w_1 \leftarrow u \rightarrow w_2$ there exists a word $v$ such that $w_1 \rightarrow v \leftarrow w_2$. So let two words $w_1$ and $w_2$ be direct successors of another word $u$ via such a $k$-period-expanding SRS $R$.

If the factors in $u$, where the rules are applied, do not overlap, then obviously in both cases the respectively other rule can be applied afterwards and one arrives at a common descendant $v$. So let two application sites $r^m$ and $s^i$ in $u$ for rules $r^m \rightarrow r^n$ and $s^i \rightarrow s^j$ overlap. Without loss of generality, let $r^m$ occur first from the left. If $s^i$ is completely inside of $r^m$, then $s$ and $r$ are conjugates as both have length $k$. The result of applying the rules in either order will be $r^{n+j-i}$.

If $s^i$ is not completely inside of $r^m$, then let us call $u'$ the factor from the start of $r^m$ till the end of $s^i$ such that $u = u_1u'u_2$ for some $u_1, u_2 \in \Sigma^*$. Now we can interpret the application of $r^m \rightarrow r^n$ as the insertion of $r^{n-m}$ just in front of $u'$; equally $s^i \rightarrow s^j$ amounts to the insertion of $s^{j-i}$ just after $u'$. Since application of these rules leaves $u'$ unchanged, the two derivations

$$u_1u'u_2 \rightarrow u_1r^{n-m}u'u_2 \rightarrow u_1r^{n-m}u's^{j-i}u_2$$

and

$$u_1u'u_2 \rightarrow u_1u's^{j-i}u_2 \rightarrow u_1r^{n-m}u's^{j-i}u_2$$

are possible, and the fact that they result in the same word with only two steps each concludes our proof. □

The proof shows even more than the lemma states: all rules can be applied from left to right, that is in an order such that the prefix left of an application site will never be altered by another rule. Thus in some sense the different rule applications are independent from each other. This will help us in showing that they preserve regularity.

**Proposition 2.4.** *$k$-period-expanding SRSs preserve regularity.*

*Proof.* Let $R$ be a $k$-period-expanding SRS. Let the longest left side of a rule from $R$ have length $km$. We will insert additional symbols from the alphabet $\Gamma := \{[w^i] : |w| = k \wedge i \leq m\} \cup \{\nabla\}$ into the words of a given language $L$. The $[w^i]$ will mark positions that are preceded by a factor $w^i$ in the original word. $\nabla$ is an auxiliary symbol, which is used to construct a deleting SRS $S$ that essentially simulates $R$. Since it is deleting it preserves regularity and thus $R^*(L)$ is regular if $L$ is.

First we describe informally the gsm mapping $g$, which introduces the symbols of $\Gamma$. Reading an input word from left to right, the gsm needs to remember at any given point the last $km$ letters of the input. If they have a suffix $w^i$, which is the left side of a rule from $R$, then the letter $[w^i]$ must be output; notice that there can be several such letters to output. After each $[w^i]$ an arbitrary number of $\nabla$ is written. Then the gsm advances and writes also the letter from $\Sigma$, which it reads, on the output.

Now we define the SRS that will work on the words produced by $g$. It simulates the rules from $R$ by inserting the newly produced symbols to the left of the corresponding symbols from $\Gamma$, and deleting, in some sense consuming one $\nabla$ in every step.

$$S := \{([w^i]\nabla, w^j[w^i]) : (w^i, w^{i+j}) \in R\}$$

This is a deleting SRS as for a precedence where $\nabla$ is greater than all the other symbols, since all the rules delete $\nabla$. Finally, to obtain $R^*(L)$ we need to delete all the symbols from $\Gamma$. This is done by the morphism

$$\delta := \begin{cases} x & \text{if } x \in \Sigma \\ \lambda & \text{if } x \in \Gamma. \end{cases}$$

Now we try to prove the inclusion $R^*(L) \subset \delta((S)^*(g(L)))$. Obviously $L = \delta(g(L))$. Further it should be clear that the rules from $S$ can simulate the rules from $R$ in the sense that if for some $w \in \Sigma^*$ we have $w \rightarrow_R w'$, then there is also $g(w) \rightarrow_S w''$ such that $w' = \delta(w'')$. So the first crucial fact here is that also further applications of rules to $w'$ can be simulated starting from $w''$; this is not obvious, because $g(w') \neq w''$. The difference is that the second word contains less symbols from $\Gamma$ since the rules from $S$ do not create these. Thus these are missing in the newly created factor. This factor and a preceding factor (to which the rule was applied) have period $k$.

The one problem here is if some periodic factor in the original word is not long enough to be the application side for a rule from $R$, but through application of shorter rules this one can become applicable. Then the corresponding symbol from $\Gamma$ is not there. See the following Example 2.5 for an illustration. In these cases an iteration of the process is necessary. In every iteration, $k$-periodic factors that allow rule applications are expanded as far as possible, in the next iteration longer rules will be applicable, too. Therefore the maximum number of iterations necessary is the number of different rules in $R$. To see this, observe that rule applications to a $k$-periodic factor can be ordered in such a way that first all applications of the rule with the shortest left side are done, then applications of the rule with the second shortest left-hand side etc. The first of these blocks of applications of the same rule will be possible in the first iteration, the second one in the second iteration and so forth. Thus we have

$$R^*(L) \subset \underbrace{\delta((S)^*(g(\ldots\delta((S)^*(g(}_{|R|\text{ times}}L)))\ldots))).$$

The inverse inclusion does not need further arguments. It is clear that rewriting the symbols of $\Gamma$ does not produce anything that is outside of $R^*(L)$ after the application of $\delta$.

In conclusion, we have shown the equality

$$R^*(L) = \underbrace{\delta((S)^*(g(\ldots\delta((S)^*(g(}_{|R|\text{ times}}L)))\ldots))),$$

and since all the finitely many operations on the right hand side preserve regularity this proves the proposition.

□

We now illustrate with an example, why so many iterations of the procedure can be necessary to fully simulate the original SRS.

**Example 2.5.** We consider the SRS $R = \{(a, a^6), (a^8, a^{15}), (a^{17}, a^{21})\}$ and the regular language $L = \{a\}$. Applying the construction from the proof of Proposition 2.4, in one iteration only a symbol for simulating the first rule is inserted, the resulting language $\delta(S^*(g(L)))$ is $\{a^{5i+1} : i \geq 0\}$. In a second iteration, symbols for the other two rules are inserted, too. However, in the word $a^{11}$ no symbol for the second rule is inserted, because the word is too short. Analysis of all possible derivations shows that therefore $a^{22} \notin \delta(S^*(g(\delta(S^*(g(L))))))$ although via $R$ the derivation $a \to a^6 \to a^{11} \to a^{18} \to a^{22}$ is possible. Thus for this three-rule system three iterations of the procedure are necessary.

Since a large class of uniformly bounded idempotency relations falls in the class of $k$-period-expanding SRSs, we obtain an immediate corollary.

**Corollary 2.6.** *String-rewriting systems* ${}^{=k}{\bowtie}^{n}_{m}$ *preserve regularity for* $k \geq 0$*,* $m > 0$*, and* $n \geq m$*.*

Looking at the SRS $S$ from the proof, we also see that the left sides of all rules consist of one letter of the form $[w^i]$ and one $\nabla$. If we simply delete all the $\nabla$ from the proof, the language generated is still the same, only $S$ is not deleting any more. Instead, now $S$ is context-free and this observation provides us with another closure property.

**Corollary 2.7.** *String-rewriting systems* ${}^{=k}{\bowtie}^{n}_{m}$ *preserve context-freeness for* $k \geq 0$*,* $m > 0$*, and* $n \geq m$*.*

Let us look a moment at the reason for the cases $m = 0$ not to be included here. The proof of Proposition 2.4 does not work, because rules with empty left side are applicable anywhere. Thus after every rule application another iteration of the process would be necessary, and there is no bound on this number. We now define a class of SRSs somewhat inverse to the $k$-period expanding ones, namely ones that reduce the length of periodic factors. Note that here right sides of length 0, i.e. deletions, are not excluded.

**Definition 2.8.** An SRS is called *$k$-period-reducing*, if for all of its rules $(\ell, r)$

(i) $\ell, r \in w^*$ for a word $w$ of length $k$ and

(ii) $r \subset_{\text{pref}} \ell$.

Also here, we can show that all systems of this class preserve regularity.

**Proposition 2.9.** *$k$-period-reducing SRSs preserve regularity.*

*Proof.* For a given regular language $L$ and a $k$-period-reducing SRSs $R$, we will define a context-free SRS $T$ such that $T^{-1}$ simulates $R$. Since the inverse context-free SRS $T^{-1}$ we construct is monadic, and since monadic SRSs preserve regularity, our claim follows.

First, we transform words from $\Sigma^+$ into a redundant representation, where every letter contains also the information about the $mk-1$ following ones, where $mk$ is the length the longest right side of a rule in $R$. This way, rewrite rules from $R$ can be simulated by ones with a right side of length only one, i.e. by inverse context-free ones.

First off we define the mapping $\phi : \Sigma^+ \mapsto ((\Sigma \cup \{\square\})^{mk})^+$ as follows. We delimit with $(\dots)$ letters from $(\Sigma \cup \{\square\})^{mk}$ and with $[\dots]$ factors of a word as usual. The image of a word $u$ is

$$u \mapsto (u[1\dots mk])(u[2\dots mk+1])\cdots(u[|u|-mk+1\dots|u|])\cdot \\ (u[|u|-mk+2\dots|u|]\square)\cdots(u[|u|]\square^{mk-1}).$$

Thus every letter contains also the information about the $mk$ following ones from the original word $u$. At the end of the word letters are filled up with the space symbol $\square$. $\phi$ is a gsm mapping and as such preserves regularity.

This encoding can be reversed by a letter-to-letter morphism $h$ defined as $h(x) := x[1]$ if $x[1] \in \Sigma$, for the other case we select for the sake of completeness some arbitrary letter $a$ and set $h(x) := a$ if $x[1] = \square$; the latter case will never occur in our context. It is clear that $h(\phi(u)) = u$ for words from $\Sigma^*$. Both mappings are extended to languages in the canonical way such that $\phi(L) := \{\phi(u) : u \in L\}$ and $h(L) := \{h(u) : u \in L\}$.

Now we define the string-rewriting system $T$ over the alphabet $(\Sigma \cup \{\square\})^{mk}$ as follows:

$$T := \{(u^i v v'), (\phi(u^j v)[1\dots|u^{j-i}|]) : (u^j, u^i) \in R \wedge |u^i v v'| = mk \wedge \\ u^i v \in \Sigma^+ \wedge v' \in \{\square\}^*\}.$$

A letter $(u^i v v')$ is replaced by the image of $u^j v$ under $\phi$ minus the suffix of letters that are already there in the image of $u^i v$. In this way, application of rules from $T$ keeps this space symbol only in the last letters of our words. If we have $w \overset{*}{\to}_R w'$, then clearly also $\phi(w') \to_T^* \phi(w)$ and thus $\phi(w) \overset{*}{\to}_{T^{-1}} \phi(w')$. This shows that $R^*(w) \subset h((T^{-1})^*\phi(w))$. For the inverse inclusion, let us take a look at what rules from $T^{-1}$ do. For any such rule $(\ell, r)$ we have $h(\ell) = u^{j-i}u[1]$ and $h(r) = u[1]$ for some rule $u^j \to u^i$ from $R$. Thus exactly the same subword is deleted. Further examination of the rules and their contexts show that also $w \overset{(u^j,u^i)}{\to} w'$ iff $\phi(w) \overset{(\ell,r)}{\to} \phi(w')$. Thus only images under $\phi$ of words in $R^*(w)$ can be reached. Example 2.10 following this proof will further illustrate this.

As all the rules of $T$ have left sides of length one and right sides of length greater than one, their inverses are all monadic, i.e. the system $T^{-1}$ is monadic. Monadic string-rewriting systems are known to preserve regularity, see for example the textbook by Book and Otto [1].

Summarizing, we can obtain $R^*(L)$ by a series of regularity-preserving operations in the following way:

$$R^*(L) = h((T^{-1})^*(\phi(L))).$$

$\square$

**Example 2.10.** Let $R$ be a 1-period reducing SRS which contains a rule $a^3 \to a^2$ and whose longest left-hand side of a rule is of length 4. Then the reduction $ba^3bc \to_R ba^2bc$ is possible. The SRS $T$ constructed in the proof of Proposition 2.9 has a rule $[a^2bc] \to [a^3b][a^2bc]$. The inverse is applicable to the word $\phi(ba^3bc) = [ba^3][a^3b][a^2bc][abc\square][bc\square\square][c\square\square\square]$ where it deletes the letter $[a^3b]$. The result is exactly $\phi(ba^2bc)$ and in this way the original rule is simulated. Here it is clearly visible how we can know from only looking at the letter $[a^2bc]$ that also the following two start with an $a$ and thus in the original word a rule with the left-hand side $a^3$ is applicable.

Once more, the consequences for idempotency systems are immediate.

**Corollary 2.11.** *String-rewriting systems* ${}^{=k}\bowtie^{n}_{m}$ *preserve regularity for* $k \geq 0$*,* $m \geq 0$*, and* $m > n$*.*

Further, we have stated already in the proof that $T$ is context-free, and thus it preserves context-freeness.

**Corollary 2.12.** *String-rewriting systems* ${}^{=k}\bowtie^{n}_{m}$ *preserve context-freeness for* $k \geq 0$*,* $m \geq 0$*, and* $m < n$*.*

*Proof.* Let $T$ be the string-rewriting system and $h$ and $\phi$ be the mappings from the proof of Proposition 2.9 constructed for ${}^{=k}\bowtie^{n}_{m}$. From the argumentation there we can see that

$$L^{{}^{=k}\bowtie^{n}_{m}} = h(T^*(\phi(L))).$$

Since $T$ is context-free and since this class of string-rewriting system preserves context-freeness, also $L^{{}^{=k}\bowtie^{n}_{m}}$ is context-free for the given combinations of parameters. $\square$

Now we can define a more general class of SRSs that can expand as well as reduce factors of period $k$ and we can show that also these preserve regularity and context-freeness.

**Definition 2.13.** Any union of finitely many $k$-period-expanding and $k$-period-reducing SRSs is called a *$k$-periodic* SRS.

**Proposition 2.14.** *$k$-periodic SRSs preserve regularity.*

*Proof.* We can combine the proofs for $k$-period-expanding and -reducing systems. First, observe that all factors of a word that have period $k$ can be considered independently in the following sense: application of a rule to one of them does not affect applicability of rules in other such factors. That is, in doing a reduction via $R$ we can look at the first such block of a word, apply all the rules that are to be applied there, then go to the second block etc.

Further, observe that such a block has the form $v^i v'$ for a word $v$ of length $k$ and a word $v'$ shorter than $k$. Application of any rule in this factor will only change the exponent $i$. Thus we can look at $k$-period-expanding rules as additions to the exponent, at $k$-period-reducing rules as subtractions.

Over integers, a series of additions and subtractions can be done in any order, the result is always the same. Since $v^+ v'$ can only represent non-negative integers, in our case we just have to be sure that the intermediary results are always non-negative; by doing first all the additions, then the subtractions, this is ensured. This shows us that we can reorder the application of rules in such a way that first all of the $k$-period-expanding ones are applied, then the $k$-period-reducing ones. Only the original order of the length-increasing rules must be preserved, because some long left side might be created only by earlier application of shorter ones.

We can partition a $k$-periodic SRS $R$ into two systems $R_\nearrow$ and $R_\searrow$, the first of which contains all the $k$-period-expanding rules while the second one contains all the $k$-period-reducing rules. Now we construct for $R_\nearrow$ the SRS $S$ from the proof of Proposition 2.4 and the corresponding mappings; for $R_\searrow$ we construct the SRS $T$ from the proof of Proposition 2.9 and the corresponding mappings. The considerations above then show us that

$$R^*(L) = h((T^{-1})^*(\phi(\underbrace{\delta((S)^*(g(\ldots\delta((S)^*(g(}_{|R_\nearrow| \text{ times}}L)))\ldots))))))$$

which proves the proposition's claim. □

In most cases, $k$-periodic SRSs will also preserve context-freeness. Only rules of the form $\lambda \to u$ must be excluded as can be seen from the various results in this above.

## 3 Outlook

When looking at the languages generated by idempotency relations without length bounds from single words, we see that almost all cases generate non-regular languages, Thus the questions treated here are not of great interest in that context. However, when the size of the alphabet is limited to two or even one letter, the picture changes [7, 9]. The reason for this is mainly that often –as in the case of duplication– there exists a finite SRS, which is equivalent to the infinite one. For these cases also the closure of regular languages under the respective SRSs is an interesting question. The cases of $\bowtie_0^1$ and $\bowtie_1^0$, however, are regular over any alphabet size as shown in work summarized by Ito [6].

## References

[1] R. Book and F. Otto: *String-Rewriting Systems*. Springer, Berlin, 1993.

[2] D.P. Bovet and S. Varricchio: *On the Regularity of Languages on a Binary Alphabet Generated by Copying Systems*. In: Information Processing Letters 44, 1992, pp. 119–123.

[3] J. Dassow, V. Mitrana and Gh. Păun: *On the Regularity of Duplication Closure*. In: Bull. EATCS 69, 1999, pp.133–136.

[4] A. Ehrenfeucht and G. Rozenberg: *On Regularity of Languages Generated by Copying Systems*. In: Discrete Applied Mathematics 8, 1984, pp. 313–317.

[5] D. Hofbauer and J. Waldmann: *Deleting String-Rewriting Systems preserve regularity*. In: Theoretical Computer Science 327, 2004, pp. 301-317.

[6] M. Ito: *Algebraic Theory of Automata and Languages*. World Scientific, New Jersey, 2004.

[7] M. Ito, P. Leupold, and K. Shikishima-Tsuji: *Closure of Language Classes under Bounded Duplication*. In: Lecture Notes in Computer Science 4036, DLT 2006, Springer, Berlin, pp. 238–247.

[8] P. Leupold: *Languages Generated by Iterated Idempotencies*. In: Theoretical Computer Science 370(1-3), 2007, pp. 170–185.

[9] P. Leupold: *General Idempotency Languages over Small Alphabets*. Accepted for publication in Journal of Automata, Languages and Combinatorics.