# Closure of Language Classes under Bounded Duplication

Masami Ito[1], Peter Leupold[2]*, and Kayoko Shikishima-Tsuji[3]

[1] Department of Mathematics, Faculty of Science
Kyoto Sangyo University
Kyoto 603-8555, Japan
ito@ksuvx0.kyoto-su.ac.jp
[2] Research Group on Mathematical Linguistics
Rovira i Virgili University
Pça. Imperial Tàrraco 1, 43005 Tarragona, Catalunya, Spain
klauspeter.leupold@urv.cat
[3] Tenri University
Tenri 632-8510, Japan
tsuji@sta.tenri-u.ac.jp

**Abstract.** Duplication is an operation generating a language from a single word by iterated application of rewriting rules $u \rightarrow uu$ on factors. We extend this operation to entire languages and investigate, whether the classes of the Chomsky hierarchy are closed under duplication. Here we treat mainly bounded duplication, where the factors duplicated cannot exceed a given length.
While over two letters the regular languages are closed under bounded duplication, over three or more letters they are not, if the length bound is 4 or greater. For 2 they are closed under duplication, the case of 3 remains open. Finally, the class of context-free languages is closed under duplication over alphabets of any size.

## 1 Duplication

In a series of recent articles, languages generated from a single word 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 $uvvw$.

The first mechanism for generating languages derived from this behaviour were the so-called duplication grammars [9],[10]. Then a great deal of interest was paid to languages generated from a word by iterated application of the duplication operation as introduced by Dassow et al. [3]. The main focus in these investigations was on determining under which conditions those languages are regular. In this context also the restriction of the duplicated factor's length to a maximum or one fixed length has been investigated [5],[6],[7],[8],[12].

The objective of this article will be the investigation of the duplication closure on entire languages rather than single words. The duplication closure of a language is the union of all the closures of the words contained in that language. Our main focus will be on the question, under which conditions this

* 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.

operation preserves regularity and context-freeness. It is rather obvious that all versions of duplication preserve context-sensitivity.

In Section 3 we establish that the class of regular languages is closed under 2-bounded duplication, but not under 4-bounded duplication; the case of 3 remains open. Further we show that over a two-letter alphabet 2-boundedness is equivalent to any longer bound and even to unbounded duplication. In combination with the preceding results this proves that over two letters regular languages are closed under general duplication and all bounded versions.

The class of context-free languages is the focus of Section 4. We establish its closure under bounded duplication, and further give a result that shows that this does not help us to answer the case of general duplication, because the $n$-bounded duplication languages of the word $abc$ form an infinite hierarchy with the unbounded duplication language as its supremum.

## 2 Definitions

We now provide the formal definitions concerning the duplication operation. For this, we take for granted elementary concepts from the theory of formal languages as exposed, for example, by Harrison [4] or Salomaa [11]. A few notations we use are: $|w|$ for the length of the word $w$, $w[i]$ for the $i$-th letter of $w$, and $w[i \dots j]$ for the factor of $w$ starting at position $i$ and ending at position $j$. A period of a word $w$ is an integer $k$ such that for all $i \leq |w| - k$ we have $w[i] = w[i+k]$. For an alphabet $\Sigma$, the set $\Sigma^n$ consists of all the words of length $n$ over this alphabet, further $\Sigma^{\leq n} := \bigcup_{i \leq n} \Sigma^i$.

An important notion that will be used is the relation $\sim_L$ over $\Sigma^* \times \Sigma^*$ for a language $L \subset \Sigma^*$, which is 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. It is well-known from the Kleene-Myhill-Nerode Theorem 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 ([11]) *A language $L$ is regular, if and only if $\sim_L$ has finite index.*

With this we come to the central notion of this article. We will use a rewriting relation to generate duplication languages. For details on string-rewriting systems we refer the reader to the book by Book and Otto [1], whose terminology we will follow here.

The relation we define will be denoted by ♡; with the origin on the bottom expanding to two equal halves, this symbol seems quite appropriate for duplication. So, in detail, the duplication relation is defined as

$$u \heartsuit v :\Leftrightarrow \exists w [u = u_1 w u_2 \wedge v = u_1 w w u_2].$$

With $\heartsuit^*$ we denote the relation's reflexive and transitive closure. We generate languages with it in the following way.

Definition 2 The duplication language generated by a word $w$ is

$$w^{\heartsuit} := \{u : w\heartsuit^{*}u\}.$$

Thus $w^{\heartsuit}$ is the language of all words that can be obtained from $w$ by a finite number of duplications. Apart from general duplication, also two restricted variants have been investigated, namely *bounded* and *uniformly bounded duplication.* These are defined for some integer $n$ as

$$u\heartsuit^{\leq n}v :\Leftrightarrow \exists w[u = u_1 w u_2 \wedge v = u_1 w w u_2 \wedge |w| \leq n]$$

and

$$u\heartsuit^{=n}v :\Leftrightarrow \exists w[u = u_1 w u_2 \wedge v = u_1 w w u_2 \wedge |w| = n]$$

respectively. So the $n$-bounded variant admits duplications of factors up to length $n$, the uniformly $n$-bounded one admits duplications of factors of length exactly $n$. The languages $w^{\heartsuit \leq n}$ and $w^{\heartsuit = n}$ are defined analogously to the unrestricted case. The latter variant we write also simply $w^{\heartsuit n}$.

We will now illustrate these definitions with two simple examples over the two-letter alphabet $\Sigma = \{a, b\}$. $(aba)^{\heartsuit}$ is the language $a\Sigma^{*}a$. $(aba)^{\heartsuit \leq 2} = a\Sigma^{*}a$. On the other hand $(aba)^{\heartsuit 2} = (ab)^{*}a$.

In the canonical way, the duplication operation is extended to sets of words, setting for such a set $W$ its language generated by duplication as

$$W^{\heartsuit} := \bigcup_{w \in W} w^{\heartsuit}.$$

This is the form, in which we will now apply duplication to languages and then investigate, whether this preserves regularity and context-freeness.

## 3 Closure of Regular Languages

We start out with the closure of regular languages. Here the size of the alphabet will play an important role, and first we treat the three-letter case, where closure is not given in most of the cases. All results for this alphabet size also carry over to bigger alphabets.

It is known that the 4-bounded duplication closure of the word $abc$ is not regular [8]. As one can see from the original proof, duplications longer than 4 do not affect the construction used, and therefore the result extends to longer bounds. Thus the class of regular languages is not closed under $n$-bounded duplication for $n \geq 4$, since singular sets are of course regular.

Proposition 3 *For $n \geq 4$ the class of regular languages is not closed under $n$-bounded duplication.*

On the other hand, it is trivial to see that 1-bounded duplication preserves regularity: the only possible change in the original word is that every letter $a$ can be blown up to any word from $a^{+}$. We now take a look at the two cases inbetween, that is length-bounds of 2 and 3.

We now fix some notation, which will be convenient in the proof that follows. For a right-syntactic congruence $\sim_L$ we denote the set of all possible right contexts of a word $u$ by $\sim_L(u) := \{w : uw \in L\}$. By $[u]_{\sim_L}$ we denote the congruence class of $u$; notice that for all $u_1, u_2 \in [u]_{\sim_L}$ we have $\sim_L (u_1) =\sim_L (u_2)$.

Proposition 4 *The class of regular languages is closed under* 2*-bounded duplication.*

*Proof.* Let $L$ be a regular language, and $\sim_L$ the corresponding right-syntactic congruence. The right-syntactic congruence $\sim_{L^{\heartsuit\leq 2}}$ we will denote more simply by $\sim$. We will show that the number of congruence classes of $\sim$ is bounded by a function of the number of congruence classes of $\sim_L$.

First notice that always $(\sim_L(u))^{\heartsuit\leq 2} \subseteq \sim(u)$, i.e. if $v$ is a possible right context of $u$ in $L$, then all words in $v^{\heartsuit\leq 2}$ are possible right contexts of $u$ in $L^{\heartsuit\leq 2}$. If the two sets are not equal, this can be caused only by some duplication transgressing the border between $u$ and $v$. Duplications of length one cannot do this, thus the only possibility is one of length two affecting the last letter of $u$ and the first letter of $v$.

If the two letters are the same, say $a$, then the result will be $a^4$, which could have been obtained also by duplicating twice the $a$ in $v$, so the result is in $v^{\heartsuit\leq 2}$. If the two letters are distinct, say $a$ and $b$, then the result of the duplication will be $abab$. If the following letter in $v$ is an $a$, then we could have obtained the same by duplicating the prefix $ba$ of $v$, so the result is in $v^{\heartsuit\leq 2}$.

Otherwise the result will be $ababc$ for some letter $c$ different from $a$. The resulting right context is not in $v^{\heartsuit\leq 2}$, so in this case a new congruence class for $u$ is created in $\sim$. More duplications on the right side will not lead to new classes, because now we have $bab$ following the final $a$ of $u$. The number of such constellations of two different letters at the border with a different one from the first one following is bounded by the total number of letters in the alphabet. Thus every congruence class of $\sim_L$ results in a finite number of congruence classes for $\sim$, except possibly for the one of words not being a prefix of a word in $L$.

Therefore it remains to show that the $u$, which are not prefixes of a word in $L$ but are prefixes of a word in $L^{\heartsuit\leq 2}$, do not generate an infinite number of new congruence classes. So let $uv \in L^{\heartsuit\leq 2}$. If there exists $u'v'$ that $u \in u'^{\heartsuit\leq 2}$ and $v \in v'^{\heartsuit\leq 2}$, then we are done. Otherwise in the generation of $uv$ from $u'v'$ there is a duplication transgressing the border between the two words.

Similarly as above, this is interesting only in the configuration $ca|b$, where $|$ denotes the border between $u'$ and $v'$ (or rather between the two intermediate words generated from them). The result of this duplication is $caba|b$. Let us call the word on the left $u''$. No further duplications transgressing the border can be necessary, since $(caba)^{\heartsuit\leq 2} b^{\heartsuit\leq 2} = (cabab)^{\heartsuit\leq 2}$. Thus for all words $u$ here we have either $[u]_\sim = [u']_\sim$ or $[u]_\sim = [u'']_\sim$. Thus also here the increase of the index of $\sim$ compared to $\sim_L$ preserves finiteness, and thus the resulting language is regular by Theorem 1, if the original language was regular. □

It appears possible to extend this proof technique to 3-bounded duplication under use of the fact that over two-letters the longest square-free word has length 3. While we leave this case open here, over an alphabet of only two letters things are not as complicated. To see this we first state a result that relates bounded and unbounded duplication. This will then allow us to state the closure of regular languages under these variants of duplication.

For the remainder of this section, → will denote the derivation relation of the string-rewriting system $R = \{a \to aa, b \to bb, ab \to abab, ba \to baba\}$, which generates the language $w^{\heartsuit\leq 2}$ for any word $w \in \{a, b\}$.

Lemma 5 *For every word* $u \in \{a, b\}^*$ *we have* $ab \xrightarrow{*} abubab$, $ab \xrightarrow{*} abuaab$, *and* $ab \xrightarrow{*} abuab$.

*Proof.* We prove this statement by induction on the length of $u$. For $|u| = 0$ the three derivations

$$\begin{aligned}
ab &\overset{ab\to abab}{\longrightarrow} abab \overset{b\to bb}{\longrightarrow} abbab = abubab\\
ab &\overset{ab\to abab}{\longrightarrow} abab \overset{a\to aa}{\longrightarrow} abaab = abuaab\\
ab &\overset{ab\to abab}{\longrightarrow} abab \qquad\qquad\;\; = abuab
\end{aligned}$$

show us that the lemma holds. So let us suppose it holds for all words, which are shorter than a number $n$. Any word $u$ of length $n$ has a factorization either as $va$ or $vb$ for a word $v$ of length $n - 1$. For this word $v$ the Lemma holds by our assumption. But then for $u = va$ the derivations

$$\begin{aligned}
ab &\xrightarrow{*} abvab \overset{ab\to abab}{\longrightarrow} abvabab = abubab\\
ab &\xrightarrow{*} abvaab \overset{a\to aa}{\longrightarrow} abvaaab = abuaab\\
ab &\xrightarrow{*} abvaab \qquad\qquad\qquad\;\; = abuab
\end{aligned}$$

and for $u = vb$ the derivations

$$\begin{aligned}
ab &\xrightarrow{*} abvbab \overset{b\to bb}{\longrightarrow} abvbbab = abubab\\
ab &\xrightarrow{*} abvbab \overset{a\to aa}{\longrightarrow} abvbaab = abuaab\\
ab &\xrightarrow{*} abvbab \qquad\qquad\qquad\; = abuab
\end{aligned}$$

show us that the lemma holds also for $u$ and thus for all words. □

Proposition 6 *Over an alphabet of two letters we have* $w^{\heartsuit\leq n} = w^{\heartsuit\leq 2}$ *and consequently* $w^{\heartsuit} = w^{\heartsuit\leq 2}$ *for all words* $w$ *and for* $n \geq 2$.

*Proof.* From Lemma 5 we know that $ab \xrightarrow{*} abuab$ holds for every word $u$, and applying this to the initial factor $ab$ in $abu$ we obtain $abu \xrightarrow{*} abuabu$. Just interchanging the letters $a$ and $b$ everything still is valid, and thus we see that also $bau \xrightarrow{*} baubau$ holds.

Now we prove that $aau \xrightarrow{*} aauaau$. If $u \in a^*$, then the statement is obviously true. Otherwise there is at least one $b$ in $u$, and therefore $u$ can be factorized as $u = a^m bv$ for some word $v$ and an integer $m \geq 0$. Now the derivation

$$aau = aa^m(ab)v \overset{\text{Lemma 5}}{\longrightarrow} aa^m abvabv \xrightarrow{*} aaa^m bvaaa^m bv = aauaau$$

shows that the statement above holds. Interchanging the letters again provides us with the dual statement $bbu \xrightarrow{*} bbubbu$.

Because any word $z$ longer than 1 has to start with either $ab$, $ba$, $aa$, or $bb$, this shows that we can always obtain by duplications of length at most 2 the word $zz$ from $z$ and thus $w^{\heartsuit^{\leq n}} \subseteq w^{\heartsuit^{\leq 2}}$. On the other hand, every duplication relation $\heartsuit^{\leq n}$ for $n \geq 2$ includes the relation $\heartsuit^{\leq 2}$ and so does $\heartsuit$. This suffices to prove that for all $n > 1$ we have $w^{\heartsuit^{\leq n}} = w^{\heartsuit^{\leq 2}}$, and $w^{\heartsuit} = w^{\heartsuit^{\leq 2}}$ immediately follows from this, because in any derivation the length of duplications used is bounded. □

Combining the results of this section we are now able to state the closure of regular languages under duplication.

Proposition 7 *The class of regular languages over two-letter alphabet is closed under $n$-bounded duplication and under general duplication.*

*Proof.* Proposition 4 states that regular languages are closed under 2-bounded duplication over any alphabet, and from Proposition 6 we see that in the two-letter case for any $n > 1$ the $n$-bounded and general duplication operations are equivalent to the 2-bounded one. □

## 4 Closure of Context-free Languages

When we speak about context-free languages, there is no difference between alphabets of size 2 and 3. It is already known that languages $w^{\heartsuit^{\leq n}}$ are always context-free [8]. By further refining the push-down automaton used in that proof, we can establish the closure of context-free languages under bounded duplication.

Proposition 8 *The class of context-free languages is closed under bounded duplication.*

*Proof.* We will show this by constructing a Push-Down Automaton in a way rather analogous to the one used in earlier work for the bounded duplication closure of a single word [8]. There the PDA reduces the results of duplications $uu$ to their origin $u$ and matches the reduced string against the original word. Here, we also have to simulate a second PDA accepting the context-free input language. This can be done, because of the two components reducing duplications and accepting the original language, the latter one does not need to access the stack ever, while the first one is working. With this sketch of the proof idea we now proceed to the technical details.

We start out from a PDA $M$, which accepts the language $L$. Let the PDA be $M = [Q, \Sigma, \Gamma, \varphi, q_o, \bot]$, where $Q$ is the set of states, $\Sigma$ the tape alphabet, and $\Gamma$ the stack alphabet. $\varphi : Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \rightarrow Q \times \Gamma^*$ is the state transition function; i.e. we allow transitions without reading input and we always take the topmost symbol off the stack replacing it by an arbitrary number of stack symbols. $q_0$ is the start state, and $\bot$ marks the stack's bottom. The acceptance mode does not really need to be specified, since any common acceptance condition will carry over to the new PDA.

We now define the PDA $A$, which accepts $L^{\heartsuit\leq n}$. The state set is $S := Q \times (\underline{\Sigma} \cup \Sigma)^{\leq n} \times \Sigma^{\leq n}$, where $\underline{\Sigma} := \{\underline{a} : a \in \Sigma\}$ is a marked copy of the tape alphabet. States $s \in S$ we will denote in the way $s = q|^{u}_{v}$, where $q \in Q$, $u \in (\underline{\Sigma} \cup \Sigma)^{\leq n}$ is called the *match*, and $v \in \Sigma^{\leq n}$ the *memory*; then $q_0|^{\lambda}_{\lambda}$ is the start state of $S$. The stack alphabet is $\Gamma' := \Gamma \cup (\underline{\Sigma} \cup \Sigma)^{\leq n}$. The tape alphabet $\Sigma$ and bottom-of-stack marker $\perp$ are as for $M$. What remains to be defined is the transition function $\delta$. We first define the part

$$\delta(q|^{\lambda}_{\lambda}, x, \gamma) := (q'|^{\lambda}_{\lambda}, \alpha) \text{ where } \varphi(q, x, \gamma) = (q', \alpha) \tag{1}$$

for $x \in \Sigma \cup \{\lambda\}$, $\gamma \in \Gamma$, and $\alpha \in \Gamma^*$. We see that when guess and memory are empty, $A$ works just as $M$; we will see that these are the only transitions changing the component from $Q$ of $A$'s states. Thus the simulation of $M$ and the undoing of duplications, which uses match and memory leaving the component from $Q$ unchanged, are done more or less independently. The next kind of transition makes a guess that the following letters on the input tape are the result of a duplication. Transitions

$$\delta(q|^{u}_{v}, x, \gamma) := (q|^{w}_{v}, u\gamma)$$

are defined for any words $u \in (\Sigma \cup \underline{\Sigma})^{\leq n}$ and $v, w \in \Sigma^{\leq n}$. Whatever is in the match is put on the stack to continue processing later. Note that the word $u$ is put on the stack as a single symbol.

Next $A$ checks whether the input continues with $ww$. This is done by matching the guess twice against the input, which is read, the first time underlining it in the guess, then undoing this. When both are matched, our PDA should continue as if there was one occurrence of $w$ left on the input tape. However, both are already read. Thus we put $w$ into the memory and read from there as if it was the input tape. Since in this construction the contents of the memory are thought to be situated in front of the input tape contents, nothing is ever read from the input tape, while the memory is not empty. For both situations all transitions are defined in parallel.

The variables used in the definition are quantified as follows: $q \in Q$, $x \in \Sigma$, $u, v, z \in \Sigma^*$, $\gamma \in \Gamma'$, $\beta \in \Gamma$, and $w \in \underline{\Sigma}^* \cdot \Sigma^* \cup \Sigma^* \cdot \underline{\Sigma}^*$ with $|w| \leq n$. Further, all catenations of words and letters are supposed to be no longer than $n$, and underlining a word from $\Sigma^*$ shall signify the corresponding word over $\underline{\Sigma}$ obtained by underlining all the individual letters.

$$\begin{array}{lll}
\delta(q|^{\underline{z}xu}_{\lambda}, x, \gamma) := (q|^{\underline{zx}u}_{\lambda}, \gamma) & \text{and} & \delta(q|^{\underline{z}xu}_{xv}, \lambda, \gamma) := (q|^{\underline{zx}u}_{v}, \gamma) \\
\delta(q|^{\underline{xu}}_{\lambda}, x, \gamma) := (q|^{x\underline{u}}_{\lambda}, \gamma) & \text{and} & \delta(q|^{\underline{xu}}_{xv}, \lambda, \gamma) := (q|^{x\underline{u}}_{v}, \gamma) \\
\delta(q|^{z\underline{xu}}_{\lambda}, x, \gamma) := (q|^{zx\underline{u}}_{\lambda}, \gamma) & \text{and} & \delta(q|^{z\underline{xu}}_{xv}, \lambda, \gamma) := (q|^{zx\underline{u}}_{v}, \gamma) \\
\delta(q|^{z\underline{x}}_{\lambda}, x, w) := (q|^{w}_{zx}, \lambda) & \text{and} & \delta(q|^{z\underline{x}}_{xv}, \lambda, w) := (q|^{w}_{zxv}, \lambda) \\
\delta(q|^{z\underline{x}}_{\lambda}, x, \beta) := (q|^{\lambda}_{zx}, \beta) & \text{and} & \delta(q|^{z\underline{x}}_{xv}, \lambda, \beta) := (q|^{\lambda}_{zxv}, \beta)
\end{array}$$

Finally, also the simulation of $M$ must be possible, when the memory is not empty. Thus for $x \in \Sigma$ we define the analogue to transitions defined in 1 for reading from the tape:

$$\delta(q|_{xv}^{\lambda}, \lambda, \gamma) := (q'|_{v}^{\lambda}, \alpha) \text{ where } \varphi(q, x, \gamma) = (q', \alpha).$$

There are no other transitions than the ones defined above. We now prove that $L^{\heartsuit \leq n} \subseteq L(A)$. For this, one observation is essential, whose truth should be immediately comprehensible after what we have already said about the way that $A$ works.

Lemma 9 *If from a state $q|_{\lambda}^{u}$ with $vw$ next on the working tape and $\gamma$ on the stack there exists an accepting computation for $A$, then from $q|_{v}^{u}$ with $w$ next on the working tape and $\gamma$ on the stack there also exists an accepting computation.*

With this we can prove $L^{\heartsuit \leq n} \subseteq L(A)$ by induction on the number of duplications used to reach a word $w \in L^{\heartsuit \leq n}$ from a word $u \in L$. While neither $u$ nor the number need to be unique, they both must exist for all words in $L^{\heartsuit \leq n}$. So let $u$ be a word such that $w \in u^{\heartsuit \leq n}$ via $k+1$ duplications. Then there exists a word $u'$ reachable from $u$ via $k$ duplications such that $u' \heartsuit^{\leq n} w$.

Let us suppose that all words, which can be generated by $k$ duplications from words in $L$, are accepted by $A$; then $u' \in L(A)$, and there exists an accepting computation of $A$ for $u'$, let us call it $\varXi$. Further let $i, \ell$ be integers such that the duplication of the factor of length $\ell$ starting at position $i$ in $u'$ results in $w$, i.e. $w = u'[1 \ldots i-1]u'[i \ldots i+\ell-1]^2 u'[i+\ell \ldots |u'|]$. Obviously $A$ can on input $w$ follow the computation $\varXi$ on the prefix $u'[1 \ldots i-1]$. Let us call the configuration reached in the step before reading the next input letter $\xi$ and let its state be $s$. Then in $s$ the memory is empty, otherwise $A$ would not read from the input tape.

Now instead of following $\varXi$ further, we guess the duplication of $u'[i \ldots i+\ell-1]$ and reduce it in the manner described above. At the end of this process we will have reached a state equal to $s$ except for the fact that its memory contains $u'[i \ldots i+\ell-1]$. On the tape we have left $u'[i+\ell \ldots |u'|]$. By Lemma 9 there is an accepting computation for this configuration if there is one for $\xi$. Since $\varXi$ is such an accepting computation, also $w$ is accepted by $A$.

Further, $A$ can obviously simulate any computation of $M$ and thus $L(M) \subseteq L(A)$, i.e. all words reachable by zero duplications are in $L(A)$. Thus also the basis for our induction is given and we have $L^{\heartsuit \leq n} \subseteq L(A)$.

We do not prove in detail that $L(A) \subseteq L^{\leq n}$. The two parts of $A$, the one deterministically reducing duplications and the one simulating the original PDA $M$ work practically independently, as the corresponding state sets are disjoint and separated by the match being filled or not. From these facts $L(A) \subseteq L^{\leq n}$ should be comprehensible rather easily. □

Of course, the same construction works for any finite set of factors that can be duplicated, and we immediately obtain a corollary.

Corollary 10 *The class of context-free languages is closed under the operation of uniformly bounded duplication.*

For general duplication this proof technique does not apply, because over three letters there is no $n$ such that $(abc)^{\heartsuit} = (abc)^{\heartsuit \leq n}$. In fact, $n$-bounded duplication grows more powerful with every increase of $n$. Here we will use the following two notions: a word $w$ is *square-free*, if it does not contain any non-empty factor of the form $uu = u^2$; $w$ is *circular square-free*, if the same holds true for $w$ written along a circle, or equivalently if $ww$ contains no square shorter than itself.

Proposition 11 *For two integers $m$ and $n$ with $17 < m < n$ the inclusion $(abc)^{\heartsuit \leq m} \subset (abc)^{\heartsuit \leq n}$ is proper.*

*Proof.* First we show that for every square-free word $u$ over three letters starting with $abc$ there exists a word $v$, such that $uv \in (abc)^{\heartsuit \leq k}$ for $k \geq 4$. This word is constructed from left to right in the following manner. The first three letters are $abc$ and thus do not need to be constructed.

The fourth letter is created by going from the third letter left to the last occurrence of this desired letter. Since $abc$ is a prefix of the word all three letters do have such an occurrence. Now the factor from this rightmost occurrence to the third letter is duplicated. In this way the fourth letter of the new word becomes the desired one. Then we move to the fifth letter, obtain it by duplicating the factor reaching back till its rightmost occurrence, and so on.

The last occurrence of any letter in the part of $u$ already constructed can be at most four positions from the last, because there are only two more different letters and the longest square-free word over two letters has length three. Of course, if in some step more than one letter of $u$ is produced, the process can advance to the next wrong one without further duplications.

We will illustrate this construction with a short example. From $abc$ we construct $abcbacb$ as a prefix. Underlining signals the factor duplicated to obtain the following word, the horizontal bar signals the end of the prefix of $abcbacb$ constructed at the respective point. $a\underline{bc} \rightarrow \underline{abcb}|c \rightarrow ab\underline{cba}|bcbc \rightarrow abcbacb|abcbc$

We now establish some bounds for the number of additional symbols produced. Since $abc$ is already there, $|u| - 3$ letters need to be constructed. In every step at most $2k - 1$ letters of $u$ can be constructed, because $u$ is square-free; thus at least one letter is added to $v$. At the same time at most $2k - 1$ letters are added to $v$, since no useless duplications are done. Thus we have $|u| - 3 \leq |v| \leq (|u| - 3)(2k - 1)$. Of course, every circular square-free word is square-free and can be constructed in this way, too. Starting from lengths of 18, such a word always exists [2].

Now we construct in this way a circular square-free word $w$ of length $n$ as a prefix of a word $wv'$ in $(abc)^{\heartsuit \leq n}$. We can expand this prefix to $w^i$ in $i - 1$ steps for any given $i \geq 1$ by the rule $w \rightarrow ww$, so all $w^i v'$ are in $(abc)^{\heartsuit \leq n}$. Further, $w^i$ contains no squares shorter than $2n$, because $w$ is circular square-free. Thus for constructing the same prefix in $(abc)^{\heartsuit \leq m}$ also the bounds $|w^i| - 3 \leq |v| \leq (|w^i| - 3)(2m - 1)$ for the corresponding suffix $v$ apply. For big enough $i$ the shortest such $v$ will be longer than $v'$. Thus such a $w^i v'$ cannot be in $(abc)^{\heartsuit \leq m}$, while it is in $(abc)^{\heartsuit \leq n}$. □

## 5 Conclusions

Thus the problem, which has received most attention in investigations on duplication remains open: Is the general duplication closure of a word over three letters always context-free? Probably this is equivalent to asking whether context-free languages are closed under general duplication. Our investigations on the length-bounded case may have shed some more light on the nature of the problem, though.

Another problem is raised by Proposition 11: Are the inclusions $(abc)^{\heartsuit\leq m} \subset (abc)^{\heartsuit\leq n}$ for $m < n$ proper also for $n \leq 17$? Or do these inclusions hold only when a circular square-free word of the corresponding length exists?

## References

1. R. Book and F. Otto: *String-Rewriting Systems.* Springer, Berlin, 1988.
2. J.D. Currie: *There are Ternary Circular Square-free Words of Length $n$ for $n \geq 18$.* In: Electric Journal of Combinatorics, 9(1) N10, 2002.
3. J. Dassow, V. Mitrana and Gh. Păun: *On the Regularity of Duplication Closure.* Bull. EATCS 69, 1999, pp. 133–136.
4. M.A. Harrison: *Introduction to Formal Language Theory.* Reading, Mass., 1978.
5. P. Leupold: *n-Bounded Duplication Codes.* Proceedings of the ICALP-Workshop on Words, Avoidability, Complexity, Turku 2004. Technical Report 2004-07, Laboratoire de Recherche en Informatique d'Amiens, Amiens 2004.
6. P. Leupold, C. Martín Vide and V. Mitrana: *Uniformly Bounded Duplication Languages.* In: Discrete Applied Mathematics Vol 146, Iss 3, 2005, pp. 301–310.
7. P. Leupold and V. Mitrana: *Uniformly Bounded Duplication Codes.* Submitted.
8. P. Leupold, V. Mitrana and J. Sempere: *Languages Arising from Gene Repeated Duplication.* In: Aspects of Molecular Computing. Essays in Honour Tom Head on his 70th Birthday. LNCS 2950, Springer Verlag, Berlin, 2004, pp. 297–308.
9. C. Martín-Vide and Gh. Păun: *Duplication Grammars.* In: Acta Cybernetica 14, 1999, pp. 101–113.
10. V. Mitrana and G. Rozenberg: *Some Properties of Duplication Grammars.* In: Acta Cybernetica 14, 1999, pp. 165–177.
11. A. Salomaa : *Formal Languages.* Academic Press, Orlando, 1973.
12. M.-W. Wang: *On the Irregularity of the Duplication Closure.* Bull. EATCS 70, 2000, pp. 162–163.