Computing Equality-Free String Factorisations

Markus L. Schmid

Fachbereich IV – Abteilung Informatikwissenschaften, Universität Trier,
D-54286 Trier, Germany, MSchmid@uni-trier.de

Abstract. A factorisation of a string is equality-free if each two factors are different; its size is the number of factors and its width is the maximum length of any factor. To decide, for a string w and a number m, whether w has an equality-free factorisation with a size of at least (or a width of at most) m are NP-complete problems. We further investigate the complexity of these problems and also study the converse problems of computing a factorisation that is to a large extent not equality-free, i. e., a factorisation of size at least (or width at most) m such that the total number of different factors does not exceed a given bound k.

Keywords: String factorisations, NP-hard string problems, FPT

1 Introduction

Many classical hard string problems can be defined in terms of factorisations of strings that satisfy certain properties. For example, the well-known problem of computing the shortest common superstring of given strings $w_1, \ldots, w_k$ (see, e. g., [1]) asks whether there exists a short string x that, for every i, $1 \leq i \leq k$, has a factorisation $u_i \cdot w_i \cdot v_i$. Since a string w is a subsequence of a string u if u has a factorisation $v_1 \cdot v_2 \cdots v_k$ and w has a factorisation $v_{i_1} \cdot v_{i_2} \cdots v_{i_n}$ with $1 \leq i_1 < \ldots < i_n \leq k$, the famous Longest Common Subsequence and Shortest Common Supersequence problems can as well be described in terms of factorisations. Another example of a string problem that has recently attracted much attention is the problem to decide for two words x and y and a given k whether they have factorisations $u_1 \cdot u_2 \cdots u_k$ and $v_1 \cdot v_2 \cdots v_k$, respectively, such that $(u_1, \ldots, u_k)$ is a permutation of $(v_1, \ldots, v_k)$, i. e., the Minimum Common String Partition problem. See [1] for a survey on the multivariate analysis of NP-hard string problems.

In this paper we are concerned with so-called equality-free factorisations, recently introduced in [2, 3]. A factorisation $u_1 \cdot u_2 \cdots u_k$ is *equality-free* if every factor is distinct, i. e., $|\{u_1, u_2, \ldots, u_k\}| = k$. In [2, 3], Condon et al. investigate the problem of deciding whether a given string w has an equality-free factorisation of *width* at most m, where the width is the maximum length of any factor.[1] A motivation for this problem comes from gene synthesis. Since it is only possible to produce short pieces of DNA (so-called *oligo fragments*) artificially, longer

[1] This problem is also mentioned in [1]; furthermore, in [6], the hardness of computing an equality-free factorisation with only palindromes as factors is investigated.

DNA sequences are usually obtained by a self-assembly of many oligos into the desired DNA sequence; thus, the task is to find the right oligos for successful self-assembly. Computing equality-free factorisations with bounded width is an abstraction of this problem: the width bound represents the necessity for short oligos and the equality-freeness models the condition that each two oligos must not be too similar in order to not hybridise with each other (see [2,3] for more details). This problem is NP-complete, even if the width bound is 2 or the alphabet is binary (see [3]). We revisit this problem and show that it is fixed-parameter tractable if both the width bound and the alphabet size are parameters.

If instead of a small width, we are looking for an equality-free factorisation with a large *size*, i. e., a large number of factors, then we obtain a different NP-complete problem (see [4]). This variant is motivated by injective pattern matching with variables (which is identical to the special case of solving word equations, where the left side of the equation does not contain variables and different variables must be replaced by different words), see [4] for more details. We show that computing equality-free factorisations with large size is fixed-parameter tractable if parameterised by the size bound. However, the question whether the problem remains hard for fixed alphabets is still open.

We also consider the converse of computing equality-free factorsations, i. e., computing factorisations that are to a large extent not equality-free (or *repetitive*). Our measure of repetitiveness is the number of different factors in the factorisation. If this number is small (in comparison to the size or width of the factorisation), then many factors are repeated. This yields an interesting combinatorial question in its own right: how many different words are needed in order to cover a given word? Furthermore, it is motivated by data compression, since a factorisation with many repeated factors can be used in order to compress a word, e. g., by using a dictionary of the different factors. We can show that deciding on whether a word w has a factorisation of width at most m and with at most k different factors is NP-complete, even if $m = 2$. On the other hand, if k or the alphabet size is a constant, then the problem can be solved in polynomial time. In contrast to this, if m is a lower bound on the size of the factorisation, then the problem can be solved in polynomial time if either m, k or the alphabet size is a constant, but it is open, whether the problem is NP-complete in general.

As a tool for proving some of our main results, we also investigate the problem of deciding whether a given word w has an equality-free factorisation with only factors from a given finite set F of words. It turns out that this problem is NP-complete even for binary alphabets. However, it is in FPT if $|F|$ is a parameter and in P if we drop the equality-freeness condition.

Due to space constraints, not all results are formally proven.

2 Basic Definitions

Let $\mathbb{N} = \{1, 2, 3, \ldots\}$. By $|A|$, we denote the cardinality of a set A. Let Σ be a finite alphabet of *symbols*. A *word* or *string* (over Σ) is a sequence of symbols from Σ. For any word w over Σ, $|w|$ denotes the length of w and ε denotes the

empty word, i. e., $|\varepsilon| = 0$. The symbol Σ^+ denotes the set of all non-empty words over Σ and $\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$. For the *concatenation* of two words w_1, w_2 we write $w_1 \cdot w_2$ or simply $w_1 w_2$. For every symbol $a \in \Sigma$, by $|w|_a$ we denote the number of occurrences of symbol a in w. We say that a word $v \in \Sigma^*$ is a *factor* of a word $w \in \Sigma^*$ if there are $u_1, u_2 \in \Sigma^*$ such that $w = u_1 v u_2$. If $u_1 = \varepsilon$ or $u_2 = \varepsilon$, then v is a *prefix* (or a *suffix*, respectively) of w. For every i, $1 \leq i \leq |w|$, by $w[1..n]$ we denote the prefix of w with length n. As a convention, in this work every set of words is always a finite set.

By the term *trie*, we refer to the well-known ordered tree data structure for representing sets of words.

Factorisations For any word $w \in \Sigma^+$, a *factorisation of* w is a tuple $p = (u_1, u_2, \ldots, u_k) \in (\Sigma^+)^k$, $k \in \mathbb{N}$, with $w = u_1 u_2 \ldots u_k$. Every word u_i, $1 \leq i \leq k$, is called a *factor* (*of* p) or simply *p-factor*. For the sake of readability, we sometimes represent a factorisation $(u_1, u_2, \ldots, u_k)$ in the form $u_1 \centerdot u_2 \centerdot \ldots \centerdot u_k$.

Let $p = (u_1, u_2, \ldots, u_k)$ be an arbitrary factorisation. We define the following parameters: $\mathsf{sf}(p) = \{u_1, u_2, \ldots, u_k\}$ (the *set of factors*), $\mathsf{s}(p) = k$ (the *size*), $\mathsf{c}(p) = |\mathsf{sf}(p)|$ (the *cardinality*) and $\mathsf{w}(p) = \max\{|u_i| \mid 1 \leq i \leq k\}$ (the *width*). A factorisation p is *equality-free* if $\mathsf{s}(p) = \mathsf{c}(p)$.

Problems We now define the different problems to be investigated in this work.

Equality-Free Factor Cover (EFFC)
Instance: A word w and a set F of words represented as a trie.
Question: Does there exist an equality-free factorisation p of w with $\mathsf{sf}(p) \subseteq F$?

Maximum Equality-Free Factorisation Size (MaxEFF-s)
Instance: A word w and a number m, $1 \leq m \leq |w|$.
Question: Does there exist an equality-free factorisation p of w with $\mathsf{s}(p) \geq m$?

Maximum Repetitive Factorisation Size (MaxRF-s)
Instance: A word w, numbers m, $1 \leq m \leq |w|$, and k, $1 \leq k \leq |w|$.
Question: Does there exist a factorisation p of w with $\mathsf{s}(p) \geq m$ and $\mathsf{c}(p) \leq k$?

The problems MaxEFF-s and MaxRF-s where m is interpreted as an upper bound on the width instead of a lower bound on the size are denoted by MinEFF-w and MinRF-w. In the remainder of the paper, the symbol m is reserved as the bound on the size or width (depending on the problem under consideration) of the factorisation and k as the bound on the cardinality of the factorisation, respectively. For any problem K from above and any fixed alphabet Σ, K_Σ denotes the problem K, where the input word is over Σ.

We shall now illustrate these definitions with an example.

Example 1. Let $p = \mathtt{aab} \centerdot \mathtt{ba} \centerdot \mathtt{cba} \centerdot \mathtt{aab} \centerdot \mathtt{ba} \centerdot \mathtt{aab}$ be a factorisation. We note that $\mathsf{sf}(p) = \{\mathtt{aab}, \mathtt{ba}, \mathtt{cba}\}$, $\mathsf{s}(p) = 6$, $\mathsf{c}(p) = 3$ and $\mathsf{w}(p) = 3$. The factorisation p is not equality-free. Furthermore, $(\mathtt{abbcbaabbc}, 6) \in$ MaxEFF-s (witnessed by $\mathtt{a} \centerdot \mathtt{bb} \centerdot \mathtt{c} \centerdot \mathtt{ba} \centerdot \mathtt{ab} \centerdot \mathtt{bc}$), whereas $(\mathtt{abbcbaabbc}, 7) \notin$ MaxEFF-s. On the other hand, $((\mathtt{aabbcc})^2, m) \in$ MinEFF-w if and only if $m \geq 2$, whereas $((\mathtt{aabbcc})^3, m) \in$ MinEFF-w if and only if $m \geq 3$.

Parameterised Complexity We consider decision problems as languages over some alphabet Γ. A *parameterisation* (*of* Γ) is a polynomial time computable

mapping $\kappa : \Gamma^* \to \mathbb{N}$ and a *parameterised problem* is a pair (Q, κ), where Q is a problem (over Γ) and κ is a parameterisation of Γ. We usually define κ implicitly by describing which part of the input is the parameter. A parameterised problem (Q, κ) is *fixed-parameter tractable* if there is an *fpt-algorithm* for it, i. e., an algorithm that solves Q on input x in time $\mathcal{O}(f(\kappa(x)) \times p(|x|))$ for recursive f and polynomial p. The class of fixed-parameter tractable problems is denoted by FPT. Note that if a parameterised problem becomes NP-hard if the parameter is set to a constant, then it is not in FPT unless P = NP. For detailed explanations on parameterised complexity, the reader is referred to [5].

3 Main Results

We begin this section with some preliminary observations that are necessary for proving some of the main results.

If in an equality-free factorisation p, we join one of the longest factors with one of its neighbours, then the resulting factorisation is still equality-free and has size $\mathsf{s}(p) - 1$.

Observation 1. A word w has an equality-free factorisation p with $\mathsf{s}(p) \geq m$, $m \in \mathbb{N}$, if and only if it has an equality-free factorisation p' with $\mathsf{s}(p') = m$.

We can check whether a factorisation p of a word w is equality-free in time $\mathcal{O}(|w|)$ by inserting all factors in a trie and checking for each factor if it is already contained in the trie. If a set F of words is given as a trie, then we can check in a similar way whether or not $\mathsf{sf}(p) \subseteq F$ in time $\mathcal{O}(|w|)$.

Observation 2. Let $w \in \Sigma^+$ and let p be a factorisation of w. It can be decided in time $\mathcal{O}(|w|)$ whether or not p is equality-free.

Observation 3. Let $w \in \Sigma^+$ and let F be a set of words over Σ represented as a trie. It can be decided in time $\mathcal{O}(|w|)$ whether or not $\mathsf{sf}(p) \subseteq F$.

The following result is straightforward, but it nevertheless contributes to our understanding of the complexity of the considered problems.

Proposition 1. *The problems* EFFC, MaxEFF-s, MinEFF-w, MaxRF-s *and* MinRF-w *are in* FPT *with respect to parameter* $|w|$.

3.1 The Problem EFFC

As mentioned in the introduction, the problem EFFC is not our main concern. However, its investigation, as we shall see, yields some valuable insights with respect to equality-free factorisations and we also obtain an algorithm that shall be used later in order to prove tractability results with respect to the problems MaxRF-s and MinRF-w.

We first show that EFFC is NP-complete, even for fixed binary alphabets.

Theorem 1. *Let* Σ *be an alphabet with* $|\Sigma| = 2$. *Then* EFFC_Σ *is* NP*-complete.*

Proof. Since we can guess a factorisation p and check in polynomial time whether it is equality-free and $\mathsf{sf} \subseteq F$, EFFC$_\Sigma$ is in NP. Let (w, m) be an instance of MinEFF-w_Σ and let F be the set of all factors of w of length at most m. We note that $|F| \leq \sum_{i=1}^{m} |w| - (i-1) \leq m \times |w|$ and F can be constructed in time $\mathcal{O}(|F| \times m)$. The word w has an equality-free factorisation p with $\mathsf{w}(p) \leq m$ if and only it has an equality-free factorisation p' with $\mathsf{sf}(p') \subseteq F$. Since MinEFF-$\mathsf{w}_\Sigma$ is NP-complete (see [3]), EFFC$_\Sigma$ is NP-complete as well. □

In addition to the alphabet size, the cardinality of the given set F of factors is another natural parameter and we can show that the hardness is not preserved if we bound $|F|$ by a constant (in contrast to bounding $|\Sigma|$ (Theorem 1)).

Theorem 2. *The Problem* EFFC *can be solved in time* $\mathcal{O}(|w|^{|F|+1})$.

Theorem 2 is obtained by an analysis of a brute-force algorithm, coupled with the observation that if an equality-free factorisation p satisfies $\mathsf{sf}(p) \subseteq F$, then its size must be bounded by $|F|$. As we shall see next, a more sophisticated approach, which relies on encoding the different factors of F as single symbols and factorisations as words over these symbols, yields an fpt-algorithm for EFFC with respect to the parameter $|F|$ (note that this does not make the algorithm of Theorem 2 obsolete, since for large $|F|$ it might still be faster than the fpt-algorithm).

Theorem 3. *The Problem* EFFC *can be solved in time* $\mathcal{O}(|w| \times (2^{|F|} - 1) \times |F|!)$.

Proof. Let $w \in \Sigma^*$ and $F = \{u_1, u_2, \ldots, u_\ell\}$ be an instance of EFFC. Furthermore, let $\Gamma = \{1, 2, \ldots, \ell\}$, let $h : \Gamma^* \to \Sigma^*$ be a morphism defined by $h(i) = u_i$, $i \in \Gamma$, and let $S = \{v \in \Gamma^+ \mid |v|_i \leq 1, i \in \Gamma\}$. There exists a word $v = j_1 j_2 \ldots j_m \in S$ with $h(v) = w$ if and only if $p = (u_{j_1}, u_{j_2}, \ldots, u_{j_m})$ is an equality-free factorisation of w with $\mathsf{sf}(p) \subseteq F$. Therefore, we can solve EFFC by checking for each word $v \in S$ whether or not $h(v) = w$, which can be done in time $\mathcal{O}(|w| \times |S|)$. We conclude the prove by observing that $S = \bigcup_{\Gamma' \subseteq \Gamma, \Gamma' \neq \emptyset} S_{\Gamma'}$, where $S_{\Gamma'} = \{v \in \Gamma'^+ \mid |v|_i = 1, i \in \Gamma'\}$. Since $|S_{\Gamma'}| = |\Gamma'|!$ and the sets $S_{\Gamma'}$ are pairwise disjoint, we have $|S| = \sum_{\Gamma' \subseteq \Gamma, \Gamma' \neq \emptyset} |\Gamma'|! \leq \sum_{i=1}^{2^\ell - 1} \ell! = (2^\ell - 1) \times \ell!$. Since $\ell = |F|$, we obtain a total running time of $\mathcal{O}(|w| \times (2^{|F|} - 1) \times |F|!)$. □

Next, we investigate the impact of the equality-freeness condition itself, i. e., we consider the problem FC, which is identical to EFFC with the only difference that the factorisation p of w with $\mathsf{sf}(p) \subseteq F$ does not need to be equality-free. This problem is similar to the problem Exact Block Cover (recently investigated by Jiang et al. in [8]), which differs from FC only in that instead of a set we are given a sequence of factors and every factor of the sequence has to be used exactly once (in particular, this coincides with the variant of Minimum Common String Partition where the partition of one of the two strings is already fixed). While Exact Block Cover is NP-complete (see [8]), FC can be solved in polynomial time by dynamic programming. This demonstrates that it is really the equality-freeness condition that makes EFFC hard and, in addition,

we obtain a useful tool to devise algorithms for solving variants of the problems MaxRF-s and MinRF-w later on in Section 3.3.

Theorem 4. *The problem* FC *can be solved in time* $\mathcal{O}(|F| \times |w|^2)$.

Proof. We define a dynamic programming algorithm. Let w be a word and $F = \{u_1, u_2, \ldots, u_\ell\}$. For every n, m, $1 \leq m \leq n \leq |w|$, let $T[n,m] = 1$ if there exists a factorisation p of size m of $w[1..n]$ with $\mathsf{sf}(p) \subseteq F$ and $T[n,m] = 0$ otherwise. Obviously, (w, F) is a positive instance of FC if and only if $T[|w|, m] = 1$ for some m, $1 \leq m \leq |w|$. We can now solve FC on instance (w, F) by computing all the $T[n,m]$, $1 \leq m \leq n \leq |w|$, in the following way.

In time $\mathcal{O}(|w| \times |F|)$, we first construct a table S with $|w|$ rows and ℓ columns with $S[n,i] = 0$, $1 \leq n \leq |w|$, $1 \leq i \leq \ell$. Then, by using the Knuth-Morris-Pratt algorithm [9], for every i, $1 \leq i \leq \ell$, we set $S[n,i] = 1$ if u_i is a suffix of $w[1..n]$. Since the Knuth-Morris-Pratt algorithm has running time $\mathcal{O}(|w|+|u_i|)$, building up this table can be done in time $\sum_{i=1}^{\ell}(|w| + |u_i|) \leq \sum_{i=1}^{\ell} 2|w| = \mathcal{O}(|F| \times |w|)$. Then, for every n, m, $1 \leq m \leq n \leq |w|$, we initialise $T[n,m] = 0$, which requires time $\mathcal{O}(|w|^2)$, and, for every i, $1 \leq i \leq \ell$, we set $T[|u_i|, 1] = 1$ if $S[|u_i|, i] = 1$, which requires time $\mathcal{O}(|F|)$. We note that, for every n, m with $2 \leq m \leq n \leq |w|$, $T[n,m] = 1$ if and only if there exists a word $u_i \in F$ that is a suffix of $w[1..n]$ (i. e., $S[n,i] = 1$) with $T[n-|u_i|, m-1] = 1$. Thus, for every n, m, $2 \leq m \leq n \leq |w|$, we can compute $T[n,m]$ in time $\mathcal{O}(|F|)$, provided that all $T[n', m-1]$, $n' < n$, have already been computed, which is satisfied if we iterate over m, $2 \leq m \leq |w|$, in an outer loop and over n, $m \leq n \leq |w|$, in an inner loop. Hence, all the elements $T[n,m]$, $1 \leq m \leq n \leq |w|$, are computed in time $\mathcal{O}(|F| \times |w|^2)$. □

3.2 The Problems MaxEFF-s and MinEFF-w

In this section, we investigate the problems MinEFF-w and MaxEFF-s. Their NP-completeness is established in [2] and [4], respectively, but in [3] it is additionally shown that MinEFF-w remains NP-complete even if the bound on the width is 2 or the alphabet is fixed and binary. In particular, this means that, unless P = NP, MinEFF-w is not in FPT with respect to parameter m or $|\Sigma|$. However, if we let both m and $|\Sigma|$ be parameters at the same time, then MinEFF-w is fixed-parameter tractable:

Theorem 5. MinEFF-w_Σ *can be solved in time* $\mathcal{O}(m^{m^2 \times |\Sigma|^m + 2} \times |\Sigma|^m)$.

Proof. Let (w, m) be an instance of MinEFF-w_Σ. For every ℓ, $1 \leq \ell \leq m$, there are $|\Sigma|^\ell$ words of length ℓ and therefore $\sum_{\ell=1}^{m} |\Sigma|^\ell \leq m \times |\Sigma|^m$ words of length at most m. Consequently, if a factorisation p satisfies $\mathsf{w}(p) \leq m$, then $\mathsf{s}(p) \leq m \times |\Sigma|^m$. Furthermore, if for an equality-free factorisation p of w we have $\mathsf{s}(p) \leq m \times |\Sigma|^m$ and $\mathsf{w}(p) \leq m$, then $|w| \leq m^2 \times |\Sigma|^m$. Hence, if $|w| > m^2 \times |\Sigma|^m$ (which can be checked in time $\mathcal{O}(m^{m^2 \times |\Sigma|^m + 2} \times |\Sigma|^m)$), then there is no equality-free factorisation p of w with $\mathsf{w}(p) \leq m$. If, on the other hand, $|w| \leq m^2 \times |\Sigma|^m$, then we can enumerate all factorisations of w that have a width of at most m and check for each such factorisation whether or not it is

equality-free in time $\mathcal{O}(|w|) = \mathcal{O}(m^2 \times |\Sigma|^m)$ (see Observation 2). Since there are at most $m^{|w|} \leq m^{m^2 \times |\Sigma|^m}$ such factorisations, the statement of the theorem follows. □

For the problem MaxEFF-s, i. e., deciding on the existence of an equality-free factorisation with a size of at least m (instead of a width of at most m), we encounter a slightly different situation. First of all, it is still an open problem whether MaxEFF-s remains NP-complete if the alphabet is fixed:

Open Problem 1. *Let Σ be an alphabet. Is* MaxEFF-s$_\Sigma$ NP*-complete?*

From an intuitive point of view, for the problem MinEFF-w, the bound on the width can conveniently be exploited in order to design gadgets for encoding an NP-hard problem (see [3] and also the proof of Theorem 12). A lower bound on the size seems to provide fewer possibilities for controlling the structure of the factorisation, which makes it difficult to express another NP-complete problem by MaxEFF-s (especially if we have only a constant number of symbols at our disposal). On the other hand, a constant alphabet does not seem to help in order to find an equality-free factorisation with a size of at least m in polynomial time.

However, if we consider m as a constant, then the problem is not NP-complete anymore; in fact, it is even fixed-parameter tractable with respect to m:

Theorem 6. *The problem* MaxEFF-s *can be solved in time* $\mathcal{O}((\frac{m^2+m}{2} - 1)^m)$.

Proof. Let (w, m) be an instance of MaxEFF-s. If $|w| \geq \Sigma_{i=1}^{m} i = \frac{m^2+m}{2}$, which can be checked in time $\mathcal{O}((\frac{m^2+m}{2} - 1)^m)$, then the factorisation $(u_1, u_2, \ldots, u_m)$ of w with $|u_i| = i$, $1 \leq i \leq m-1$, and $|u_m| = |w| - |u_1 u_2 \ldots u_{m-1}|$ is equality-free, since each two factors have a different length. If, on the other hand, $|w| \leq \frac{m^2+m}{2} - 1$, then we can enumerate all factorisations of size m of w in time $\mathcal{O}(|w|^{m-1})$ and, by Observation 2, check in time $\mathcal{O}(|w|)$ for each such factorisation whether or not it is equality-free. Since w has an equality-free factorisation of size at least m if and only if it has an equality-free factorisation of size exactly m (see Observation 1), this solves the problem MaxEFF-s in time $\mathcal{O}(|w|^m) = \mathcal{O}((\frac{m^2+m}{2} - 1)^m)$. □

3.3 The Problems MaxRF-s and MinRF-w

In this section, we investigate the problem of finding a factorisation of a word w with as few different factors as possible. Since (w) is always a solution, we also impose an upper bound on the width of the factorisation or a lower bound on its size. In a sense, a factorisation p of this kind is to a large extent repetitive, since if k is much smaller than $\mathsf{s}(p)$ or $\frac{|w|}{\mathsf{w}(p)}$, then many factors must be repeated.

We shall see that if k or $|\Sigma|$ are constants, then both MaxRF-s and MinRF-w can be solved in polynomial time. If, on the other hand, m is a constant, then MaxRF-s can be solved in polynomial time as well, whereas MinRF-w is NP-complete even for $m = 2$. Unfortunately, we are not able to answer whether MaxRF-s is NP-complete in general.

We now first investigate the problem MaxRF-s.

Theorem 7. *The problem* MaxRF-s *can be solved in time* $\mathcal{O}(k^2 \times |w|^{2k+3})$.

Proof. Let (w, m, k) be an instance of MaxRF-s with $m \leq |w|$ and $k \leq |w|$ (otherwise, it would be a negative instance). Let $F_w = \{u \mid u \text{ is a factor of } w\}$. For every $F \subseteq F_w$ with $|F| \leq k$, we run the algorithm defined in the proof of Theorem 4 on input (w, F). If $T[|w|, \ell] = 1$, for an ℓ, $m \leq \ell \leq |w|$, then there is a factorisation p of w with $\mathsf{s}(p) \geq m$ and $\mathsf{sf}(p) \subseteq F$; since $|F| \leq k$, this implies $\mathsf{c}(p) \leq k$. To carry out this procedure, we have to enumerate all subsets $F \subseteq F_w$ with $|F| \leq k$. Since $|F_w| \leq |w|^2$, for every ℓ, $1 \leq \ell \leq k$, there are at most $|F_w|^\ell \leq |w|^{2\ell}$ subsets $F \subseteq F_w$ with $|F| = \ell$. Thus, there are $\sum_{i=1}^{k} |w|^{2i} \leq k \times |w|^{2k}$ subsets to investigate. For each subset F, we run the algorithm of the proof of Theorem 4 in time $\mathcal{O}(|F| \times |w|^2)$, and check for every ℓ, $m \leq \ell \leq |w|$, whether or not $T[|w|, \ell] = 1$, which requires time $\mathcal{O}(|w|)$. Hence, the total running time of this procedure is $\mathcal{O}(k \times |w|^{2k} \times k \times |w|^3) = \mathcal{O}(k^2 \times |w|^{2k+3})$. □

From Theorem 7 we can conclude with moderate effort that MaxRF-s can also be solved in time that is exponential only in m or $|\Sigma|$. To this end, we observe that if, for an instance (w, m, k) of MaxRF-s, we have $k \geq |\Sigma|$, then splitting w in only factors of length 1 yields a factorisation p with $\mathsf{c}(p) \leq |\Sigma| \leq k$ and $\mathsf{s}(p) = |w| \geq m$, and if $k \geq m$, then any factorisation p of w of size m satisfies $\mathsf{c}(p) \leq m \leq k$ and $\mathsf{s}(p) \geq m$. If, on the other hand, k is bounded by $|\Sigma|$ or m, then the procedure used in the proof of Theorem 7 has a running time that is exponential only in $|\Sigma|$ or m, respectively, which yields the following results:

Theorem 8. *Let* Σ *be an alphabet. Then the problem* MaxRF-s_Σ *can be solved in time* $\mathcal{O}(|\Sigma|^2 \times |w|^{2|\Sigma|+1})$.

Theorem 9. *The problem* MaxRF-s *can be solved in time* $\mathcal{O}(m^2 \times |w|^{2m+1})$.

The probably most interesting question, which, unfortunately, is still open is whether the general version of MaxRF-s can also be solved in polynomial time.

Open Problem 2. *Is* MaxRF-s NP*-complete?*

We now turn to the problem MinRF-w. In an analogous way as done in the proof of Theorem 7, we can show that MinRF-w can be solved in time exponential only in k, too. The only difference is that instead of running the algorithm of Theorem 4 for every subset of the set of all factors of w, it is sufficient to only consider all subsets of the set of all factors of w that have a length of at most m.

Theorem 10. MinRF-w *can be solved in time* $\mathcal{O}(k^2 \times m^k \times |w|^{k+3})$.

In a similar way as Theorem 8 follows from Theorem 7, i. e., by bounding k in terms of $|\Sigma|$, we can conclude from Theorem 10 the next result.

Theorem 11. *Let* Σ *be an alphabet. Then the problem* MinRF-w_Σ *can be solved in time* $\mathcal{O}(|\Sigma|^2 \times m^{(|\Sigma|-1)} \times |w|^{|\Sigma|+2})$.

While for problem MaxRF-s it was also possible to bound k in terms of m, for MinRF-w, we can only observe that (w, m, k) must be a positive instance if $k \geq \lceil \frac{|w|}{m} \rceil$, but in case $k < \lceil \frac{|w|}{m} \rceil$, the algorithm of the proof of Theorem 10 has a running time exponential in $|w|$ and it does not seem possible to solely bound k in terms of m. We now justify this intuition by showing that MinRF-w is NP-complete, even if $m = 2$. First, we recall the hitting set problem (see [7]):

Hitting Set (HS)
Instance: $U = \{x_1, \ldots, x_\ell\}$, $S_1, \ldots, S_n \subseteq U$ and $q \in \mathbb{N}$.
Question: Does there exist $T \subseteq U$ with $|T| \leq q$ and $T \cap S_i \neq \emptyset$, $1 \leq i \leq n$?

We now give a reduction from HS to MinRF-w with $m = 2$. Let $(U, S_1, \ldots, S_n, q)$ be an instance of HS. We assume that, for every i, j, $1 \leq i < j \leq n$, $|S_i| = |S_j| = r$ (note that HS reduces to the variant where all sets S_i have the same cardinality r by adding $r - |S_i|$ new elements to every S_i). For the sake of concreteness, we assume $S_i = \{y_{i,1}, y_{i,2}, \ldots, y_{i,r}\}$, $1 \leq i \leq n$. We define an alphabet $\Sigma = U \cup \{\$_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq r-1\} \cup \{¢\}$ and a word $w = ¢¢\, v_1\, ¢\, v_2\, ¢ \ldots ¢\, v_n\, ¢$, where, for every i, $1 \leq i \leq n$, $v_i = y_{i,1}\$_{i,1}y_{i,2}\$_{i,2} \ldots \$_{i,r-1}y_{i,r}$. The following lemma states that this transformation from an HS instance to a word over Σ is in fact a reduction from HS to MinRF-w.

Lemma 1. *There exists a set $T \subseteq U$ with $|T| \leq q$ and $T \cap S_i \neq \emptyset$, $1 \leq i \leq n$, if and only if w has a factorisation p with $\mathsf{w}(p) \leq 2$ and $\mathsf{c}(p) \leq n(r-1) + q + 1$.*

Proof. We start with the *only if* direction and assume that there exists a set $T \subseteq U$ with $|T| \leq q$ and $T \cap S_i \neq \emptyset$, $1 \leq i \leq n$. We now construct a factorisation p of w with the desired properties. We let every single occurrence of ¢ be a factor of p; thus, it only remains to split every v_i, $1 \leq i \leq n$, into factors of size at most 2, which is done as follows. For every i, $1 \leq i \leq n$, let j_i, $1 \leq j_i \leq r$, be arbitrarily chosen such that $y_{i,j_i} \in T$ (since $T \cap S_i \neq \emptyset$, $1 \leq i \leq n$, such j_i exist). Then, for every i, $1 \leq i \leq n$, we factorise v_i into

$$y_{i,1}\$_{i,1} \cdot y_{i,2}\$_{i,2} \cdot \ldots \cdot y_{i,j_i-1}\$_{i,j_i-1} \cdot y_{i,j_i} \cdot \$_{i,j_i}y_{i,j_i+1} \cdot \ldots \cdot \$_{i,r-1}y_{i,r}\,.$$

Obviously, this results in a factorisation p of w with $\mathsf{w}(p) \leq 2$. Furthermore, $\mathsf{sf}(p)$ contains the factor ¢, at most $|T|$ factors x with $x \in T$ and, for every i, $1 \leq i \leq n$, j, $1 \leq j \leq r-1$, a distinct factor of length 2 that contains the symbol $\$_{i,j}$ (the distinctness of these factors follows from the fact that each symbol $\$_{i,j}$ has only one occurrence in w). This implies that $\mathsf{c}(p) \leq 1 + |T| + n(r-1) \leq 1 + q + n(r-1)$, which concludes the *only if* direction of the proof.

In order to prove the *if* direction, we assume that there exists a factorisation p of w with $\mathsf{w}(p) \leq 2$ and $\mathsf{c}(p) \leq 1 + q + n(r-1)$. We now modify p step by step such that every modification maintains $\mathsf{w}(p) \leq 2$ and $\mathsf{c}(p) \leq 1 + q + n(r-1)$. Since w starts with ¢¢ and $\mathsf{w}(p) \leq 2$, we can conclude that ¢ or ¢¢ is a factor of p. If ¢¢ is a factor of p, then we can split it into ¢ · ¢ without increasing $\mathsf{c}(p)$, since the factor ¢¢ is then not a factor of p anymore and we get at most ¢ as a new factor. Every factor of p that contains the symbol ¢ is either the factor ¢ or of the form x¢ or ¢x for some $x \in U$. If, for an $x \in U$, we split all occurrences of

factor $x\¢$ in p into $x \centerdot \¢$, then we may produce the new factor x (recall that $\¢$ is already a factor), but we also necessarily lose $x\¢$ as a factor; thus, $\mathsf{c}(p)$ does not increase. If we apply this modification with respect to all $x \in U$ and all factors $x\¢$ and $\¢x$, then we obtain a factorisation in which every single occurrence of the symbol $\¢$ in w is also a factor of p, $\mathsf{w}(p) \leq 2$ and $\mathsf{c}(p) \leq 1+q+n(r-1)$. For every i, $1 \leq i \leq n$, $|v_i|$ is odd, which implies that the factorisation of v_i (according to p) must contain a factor of length 1 and, by the structure of v_i, this factor must be of the form $x \in U$. This particularly implies that for the set T of all elements of U that occur as a factor in p, we must have $T \cap S_i \neq \emptyset$, $1 \leq i \leq n$. Now $\mathsf{sf}(p)$ contains $\¢$, all $n(r-1)$ factors containing a symbol $\$_{i,j}$, $1 \leq i \leq n$, $1 \leq j \leq r-1$, and the factors in T. Thus, $\mathsf{c}(p) = 1 + n(r-1) + |T| \leq 1 + n(r-1) + q$, which implies $|T| \leq q$. □

We note that the MinRF-w instance $(w, 2, n(r-1)+q+1)$ can be constructed from the HS instance $(U, S_1, \ldots, S_n, q)$ in polynomial time and that MinRF-w is in NP (we can guess and verify a factorisation). Hence, from the NP-completeness of HS (see [7]) and Lemma 1, we can conclude the following:

Theorem 12. *The problem* MinRF-w *is* NP*-complete even if* $m \leq 2$.

References

[1] Bulteau, L., Hüffner, F., Komusiewicz, C., Niedermeier, R.: Multivariate algorithmics for NP-hard string problems. EATCS Bulletin 114, 31–73 (2014)

[2] Condon, A., Maňuch, J., Thachuk, C.: Complexity of a collision-aware string partition problem and its relation to oligo design for gene synthesis. In: Proc. 14th Annual International Computing and Combinatorics Conference, COCOON 2008. LNCS, vol. 5092, pp. 265–275 (2008)

[3] Condon, A., Maňuch, J., Thachuk, C.: The complexity of string partitioning. In: Proc. 23th Annual Symposium on Combinatorial Pattern Matching, CPM 2012. LNCS, vol. 7354, pp. 159–172 (2012)

[4] Fernau, H., Manea, F., Mercaş, R., Schmid, M.L.: Pattern matching with variables: Fast algorithms and new hardness results. In: Proc. 32nd Symposium on Theoretical Aspects of Computer Science, STACS 2015. Leibniz International Proceedings in Informatics (LIPIcs), vol. 30, pp. 302–315 (2015)

[5] Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer-Verlag New York, Inc. (2006)

[6] Gagie, T., Inenaga, S., Karkkainen, J., Kempa, D., Piatkowski, M., Puglisi, S.J., Sugimoto, S.: Diverse palindromic factorization is NP-complete. Tech. Rep. 1503.04045 (2015), http://arxiv.org/abs/1503.04045

[7] Garey, M.R., Johnson, D.S.: Computers And Intractability. W. H. Freeman and Company (1979)

[8] Jiang, H., Su, B., Xiao, M., Xu, Y., Zhong, F., Zhu, B.: On the exact block cover problem. In: Proc. 10th International Conference on Algorithmic Aspects in Information and Management, AAIM 2014. LNCS, vol. 8546, pp. 13–22 (2014)

[9] Knuth, D.E., Morris, J.H., Pratt, V.R.: Fast pattern matching in strings. Communications of the ACM 6(2), 323–350 (1977)