Tiling Periodicity [⋆]

Juhani Karhumäki[1], Yury Lifshits[2], and Wojciech Rytter[3,4]

[1] Turku University, Finland, karhumak@utu.fi
[2] Steklov Institute of Mathematics at St.Petersburg, Russia,
yura@logic.pdmi.ras.ru
[3] Department of Mathematics and Informatics, Copernicus University, Torun, Poland
[4] Warsaw University, Poland, rytter@mimuw.edu.pl

Abstract. We contribute to combinatorics and algorithmics of words by introducing new types of periodicities in words. A *tiling period* of a word w is partial word u such that w can be decomposed into several disjoint parallel copies of u, e.g. $a \diamond b$ is a tiling period of $aabb$. We investigate properties of tiling periodicities and design an algorithm working in $O(n \log(n) \log\log(n))$ time which finds a tiling period of minimal size, the number of such periods and their compact representation. The combinatorics of tiling periods differs significantly from that for classical full periods, for example unlike the classical case the same word can have many different primitive tiling periods. We consider also a related new type of periods called in the paper *multi-periods*. As a side product of the paper we solve an open problem posted by T. Harju.

1 Introduction

The number p is a *full period* (period, in short) of a word w of length n iff $p|n$ and $w_i = w_{i+p}$ whenever both sides are defined. Define by $period(w)$ the shortest nonzero full period of w.

In this paper we extend the notion of a full period. Namely, we are interested in tilings of a word where the tiles themselves may contain "transparent" letters. A **tiler** (or partial word) is a word over $\Sigma \cup \{\diamond\}$ alphabet, where $\diamond$ is a special transparent (or undefined) letter. In other words, a tiler is a sequence of connected words (blocks) with gaps between the blocks. The **size** of a tiler is the number of defined symbols.

Imagine that we have several copies of a tiler printed on transparencies. Then, this tiler is a period for some word, if we can put these copies into the stack such that they form a single connected word without overlapping of visible letters. Thus, a tiler S is called a **tiling period** of an (ordinary) word T if we can split T into disjoint parallel copies of S satisfying the following: **(a)** All defined (visible) letters of S-copies match the text letters; **(b)** Every text letter is covered by

[⋆] Support by grant N 8206039 of the Academy of Finland for the first author, by grants INTAS 04-77-7173 and NSh-8464.2006.1 for the second author, and by grant of the Polish Ministry of Science and Higher Education N 206 004 32/0806 is gratefully acknowledged.

exactly one defined (visible) letter. Similarly we define a tiling period x of a tiler x: x consists of several disjoint copies of y. The word (a tiler) is **primitive** if it has no proper tiling period. The tiling period of x is **minimal** iff it has minimal size.

Example. For example $a\ a\ \ \diamond \diamond\ b\ b$ is a tiling period of $aaaabbbbaaaabbbb$ and $a \diamond b \diamond \diamond \diamond \diamond \diamond c \diamond d$ is a tiling period of $aabbaabbccddccdd$. Both have size 4, the first one is not primitive.

Problem. We investigate basic properties of tiling periodicity and comparing it to the classical notion. We address the following questions: (1) How to enlist all possible tiling periods? (2) Does every word have a unique *primitive* tiling period? (3) How to find all tiling periods of minimal size? (4) How many periods can a word of length n have? (5) What is the relation between the primitive classical and the primitive tiling periods? We have several reasons to be interested in tiling periodicity. Firstly of all, it is a natural generalization of classical notion, i.e. any full period is also a tiling period. Secondly, in the case when a tiling period is relatively small, we can describe a long word by just its tiling period and the length. Hence, we get a new class of words with low Kolmogorov complexity. Tiling periodicity might be useful for the new text compression methods (especially for generalizing run-length encoding). Note here, that the ratio between the "size" of tiling period and the length of classical full period may be arbitrarily small. Next, the notion of tiling periodicity provides a geometrical intuition about structure of the text. We have a conjecture that tiling periodicity is not expressible by word equations. Yet another motivation for studying tiling periodicity is a hope for applications in pattern discovery in some real data.

There are natural sources of tiling periodicity when considering multidimensional $n_1 \times \cdots \times n_d$ rectangles. Let every integer point in it be colored. We sort all points in lexicographical order of their coordinates and write down all their colors in a single sequence. Assume that the initial rectangle was tiled by a smaller one $m_1 \times \cdots \times m_d$ where $m_1 | n_1, \ldots, m_d | n_d$ and every copy of the smaller rectangle was colored in the same way. Then the color sequence for the bigger rectangle has a tiling period. We comment this example later after introducing some useful terminology. Automatically generated texts and XML-files might be another possible sources of tiling periodicity.

Our results. Tilling periodicity looks very simple and natural, but up to our knowledge it was never formulated before in its whole generality. We introduce a partial order on tiling periods and discover that contrary to the classical case there might be several incomparable primitive tiling periods. This helps to disprove a *common subtiler* conjecture by Tero Harju. However we prove that every primitive tiling period of a word T is also a tiling period of the primitive full period of T. This property tells us that tiling periodicity lives "inside" classical one. Finally, we present an algorithm which in $O(n \log(n) \log\log(n))$ time finds a tiling period of minimal size, the number of such periods and their compact representation.

We present a complete hierarchy of all possible tiling periods. In particular we get a recursive formula for computing function $L(n)$ that gives the maximal number of tiling periods for a unary word of length n. The value $L(n)$ might be even more than the text length. Actually, Bodini and Rivals obtain this recursive formula a few months earlier: their paper [3] was submitted in January 2006, while our results were reported only in May 2006 [10]. Here we keep our proof since (comparing to [3]) it construct explicit one-to-one correspondence between tilers and length factorizations, and introduces *levels* in the set of all tilers.

Related results Prior to other work only tilings of a *unary* word were considered [3]. In the paper [13] authors present an algorithm for finding all tilers that have at least q (quorum parameter) matches with the text. Another related notion is *cover* [1]: a word C is a cover for a word T if any letter of T belongs to some occurrence of C in T. One of the most important results related to periodicity is a theorem by Fine and Wilf [5]. They studied the necessary and sufficient condition under which from p-periodicity and q-periodicity we can derive $\gcd(p, q)$-periodicity (recall that gcd is the greatest common divisor). We tried to prove the similar property for tiling periods. Surprisingly, it does not hold. Even two tiling periods of the same text may have no common subperiod. Our attempt to generalize the notion of periodicity is not the first one. Recently, Simpson and Tijdeman generalized Fine and Wilf theorem for multidimensional periodicity [17]. Berstel and Boasson [9] followed by Shur and Konovalova (Gamzova) [16, 15] and Blanchet-Sadri [2] extensively studies partial words and their periodicity properties. However, overlapping of periods (where every letter of the text is covered exactly once) were not considered and therefore in their terms only partial words may have partial periods (i.e. periods with gaps). In the paper [8] borders of a partial word are studied. Katona and Szàsz introduced in [11] a sort of tiling periodicity in two dimensions. They consider tilers consisting only of two letters. The notion of periodicity was generalized for infinite words under the name *almost periodic sequences*, see e.g. [14].

2 Properties of Tiling Periodicity

We say that one tiler S is *smaller* than another tiler Q, and write $S \prec Q$, if S is a proper tiling period of Q. In the case of full periodicity any text has a single primitive full period. We are interested in the following question: is it true that for every ordinary word there exists a unique primitive tiling period (i.e. it is smaller than any other tiling period)? It can be reformulated in an alternative way: do any two tiling periods have a common tiling “subperiod”? Surprisingly, the answer is negative. Figure 1 presents the shortest known example T (24 letters) with two incomparable periods (the proof of their primitiveness is omitted in this version).

We can get more incomparable tiling periods. Let $T_{A_1B_1}$ and $T_{A_2B_2}$ be obtained from T by just using different letters. We construct the text T_2 by replacing every A in T by $T_{A_1B_1}$ and every B in T by $T_{A_2B_2}$. Now we can describe four incomparable tiling periods for T_2. The text T_2 has length 24^2. Group all letters to 24 blocks of 24 letters. Choose 12 of blocks using one of two partial words above. Then inside each block also keep only 12 letters using either first or second tiling period (the same in every block). We can repeat the construction several times. The text T_k has size 24^k and has 2^k incomparable periods. Asymptotically it has $n^{\frac{\log 2}{\log 24}} > \sqrt[5]{n}$ incomparable primitive tiling periods.

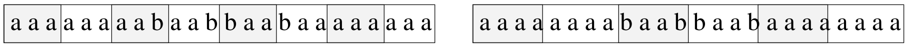
a a a a a a a a b a a b b a a b a a a a a a a a
a a a a a a a a b a a b b a a b a a a a a a a a

Fig. 1. Two primitive tiling periods of an example word, both of size 12.

In 2003 in his lecture course [6] Tero Harju asked the following question. Assume that some colored cellular figure has tiling by one pattern and by another one. Is it always true that there exists the third one such that both first two can be tiled by that third one? This question is motivated by studying defect effect in combinatorics of words [7]. Our example shows that the answer is negative even for one-dimensional (but disconnected!) figures.

We consider now the situation when the number of tilers in a word is maximal, this happens for a unary word of length n (i.e only one character is used). Then we reformulate tiling periodicity in the algebraic terms (like $s_i = s_{i+p}$ for full periodicity). This reformulation helps us to prove that any primitive tiling period is smaller than the primitive full period and to compute all tiling periods of minimal size.

Lemma 1. *Take any tiling period P. Then all continuous blocks in P are of the same size s and the length of any gap in P is a multiple of s.*

Proof. Indeed, in the text tiling the second copy of P is shifted by the size of the first block s_1. Hence, for avoiding overlapping all other blocks are smaller or equal than s_1. Assume now, that the block b is the first one which is *strictly* smaller than s_1. Then the gap between b in the first copy of P and b in the second copy of P is smaller than s_1. Hence, that gap cannot be filled by anything. Every gap in P is filled by several blocks of some copies of P. Hence it must be a multiple of s.

Theorem 1. *There is a one-to-one correspondence between tiling periods of a unary word and decompositions $n = n_1 \cdot \ \ldots \ \cdot n_k$, where $n_2, \ldots, n_k \geq 2$.*

Proof. At first, we describe a set of tiling periods (hierarchial construction) for the word of length n over the unary alphabet. Then we prove that this set is complete, i.e. contains all possible tiling periods.

We divide the set of all periods in levels. Every period in our system is associated with a special *code.* The whole text itself is the only period from

0-level and has the code n. For every decomposition $n = n_1 \cdot n_2$ with $n_2 \geq 2$ the block of size n_1 is a period from 1-level with a code $n_1 \cdot n_2$. The 0-level and 1-level actually represent all classical (i.e. connected) full periods.

We now explain how to construct a period Q of $k+1$-level from a period P of $k-1$-level with the code $n_1 \cdot \ldots \cdot n_k$. Take any decomposition $n_1 = m_1 \cdot m_2 \cdot m_3$ with $m_2, m_3 \geq 2$. Through all our construction the first number of a code is equal to block size of a tiling period (all continuous blocks have equal sizes in our constructions). To construct the new tiler Q we take every block of P, divide it into m_3 groups of size $m_1 \cdot m_2$, and in every such group keep only first m_1 letters. We can notice that P can be tiled by m_2 copies of Q (with shifts $0, m_1, \ldots, m_1 \cdot (m_2 - 1)$). Hence, Q is also a tiling period for the text with the code $m_1 \cdot m_2 \cdot m_3 \cdot n_2 \cdot \ldots \cdot n_k$. Note, that our construction maintains the inequality $n_2, \ldots, n_k \geq 2$ for all codes.

We now prove that any tiling period is included in our construction. Consider a tiler P. Let s be the block size and $g \cdot s$ be the length of the first gap. The *plain power* of P is defined as the union of P-copies shifted by $0, s, 2s, \ldots, g \cdot s$.

Claim: The plain power of any tiling period P is also a tiling period.

Proof: take a text splitting in P-copies. Let s be the block size and $g \cdot s$ be the length of the first gap. We divide all copies of P into groups of $g+1$ copies in the following way. Consider all P-copies from the left to the right. The gap between first two blocks of the first copy can be filled *only by first* blocks of other P-copies. These copies together with the first one form the first group. Now assume that we already formed several such groups. Consider the first copy of P unused so far. Look at its first gap. All P-copies from the previous groups has long (at least $s \cdot (g+1)$) continuous blocks. Hence, this gap is also filled by new, still unused P-copies. Since we process all copies from the left one to the right one, all of them contribute to this gap filling by their first block. Therefore these copies form the next group we need. Every group itself is exactly a plain power of P. Hence, the initial text has also splitting in the plain power copies. The claim is proved.

Assume now that there exists some P outside of our construction. Then there also exists some tiling period P' such that (1) it is outside of our hierarchy and (2) its plain power Q is included in our hierarchy. Let us derive a contradiction from that. Indeed, let $n_1 \cdot \ldots \cdot n_k$ be the code of Q, let s be the block size of P' and $s \cdot g$ be the length of its first gap. Then n_1 is the block size of Q and it is a multiple of $s \cdot (g+1)$. Let $m_1 = s, m_2 = g+1, m_3 = n_1/(s \cdot (g+1))$. Now we see that P' is in fact included in our hierarchy as a tiling period with the code $m_1 \cdot m_2 \cdot m_3 \cdot n_2 \cdot \ldots \cdot n_k$. Therefore, our hierarchy is complete.

Corollary 1. Let $L(n)$ be the number of tiling periods for the word of length n over a unary alphabet. The theorem above states that $L(n)$ is equal to the number of factorizations $n = n_1 \cdot \ldots n_k$, where $n_2, \ldots, n_k \geq 2$. By grouping all decompositions by the rightmost factor we obtain the recurrence: $L(1) = 1$; $L(n) = 1 + \sum_{d|n, d \neq n} L(d)$.

Remark. Two related sequences are included in the On-line Encyclopedia of Integer Sequences [18] maintained by N.J.A. Sloane. The sequence A067824 is defined by the formula in Corollary 1. The sequence A107736 is the number of polynomials p with coefficients in $\{0,1\}$ that divide x^n-1 and such that $(x^n-1)/((x-1)p(x))$ has all coefficients in $\{0,1\}$. But this is exactly the number of tiling periods for the unary text of the length n. Indeed, multiplying p by some polynomial with coefficients in $\{0,1\}$ we are trying to split n "ones" in several parallel copies of p without overlapping. Till August 2006 Encyclopedia indicated that "A067824 and A107736 agree at least on first 300 terms, but no proof of equivalence is known". After recent work [3], Theorem 1 and Corollary 1 give a new independent proof of their equality. Indeed, the number of polynomials is equal to the number of tiling periods, the number of tiling periods is equal to the number of factorizations (Theorem 1), the number of factorizations satisfies the recursive formula (Corollary 1). The function $L(n)$ and the number of factorizations also appear in Knuth's book [12]. However, no closed formula for $L(n)$ is known so far.

Definition. Assume we count positions starting from 0. For a divisor p of n, by p-block we mean a subword of the form $x[i\cdot p..(i+1)\cdot p-1]$, where $0 \le i \le (n-1)/p$. We say that a word $T = T_0 \dots T_{n-1}$ has a **multi-period** (a,b) (or a period a ranged by b) if $b|n$, and all b-blocks have a full period a. Observe that the word corresponding to this period can be different in each block. Classical full period p coincides with the multi-period (p,n).

Lemma 2. *A word T has a tiling period with the code $n_1\cdot \dots \cdot n_{2k}$ or $n_1\cdot \dots \cdot n_{2k+1}$ iff it has multi-periods respectively*

$$(n_1, n_1n_2), \dots, (\prod_{i=1}^{2k-1} n_i, \prod_{i=1}^{2k} n_i) \quad \text{or} \quad (n_1, n_1n_2), \dots, (\prod_{i=1}^{2k-1} n_i, \prod_{i=1}^{2k} n_i).$$

Proof. We first describe **Step 1:** from tiling period P to multiperiodicity. We use induction over tiler's level. Consider the corresponding text tiling. Recall that all copies of P can be divided into groups of size n_2 with internal shifts $0, n_1, \dots, n_1(n_2-1)$. If we divide the whole text into the blocks of size n_1n_2, every block is covered by P-copies from the same groups, and therefore it is n_1-periodic (inside the block). We proved (n_1, n_1n_2)-multiperiodicity. All others follow from the induction hypothesis for the plain power of P.
Step 2: from multi-periods to a tiling periodicity.

Consider the *top* multi-period $(\prod_{i=1}^{2k-1} n_i, \prod_{i=1}^{2k} n_i)$. Let us consider the text tiling by tiler Q with the code $(\prod_{i=1}^{2k-1} n_i) \cdot n_{2k}$ for the first statement of lemma and Q with the code $(\prod_{i=1}^{2k-1} n_i) \cdot n_{2k} \cdot n_{2k+1}$ for the second case. Directly from this top multiperiodicity every copy of Q has the same letters on the same places. Hence, Q is a tiling period for the text. Continuing the reasoning, with the help of the second multiperiodicity we find another tiling period R inside Q. Finally, the last multiperiodicity gives us the tiling period with the code we promised in the lemma's statement.

Lemma 3. *If the text T has a full period p and a multi-period (a,b), then either $b|p$ or the text has also full period $gcd(a,p)$.*

Proof. Take any letter T_i. We are going to make several moves of size $\pm p$ and $+a$ for reaching position $i+\gcd(a,p)$. We want to be on the same character every time and hence we can not make a move of size $+a$ from the last a-blocks in every b-block. As we know from extended Euclid algorithm, there exist integers k and l such that $\gcd(a,p)=ka-lp$. We will use the following *greedy* strategy. If we are able to make a move of size $+a$ we do this. Otherwise we try to make several moves of size $\pm p$. After making exactly k moves of size $+a$ we just make all the remaining moves of size $\pm p$ and we are done. We cannot follow greedy strategy only if for some position $j<p$ from all points $j, j+p, \dots j+(\frac{n}{p}-1)p$ we cannot jump $+a$. This means that all that points belong to the last a-groups in b-blocks. Suppose now that p is not divided by b. Then $u=\gcd(p,b)\le\frac{b}{2}$. Since $p|n$ and $b|n$, among the numbers $j \bmod b, \dots, j+(\frac{n}{p}-1)p \bmod b$, there exists all residues h modulo b such that $h\equiv j (\bmod\ u)$. Hence, one of these values is smaller than u which does not exceed $b/2$. But this means that the corresponding point does not belong to the last a-group in the b-block.

Theorem 2. *Any primitive tiling period Q of word T is also a tiling period for the primitive full period of T.*

Proof. We use induction over the text length. In the case of the one-letter-text theorem is true. Let now p be the length of the full period and (a,b) be the top multiperiodicity from the code of Q (here we use Lemma 1). From $p=n$ theorem follows immediately. Assume now that $p<n$. By hierarchial construction Q consists of some letters from the first a-block in every b-blocks in the text. Let us apply Lemma 2. If $b|p$, then we can produce a new tiling period Q' restricting Q to the first p symbols in the text. By p-periodicity Q can be split in several parallel copies of Q' with shifts $0,p,2p,\dots,(\frac{n}{p}-1)p$. We get the contradiction against the primitivity of Q. Therefore the text has full period $\gcd(a,p)$. But p is the minimal full period. Hence, $p|a$. Since the text is p-periodic, it is also a-periodic and b-periodic. That means that we can again restrict Q only for the first a-block. Either we get a new smaller tiling period Q' (and that gives us contradiction) or $b=n$ and all letters of Q belong to the first a-block. We now see that both full period p and the tiling period Q are periods for the first a-block. Since $a\le b/2\le n/2$, we can apply induction hypothesis.

Corollary 2. Take any tiling period Q and any full period p. Then they have a common “tiling subperiod”.

Proof. Consider a primitive tiling period Q' that is smaller than Q. Consider the primitive full period p'. From folklore we know that $p'|p$. By Theorem 2 Q' is a tiling period for the first p'-block of the text. Hence, Q' is the required tiling subperiod for Q and p.

Remark. Using the technique from Lemma 1, Lemma 2 and Theorem 2 it is possible to prove that the primitive tiling period is unique for $n=2^k$. We just suggest the proof scheme here. Take two primitive incomparable tiling periods. Consider their top multiperiodicities. Apply the reasoning of Lemma 2. Either one of the periods is not primitive or these multiperiodicities are the same. Going down we prove either their equivalence or their non-primitivity.

3 Algorithm *Compute-Minimal-Tilers*

We define the size of tiling period as the number of defined letters in it. In the algorithm we use the fact that for every tiling period there is a corresponding chain of *embedded* multiperiodicities $(a_1, b_1), \ldots, (a_k, b_k)$. By "embedded" we mean that $b_i | a_{i+1}$ holds for every i. Notice that the size of a tiling period is equal to $n \prod_{i=1}^{k} \frac{a_i}{b_i}$

```
ALGORITHM Compute-Minimal-Tilers (w)
  1. Construct the acyclic tiling graph $\mathcal{G}$ of multi-periods.
     (a) $V = \{(a,b) : 1 \le a < b \le n,\ \&\ a|b\ \&\ b|n\ \&\ MultiPeriod(a,b)\ )$
     (b) $E = \{(a,b) \to (c,d) : b|c\}$.
     (c) Assign the weight $b/a$ to every node $(a,b) \in V$.
  2. Find in $\mathcal{G}$ a path $\pi$ having maximal product $val(\pi)$ of its node-weights;
  3. Output $tiler(w,\pi)$ (of size $\frac{n}{val(\pi)}$).
```

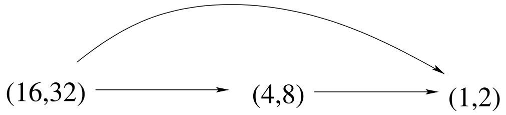

Fig. 2. The graph $\mathcal{G}$ for the word $w = aabbaabbccddccddaabbaabbccddccdd$. The path with maximal product of weights is $\pi :\ (16,32) \to (4,8) \to (1,2)$. We have $val(\pi) = 8$ and $tiler(w,\pi) = a\diamond b\diamond\diamond\diamond\diamond\diamond\diamond c\diamond d$. The size of $tiler(w,\pi)$ equals $\frac{32}{8}$, since $n = |w| = 32$.

We describe how to construct the tiler $tiler(w,\pi)$ corresponding to a path π of multi-periods:

$$\pi = (p_1, q_1) \to (p_2, q_2) \to (p_3, q_3) \to \ldots (p_k, q_k).$$

For each p_i-block of w we replace all symbols which are not in the first q_i block in this block by $\diamond$. Then from the resulting word we remove all ending $\diamond$'s.

Remark. The following $O(n)$ size *max-paths graph* representation $\mathcal{G}' \subseteq \mathcal{G}$ of all tiling periods of minimal size can be constructed with additional linear work (since the graph is extremely small). For each vertex v of $\mathcal{G}$ compute all outgoing edges which are on a maximal path starting from v. The graph of all edges on maximal-product paths represents all tiling periods of minimal size. In particular we can compute easily the number of such periods.

We describe now an efficient implementation. Let w be a word of length n and let $Divisors(n)$ denotes the set of divisors of n. The subwords are given by their starting-ending positions in w. Recall that by p-block we always mean a subword of length p with a starting position being multiple of p. Denote by $period(w)$ the size of the smallest full period of a word w. The basic operation in the algorithm *Compute-Minimal-Tilers* is the boolean function $MultiPeriod(q,p)$, called *main query*, which can be expressed in terminology of the blocks as follows:

MultiPeriod(q,p):

for given $p,q \in Divisors(n)$ check if each p-block has a full period q.

Lemma 4. *Assume we can preprocess the word in time $O(F(n))$ to compute each query MultiPeriod in logarithmic time. Then the algorithm works in time $O(n + F(n))$,*

Proof. By $d(n)$ we denote the number of divisors of n. It is known that $d(n) = O(n^\epsilon)$, for any constant $\epsilon > 0$. Hence the number of nodes and edges is $O(n)$. The construction of the graph can be done in linear time after preprocessing. Computation of a maximal path in time $O(n)$ is very easy, since it is an acyclic graph with linear number of edges.

Theorem 3. [Fast-Preprocessing]
We can preprocess the word in $O(n \log n \log\log n)$ time in such a way that each MultiPeriod query can be answered in constant time.

The algorithm *Compute-Minimal-Tilers* is doing a sublinear number of MultiPeriod queries, hence the theorem implies immediately the following speed-up result.

Theorem 4. *The minimal-size tiler of a word can be found in $O(n \log n \log\log n)$ time.*

We now show the proof of Theorem 3. Firstly we introduce and concentrate on *small queries*. A **small query** operation is to compute for a given subword u of w (given by interval in w) the value $period(u)$.

We say that a natural number is a *2-power* number if it is a power of two. We use the idea of *basic factors*: the subwords with 2-power lengths. Denote by $subword_k(i)$ the subword of size 2^k starting at position i in a given word w. We can add suitable number of endmarkers to guarantee that for each original position we have subword of the corresponding length.

Define the table $NEXT$ such that for each $0 \le k \le \log n,\ 0 \le i < n$ the value of $NEXT[k,i]$ is the first position $j > i$ such that $subword_k(i) = subword_k(j)$. If there is no such j then $NEXT[k,i] = -1$.

Our basic data structure is the dictionary of basic factors, we refer to [4] for detailed definition. Denote this data structure by $DBF(w)$. For each position i in w and $0 \leq k \leq \log n$ we have a unique label $NAME[k,i] \in [1..n]$ such that $subword_k(i) = subword_k(j) \Leftrightarrow NAME[k,i] = NAME[k,j]$.

Lemma 5. *The tables NAME and NEXT can be computed in $O(n \log n)$ time.*

Proof. The table $NAME$ is the basic part of $DBF(w)$ and can be computed within required complexity using Karp-Miller-Rosenberg algorithm, see [4].

We show how to compute the table $NEXT$. Let us fix $k \leq \log n$. We sort lexicographically the pairs $(NAME[k,i], i)$. Then the block of elements in the sorted sequence with the same first component r gives the increasing sequence of positions i with the same value $NAME[k,i] = r$. Let $SORTED_1[j],\ SORTED_2[j]$ be the first and second component of the j-th pair in the sorted sequence.

We execute:

```
for $i := 0$ to $n-1$ do NAME[k,i]:=-1;
for $j := 1$ to $n-1$ do
    if $SORTED_1[j-1] = SORTED[j]$ then
        $NEXT[k, SORTED_2[j-1] := SORTED_2[j]$;
```

The radix sorting of pairs of integers can be done in linear time for each k. We have logarithmic number of k's, hence the whole computation of $NEXT$ takes $O(n \log n)$ time. This completes the proof.

Lemma 6. [Small Queries]
Assume the tables NAME and NEXT are already computed. Then for any subword u of w (given by interval in w) we can compute period(u) in $O(\log n)$ time.

Proof. We show now how we compute *period*(u) for $u = w[p..q]$. We can check if u has a full period of length $u/2$ or $u/3$ in constant time, since we can check equality of constant number of subwords in constant time. The DBF data structure allows to check in constant time equality of subwords which lengths are not necessarily 2-powers (decomposing them into ones which are, possibly overlapping each other).

Now let us go to smaller candidate periods, assume *period*$(u) \leq |u|/4$. Let us take the prefix v of u which size is a largest power 2^k such that $|u|/4 \leq 2^k \leq |u|/2$.

Claim. Let $u = w[p,q]$, if *period*$(w[p..q]) \leq |v|/4$ then *period*(v) is equal to $NEXT[k,p] - p$.

The claim follows from the fact that period(u) in this case is a size of a primitive word v such that u is a full power of v. This primitive word can start an occurrence only at positions which are multiples of v, due to primitivity. Hence the first such internal position after p should be equal to $|v| =$ *period*(u). We can verify this “candidate period” in $O(\log n)$ time using NAME table. This completes the proof of the lemma.

We define now the following data structure. For each $p \in Divisors(n)$ define $LCM[p]$ as the least common multiple of the smallest full periods of all p-blocks of w.

Lemma 7. *We can precompute the table LCM in $O(n \log n \log\log n)$ time.*

Proof. Assume we constructed tables $NEXT$ and $NAME$. We can do it within required complexity due to Lemma 5. Then for each $p \in Divisors(n)$ we can compute the set of periods of p-blocks in time $O(p \log n)$ due to Lemma 6. Then we compute the lowest common multiple of all these periods in time $O(p \log n)$ for each p. Now the thesis follows from the well known number-theory fact that

$$\sum_{p \in Divisors(n)} p \;=\; O(n \log\log n)$$

This completes the proof of the lemma.

We can finish now the proof of Theorem 3. We know that $MultiPeriod(p, q) = true$ iff $LCM[p]$ is a divisor of q. This property can be checked in constant time using the precomputed values of $LCM[p]$. Consequently, this completes the proof of Theorem 3.

4 Directions for Further Work

There are a lot of natural, important and perhaps not so difficult questions we can suggest for further work on tiling periodicity. They are summarized in the list below.

1. Introduce and study *not full* tiling periodicity, approximate tiling periodicity, tilings by two (or more) partial words. This kind of tilings might be even more useful for compression purposes.
2. Calculate how often do words have proper tiling periods for various models of random words. Compare the answer with the classical case. Characterize the equivalent of primitive words.
3. Is it true that all primitive tiling periods are minimal-size tiling periods?

References

1. A. Apostolico, M. Farach, and C. S. Iliopoulos. Optimal superprimitivity testing for strings. *Inf. Process. Lett.*, 39(1):17–20, 1991.
2. F. Blanchet-Sadri. Periodicity on partial words. *Computers and Mathematics with Applications*, 47(1):71–82, 2004.
3. O. Bodini and E. Rivals. Tiling an interval of the discrete line. In *CPM'06*, LNCS 4009, pages 117–128. Springer-Verlag, 2006.
4. M. Crochemore and W. Rytter. *Jewels of stringology: text algorithms*. World. Sc., 2003.

5. N. Fine and H. Wilf. Uniqueness theorems for periodic functions. *Proc. Amer. Math. Soc.*, 16:109–114, 1965.
6. T. Harju. Defect theorem, lecture notes of ”Combinatorics of words” Tarragona course, 2002/2003.
7. T. Harju and J. Karhumäki. Many aspects of defect theorems. *Theor. Comput. Sci.*, 324(1):35–54, 2004.
8. C. S. Iliopoulos, M. Mohamed, L. Mouchard, K. Perdikuri, W. F. Smyth, and A. K. Tsakalidis. String regularities with don't cares. *Nord. J. Comput.*, 10(1):40–51, 2003.
9. L. Boasson J. Berstel. Partial words and a theorem of Fine and Wilf. *Theor. Comput. Sci.*, 218(1):135–141, 1999.
10. J. Karhumäki and Y. Lifshits. Tiling periodicity, May 2006. Dagstuhl seminar “Combinatorial and Algorithmic Foundations of Pattern and Association Discovery”, `http://kathrin.dagstuhl.de/files/Materials/06/06201/06201.LifshitsYury1.Slides.pdf`.
11. G. O. H. Katona and D. O. H. Szász. Matching problems. *J. of Combinatorial Theory Ser B*, 10(1):60–92, 1971.
12. D. Knuth. *The Art of Computer Programming, Volume 4, Fascicle 3: Generating All Combinations and Partitions.* Addison-Wesley, 2005.
13. N. Pisanti, M. Crochemore, R. Grossi, and M.-F. Sagot. Bases of motifs for generating repeated patterns with wild cards. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 2(1):40–50, 2005.
14. Y. Pritykin and M. Raskin. Almost periodicity and finite automata. Technical Report available at `http://lpcs.math.msu.su/~pritykin/files/apfinaut.zip`, 2007.
15. A. M. Shur and Y. V. Gamzova. Partial words and the periods' interaction property. *Izvestiya RAN*, 68(2):199–222, 2004.
16. A. M. Shur and Y. V. Konovalova. On the periods of partial words. In *MFCS'01*, LNCS 2136, pages 657–665. Springer-Verlag, 2001.
17. R. J. Simpson and R. Tijdeman. Multi-dimensional versions of a theorem of Fine and Wilf and a formula of Sylvester. *Proc. Amer. Math. Soc.*, 131:1661–1667, 2003.
18. N. J. A. Sloane. The on-line encyclopedia of integer sequences. `http://www.research.att.com/~njas/sequences`.