Available online at www.sciencedirect.com

ScienceDirect

Theoretical Computer Science 370 (2007) 170–185

Theoretical Computer Science

www.elsevier.com/locate/tcs

Languages generated by iterated idempotency

Peter Leupold

Research Group in Mathematical Linguistics, Rovira i Virgili University, Pça. Imperial Tàrraco 1, 43005 Tarragona, Spain

Received 7 February 2006; received in revised form 4 July 2006; accepted 25 October 2006

Communicated by M. Ito

Abstract

Duplication languages are generated from an initial word by iterated application of string-rewriting rules of the form $u \to uu$. In several recent articles such languages have been investigated with a main focus on finding their placement in the Chomsky hierarchy. We generalize the generating rules to $u^m \to u^n$ with arbitrary m and n.

When the length of the factor u is a fixed number, most cases result in regular languages. If there is just some bound on the length, then often non-regular but always context-free languages are generated. The regularity conditions for both variants are fully characterized, and confluence for the underlying rewrite relations is determined. For the unrestricted case only some results are presented which carry over from restricted variants.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Formal languages; Idempotency; String-rewriting

0. Idempotency and duplication

In a series of recent articles, languages generated by iteration of the duplication operation have been investigated. This operation was inspired by a behaviour observed in strands of DNA: certain factors of such sequences can be duplicated within their strand forming a so-called tandem repeat; from a formal language point of view, a word uvw is transformed into $uvvw$.

This behaviour first inspired so-called duplication grammars [20,21]. Then a great deal of interest was paid to languages generated from a word by iterated application of the duplication operation in the form of rewriting rules $u \to uu$ acting on factors [8,24]. In this context also the restriction of the duplicated factors' length to a maximum or to one fixed length have been investigated [17,15,14,16]. The main focus in all this work has been on determining whether the languages generated are regular or not.

From an algebraic point of view, the basic feature underlying duplication is the idempotency $u \equiv u^2$, however read only from left to right. The first and second power on the left and right hand side respectively are motivated by the duplications observed in DNA strands. However, from a purely mathematical standpoint there is no reason to restrict our attention only to this special case. Starting out from this thought we will investigate the languages generated from one word by iterated application of general idempotency rules $u^m \equiv u^n$ for arbitrary integers m and n; a rule here is the interpretation of $u^m \equiv u^n$ as a string-rewriting rule $u^m \to u^n$.

E-mail address: klauspeter.leupold@urv.cat.

0304-3975/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2006.10.021

Idempotencies have already received a great deal of interest through a problem stated by Burnside in 1902 [6]: Is every group finite, which satisfies the identity $x^r = 1$ and has a finite set of generators? Himself he gave a positive answer for $r \in \{1, 2, 3\}$. Since then many cases have been solved, others remain open. For odd $r \geq 665$, for example, the group generated is infinite as shown by Adian [1]. In the 1980s and 1990s interest in the topic flared up again, a nice overview of the history and the results obtained in that period has been given by Dershowitz [9].

A first direct connection to formal languages was established by the so-called non-counting classes as presented by Brzozowski [5]. These are the congruence classes of the non-counting relations, and the main question was whether they are regular. A related chapter can be found in Lothaire's book on combinatorics of words [18]. The main result presented there is the theorem of Green and Rees [11], which states that the relations $ww \equiv w$ for a finite alphabet always have a finite number of equivalence classes. This is contrasted to the fact that in general every square-free word defines a separate equivalence class for $w^m \equiv w^n$ for $m, n \geq 2$. Thus over at least three letters these relations have an infinite number of equivalence classes.

We will now investigate the languages generated by general idempotency rules in the way described above, and in this we will follow very much the lines of the research on duplication languages. Foremost we will try to establish the placement of these languages within the Chomsky hierarchy. After compiling some existing results about special cases such as duplication and insertion in Section 2, we will first investigate the most tightly restricted case, uniformly bounded idempotency, in Section 3. The requirement that all rewritten factors have the same length results in confluent rewriting systems and regular languages in many cases. The conditions under which this is the case are fully characterized.

Imposing just an upper bound on this length results in non-regularity already in cases like the duplication one as we will see in Section 4. Here also the cardinality of the alphabet plays an important role, and frequently the rewriting systems are not confluent any more. If we drop also this bound and allow arbitrary idempotency rules, things become significantly more complicated, and much less is known about these cases. The problem that has received most attention in this context, the question of whether general duplication languages are context-free or not, remains open. What is known we will summarize in Section 5.

1. Preliminaries

To start with, we recall some standard notions concerning formal languages. For everything not defined here we refer the reader to the standard textbooks on the topic by Salomaa [23] or Harrison [12]. In particular, we will denote by REG, LIN, and CF the classes of regular, linear, and context-free languages respectively.

One notion that will be used repeatedly is the relation $\sim_L$ over $\Sigma^* \times \Sigma^*$ for a language $L \subset \Sigma^*$, which is called the *syntactic right congruence* and is defined as follows:

$$u \sim_L v :\leftrightarrow \forall w \in \Sigma^*(uw \in L \leftrightarrow vw \in L).$$

This is obviously an equivalence relation. A variant of the well-known theorem of Myhill states that a language L is regular, if and only if the corresponding relation $\sim_L$ has a finite number of equivalence classes; this number is called the *index* of $\sim_L$.

Theorem 1. *A language L is regular, if and only if $\sim_L$ has finite index.*

The most important notions from the field of combinatorics of words will concern periodicity. Let $w[i]$ denote the i-th letter of a word w for $1 \leq i \leq |w|$, where $|w|$ is w's length. The notation $w[i \dots j]$ is used to refer the part of a word starting at the i-th position and ending at the j-th position. A word w has a positive integer k as a *period*, if for all i, j such that $i \equiv j (\bmod k)$ we have $w[i] = w[j]$, if both $w[i]$ and $w[j]$ are defined. Two words u and v are *conjugates*, if there exists a factorization $u = u_1u_2$ such that $v = u_2u_1$. A word of the form $uu = u^2$ is called a *square*, u^3 is a *cube*.

A word u is a *prefix* of w if there exists an $i \leq |w|$ such that $u = w[1 \dots i]$; if $i < |w|$, then the prefix is called *proper*. A *suffix* is a word u such that $u = w[i..|w|]$, and a *factor* is any word such that there exist i and j such that $u = w[i..j]$. A *scattered subword* of w, in contrast, is a word u for which there exist integers $i_1 < i_2 \cdots < i_{|u|}$ such that for all $j \in \{1, 2 \dots |u|\}$ there is $u[j] = w[i_j]$. For further details on these concepts the reader should consult the standard textbooks by Lothaire [18,19].

Now we define the central notion of this article: idempotency relations and the languages that are generated by them.

Definition 2. For an alphabet Σ and two natural numbers m and n the relation $\bowtie_m^n$ over $\Sigma^* \times \Sigma^*$ is defined as

$$u \bowtie_m^n v :\Leftrightarrow \exists w[w \in \Sigma^+ \wedge u = u_1 w^m u_2 \wedge v = u_1 w^n u_2].$$

With $(\bowtie_m^n)^*$ we denote the relation's reflexive and transitive closure and define the language it generates from a given word w as

$$w^{\bowtie_m^n} := \{u : w(\bowtie_m^n)^* u\}.$$

If the factor, whose number of occurrences is changed, is bounded in length or required to have a certain length k, then the corresponding relations are denoted by ${}^{\leq k}\bowtie_m^n$ and ${}^{=k}\bowtie_m^n$, formally defined as

$$u^{\leq k}\bowtie_m^n v :\Leftrightarrow \exists w[w \in \Sigma^+ \wedge u = u_1 w^m u_2 \wedge v = u_1 w^n u_2 \wedge |w| \leq k] \text{ and}$$
$$u^{=k}\bowtie_m^n v :\Leftrightarrow \exists w[w \in \Sigma^+ \wedge u = u_1 w^m u_2 \wedge v = u_1 w^n u_2 \wedge |w| = k].$$

The languages generated we denote by $w^{\leq k \bowtie_m^n}$ and $w^{=k \bowtie_m^n}$.

A few simple examples shall give a first taste of what these definitions result in. We will not prove their correctness here, though — this might be a nice exercise to become familiar with the way the idempotency rules in question work.

Example 3. Over two letters, duplications can generate just about any factor in any place as the example $(aba)^{\bowtie_1^2} = a\{a,b\}^* b\{a,b\}^* a$ shows. In the case of $(abcbcbab)^{=2\bowtie_2^4} = a(bcbc)^+ bab$ the rules can be applied only on square factors, and in $abcbcbab$ there are only two, which overlap and are even conjugates; thus only one of them needs to be considered and the language generated consists simply of the words reached by iterated catenation of this factor.

For length-reducing rules the languages generated are, of course, finite, like in the case of $(abcbabcbc)^{\bowtie_2^1} = \{abc, abcbc, abcbabc, abcbabcbc\}$; here in a first step either the prefix $(abcb)^2$ or the suffix $(bc)^2$ can be reduced, only the former case results in a word with another square, which can be reduced to abc. This example already shows that one word can in general be reduced to more than one irreducible word, i.e. the reduction is not converging towards a unique endpoint.

A first, rather obvious consequence of these definitions is that the languages defined by idempotencies not reducing the factor's length are always contained in the corresponding duplication languages.

Proposition 4. *For integers $k, m > 0$ and $n \geq m$, a word w and a condition $c \in \{\lambda, \leq k, = k\}$ we always have $w^{c\bowtie_m^n} \subseteq w^{c\bowtie_1^2}$.*

Quite frequently we will view such idempotency relations as string-rewriting systems in the sense of Book and Otto [3]. For ease of notation we will also write $u \to v$ instead of for example $u^{=k} \bowtie_m^n v$, if the three parameters of the idempotency relation are clear from the context. We now recall the concepts from rewriting theory that will be most important in our context.

Definition 5. A string-rewriting system R on Σ is 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 more simply $\to$, and $\overset{*}{\to}$ denotes the relation's reflexive and transitive closure.

Such a relation (reduction) $\to$ is called *confluent*, iff for all $w, w_1, w_2 \in \Sigma^*$ always $w_1 \overset{*}{\leftarrow} w \overset{*}{\to} w_2$ implies the existence of some w' such that $w_1 \overset{*}{\to} w' \overset{*}{\leftarrow} w_2$. Here we will use $v \leftarrow u$ as another notation for $u \to v$, where it is more convenient. Further, $\to$ is *noetherian* (also called *terminating*), iff there is no infinite sequence $u_0, u_1, \ldots$ such that $u_i \to u_{i+1}$ for all $i \geq 0$. The relation is *convergent* iff it is both confluent and noetherian.

A string w is *irreducible* iff there is no rule $(\ell, r) \in R$ such that ℓ is a factor of w, i.e. no rule can be applied on w. An irreducible string v such that $u \overset{*}{\to} v$ is called a *normal form* of u.

So obviously a string-rewriting system defined by an idempotency relation $\bowtie_m^n$ is noetherian, iff $n \leq m$. All other cases result in non-convergent systems though in many cases they will be confluent.

2. Known results about special cases

As mentioned already in the introduction, the most intensively investigated case of idempotency-generated languages so far seems to be the duplication closure, i.e. the case of languages generated by rules $u \rightarrow u^2$. First we present two regular cases.

Proposition 6 (*[15]*). *For every word w and integer $k \geq 0$ the language $w^{\overset{=k}{\bowtie}{}^2_1}$ is regular.*

Proposition 7 (*[8]*). *For every word $w \in \{a, b\}^*$ the language $w^{\bowtie^2_1}$ is regular.*

Over an alphabet of more than two letters we can get beyond regularity in the general and even in most length-bounded cases.

Proposition 8 (*[17]*). *For every integer $k \geq 4$ the language $abc^{\overset{\leq k}{\bowtie}{}^2_1}$ is not regular.*

These cases of non-regularity were shown by refinements in the proof techniques used for obtaining the chronologically first result of this kind.

Proposition 9 (*[24]*). *The language $abc^{\bowtie^2_1}$ is not regular.*

These results raise the question of an upper bound for the languages generated by bounded and general duplication. In the bounded case, context-freeness of the languages generated has been proved, in the general case it remains an open problem.

Proposition 10 (*[17]*). *For every every word w and integer $k \geq 0$ the language $w^{\overset{\leq k}{\bowtie}{}^2_1}$ is context-free.*

It must be mentioned here that some of these results were already obtained earlier in investigations dealing with so called copy systems. Obviously the work on duplication has so far been done without any knowledge of this field. These copy systems are actually defined in exactly the same way as our idempotency languages for $\bowtie^2_1$, only the symbol for the relation differs. Ehrenfeucht and Rozenberg wrote the initial article on copy systems [10] and proved a result implying Proposition 9. In a following article [4], Bovet and Varricchio did the same for Proposition 7.

Another special case of idempotency languages is that of arbitrary insertion or deletion of factors, which correspond to the relations $\bowtie^1_0$ and $\bowtie^0_1$, respectively. These have been investigated under the names of 1-insertion and deletion and a compilation of the results obtained can be found in the book by Ito [13]. The most interesting ones are the following.

Proposition 11. *For every regular language L, the language $\bigcup_{w \in L} w^{\bowtie^1_0}$ is regular.*

Proposition 12. *For every regular language L, the language $\bigcup_{w \in L} w^{\bowtie^0_1}$ is regular.*

From Proposition 11 we immediately obtain the regularity of languages $w^{\bowtie^1_0}$; Proposition 12 states implicitly the regularity of $w^{\bowtie^0_1}$, which is, of course, trivial, because these languages are even finite.

Here we also want to mention that in the field of of DNA computation similar mechanisms have been investigated under the name of Insertion–Deletion systems [22]. Using only insertion or only deletion also here amounts to applying an idempotency rule. However, while some variants without any deletion operations were considered, always context-sensitive insertion has been the focus of attention. Therefore it seems that all existing results cannot help in our context.

3. Uniformly bounded idempotency

The first variant of idempotency-generated languages we will deal with is the one where the idempotencies are most strongly restricted: all words defining idempotency rules must have the same length. This implies serious limitations for the languages generated; for example, their words can only be of certain lengths: the language $ababa^{\overset{=2}{\bowtie}{}^5_2} = ababa((ba)^3)^*$, for example, consists only of words of lengths $5 + 6i$ for integers i.

If we deal with words over an alphabet of only one letter, then, as one might expect, this strict restriction results in the languages generated being rather simple, namely ultimately periodic and therefore regular. This result is implied by the later one on two-letters; we still prove it explicitly, because the proof is easier in this case and it provides us with a concrete expression for the language generated.

Proposition 13. *Over a one-letter alphabet $\{a\}$ for every nonempty word w and integers $k, m, n \geq 0$ the language $w^{=k\bowtie_m^n}$ is regular.*

Proof. If $m \geq n$, then the language generated is finite and thus also regular. For $m < n$ there exists only one possible rewrite rule, namely $(a^k)^m \to (a^k)^n$, and with every application exactly $k \cdot (n-m)$ copies of the letter a are inserted. The place of application does not matter since catenation is commutative over just one letter. Thus $w^{=k\bowtie_m^n} = w(a^{k\cdot(n-m)})^*$. $\square$

While in most cases also for bigger alphabets the languages generated remain regular, the proofs of this will be somewhat more involved. For the rest of this section we will assume an alphabet Σ containing at least two letters. It is still rather easy to see that insertion of arbitrary words generates only regular languages, see also Proposition 11, where, however, unrestricted insertion is treated.

Proposition 14. *For every word w and an integer $k \geq 0$ the language $w^{=k\bowtie_0^1}$ is regular.*

Proof. In this case, at any point arbitrary words from the set Σ^k can be inserted into the original word. Thus the language generated is described by the regular expression $\phi := (\Sigma^k)^* w[1](\Sigma^k)^* w[2](\Sigma^k)^* \ldots (\Sigma^k)^* w[|w|](\Sigma^k)^*$.

The only consideration necessary to see this is the following: let some word $u = u_1u_2$ be inserted, and later a second one v between the two factors u_1 and u_2; choose the factorization v_1v_2 of v for which $|u_1v_1| = |v_2u_2| = k$. Then the same word would have been reached by first inserting u_1v_1, and then v_2u_2 just behind it. Thus insertions of one factor inside another need not be considered and catenation of factors from Σ^n in the way described in ϕ suffices to generate the entire language $w^{=k\bowtie_0^1}$. $\square$

When n becomes greater than 1, instead of arbitrary words we insert words which already have some internal structure, namely they are squares, cubes etc., i.e. they are always non-primitive. Then the insertion cannot be replaced by simple catenation and we obtain also non-regular languages.

Example 15. Let $L \subset \{a, b\}^*$ be the language generated from λ by insertion of squares of words of length 2, i.e. $L = \lambda^{=2\bowtie_0^2}$. Then we show that $L \cap (bbaa)^+(aabb)^+ = \{(bbaa)^n(aabb)^n : n \geq 0\}$, and this language is clearly not regular.

Every word in $\{(bbaa)^n(aabb)^n : n \geq 0\}$ can be generated from λ by first putting b^4, then a^4 in its center, and so on.

On the other hand, every word in $(bbaa)^+(aabb)^+$ and therefore also every word in $L \cap (bbaa)^+(aabb)^+$ contains only one square of a word of length 2, namely the a^4 in the center. Removing it, b^4 forms a unique such square. Thus a reduction to λ is possible only if the numbers of $bbaa$ and $aabb$ correspond, and this shows that all words in this intersection must belong to the set $\{(bbaa)^n(aabb)^n : n \geq 0\}$.

This example does not represent some special case, rather non-regularity always holds over an alphabet of at least two letters, more precisely speaking the languages generated are not even linear.

Proposition 16. *For every word w and integers $k \geq 2$, and $n \geq 2$ the language $w^{=k\bowtie_0^n}$ is not linear but context-free.*

Proof. Analogously to the language obtained by intersection in Example 15 we can always filter out a non-linear component over two letters. So for an arbitrary relation $=k\bowtie_0^n$ let us consider the language

$$\lambda^{=k\bowtie_0^n} \cap (b^2a^{nk-1})^+(ab^{nk-2})^+(ba^{nk-1})^+(ab^{nk-1})^+$$

obtained by intersection of $\lambda^{=k\bowtie_0^n}$ with a regular language.[1] This results in the non-linear

$$L = \{(b^2a^{nk-1})^i(ab^{nk-2})^i(ba^{nk-1})^+j(ab^{nk-1})^j : i, j \geq 0\}.$$

The reasoning for seeing this is the same as in Example 15. Clearly L is a subset of the intersection by derivations

$$\lambda \to b^{nk} \to b^2a^{nk}b^{nk-2} \to b^2a^{nk-1}b^{nk}ab^{nk-2} \to \cdots$$

for the first component, and by an analogous derivation for the second component.

[1] Thanks are due to an anonymous referee for suggesting this regular language.

To see that the language obtained by intersection is contained in L, we observe that all words in $(b^2a^{nk-1})^+(ab^{nk-2})^+(ba^{nk-1})^+(ab^{nk-1})^+$ are in the intersection, iff they have λ as a normal form under the relation ${}^{=k}\bowtie^0_n$. For obtaining the normal form of any word in $(b^2a^{nk-1})^+(ab^{nk-2})^+(ba^{nk-1})^+(ab^{nk-1})^+$ the only applicable rule is $a^nk \to \lambda$, which is applicable on two sites. Application at either site creates b^nk there and applying $b^nk \to \lambda$ takes us back into the original language. At no stage is any rule transgressing the border between the two first and two last iterations possible. So the reduction goes independently in both components, and the word can only be reduced to λ if the exponents are as in L.

Now we show the inclusion of languages $w^{{}^{=k}\bowtie^n_0}$ in CF by sketching the construction of a context-free grammar generating $w^{{}^{=k}\bowtie^n_0}$ for some word w and non-negative integers k and n. It has only two non-terminals S and T. For the start symbol S, there is the unique rule $(S, Tw[1]Tw[2]T \dots Tw[|w|]T)$. The rest of the rules consist of the set $\{(T, T(x_1Tx_2T \dots Tx_kT)^n : x_1, x_2, \dots, x_k \in \Sigma)\}$ and the deleting rule (T, λ). It should be rather obvious that this grammar generates exactly the desired language with the ubiquitous T permitting insertion at any position, while it can be deleted whenever no further insertions occur. □

Before we investigate the remaining cases, we will now state a useful property of uniformly bounded idempotency languages. It simplifies the construction of regular expressions for such a language and thus will be used implicitly further down.

Lemma 17. *Let $k, m, n > 0$ with $n \geq m$ and let the word $w \in \Sigma^*$ have period k. Then $w^{{}^{=k}\bowtie^n_m} = w[1 \dots |w| - k](w[|w| - k + 1 \dots |w|]^{n-m})^+$.*

Proof. We prove the claim by induction on the number of rewrite rules that have been applied to obtain a word in $w^{{}^{=k}\bowtie^n_m}$. Clearly the induction basis $w \in w[1 \dots |w| - k](w[|w| - k + 1 \dots |w|]^{n-m})^+$ holds. So let $w_1 \; {}^{=k}\bowtie^n_m \; w_2$ with $w_1 \in w[1 \dots |w| - k](w[|w| - k + 1 \dots |w|]^{n-m})^+$. Then w_2 can be obtained from w_1 by application of one idempotency rule on a factor v^m of w_1 with $|v| = k$. So v has period k. Therefore the period k of the word w_1 is preserved, and of course the last k letters of w_1 also remain unchanged. Thus we have $w_2 \in w[1 \dots |w| - k](w[|w| - k + 1 \dots |w|]^{n-m})^+$. Together with trivial length considerations for the exponent $(n - m)$ this suffices to prove the claim. □

Lemma 18. *For $k, m, n \geq 0$ with $n \geq m$ the relation ${}^{=k}\bowtie^n_m$ is confluent.*

Proof. It is known that the diamond property implies confluence [2]. Therefore it suffices to show that this property $w_1 \leftarrow u \rightarrow w_2 \Rightarrow \exists v(w_1 \rightarrow v \leftarrow w_2)$ holds for the relation ${}^{=k}\bowtie^n_m$. So let two words w_1 and w_2 be direct successors of another word u.

If the factors in u, where the rules are applied, do not overlap, then obviously in both cases the other respective rule can be applied afterwards and one arrives at a common v. So let two application sites r^m and s^m overlap in u. Without restriction of generality let r^m occur first from the left, and call u' the factor from the start of r^m till the end of s^m such that $u = u_1u'u_2$ for some $u_1, u_2 \in \Sigma^*$.

Now we can interpret the application of $r^m \to r^n$ as the insertion of r^{n-m} just in front of u'; equally $s^m \to s^n$ amounts to the insertion of s^{n-m} just after u'. Since application of these rules leaves u' unchanged, the two derivations

$$u_1u'u_2 \to u_1r^{n-m}u'u_2 \to u_1r^{n-m}u's^{n-m}u_2$$

and

$$u_1u'u_2 \to u_1u's^{n-m}u_2 \to u_1r^{n-m}u's^{n-m}u_2$$

are possible, and the fact that they result in the same word concludes our proof. □

So all the length-increasing variants are confluent. For length-reducing rules, however, this is true only in some cases.

Lemma 19. *For $k \geq 2$ the relation ${}^{=k}\bowtie^0_1$ is not confluent.*

Proof. Let w be a word of length $k + 1$. Then the parameters of the relation allow the application of a rewrite rule exactly on two sites: w's prefix and suffix of length k; these will leave the last, respectively the first, letter of w as an irreducible remainder, and these are in general not equal. □

Lemma 20. *For $k \geq 2$, $m > n$, and $n \geq 1$ the relation ${}^{=k}{\bowtie}^n_m$ is confluent.*

Proof. As in the proof of Lemma 18 it suffices to show that the diamond property holds, i.e. $w_1 \leftarrow u \rightarrow w_2 \Rightarrow \exists v(w_1 \rightarrow v \leftarrow w_2)$ for the relation ${}^{=k}{\bowtie}^n_m$.

Since $m > n$, rewrite rules reduce repetitive factors to ones of lower repetitiveness but at least one copy of the repeated word of length k remains, because $n \geq 1$. Therefore the diamond property holds obviously, if the application sites of two rewrite rules do not overlap by more than k symbols.

If, on the other hand, there are two powers of order m overlapping in more than k symbols, then the entire sequence has period k, and thus the application of either rule results in the same word, thus already $w_1 = w_2$. □

Now we are able to fully characterize the conditions under which uniformly bounded idempotency relations are confluent. Lemmata 18–20 leave open only the cases where $k = 1$ and $k = 0$. But for these cases confluence is obvious for any m and n.

Proposition 21. *The relation ${}^{=k}{\bowtie}^n_m$ is confluent except for the case where $k \geq 2$, $m = 1$, and $n = 0$.*

These results will help us in proving the regularity of a big family of cases.

Proposition 22. *For every nonempty word w, integers $k, n \geq 0$, and $m \geq 1$ the language $w^{{}^{=k}{\bowtie}^n_m}$ is regular.*

Proof. Because different parameters can result in a quite different behaviour of the relations ${}^{=k}{\bowtie}^n_m$, we distinguish several cases.
Case 1: $m \geq 1$ *and* $n \leq m$. The relation is length-reducing or the identity, the resulting language is obviously finite and therefore regular.
Case 2: $n = 2, m = 1$. This is the special case of uniformly bounded duplication and is therefore covered by Proposition 6.
Case 3: $n > 2, m = 1$. The crucial fact to note is that the applications of the idempotency rules can be done strictly from left to right; i.e. it can be done in a way such that at most the last k positions produced in the last step are affected in the following one. To see this it suffices to recall that according to Proposition 21 the relation is confluent here, and as shown in the proposition's proof it even fulfills the diamond property.

This implies that every word $u \in w^{{}^{=k}{\bowtie}^n_m}$ can be constructed by successive applications of idempotency rules in such a way that at any stage it can be factored as rst, where rs is already a prefix of u, s will be replaced by s^n in the next step, and st is a suffix of the original word w. This tells us that for any prefix u' of a word in $w^{{}^{=k}{\bowtie}^n_m}$ there exists a word v, which is a suffix of w such that $u'v \in w^{{}^{=k}{\bowtie}^n_m}$. It remains to show that this allows us to give a bound for the number of equivalence classes of the syntactic right congruence $\sim$ for the language $w^{{}^{=k}{\bowtie}^n_m}$.

All words u such that there exists no v such that $uv \in w^{{}^{=k}{\bowtie}^n_m}$ constitute one such class C. Now let such a factor v exist, i.e. u is a prefix of a word in the language. As shown above, this word can be constructed from left to right.

This means that there exists a word v fulfilling the above property, which is at the same time a suffix of w except for maybe its first $(n - m)k - 1$ letters produced in the last application of an idempotency rule. Of course, there are only finitely many suffixes of w and only finitely many words of length $(n - m)k - 1$. As the possible right contexts of all equivalence classes of $\sim$ (except for C) have to contain at least one such suffix, their number is bounded exponentially by the number of suffixes of w and the number $(n-m)k-1$, to be more exact, $|\Sigma|^{|w|+(n-m)k-1}$ is a bound. Therefore the syntactic right congruence is of finite index and by Theorem 1 the language $w^{{}^{=k}{\bowtie}^n_m}$ is regular.
Case 4: $n > m, m = 2$. First let us look at the rules $u^2 \rightarrow u^n$ as insertions of u^{n-2} between the two original occurrences of u. This illustrates that idempotency rules affecting factors overlapping by no more than k symbols can be looked at independently. Further, note that due to the fixed length of such words u, every border between letters in the original word w can be the center of at most one relevant factor uu.

Now we construct the regular expression R from w as follows. Going from left to right, every square uu with $|u| = k$ is replaced by $u(u^{n-2})^*u$. Clearly the language described by R is a subset of $w^{{}^{=k}{\bowtie}^n_m}$. However, two squares of length $2k$ overlapping in more than k letters might allow applications of idempotency rules in ways not described by this expression.

To see that this is not the case, we first notice the fact that two such factors uu and vv overlapping in more than k letters imply that u and v are conjugates, because v is an internal factor of uu. This means that the entire factor

of w spanning these two squares has period k. Therefore it does not matter, whether u^{n-2} or v^{n-2} is inserted at the respective place, the result is the same, see also Lemma 17. Thus this case is described by R, too, and consequently the language $w^{=k \bowtie^n_m}$ is described exactly and is therefore regular.
Case 5: $n > m, m > 2$. Essentially the same reasoning as in case 4 applies. □

4. Bounded idempotency

Compared to uniformly bounded rules, general length-bounded rules allow many more possibilities, namely to have the application site for one inside the site for another rule. This feature makes the languages generated non-regular in many cases. However, as in the preceding section we will first off treat a few cases, whose regularity is rather easy to prove.

Proposition 23. *Over a one-letter alphabet* $\{a\}$ *for every nonempty word* w *and integers* $k, m, n \geq 0$ *the language* $w^{\leq k \bowtie^n_m}$ *is regular.*

Proof. With a reasoning very much along the lines of the proof of Proposition 13 we can see that for $m < n$

$$w^{\leq k \bowtie^n_m} = w(a^{(n-m)})^*(a^{2\cdot(n-m)})^* \cdots (a^{k\cdot(n-m)})^*. \quad \square$$

For a greater alphabet the language generated is also regular, if we look at the insertion of words with no inner structure u^n for $n \geq 2$.

Proposition 24. *For every word* w *and integer* $k \geq 0$ *the language* $w^{\leq k \bowtie^1_0}$ *is regular, and further* $w^{\leq k \bowtie^1_0} = w^{\leq 1 \bowtie^1_0}$ *for* $k \geq 1$.

Proof. The case of $k = 0$ is trivial. For greater k, insertions of length one, i.e. of single letters are always possible at any position, and between the letters of the original word any word can be generated. Thus any word in $w^{\leq k \bowtie^1_0}$ can be generated by insertions of length only one and the resulting language consists exactly of all the words having w as a scattered subword — a condition that can easily be checked by a finite automaton. □

Along quite similar lines as originally used by Wang for unbounded duplication [24] we will now prove that for many relations the languages generated are not regular. For this we will use some concepts from pattern avoidance. There a word is called *n-free*, if it does not contain any repetition of order n, i.e. no factor u^n, where u is a non-empty word and n a natural number. 2-free is also called *square-free*. Further, a word is called n^+*-free*, if it does not contain any repetition of order greater than n; for example, being 2^+-free means containing no factor $auaua$ for a letter a and a (possibly empty) word u. The two fundamental results we will use are the well-known facts that over two letters there exists no infinite square-free word, but a 2^+-free one, and that over three letters there is an infinite square-free word [18].

Proposition 25. *Over an alphabet of three letters, for every word* w *and integers* $k \geq 1$ *and* $n \geq 2$ *the language* $w^{\leq k \bowtie^n_0}$ *is not regular.*

Proof. We prove that $\lambda^{\leq k \bowtie^n_0}$ is not regular. First we show that for every square-free word u there exists a word v such that $uv \in \lambda^{\leq k \bowtie^n_0}$. It is rather straight-forward to construct u letter by letter, while v will consist of all the letters produced, which do not form part of u. We start with λ and first insert $u[1]^n$, then after the first letter of u insert $u[2]^n$ etc. In this v takes up all the letters not needed for u, but which are produced by the rules. By this method we obtain an upper bound on the length of the smallest such v, namely $|v| \leq |u|(n-1)$, because exactly $n-1$ letters of v are produced in every step.

Now we establish a lower bound on the length of words v such that $uv \in \lambda^{\leq k \bowtie^n_0}$. Since u is square-free, every insertion can produce at most $2k-1$ symbols of it, otherwise there would be a square in u. It is also impossible for letters after the $(n-1)$-st position to become part of u later by insertions in front of them: then these would leave a square within u. So still in the optimal case of always producing $2k-1$ letters of u in every step, we have that $|v| \geq \frac{|u|}{2k-1}((n-2)k+1)$.

Summarizing, for every square-free word u there exists a word v such that $uv \in \lambda^{\leq k \bowtie^n_0}$, and for the shortest such v we have $\frac{|u|}{2k-1}((n-2)k+1) \leq |v| \leq |u|(n-1)$, where the lower bound is optimal. Now, over three letters there exists an infinite square-free word. Let $u_1, u_2, u_3 \ldots$ be a sequence of prefixes of such a word with $\frac{|u_{i+1}|}{2k-1}((n-2)k+1) > |u_i|(n-1)$ and let v_i be the shortest word such that $u_i v_i \in \lambda^{\leq k \bowtie_0^n}$ for all $i \geq 1$. Then clearly $u_j v_i \notin \lambda^{\leq k \bowtie_0^n}$ for all $j > i$. This means that the equivalence classes of the u_i in the syntactical congruence of $\lambda^{\leq k \bowtie_0^n}$ are pairwise different, so there is an infinite number of such classes. According to Theorem 1 the language $\lambda^{\leq k \bowtie_0^n}$ cannot be regular. □

Before we treat the cases, where $m = 1$, we compile some properties of the underlying relations, which will then allow us to prove the non-regularity of several cases.

Lemma 26. *Over a two-letter alphabet $\{a, b\}$ for every 2^+-free word u starting with ab and every integer $n > 1$ there exists a word v, such that $uv \in (ab)^{\leq 3 \bowtie_1^n}$ and $|v| \leq (3(n-1)+2)(|u|-2)$.*

Proof. u being 2^+-free implies that there is no factor xxx for any letter $x \in \Sigma$. Thus the alphabet's containing only two elements guarantees that after at most 2 positions in u letters repeat, i.e. for every position in u its letter is repeated at most three positions later. Thus we can construct a word having u as a prefix in the following way: starting from ab, always one letter more of u is constructed per step. We take the shortest suffix z of the already constructed part of u starting with the next letter needed. On it we apply the rule $z \to z^n$ putting the required letter in the position. As shown above, the maximum length of z is 3 and thus all rules belong to ${}^{\leq 3}\bowtie_1^n$. This process takes exactly $|u| - 2$ steps, and in each one at most $3(n-1)+2$ additional letters are introduced, which proves the length bound on v. □

However, the length of the word v in Lemma 26, i.e. in some sense the amount of garbage produced during the generation of u, cannot be reduced to arbitrarily small numbers.

Lemma 27. *Over a two-letter alphabet $\{a, b\}$ for every 2^+-free word u starting with ab and every integer $n \geq 3$ there exists no word v, such that $uv \in (ab)^{\leq 3 \bowtie_1^n}$ and $|v| \leq \frac{|u|-2}{2k}$.*

Proof. u is obtained from ab by the application of rules $z \to z^n$. Since u is 2^+-free and $n \geq 3$ every such rule must produce at least one additional symbol outside of u, therefore contributing to v. At the same time each rule produces at most $2k$ letters of u such that at least $\frac{|u|-2}{2k}$ rules must be applied. Therefore there are at least $\frac{|u|-2}{2k}$ symbols in v. □

Lemma 28. *Over a three-letter alphabet $\{a, b, c\}$ for every square-free word u starting with abc and every integer $n > 1$ there exists a word v, such that $uv \in (abc)^{\leq 4 \bowtie_1^n}$, and for the shortest such word we have $\frac{|u|-3}{7} \leq |v| \leq (4(n-1)+3)(|u|-3)$.*

Proof. uv can be constructed starting from abc in a way very similar to that of the proof of Lemma 26. Only here between two consecutive occurrences of the same letter in u there can be three other letters, because 3 is the length of the longest square-free word over two letters. Therefore the longest z such that rules $z \to z^n$ are applied is 4 letters long, and that gives us the upper bound on the length of v. The lower bound is obtained in a manner analogous to the proof of Lemma 27. □

Proposition 29. *Over a two-letter alphabet for every word w and integers $k, n \geq 3$ the language $w^{\leq k \bowtie_1^n}$ is not regular, while $w^{\leq k \bowtie_1^2}$ is.*

Proof. The non-regularity of $w^{\leq k \bowtie_1^2}$, i.e. for the case of duplication, is already stated in Proposition 8. For $n \geq 3$ Lemmata 26 and 27 show us that for every 2^+-free word u starting with ab and every integer $n \geq 3$ there exists a word v, such that $uv \in (ab)^{\leq 3 \bowtie_1^n}$ and $\frac{|u|-2}{2k} \leq |v| \leq (3(n-1)+2)(|u|-2)$, i.e. the length of a minimal v is bounded from above and below.

Now we take an infinite 2^+-free word starting with ab and produce a sequence of prefixes $(u_i)_{i \geq 1}$ such that $\frac{|u_{i+1}-2|}{2k} > (|u_i|-2)2k$. Then the v_i from the construction in the proof of Lemma 26 is such that $u_i v_i \in w^{\leq k \bowtie_1^n}$, while $u_{i+1} v_i \notin w^{\leq k \bowtie_1^n}$ due to length considerations. Therefore all our words u_i are pairwise in different equivalence classes of the syntactical right congruence of $w^{\leq k \bowtie_1^n}$, and by Theorem 1 the language cannot be regular. □

With more than two letters, the special case of $n = 2$ is also no longer regular.

Proposition 30. *Over a three-letter alphabet for every word w and integers $k \geq 4$ and $n > 1$ the language $w^{\leq k \bowtie^{n}_{1}}$ is not regular.*

Proof. A construction analogous to the proof of Proposition 29 using the length bounds from Lemma 28 proves this statement. For more details the reader can also consult the proof for the special case of bounded duplication [17]. $\square$

Having found lower bounds for complexity in the interesting cases where $m = 1$, we can now state an upper bound as well, which determines the exact place of the languages $w^{\leq k \bowtie^{n}_{m}}$ in the Chomsky Hierarchy, also for $m > 1$.

Proposition 31. *For every word w, and for integers $k, m, n \geq 0$ the language $w^{\leq k \bowtie^{n}_{m}}$ is context-free.*

Proof. This can be shown by constructing a push-down automaton in a way rather analogous to the one used for bounded duplication in the original proof of Proposition 10, see [17]. The definition will be elaborate, but then, by contrast, the desired properties will be rather straightforward to prove.

First we introduce a marked copy of our alphabet Σ, which we will call $\Sigma_{\langle\rangle}$ and whose letters we will denote in the following way: $\Sigma_{\langle\rangle} := \Sigma \cup \{\langle a\rangle \mid a \in \Sigma\}$; for a word $u \in \Sigma^*$ we will denote by $\langle u\rangle$ the corresponding word over $\{\langle a\rangle \mid a \in \Sigma\}$. Further we define $L^{\leq \ell} := \{u \in L \mid |u| \leq \ell\}$ for any language L and integer ℓ.

With this we come to the definition of the push-down automaton accepting the language $w^{\leq k \bowtie^{n}_{m}}$ as

$$A = \left(Q, \Sigma, \Gamma, \delta, \begin{bmatrix} \lambda \\ \lambda \\ w \end{bmatrix}, \bot, \left\{\begin{bmatrix} \lambda \\ \lambda \\ \lambda \end{bmatrix}\right\}\right).$$

The first and third components are the state set

$$Q = \left\{\begin{bmatrix} \mu \\ v \\ z \end{bmatrix} \mid \mu \in (\Sigma_{\langle\rangle}^* \cdot \Sigma^*)^{\leq k\cdot n}, v \in (\Sigma^*)^{\leq k\cdot m}, z \in \mathrm{suff}(w)\right\},$$

and the push-down alphabet

$$\Gamma = \{\bot\} \cup \left\{\overline{\mu} \mid \begin{bmatrix} \mu \\ v \\ z \end{bmatrix} \in Q, v \in (\Sigma^*)^{\leq k\cdot m}, z \in \mathrm{suff}(w)\right\},$$

where $\mathrm{suff}(w)$ denotes the set of suffixes of the word w including λ and w itself.

The transition function δ is a mapping from $Q \times (\Sigma \cup \{\lambda\}) \times \Gamma$ into $Q \times \Gamma^*$; i.e. we allow reading and changing the stack without reading from the input tape, and in every step the topmost stack symbol is read and has to be put back to remain on the stack, if it is supposed to remain there. We will not define the transition function δ right away, but wil start out with a preliminary one called δ'.

In the definition of δ', the following variables are always quantified universally over the following domains: $p \in (\Sigma^*)^{\leq k}$, $u \in (\Sigma^*)^{\leq k\cdot n}$, $v \in (\Sigma^*)^{\leq k\cdot m}$, $z \in (\Sigma^*)^{\leq |w|}$, $\mu \in (\Sigma_{\langle\rangle}^*)^{\leq k\cdot n}$, $\eta \in (\Sigma_{\langle\rangle}^* \cdot \Sigma^*)^{\leq k\cdot n}$, $\gamma \in \Gamma$, and $x \in \Sigma$. Now, δ' is defined as follows:

(i) $\delta'\left(\begin{bmatrix} \lambda \\ \lambda \\ xz \end{bmatrix}, x, \bot\right) = \left(\begin{bmatrix} \lambda \\ \lambda \\ z \end{bmatrix}, \bot\right)$ and $\delta'\left(\begin{bmatrix} \lambda \\ xu \\ xz \end{bmatrix}, \lambda, \bot\right) = \left(\begin{bmatrix} \lambda \\ u \\ z \end{bmatrix}, \bot\right)$

(ii) $\delta'\left(\begin{bmatrix} \mu x u \\ \lambda \\ z \end{bmatrix}, x, \gamma\right) = \left(\begin{bmatrix} \mu\langle x\rangle u \\ \lambda \\ z \end{bmatrix}, \gamma\right)$ and $\delta'\left(\begin{bmatrix} \mu x u \\ xv \\ z \end{bmatrix}, \lambda, \gamma\right) = \left(\begin{bmatrix} \mu\langle x\rangle u \\ v \\ z \end{bmatrix}, \gamma\right)$

(iii) $\delta'\left(\begin{bmatrix} \langle p\rangle^n \\ \lambda \\ z \end{bmatrix}, x, \overline{\eta}\right) = \left(\begin{bmatrix} \eta \\ p^m x \\ z \end{bmatrix}, \lambda\right)$, $\delta'\left(\begin{bmatrix} \langle p\rangle^n \\ xv \\ w \end{bmatrix}, \lambda, \overline{\eta}\right) = \left(\begin{bmatrix} \eta \\ p^m xv \\ w \end{bmatrix}, \lambda\right)$.

For all triples $(q, x, \gamma) \in Q \times (\Sigma \cup \{\lambda\}) \times \Gamma$ not listed above, we put $\delta'(q, x, \gamma) = \emptyset$.

In what follows we will call the three strings occurring in the PDA states from bottom to top *original*, *memory*, and *guess* respectively. The transitions from set (i) match the original against either the input tape or the memory. In set (ii) a letter is marked in the guess, if it is read, and (iii) puts the top of the stack into the guess, if this has completely been matched against the input tape; at the same time, if the match contains some $\langle p\rangle^n$, then p^m is put in the memory. This signals that the right side of a rule $p^m \to p^n$ has been read, and now the computation should continue as before, just on a string having the factor p^m instead of p^n. Since the entire factor has already been read, p^m is put into the memory.

Storing several such factors p^m in rapid succession might imply a danger of exceeding the length bound for this memory, possibly leading to an infinite set of states. However, during any reduction, which in the end puts $k \cdot m < k \cdot n$ letters into the memory, either $k \cdot n$ letters from the memory or all letters of the memory (provided the memory is shorter than $k \cdot n$) have been read, since reading from memory has priority: tape symbols are read only if the memory is empty. This gives us a bound on the length of words in the memory which is $k \cdot n$.

To obtain the transition function δ of our automaton, we now add the possibility to interrupt at any point the computation of δ' and change to a state that starts the reduction of another rule application. This is done by putting the content of the match onto the stack (as one single stack symbol), guessing which idempotency rule was applied at the current point, and by putting the rule's right side into the match. From there it will be matched against the input word, thus verifying the correctness of the guess.

So we define for all $\begin{bmatrix}\eta\\ v\\ z\end{bmatrix} \in Q \setminus F$, and $\gamma \in \Gamma$

$$\delta\left(\begin{bmatrix}\eta\\ v\\ z\end{bmatrix}, \lambda, \gamma\right) = \delta'\left(\begin{bmatrix}\eta\\ v\\ z\end{bmatrix}, \lambda, \gamma\right) \cup \left\{ \left(\begin{bmatrix}p^n\\ v\\ z\end{bmatrix}, \overline{\eta}\gamma\right) \middle| p \in (\Sigma^+)^{\leq k} \right\}.$$

The PDA being completely defined, we can now state an important property of A.

Property. *If there is an accepting computation of A starting from the configuration $\left(\begin{bmatrix}\eta\\ \lambda\\ z\end{bmatrix}, v, \alpha\right)$, then there is also an accepting computation starting from any configuration $\left(\begin{bmatrix}\eta\\ v_1\\ z\end{bmatrix}, v_2, \alpha\right)$ where $v = v_1v_2$, $|v_1| \leq kn$.*

Proof of the Property. We will use induction on the length of v_1. For the empty string the two states are identical and the property holds trivially. So suppose $|v_1| = \ell$ and the property holds for all strings shorter than ℓ. Now we start in the configuration $\left(\begin{bmatrix}\eta\\ \lambda\\ z\end{bmatrix}, v, \alpha\right)$. If the transition applied is from δ', then it will read v's first letter because the memory is empty. Let us call the configuration reached in this way Δ.

For every one of these reading transitions, in the same clause of the definition of δ' a parallel one is defined that reads from the memory instead of the input tape. Thus it can be applied in the configuration $\left(\begin{bmatrix}\eta\\ v_1\\ z\end{bmatrix}, v_2, \alpha\right)$ where $v = v_1v_2$, $|v_1| \leq n$. Any of these transitions reading from the memory decreases the length of the string in the memory by one. Thus by checking the parallel transitions in the definition of δ' one sees that the configuration reached will be equivalent to Δ by our induction hypothesis.

In the case that the first transition applied is a guessing one, we can also do the equivalent for the configuration with v_1 in memory and start our reasoning from the first transition reading from the input tape after the configuration given in the property's statement. This concludes the proof of the above property. $\square$

Now we can prove the proposition by induction on the number of rewrite rules applied to create a word in $w^{\leq k}\bowtie^n_m$. We start by noting that the original word w is obviously accepted. Now we suppose that all words reached from w by $\ell - 1$ rule applications are also accepted and look at a word s reached by m applications. Clearly there is a word s' reached from w by $\ell - 1$ rule applications such that s is the result of rewriting one part of s'. By the induction hypothesis, s' is accepted by A, which means that there is a computation, we call it Ξ, which accepts s'. Let $s'[l_1 \ldots l_2] = t^m$ be the segment of s' which is replaced in the final step of producing s. Therefore

$$s = s'[1 \ldots l_1 - 1]s'[l_1 \ldots l_2]t^{n-m}s'[l_2 + 1 \ldots |s'|].$$

Because A reads in an accepting computation every input letter exactly once, there is exactly one step in Ξ, where $s'[l_2]$ is read. Let this happen in a state $\begin{bmatrix}\mu\\ \lambda\\ z\end{bmatrix}$ (recall that input letters are read exclusively when the memory is empty) with $s'[l_2 \ldots |s'|] = s[l_2 + (n-m)\frac{|t|}{m} \ldots |s|]$ left on the input tape and α the stack contents. Now instead of continuing Ξ we go without reading the input tape to state $\begin{bmatrix}s'[l_1 \ldots l_2]t^{n-m}\\ \lambda\\ z\end{bmatrix}$ and push $\overline{\mu}$ onto the stack. Then we reduce with transitions of δ' the result of rewriting the factor $s'[l_1 \ldots l_2]$ (i.e. $s'[l_1 \ldots l_2]t^{n-m}$) by matching it against the letters read on the input tape.

After matching this, we arrive at a configuration having a state containing μ in the guess, $s'[l_1 \dots l_2]$ in the memory, and z in the pattern, $s'[l_2 + 1 \dots |s|]$ left on the tape, and the stack contents as before. By the property above, there exists an accepting computation starting with this configuration, hence s and consequently all words in $w^{\leq k \bowtie^n_m}$ are accepted by A.

On the other hand, all words accepted by A are clearly in $w^{\leq k \bowtie^n_m}$, because the deterministic δ' leaves no choice but to reduce the results of rule applications against the input word, and the other transitions can only guess such applications. Thus we are done. □

The properties of languages generated by bounded idempotency which we stated for the non-regularity proofs earlier also allow us to conclude that in many cases the inclusions $w^{\leq k \bowtie^n_1} \subset w^{\leq k+1 \bowtie^n_1}$ are proper. For this, however, we first need to recall the notion of circular pattern avoidance. A word w is said to be *circular square-free*, iff it is square-free and so are all its conjugates. This means that one can arrange the word in a circle with the first letter following the last, and nowhere along the circle is there a square. We explicitly state an immediate consequence of this definition.

Lemma 32. *For a circular square-free word w the word ww contains no square shorter than ww itself.*

Circular cube-freeness is defined analogously. It is known that over a three letter alphabet there exist circular square-free words of any length greater than 17, and over two letters there exist circular cube-free words of any given length [7].

Proposition 33. *For every word w over two letters all inclusions $w^{\leq k \bowtie^n_1} \subset w^{\leq k+1 \bowtie^n_1}$ are proper for $n \geq 3$ and $k \geq 2$.*

Proof. From Lemma 26 we know that for every 2^+-free word u starting with ab and every integer $n \geq 2$ there exists a word v, such that $uv \in (ab)^{\leq 3 \bowtie^n_1}$. At some point of w a change from one letter to another must occur. So there we can construct any circular cube-free word. Let us construct such a word u of length $k + 1$ for some fixed k. In the next step we can apply here the rule $u \to u^n$.

The resulting factor u^n can also be produced by shorter rules, but Lemma 27 also shows that there is a lower bound on the number of additional symbols produced in this process. Thus by further applying the rule $u \to u^n$ we can reach a word, where the block of u^+ is so long in relation to the rest of the word, that it is impossible to produce the same word only with rules where the left side is not longer than k, since by a generalization of Lemma 32 to blocks u^n instead of just u^2 no shorter rule can have been applied anywhere within this block. □

Proposition 34. *For every word w over three or more letters there exists a k_w such that all inclusions $w^{\leq k \bowtie^n_1} \subset w^{\leq k+1 \bowtie^n_1}$ are proper for $n \geq 2$ and $k \geq k_w$; if w has a factor abc, then $k_w = 18$ will work.*

Proof. k_w will be the maximum of two values. One is the smallest number such that with rules of left sides of this length we can produce a factor of the form abc in w. For example, for the word $aabbbbbbccc$ the value is 9: with one rule we can produce $aabbbbbbcaabbbbbbccc$ and with another one $aabbbbbbc$**abc**$aabbbbbbccc$. The second value is 18, since starting from this length there exist circular square-free words of any given length.

From Lemma 28 we know that over a three-letter alphabet for every square-free word u starting with abc and every integer $n \geq 2$ there exists a word v, such that $uv \in (abc)^{\leq 4 \bowtie^n_1}$. Thus also every circular square-free word can be constructed. Starting from lengths of 18, such a word always exists, and starting from k_w we also can suppose that a word in $w^{\leq k \bowtie^n_1}$ contains a factor of the form abc. Since Lemma 28 also provides a lower bound on the length of the additional v which is produced, the same proof technique as for Proposition 33 applies. □

Also for bounded idempotency relations, we now take a look at the conditions under which they are confluent.

Proposition 35. *For all $k, n \geq 1$ the relation ${}^{\leq k}\bowtie^n_0$ is confluent.*

Proof. Let $u, v \in w^{\leq k \bowtie^n_0}$ for a word w. This means that u and v can both be obtained from w by inserting n-th powers of words of length no greater than k between the letters of w. So, marking the original letters of w by underlining them, we have $u = u_1\underline{w[1]}u_2\underline{w[2]}\dots u_{|w|}\underline{w[|w|]}u_{|w|+1}$ and $v = v_1\underline{w[1]}v_2\underline{w[2]}\dots v_{|w|}\underline{w[|w|]}v_{|w|}$ for some words

$u_1, u_2 \ldots u_{|w|+1}, v_1, v_2, \ldots v_{|w|+1} \in \Sigma^*$. Now clearly

$$u_1 v_1 \underline{w[1]} u_2 v_2 \underline{w[2]} \ldots u_{|w|} v_{|w|} \underline{w[|w|]} u_{|w|+1} v_{|w|+1} \in u^{\leq k \bowtie_0^n} \cap v^{\leq k \bowtie_0^n},$$

which proves the confluence of ${}^{\leq k}\bowtie_0^n$. □

Proposition 36. *For all $k < 3$ and $n \geq 1$ the relation ${}^{\leq k}\bowtie_1^n$ is confluent.*

Proof. We show that the diamond property holds, i.e. $w_1 \leftarrow u \rightarrow w_2 \Rightarrow \exists v (w_1 \rightarrow v \leftarrow w_2)$. For $k < 2$ this is obvious. The same is true for $k = 2$ if the two application sites of the rules do not overlap. The few possible cases for $k = 2$ can now be checked in an exhaustive manner to have the diamond property.

For $n \geq 2$ the derivations $abc \rightarrow (ab)^n c \rightarrow (ab)^n c (bc)^{n-1} \leftarrow a(bc)^{n-1} \leftarrow abc$ treats the case of two rules with left sides of length two. If they are of length one and two, and then $ab \rightarrow ab^n \rightarrow (ab)^n b^{n-1} \leftarrow (ab)^n \leftarrow ab$ proves the diamond property. Of course, if not all of the letters involved are different, then things become even easier. □

The argumentation shows that, informally speaking, for non-confluence it has to be possible to have the application site of one rule properly inside the other one. The shortest possible lengths for this are one and three, and these already suffice, however only over at least three letters.

Proposition 37. *For all $k \geq 3$ and $n \geq 2$ the relation ${}^{\leq k}\bowtie_1^n$ is not confluent over an alphabet of three or more letters.*

Proof. From the word $ab^{k-2}c$ one can obtain in one step $u = ab^{k+n-3}c$ and also $v = (ab^{k-2}c)^n$. Notice that v contains an occurrence of a after one of c, and thus all words obtained by application of further rules will do so.

At the same time in u the unique occurrences of a and c are separated by at least $k - 1$ letters b. Thus no application of a rule from ${}^{\leq k}\bowtie_1^n$ can include a as well as c. Since this central block of b is conserved, no word with an a after a c can be reached. Thus $u^{\leq k \bowtie_1^n} \cap v^{\leq k \bowtie_1^n} = \emptyset$, which proves our claim. □

Proposition 38. *Over a two-letter alphabet, the relation ${}^{\leq k}\bowtie_1^2$ is confluent for all $k \geq 1$.*

Proof. The cases where $k = 1$ are obvious. So let us suppose that we have $u \overset{*}{\leftarrow} w \overset{*}{\rightarrow} v$. Notice that all words in $w^{\leq k \bowtie_1^2}$ start and end with the same letters, let them be a and b respectively. Then a characteristic feature of every such word is its number of changes from a to b. Let this number be i for u and j for v. Unless they are equal, without restriction of generality let i be the greater number. We now start from the word v and select any occurrence of ab in it. This we duplicate $i - j$ times, the resulting word v' now has i changes from a to b, just as u.

In a next step we look at u and v' and compare the length of the initial blocks of a. In the shorter one we duplicate the initial a so often that the block of a becomes as long as the other one. Then the same is done for the first block of b and so on for all blocks. Clearly the resulting word is in $u^{\leq k \bowtie_1^2} \cap v^{\leq k \bowtie_1^2}$, which proves the confluence of ${}^{\leq k}\bowtie_1^2$. Note that we have used only rules where $k = 1$ or $k = 2$. □

To show the confluence of ${}^{\leq k}\bowtie_1^n$ for greater n, the construction method used for $n = 2$ cannot be applied. Unlike in the construction in the proof of Proposition 38, two blocks of one letter cannot be made to have the same length in general as the following lemma shows.

Lemma 39. *Let $w \in \{a, b\}^*$, $k \geq 1$ and $n > 2$. For all words $u \in w^{\leq k \bowtie_1^n}$ the number of changes from a to b, the number of changes from b to a, and the numbers $|u|_a$ and $|u|_b$ are constant modulo $(n - 1)$.*

Proof. If the site of a rule application contains no change from a to b, then the number of such changes for the entire word stays the same. If, on the other hand, it contains i changes, they will be replaced by $n \cdot i$ ones, the number increases by $(n - 1) \cdot i$. Similarly, a rule whose left side contains i letters a replaces them by $n \cdot i$ new ones, also here the number increases by $(n - 1) \cdot i$. □

Nonetheless we conjecture that these relations are still confluent, but some different reasoning will be necessary.

Proposition 40. *For all $k \geq 3$, $m \geq 2$, $k > m$ and $n > m$ the relation ${}^{\leq k}\bowtie_m^n$ is not confluent.*

Proof. We start from the word $(a^m b)^m$. The entire word is the left side of a rule resulting in $(a^m b)^n$, which has more than m letters b. On the other hand the rule $a^m \rightarrow a^n$ can be applied to any of the blocks a^m; if this is done to any of these blocks except the first one, it is quite clear that after this it will only be possible to apply rules producing more a. Thus the number of b will always remain lower than n, which suffices to prove our claim. □

5. Unbounded idempotency

When dropping all restrictions on the idempotencies, a fundamental difference to the cases treated up to this point is that we have infinitely many rewrite rules, whereas so far due to the length restrictions there have been only finitely many. In some simple cases there are finite sets equivalent in generating power, but not in general as shown already by Propositions 33 and 34. We will first list a number of results that carry over more or less directly from previous ones.

Over a one-letter alphabet, we can see that all languages generated are regular as a direct corollary of Proposition 23.

Proposition 41. *Over a one-letter alphabet $\{a\}$ for every nonempty word w and integers $m, n \geq 0$ the language $w^{\bowtie_m^n}$ is regular.*

As already stated in Section 2, the cases of insertion and deletion, i.e. the languages $w^{\bowtie_0^1}$ and $w^{\bowtie_1^0}$, are both regular, see Propositions 11 and 12. Non-regularity can be established in several cases in similar ways to the proofs for bounded idempotencies, only the length bounds change. We first fix these in a few lemmata.

Lemma 42. *Over a two-letter alphabet $\{a, b\}$ for every 2^+-free word u starting with ab and every integer $n > 1$ there exists a word v, such that $uv \in (ab)^{\bowtie_1^n}$ and $|v| \leq (3(n-1)+2)(|u|-2)$.*

Proof. The same construction as in the proof of Lemma 26 applies. □

While the upper bound carries over, the lower bound is significantly lower than the one for the bounded case stated in Lemma 43. The length bound provided is not tight, but suffices for our purposes.

Lemma 43. *Over a two-letter alphabet $\{a, b\}$ for every 2^+-free word u starting with ab and every integer $n \geq 3$ there exists no word v, such that $uv \in (ab)^{\leq 3 \bowtie_1^n}$ and $|v| \leq \log_2(|u|/3)$.*

Proof. u is obtained from ab by the application of rules $z \to z^n$. Since u is 2^+-free and $n \geq 3$ every such rule must produce at least one additional symbol outside of u, therefore contributing to v. At the same time each rule produces at most $2|\ell|$ letters of u, where ℓ is the rule's left side. Thus at least $\log_2(|u|/3)$ rules must be applied, since our starting word has length 3 and each idempotency rule can at most double the length of the subword of u already produced. Consequently, v is at least $\log_2(|u|/3)$ symbols long. □

Lemma 44. *Over a three-letter alphabet $\{a, b, c\}$ for every square-free word u starting with abc and every integer $n > 1$ there exists a word v, such that $uv \in (abc)^{\bowtie_1^n}$ and $\log_2(|u|/3) \leq |v| \leq (|u|-3)(4(n-1)+3)$.*

Proof. uv can be constructed starting from abc as in the proof of Lemma 28. Only the lower bound for the length here corresponds to the one from Lemma 43. □

Proposition 45. *Over a two-letter alphabet for every word w and integers $n \geq 3$ the language $w^{\bowtie_1^n}$ is not regular, while $w^{\bowtie_1^2}$ is.*

Proof. The regularity of $w^{\bowtie_1^2}$, i.e. for the case of duplication, was proven by Dassow et al. [8]. For $n \geq 3$ Lemmata 42 and 43 allow a proof completely analogous to the one of Proposition 29. □

With more than two letters, also here the special case of $n = 2$ is no longer regular; the proof can again be done by the same method as for bounded idempotencies and the length bounds from Lemma 44.

Proposition 46. *Over a three-letter alphabet for every word w and an integer $n > 1$ the language $w^{\bowtie_1^n}$ is not regular.*

A well-known open problem, on the other hand, is the question of context-freeness in the case of $\bowtie_1^2$ general duplication. While it is known that the languages generated over three letters are not regular, see Proposition 46, their context-freeness has neither been proved nor disproved. The same is true for the related cases of $\bowtie_1^n$ with $n > 2$.

Now we again turn our attention to the confluence of the relations under question.

Proposition 47. *The relations $\bowtie_0^n$ for $n \geq 0$ and $\bowtie_1^0$ are confluent.*

Proof. The confluence of $\bowtie_1^0$ is trivial, because here every word can be reduced to λ and this is the only irreducible word. For the confluence of $\bowtie_0^n$, on the other hand, the same proof as for the bounded case in Lemma 35 applies. □

Proposition 48. *The relations $\bowtie_m^0$ are not confluent for $m \geq 2$.*

Proof. We consider the word $a^m b(a^{m-1}b)^{m-1}$. It contains two factors, which are powers of order m, namely a^m and $(a^{m-1}b)^m$. Reducing the first one results in $b(a^{m-1}b)^{m-1}$, reducing the second one results in a; both are irreducible, and thus the reduction relation is not confluent. □

Proposition 49. *The relations $\bowtie_m^1$ are not confluent for $m \geq 2$.*

Proof. For the case $n = 2$ already Example 3 provides the appropriate counterexample of $(abcbabcbc)^{\bowtie_2^1} = \{abc, abcbc, abcbabc\}$. This can be generalized by using for a given n the word $(abcb)^m c(bc)^{m-2}$, which can be reduced to the two irreducible words abc and $(abcb)^{m-1}abc$. □

We also want to mention a class of languages with some connection to the relations $\bowtie_m^{m+1}$. A language L is called *non-counting*, iff there is an integer $i \geq 0$ such that for every $y \in \Sigma^+$ and $x, z \in \Sigma^*$, we have $xy^i z \in L$ iff $xy^{i+1}z \in L$. This definition has many parallels to the one of the relations $\bowtie_m^{m+1}$. Clearly m is a constant such that $xy^i z \in w^{\bowtie_m^{m+1}}$ iff $xy^{i+1}z \in w^{\bowtie_m^{m+1}}$. This allows us to directly conclude the following.

Proposition 50. *For every $m \geq 0$ and every word w the language $w^{\bowtie_m^{m+1}}$ is non-counting.*

With this we close this section. The results on unbounded cases are much less than for the bounded ones; mainly we have only those ones that carry over in some way from the case of bounded length. Thus much remains to be done in this direction.

6. Lines for further research

Although many cases have been treated, open problems about languages generated by idempotency relations abound. We will list some of the more general and interesting ones.

For the two bounded cases, very often confluence of the relation and regularity of the language generated coincide. Maybe a general statement relating these two properties can be found, or at least one, which widely applies. Instead of confluence the stronger diamond property might also be a good candidate for such a relation.

The least number of results has been established in the unbounded case. Especially the question of whether the relations $\bowtie_1^n$ generate only context-free languages remains open; for $n = 2$, the duplication case, this has already received a great deal of interest.

Another topic known already from duplication are roots, i.e. the languages of normal forms generated by applying rules $u^m \to u^n$ with $m < n$. Interesting questions in this field include: Is the root unique for a given word? Does a language have a finite/regular root, and is this property decidable? The first question has been answered here for a few special cases, where such relations have been shown to be confluent or not, but much work remains to be done.

Finally, it should be remarked that the name idempotency relations is slightly inaccurate, because we use the relations $u^m \equiv u^n$ only in the direction from left to right and not as a real equivalence. But in principle nothing stands against doing that, in our diction this would be using the relations $\bowtie_m^n \cup \bowtie_n^m$. Whereas in our case the languages generated are always partially ordered with the original word as minimum (taking "‘is reachable by applying rules"’ as underlying relation), this property would be lost here. Nonetheless, similar investigations to ours could be carried out about languages generated by this type of relation.

Acknowledgements

This work was done, while the author was funded by the Spanish Ministry of Culture, Education and Sport under the Programa Nacional de Formación de Profesorado Universitario (FPU). Further, he was very kindly hosted by Kyoto Sangyo University during part of the elaboration.

References

[1] S.I. Adian, The Burnside Problem and Identities in Groups, Springer-Verlag, Berlin, 1979. Translated from Russian.

[2] F. Baader, T. Nipkow, Term Rewriting and All That, Cambridge University Press, Cambridge, 1998.

[3] R. Book, F. Otto, String-Rewriting Systems, Springer, Berlin, 1988.

[4] D.P. Bovet, S. Varricchio, On the regularity of languages on a binary alphabet generated by copying systems, Information Processing Letters 44 (1992) 119–123.

[5] J. Brzozowski, Open problems about regular languages, in: R.V. Book (Ed.), Formal Language Theory — Perspectives and Open Problems, Academic Press, New York, 1980.

[6] W. Burnside, On an unsettled question in the theory of discontinuous groups, Quarterly Journal of Pure and Applied Mathematics 33 (1902) 230–238.

[7] J.D. Currie, S.D. Fitzpatrick, Circular words avoiding patterns, in: DLT 2002, in: Lecture Notes in Computer Science, vol. 2450, Springer-Verlag, Berlin, 2003.

[8] J. Dassow, V. Mitrana, Gh. Păun, On the regularity of duplication closure, Bulletin of the European Association for Theoretical Computer Science 69 (1999) 133–136.

[9] N. Dershowitz, Semigroups Satisfying $x^{m+n} = x^n$, in: Lecture Notes in Computer Science, vol. 656, Springer-Verlag, Berlin, 1993, pp. 307–314.

[10] A. Ehrenfeucht, G. Rozenberg, On regularity of languages generated by copying systems, Discrete Applied Mathematics 8 (1984) 313–317.

[11] J.A. Green, D. Rees, On semigroups in which $x^r = x$, Proceedings of the Cambridge Philosophical Society 48 (1952) 35–40.

[12] M.A. Harrison, Introduction to Formal Language Theory, Reading, Mass, 1978.

[13] M. Ito, Algebraic Theory of Automata and Languages, World Scientifc, New Jersey, 2004.

[14] P. Leupold, n-Bounded duplication codes, in: Proceedings of the ICALP-Workshop on Words, Avoidability, Complexity, Turku 2004. Technical Report 2004-07, Laboratoire de Recherche en Informatique d'Amiens, Amiens 2004.

[15] P. Leupold, C. Martín Vide, V. Mitrana, Uniformly bounded duplication languages, Discrete Applied Mathematics 146 (3) (2005) 301–310.

[16] P. Leupold, V. Mitrana, Uniformly bounded duplication codes, RAIRO Informatique Théorique (in press).

[17] P. Leupold, V. Mitrana, J. Sempere, Languages arising from gene repeated duplication, in: Aspects of Molecular Computing, in: LNCS, vol. 2950, Springer Verlag, Berlin, 2004, pp. 297–308. Essays in Honour Tom Head on his 70th Birthday.

[18] M. Lothaire, Combinatorics on Words, Addison-Wesley, Reading, MA, 1983.

[19] M. Lothaire, Algebraic Combinatorics on Words, Cambridge University Press, Cambridge, 2002.

[20] C. Martín-Vide, Gh. Păun, Duplication grammars, Acta Cybernetica 14 (1999) 101–113.

[21] V. Mitrana, G. Rozenberg, Some properties of duplication grammars, Acta Cybernetica 14 (1999) 165–177.

[22] Gh. Păun, G. Rozenberg, A. Salomaa, DNA Computing — New Computing Paradigms, Springer Verlag, Berlin, 1998.

[23] A. Salomaa, Formal Languages, Academic Press, Orlando, 1973.

[24] M.-W. Wang, On the irregularity of the duplication closure, Bulletin of the European Association for Theoretical Computer Science 70 (2000) 162–163.