The complexity of the Multiple Pattern Matching Problem for random strings[*]

Frédérique Bassino[†] Tsinjo Rakotoarimalala[‡] Andrea Sportiello[§]

November 1st, 2017

Abstract

We generalise a multiple string pattern matching algorithm, proposed by Fredriksson and Grabowski [J. Discr. Alg. 7, 2009], to deal with arbitrary dictionaries on an alphabet of size s. If r_m is the number of words of length m in the dictionary, and $\phi(r) = \max_m \ln(s\, m\, r_m)/m$, the complexity rate for the string characters to be read by this algorithm is at most $\kappa_{\text{UB}}\, \phi(r)$ for some constant κ_{UB}. Then, we generalise the classical lower bound of Yao [SIAM J. Comput. 8, 1979], for the problem with a single pattern, to deal with arbitrary dictionaries, and determine it to be at least $\kappa_{\text{LB}}\, \phi(r)$. This proves the optimality of the algorithm, improving and correcting previous claims.

Furthermore, we establish a tightness result for dictionaries with the same set $\{r_m\}$: the worst-case, average-case, and best-case complexities (the latter, up to a finite fraction of the dictionaries) are all equal, up to a finite multiplicative constant.

1 The problem

1.1 Definition of the problem Let Σ be an alphabet of s symbols, $\xi = (\xi_1, \dots, \xi_n) \in \Sigma^n$ a word of n characters (the input *text string*), $D = \{w_1, \dots, w_k\}$, $w_i \in \Sigma^*$ a collection of k words (the *dictionary*). We say that w occurs in ξ at position j if $w \equiv \xi_j \xi_{j+1} \cdots \xi_{j+|w|-1}$. Let $r_m(D)$ be the number of words of length m in D. We call $\boldsymbol{r}(D) = \{r_m(D)\}_{m \geq 1}$ the *content* of D, a notion of crucial importance in this paper.

The *multiple string pattern matching problem* (MPMP), for the datum (D, ξ), is the problem of identifying all the occurrences of the words in D inside the text ξ (cf. Figure 1).

A version of the problem can be defined in the case of infinite dictionaries ($k = \infty$), where, as we may assume that all the words of the dictionary are distinct, we can suppose that $r_m \leq s^m$. The analysis of the present paper treats in an unified way the case of finite and infinite dictionaries. Of course, in the case of an infinite dictionary, the matter of how to relate our estimates of character accesses to the ordinary time-complexity of the algorithms is more delicate, but, as an example of application, we can think to infinite dictionaries determined by a rational grammar, for which the associated automaton provides an effective tool for recognising words in the text.

The first seminal works have concerned single-word dictionaries. Results included the determination of the average complexity of the problem, the design of efficient algorithms (notably Knuth-Morris-Pratt and Boyer-Moore), and have led to the far-reaching definition of Aho-Corasick automata [1, 4, 11, 14].

The computational complexity of the problem, for a given dictionary D, in worst case among texts ξ of length n, is a problem solved by Rivest long ago [13], with a deceivingly simple answer (within our complexity paradigm, based on text accesses, see below): for each word, there exist texts that need to be read entirely.

On the other side, the *average-case* analysis, over texts ξ sampled uniformly in Σ^n (when n tends towards infinity) is an interesting problem, and its determination for any given dictionary D is the ideal goal of this paper.

Unfortunately, the *exact* determination of the complexity for an arbitrary dictionary seems a formidable task. There exists a wide amount of explicit results, for the simpler situation in which one analyses one *given* algorithm (instead of the intrinsic complexity of the problem), or, even simpler, just the statistical properties of the occurrences in the text (see, for example, the reviews in [12, ch. 7] [10, ch. 4] [2, ch. 1 and 2]), from which it is evinced that the dependence of the complexity from the dictionary shall be subtle, and involves all sorts of mutual correlations among its words. For this reason, we have to revert to a more concrete challenge. More precisely, our goal is as follows.

- Analogously to what is done in Knuth, Morris and Pratt [11] and in Yao [14], we do not analyse the time complexity, but rather the complexity in terms of *character accesses* to original text.
- The asymptotic complexity is $\Theta(n)$, more precisely there exists some constant $\Phi(D)$ such that the

[*] Supported by French ANR projects SuaDicc and MetaConc.

[†] LIPN, Université Paris 13, Sorbonne Paris Cité, France.
[‡] LIPN, Université Paris 13, Sorbonne Paris Cité, France.
[§] LIPN, Université Paris 13, Sorbonne Paris Cité, France.

ξ	5 10 15 20 25 30 35 40 45	
D	110010101100100001001010110101010001110010110	output
0010	..*---....*---.*--*---................*---...	$3, 11, 16, 19, 39$
01010	...*----...........*----..*-*----............	$4, 20, 27, 29$
1011001	......*------................................	7

Figure 1: Typical output of a multiple string pattern matching problem. In this case $s = 2$ and $\boldsymbol{r}(D) = (0, 0, 0, 1, 1, 0, 1, 0, \ldots)$.

quantity of interest is $\Phi(D)\, n + o(n)$. Again, analogously to previous literature, instead of determining the quantity $\Phi(D)$ exactly, we determine the *functional dependence* of Φ from D, up to a finite multiplicative constant, and thus produce upper and lower bounds whose ratio is uniformly bounded.

We now describe in more detail the subtleties of the forementioned complexity paradigm.

Let $m_{\min}$ be the length of the shortest pattern in D. In particular, the number of character accesses is at most n, the case in which the whole text is read, and is $n - \mathcal{O}(1)$ in worst case.[1] It is also at least $n/m_{\min} + \mathcal{O}(1)$, because one needs to read at least one character out of $m_{\min}$ consecutive ones. Another trivial consequence is that we can restrict to the case $m_{\min} \geq 2$, as, if we have even a single word in D of length 1, we know in advance that the whole text needs to be read.

The average number of characters to be read for texts of length n is a super-additive sequence (because solving an instance ξ of size $n_1 + n_2$ is harder than solving the two instances, of sizes n_1 and n_2, associated to the appropriate prefix and suffix of ξ). As a consequence of Fekete's Subadditive Lemma, the limit fraction of required character comparisons exists and is equal to the lim-sup of the same quantity. The Fekete's Lemma in itself leaves open the possibility that the limit is infinite, however from the forementioned trivial upper bound we can conclude that this limit is a finite constant associated to the dictionary.

We introduce the quantity $\Phi(D)$, by defining this fraction to be $\Phi(D)/\ln s$, and we have thus determined that

$$\frac{\ln s}{m_{\min}(D)} \leq \Phi(D) \leq \ln s\,.$$

The presence of the logarithmic prefactor is a useful convention, associated to the fact that reading N characters in a s-symbol alphabet gives a $N \log_2(s)$ amount of Shannon information entropy (we further use natural basis for logarithms, in order to simplify the calculations). An advantage of the rescaled quantity is the fact that it has a form of stability under reduction of the text into q-grams (i.e., a text of length qn on an alphabet Σ of s symbols has the same Shannon entropy, $(nq)\log_2(s) = n \log_2(s^q)$, of the text of length n constituted of symbols in Σ^q). Our ideal task would be the determination of $\Phi(D)$.

Yao [14] determines in particular that, if D consists of a single word of length m, when $1 \ll m \ll n$, the complexity above is given by a certain expression in m and s, up to a finite multiplicative constant, for *almost all patterns*, i.e. for almost all of the s^m possible words of the given length. Patterns with a *different* complexity, in fact, always have a *smaller* complexity, i.e. it is not excluded that there are a few atypically-simpler words, while it is established that there are no (significantly) atypically-harder ones. In analogy to this result, it is natural to imagine that, as we will see in the following, almost all dictionaries with the same content $\boldsymbol{r}$ have the same complexity up to finite multiplicative factors, and the few remaining ones have a smaller complexity. For this reason we find it interesting to define

$$\Phi_{\min}(\boldsymbol{r}) := \min_{D} \Phi(D)\,; \tag{1.1}$$

$$\Phi_{\text{aver}}(\boldsymbol{r}) := \mathbb{E}_D\big(\Phi(D)\big)\,; \tag{1.2}$$

$$\Phi_{\max}(\boldsymbol{r}) := \max_{D} \Phi(D)\,; \tag{1.3}$$

where min, max and average are taken over D such that $\boldsymbol{r}(D) = \boldsymbol{r}$, and one of our aims is to prove that $\Phi_{\text{aver}}(\boldsymbol{r})$ and $\Phi_{\max}(\boldsymbol{r})$ functions have the same behaviour up to a multiplicative constant (while this is not the case for $\Phi_{\min}(\boldsymbol{r})$).

Also the exact determination of $\Phi_{\text{aver}}(\boldsymbol{r})$ and $\Phi_{\max}(\boldsymbol{r})$ is an overwhelming task. We will content ourself of a determination of these functions up to a multiplicative constant, i.e. the identification of a function $\phi(\boldsymbol{r})$, and a constant κ, such that

$$\frac{\phi(\boldsymbol{r})}{\kappa} \leq \Phi_{\text{aver}}(\boldsymbol{r}) \leq \Phi_{\max}(\boldsymbol{r}) \leq \kappa\, \phi(\boldsymbol{r})\,. \tag{1.4}$$

Yet again, this is not dissimilar to what is done in the seminal Yao paper.

[1] E.g. when ξ is composed of a concatenation of words from D, possibly up to the last k characters, with $k < m_{\min}$.

Note that the average in $\Phi_{\rm aver}(\boldsymbol{r})$ allows for the repetition of the same word in the dictionary, and for the presence of pairs of words in the dictionary which are one factor of the other. These problems are thus trivially reduced to problems on a smaller dictionary. If we call $\Phi'_{\rm aver}(\boldsymbol{r})$ the average performed while excluding these degenerate cases, from the monotonicity of the complexity as a function of D (w.r.t. inclusion) we get that $\Phi_{\rm aver}(\boldsymbol{r}) \leq \Phi'_{\rm aver}(\boldsymbol{r}) \leq \Phi_{\max}(\boldsymbol{r})$, so that our theorem, as a corollary, establishes the behaviour of the more interesting quantity $\Phi'_{\rm aver}(\boldsymbol{r})$. Observe that it is legitimate to do this even while still keeping the bound $r_m \leq s^m$, which occurs only in problems without repetitions, because we are comparing the two types of averages while the content $\boldsymbol{r}$ is kept fixed.

We shall comment on the fact that finding the complexity up to a multiplicative constant is not an easy task *a priori*. Suppose that we have a concrete working strategy to determine an upper bound for a given vector $\boldsymbol{r}$, and a second strategy to determine a lower bound for given $\boldsymbol{r}$. Then one may naïvely think that κ is related to the maximum, over all $\boldsymbol{r}$, of the associated ratio. This idea is wrong, because the space of possible $\boldsymbol{r}$ is not a compact, and the 'maximum' of this ratio is in fact a 'supremum', which may well be infinity. So, any valid lower and upper bounds provide an estimate of the complexity for a given content $\boldsymbol{r}$ up to a constant, but we need "good" bound strategies in order to capture the full functional dependence of the complexity from the content parameters $\boldsymbol{r} = \{r_m\}$.

1.2 The result and its implications Let

$$\phi(\boldsymbol{r}) := \max_m \frac{1}{m} \ln(s\, m\, r_m)\,; \tag{1.5}$$

$$\tilde{\phi}(\boldsymbol{r}) := \phi(\boldsymbol{r}) + \frac{1}{2s\, m_{\min}}\,. \tag{1.6}$$

Note that, even in the case of infinite dictionaries D, with all words of length at least $m_{\min}$, $\phi(\boldsymbol{r}(D))$ must evaluate to a finite value, as it is bounded by $\ln s + \frac{1}{m_{\min}} \ln(s\, m_{\min})$ (because $r_m \leq s^m$).

Our aim is to prove the following theorem, valid in the interesting regime $s \geq 2$ and $m_{\min} \geq 2$.

THEOREM 1.1. *Let $\kappa_s = 5\frac{\sqrt{s}}{\ln s}$. For all contents $\boldsymbol{r}$, the complexity of the MPMP on an alphabet of size s satisfies the bounds*

$$\frac{\tilde{\phi}(\boldsymbol{r})}{\kappa_s} \leq \Phi_{\rm aver}(\boldsymbol{r}) \leq \Phi_{\max}(\boldsymbol{r}) \leq 2\tilde{\phi}(\boldsymbol{r})\,. \tag{1.7}$$

As a corollary, by using

$$0 \leq \frac{1}{m_{\min}} \leq \frac{1}{\ln 4} \frac{\ln(s\, m_{\min} r_{m_{\min}})}{m_{\min}} \leq \frac{1}{\ln 4}\phi(\boldsymbol{r})\,, \tag{1.8}$$

from (1.7) we can deduce bounds in a weaker but functionally simpler form

$$\frac{\phi(\boldsymbol{r})}{\kappa_s} \leq \Phi_{\rm aver}(\boldsymbol{r}) \leq \Phi_{\max}(\boldsymbol{r}) \leq \left(2 + \frac{1}{\ln 4}\right)\phi(\boldsymbol{r})\,,$$

i.e. the whole functional dependence from the dictionary is captured by the function ϕ, up to a multiplicative factor depending only on s, and bounded by $\sim 13.6\frac{\sqrt{s}}{\ln s}$.

The reader may be wondering why the function $\phi(\boldsymbol{r})$ is the 'good one' to capture the behaviour of the complexity. We cannot give a full intuition of this feature without entering in the details of the calculations. Nonetheless, we can remark that $\phi(\boldsymbol{r})$ is defined as a maximum over m, of a function of m and r_m alone. As a result, from the statement of the theorem we deduce the following striking property. Let D be a dictionary for the generic MPMP, with content $\boldsymbol{r} = \{r_m\}_{m\geq 2}$. Let m^* be any argmax of the function entering the definition of ϕ, for such a $\boldsymbol{r}$, and let D' be the dictionary corresponding to the restriction of D only to the words with length m^*, plus a single word of length $m_{\min}$ (if $m_{\min} \neq m^*$). As the functions $\phi(\boldsymbol{r}(\cdot))$ and $1/m_{\min}$ evaluate to the same values for D and D', we have thus determined that $\Phi(D)/\Phi(D') \leq \kappa^2$, i.e. the pruned dictionary has a complexity not sensibly smaller than the original one, despite the fact that the space of possible D's is not compact, and, even worse, D may consist of an infinite dictionary.

Let us make one last comment on the structure of the theorem. Our goal is the determination of the functional dependence of $\Phi(D)$ *up to a finite multiplicative constant.* From the forementioned trivial bounds $\frac{1}{m_{\min}} \ln(s) \leq \Phi(D) \leq \ln(s)$, we see that this goal only makes sense when the ratio between the two trivial bounds is large, and there is no point in analysing dictionaries in which $m_{\min}$ is smaller than this constant.

As a further corollary, there is no point in considering dictionaries which do not have complexity $o(1)$ w.r.t. some size parameter. It is not hard to see that such dictionaries exist, even with simple arguments unrelated to our analysis.[2]

The fact that 'interesting' dictionaries have all words 'sufficiently long' leads to a *leitmotif* of our analysis (which also appears transversally in most of the mentioned literature). In the construction of the bounds we are led to analyse complicated functions $f(m)$, e.g. the solution of a transcendental equation. In order to work out our results, we will just need that we can

[2] For example, in searching the word $\mathtt{aa}\cdots\mathtt{a}$ of length m, in a two-symbol alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}\}$, by the Boyer–Moore algorithm, we get the asymptotic complexity $\frac{2}{m-1} + \mathcal{O}(2^{-m})$, which is an upper bound to the exact complexity of this dictionary.

estimate upper and/or lower bounds to this function, which are 'effective' in a regime of m large, even if they are quite poor for m too small. As we will see, most functions $f(m)$ occurring in this problem will allow for a perturbative expansion in powers of $(\ln m)/m$.

1.3 Previous results *Single-word dictionaries.* The statement of Theorem 1.1 restricted to single-word dictionaries is related to the result of the seminal paper of Yao [14]. In this case, in ϕ there is no need to take a max, and $r_{m'} = 1$ for $m = m'$ and 0 otherwise. While our theorem gives that, for such a $\boldsymbol{r}$, in worst case among words of length m,

$$\frac{1}{\kappa_s}\left(\frac{\ln(s\,m)+\frac{1}{s}}{m}\right) \leq \Phi(\boldsymbol{r}) \leq 2\left(\frac{\ln(s\,m)+\frac{1}{s}}{m}\right),$$

Yao determines that for almost all words of length m, and treating s as a constant, the complexity is $\Theta(\frac{\ln(m)}{m})$. In order to establish this result, Yao introduces a notion of *certificate*. The set $C \subseteq \{1,\dots,n\}$ is a certificate for the pair (D,ξ) if the knowledge of $\{\xi_j\}_{j\in C}$ completely determines the output of the problem (in fact, Yao defines certificates only for single-word dictionaries, but the generalised notion is evinced in a natural way). Call $\mathcal{C}(D,\xi)$ the set of certificates, and define

$$\Phi_{\mathrm{cert}}(D,\xi) := \frac{\ln s}{n} \min_{C\in\mathcal{C}(D,\xi)} |C|\,, \tag{1.9}$$

together with the average quantities

$$\Phi_{\mathrm{cert}}(D) := \mathbb{E}_\xi\big(\Phi_{\mathrm{cert}}(D,\xi)\big)\,; \tag{1.10}$$

and

$$\Phi_{\mathrm{cert}}(\boldsymbol{r}) := \mathbb{E}_{D\,:\boldsymbol{r}(D)=\boldsymbol{r}}\big(\Phi_{\mathrm{cert}}(D)\big)\,. \tag{1.11}$$

This is another natural notion of complexity for MPMP, that we shall call the *certificate complexity* of the instance. It is the smallest possible complexity of the given instance, or, in other words, it is the smallest complexity among all possible runs of the probabilistic algorithm, that reads the characters of the text one by one in a random order, up to when a certificate is reached.

This notion is a lower bound to the 'true' complexity of the instance, just because any algorithm (and any run of a non-deterministic algorithm) may halt only when a certificate is reached. Intuitively, reaching the value $\Phi_{\mathrm{cert}}(D,\xi)$ supposes not only (or better, not necessarily) the optimality of the algorithm, but also an infinite amount of 'luck' in the choice of the characters to read, given the hidden text ξ, from which the inequality would follow.

Thus the analysis of Φ_{cert} provides a natural proof strategy for what concerns the estimate of a lower bound, that Yao succeeds in pursuing for the single-word case, and that, in this paper, we adapt to the case of general dictionaries.

Uniform dictionaries. Let us say that a dictionary D is *uniform* if all the words have the same length. Fredriksson and Grabowski [7] describe a MPMP algorithm (in the following, the *FG algorithm*) adapted to deal with uniform dictionaries, and analyse the resulting upper bound. This algorithm is possibly never optimal (and known to be non-optimal on certain simple dictionaries), but the loss is by a relatively small factor, and, in many applications, is compensated by a great simplicity both in programming and in the statistical analysis. Based on previous results of Fredriksson and Navarro [8], that purportedly estimated lower bounds for uniform dictionaries, the authors claimed their algorithm to be optimal up to multiplicative factor, in its domain of contents $\boldsymbol{r}$, and provided a complexity compatible with the corresponding specialisation of our theorem above. However, the results of [8] are invalidated by a major flaw[3] (and thus the optimality result in [7] remained unproven before the present paper). For this reason, our lower-bound estimate, for dictionaries of arbitrary content, is completely different from the one in [8], while it is in fact mostly an adaptation of the original strategy of Yao [14] for single-word dictionaries.

Plan of the paper. The adaptation of the FG algorithm is outlined in Section 2. While the algorithm in itself is only mildly modified, both the analysis of the upper bound, and the crucial determination of the optimal value for a parameter of the algorithm, are sensibly different in our general context. The pertinent analysis is in Section 3.

The lower bound is presented in Section 4. This is the most complex part of our work, and the original ideas of Yao need to be combined with further ingredients. In particular, Section 4.2 contains the subtle reasonings which allow to establish a lower-bound strategy, while Sections 4.3 and 4.4 contain the main steps in the rather involved calculations.

2 The Fredriksson-Grabowski pattern-matching algorithm

As we mentioned, ideally we want an algorithm with a degree of flexibility, such that, in the interesting cases $m_{\min} \gg 1$, a fraction $\ll 1$ of the text is accessed, i.e.

[3] See the annotation at `https://www.dcc.uchile.cl/~gnavarro/fixes/tcs04.html`, based on a personal communication of Ralf Stiebe to G. Navarro.

some ‘big jumps’ are performed. The idea of Fredriksson and Grabowski [7] is to encode this flexibility in a single parameter q associated to a *dilution rate* of the words in the dictionary.

For a fixed integer q, a text ξ and a dictionary $D = \{w_j\}_{1 \leq j \leq k}$, define the *diluted text* $\xi^{(q)} = (\xi_q, \xi_{2q}, \xi_{3q}, \dots)$ and the *diluted dictionary* $D^{(q)} = \{w_{j,p}\}_{1 \leq j \leq k,\, 0 \leq p < q}$. For a word w_j of length m, call $w_j^1, \dots, w_j^m$ its m characters. If $|w_j| = m_j$, $w_{j,p}$ is defined as

$$w_{j,p} = w_j^{m_j-p-hq} \cdots w_j^{m_j-p-q} w_j^{m_j-p},$$

with $h = \lfloor \frac{m_j-p-1}{q} \rfloor$. For example, if $q = 3$ and $w_1 = \texttt{thepattern}$, the three words $w_{1,0}$, $w_{1,1}$ and $w_{1,2}$ are obtained by the sieve construction

t	h	e	p	a	t	t	e	r	n	
t	.	.	p	.	.	t	.	.	n	tptn
.	.	e	.	.	t	.	.	r	.	etr
.	h	.	.	a	.	.	e	.	.	hae

Introducing the notation $[m/q]_p := \lfloor \frac{m-p-1}{q} \rfloor + 1$, the lengths of the diluted words read $|w_{j,p}| = [|w_j|/q]_p$.

Let us denote by FG(q) the algorithm of parameter q, and by $\Phi_{\mathrm{FG}(q)}(D)$ the upper bound to $\Phi(D)$ obtained by analysing this algorithm. The FG algorithm of parameter q, in its simplest version, consists in reading the characters of $\xi^{(q)}$ in sequence, and searching for words in the dictionary $D^{(q)}$. Once an occurrence has been found, it reads the missing $m\frac{q-1}{q} + \mathcal{O}(1)$ characters in order to verify if the original word was present. This idea can be improved at various extents. For example, once we have agreed to solve the MPMP $(D^{(q)}, \xi^{(q)})$, nothing prevents from using a ‘good’ algorithm for this, and possibly, just recursively the FG algorithm, instead of switching to the naïve read-all algorithm. Furthermore, instead of just filling the whole gap, we can just proceed to read the characters of the gap one by one, and stop the procedure if a contradiction with the candidate word is found (for future reference, let us call this *sharp gap filling*). However it turns out that, at the aim of determining $\Phi(D)$ up to multiplicative constants, these improvements are not crucial. See Figure 2 for an example.

In the (non-improved) FG(q) algorithm, we read at least a fraction $1/q$ of the text, and, as it will turn out, for the optimal value of q the total fraction is slightly above $1/q$. If a dictionary D has its shortest word of length $m_{\min}$, the complexity of any algorithm is at least $1/m_{\min}$, so, under the *ansatz* above, there is no point in searching among values $q > m_{\min}$, and in particular there are no empty words in $D^{(q)}$.

The exact value of $\Phi_{\mathrm{FG}(q)}(D)$ is hardly expressed by a closed formula, because we may be forced to fill the same gap by more than one occurrence of diluted words. These characters shall count only once in the complexity, but the interplay of different words in the dictionaries is complicated to analyse. So, we will content ourselves with the upper bound to this complexity obtained from the union bound (Boole’s inequality[4]) on these occurrences:

Proposition 2.1.

$$\Phi_{\mathrm{FG}(q)}(D) \leq \frac{\ln s}{q} \Bigg(1 + \sum_{\substack{m \geq 2 \\ 0 \leq p < q}} r_m s^{-[m/q]_p} (m - [m/q]_p) \Bigg)$$

Proof. The formula above is deduced from an easy reflection on the mechanisms of the algorithm. We read one in q characters, to solve the diluted problem with the trivial read-all algorithm (from which the $1/q$ factor).[5] Then, at each position of the diluted string, if we find a candidate subpattern $w_{j,p}$, we read the missing $|w_j| - |w_{j,p}|$ characters (from which the $(m - [m/q]_p)$ factor). This gives an overcounting (by union bound) when we find more than one subpattern at the same position, or at a short distance along the diluted text, but it makes the probabilities of finding subpatterns independent, and thus just given by the obvious factor $s^{-|w_{j,p}|}$. □

Note how, just by the use of the read-all algorithm on the diluted subproblem, and the union bound on the occurrences in this subproblem, we have determined an upper bound on $\Phi_{\mathrm{FG}(q)}(D)$ that depends on D only through its content $\boldsymbol{r}(D)$. This upper bound on the complexities of all variants of the FG algorithm discussed above is in fact the *exact* complexity of the most naïve version of the algorithm, the one in which, if more than one word of $D^{(q)}$ is found at a given diluted position, the missing characters are accessed multiple times, and these multiple accesses are counted with their multiplicities.[6]

If we perform the ‘sharp gap filling’ enhancement discussed above, i.e. we stop to read characters in the verification of a given candidate sub-pattern whenever we obtain a contradiction, we can replace the factor $m' := m - [m/q]_p$ in Proposition 2.1, i.e. the maximal number of read characters for the diluted pattern, by

[4] I.e., the fact $\mathrm{prob}(\mathcal{E}_1 \vee \cdots \vee \mathcal{E}_k) \leq \mathrm{prob}(\mathcal{E}_1) + \cdots + \mathrm{prob}(\mathcal{E}_k)$.

[5] As we said, for the *ordinary* complexity paradigm, this problem is *not* trivial, as it requires the use, for example, of the Aho–Corasick automaton. However, for what concerns the number of accesses to the text, reading all the characters is just the worst possible strategy, and thus a trivial upper bound.

[6] Note that, for such a paradigm, in principle the trivial upper bound $(\ln s)\Phi_{\mathrm{FG}(q)}(D) \leq n$ may be violated, and it will be our care to choose appropriate values of q such that the bound is tight.

$$D=\{w\}=\{\texttt{thepattern}\}\ ;\quad q=3;\quad D^{(q)}=\{\texttt{tptn},\texttt{hae},\texttt{etr}\}$$

```
ξ = . . x . . h . . a s t e r n t . a r n . r . .
            t h e p a s t e r n
                      t h e p a t t a r n
```

Figure 2: A typical realisation of the Fredriksson–Grabowski algorithm. Dotted entries will not be read. In its basic implementation, analysed here, it is the `a` at the far right that excludes the decimated pattern `etr`. A more efficient variant, but more complicated to analyse, would find that the pattern `etr` is excluded by the ξ_i's already read at the previous step.

the *expectation* of read characters, which is

$$\tfrac{s-1}{s}\left(1+\tfrac{2}{s}+\tfrac{3}{s^2}+\cdots+\tfrac{m'-1}{s^{m'-2}}\right)+\tfrac{m'}{s^{m'-1}}=\tfrac{s(1-s^{-m'})}{s-1}$$

getting the improved expression

Proposition 2.2.

$$\Phi_{\mathrm{FG}(q)}(D)\leq\frac{\ln s}{q}\left(1+\tfrac{s}{s-1}\sum_{\substack{m\geq 2\\ 0\leq p<q}} r_m\big(s^{-[m/q]_p}-s^{-m}\big)\right)$$

3 The upper bound

We shall now analyse the expression on the RHS in Proposition 2.1 – or better, the one in the stronger Proposition 2.2, improved by the sharp gap filling – in order to produce a more compact expression for an upper bound to the complexity, and, in agreement with Theorem 1.1, prove the following

Proposition 3.1. *With $\phi(\boldsymbol{r})$ as in equation (1.5), we have*

$$\Phi_{\max}(\boldsymbol{r})\leq 2\phi(\boldsymbol{r})+\frac{1}{s\,m_{\min}}\,. \tag{3.12}$$

As we have $\Phi(D)\leq\Phi_{\mathrm{FG}(q)}(D)$ for any value of q, we have in particular $\Phi(D)\leq\Phi_{\mathrm{FG}(q(\boldsymbol{r}(D)))}(D)$ for any integer-valued function $q(\boldsymbol{r})$, the best choice being the value of q minimising the expression on the RHS in Proposition 2.2, for the given $\boldsymbol{r}$. We shall provide a function $q(\boldsymbol{r})$ which, although not being exactly the minimum, is sufficiently close to this value that our upper and lower bounds, as functions of $\boldsymbol{r}$, match up to a finite multiplicative constant. However, this condition can only be verified *a posteriori*, once that a lower bound, of functional form analogous to the RHS of (3.12), is established, so in this section we shall content ourselves with the analysis of the resulting expression, leaving our choice of $q(\boldsymbol{r})$ as an *ansatz*.

Proof of Proposition 3.1. Let us start from the improved expression in Proposition 2.2. Let us drop the summands $-s^{-m}$, and get the slightly larger bound

$$\Phi_{\mathrm{FG}(q)}(D)\leq\frac{\ln s}{q}+\frac{s\ln s}{s-1}\sum_m r_m\frac{1}{q}\sum_{p=0}^{q-1}s^{-[m/q]_p}\,. \tag{3.13}$$

Let us concentrate on the inner-most expression

$$f(s;m,q):=\frac{1}{q}\sum_{p=0}^{q-1}s^{-[m/q]_p}\,. \tag{3.14}$$

Writing $m=aq+b$ with $0\leq b<q$, this just reads

$$f(s;aq+b,q):=s^{-a}\frac{bs^{-1}+q-b}{q}\,. \tag{3.15}$$

This function can be analytically extended, in terms of

$$g(s,x):=s^x\big(xs^{-1}+1-x\big) \tag{3.16}$$

because

$$f(s;m,q)=s^{-\frac{m}{q}}g(s,b/q)\,. \tag{3.17}$$

For $s>0$, the function $g(s,x)$ is concave on $[0,1]$, its only maximum is at $x^*(s)=1+\frac{1}{s-1}-\frac{1}{\ln s}$, where one has $g^*(s)=g(s,x^*(s))=\frac{s-1}{\ln s}\exp\big(\frac{\ln s}{s-1}-1\big)$. In particular, for $s\geq 2$ we have $g^*(s)\leq\frac{2}{e}\frac{s-1}{\ln s}$. Substituting into (3.13) gives

$$\Phi_{\mathrm{FG}(q)}(D)\leq\frac{\ln s}{q}+\frac{2s}{e}\sum_m r_m s^{-\frac{m}{q}}\,. \tag{3.18}$$

The change of variables $\rho=\frac{\ln s}{q}$ gives

$$\begin{aligned}\Phi_{\mathrm{FG}(q)}(D)&\leq\rho+\frac{2s}{e}\sum_m r_m e^{-m\rho}\\&=\rho+\frac{2}{es}\sum_{m\geq m_{\min}}\frac{1}{m^2}e^{-m\left(\rho-\frac{\ln(s^2m^2r_m)}{m}\right)}\,.\end{aligned} \tag{3.19}$$

Now choose the candidate value $\rho=2\phi(\boldsymbol{r})$, that is, $q(\boldsymbol{r})=\ln s/(2\phi(\boldsymbol{r}))$, with $\phi(\boldsymbol{r})$ as in equation (1.5).[7]

[7] Up to a round-off due to q being an integer. We neglect here the (easy) corrections coming from this feature.

We have $\rho \geq \frac{2\ln(smr_m)}{m} \geq \frac{\ln(s^2m^2r_m)}{m}$ for all m, so that

$$\sum_{m \geq m_{\min}} \frac{e^{-m\left(\rho - \frac{\ln(s^2m^2r_m)}{m}\right)}}{m^2} \leq \sum_{m \geq m_{\min}} \frac{1}{m^2}$$
$$\leq \frac{\pi^2-6}{3}\frac{1}{m_{\min}}\,,$$

(where we used the result of Appendix A). As $\frac{2}{e}\frac{\pi^2-6}{3} < 1$, we can write

$$\Phi_{\mathrm{FG}(\ln s/(2\phi(\boldsymbol{r})))}(\boldsymbol{r}) \leq 2\phi(\boldsymbol{r}) + \frac{1}{s\,m_{\min}} = 2\tilde{\phi}(\boldsymbol{r})\,.$$

This allows us to conclude. □

4 The lower bound

In this section we establish a lower bound on the complexity, averaged over all dictionaries of content $\boldsymbol{r}$, of the form

Proposition 4.1. *For $s \geq 2$, define $\kappa_s = 5\frac{\sqrt{s}}{\ln s}$. In the MPMP on an alphabet of size s,*

$$\Phi_{\mathrm{aver}}(\boldsymbol{r}) \geq \Phi_{\mathrm{cert}}(\boldsymbol{r}) \geq \frac{1}{\kappa_s}\left(\phi(\boldsymbol{r}) + \frac{1}{2s\,m_{\min}}\right)$$

This is done in four main steps, one per subsection: (1) we establish that analysing the problem on a finite text provides a bound in the appropriate direction; (2) we discuss in detail why the certificate complexity is a lower bound to the minimal complexity of an algorithm, and why its average over dictionaries can be effectively estimated; (3) we relate the involved estimate to a classical problem in enumerative combinatorics: evaluating the fraction of maps which are surjections; (4) we provide calculations which allow to transform the complicated variational expression for the bound in terms of a simple function.

4.1 Reduction to a finite text In evaluating a lower bound on the information complexity of MPMP, it is legitimate to replace the exact expression by simpler quantities which bound it from below. The goal is to reach a concretely evaluable expression, without losing the leading functional dependence from D (up to multiplicative constants), or, better, the dependence from $\boldsymbol{r}(D)$.

The output of a MPMP is a list of pairs $S(D,\xi) = \{(\ell,h)\}$, with $1 \leq h \leq k$ and $1 \leq \ell \leq n - |w_h| + 1$, such that w_h occurs at position ℓ in the text. Call $X_0 = \{(\ell,h)\,|\,1 \leq h \leq k;\, 1 \leq \ell \leq n - |w_h| + 1\}$ the list of possible pairs described above. If we specify in advance a subset $X \subseteq X_0$, and decide that we aim at finding all the occurrences within this subset, we have a reduced problem with complexity, say, $\Phi(D|X)$ (while the complexity of the original problem is $\Phi(D) \equiv \Phi(D|X_0)$). The problem associated to the subset X is evidently simpler than the original one, at least within our notion of complexity (in terms of text characters to be read): a solution $S(D,\xi)$ of the full problem provides a solution of the reduced one, just by taking the intersection $S(D,\xi) \cap X$, and this process does not involve further accesses to the text, thus we have $\Phi(D|X) \leq \Phi(D)$ for all $X \subseteq X_0$. A similar argument applies to the certificate complexity. Thus, this observation provides a useful strategy for producing lower bounds for both versions of the complexity.

As it was the case for the upper bound and the parameter q, we need a free real parameter in order to capture the functional dependence from D. In analogy to Yao procedure, a good choice is an integer L, which determines a set $X = X(L)$ as follows:

$$X(L) = \left\{(\ell,h)\,\middle|\,\left\lfloor\frac{\ell+1}{L}\right\rfloor = \left\lfloor\frac{\ell+|w_h|}{L}\right\rfloor\right\}\,. \tag{4.20}$$

In simple words, the text, originally of length n, is divided into blocks of size L (with the possible exception of one last block of size $< L$), and we search for occurrences contained within one block. As a result of this bound, we have factorised the problem on the blocks, and the calculation of (the bound on) the complexity rate, i.e. the limit of $1/n$ times the complexity, for $n \to \infty$, is given by $1/L$ times the whole average complexity of single blocks, again both for ordinary and certificate complexities.

$$\Phi(D) \geq \mathbb{E}_{\xi\in\Sigma^L}\Phi(D,\xi) \qquad \forall\ L\,; \tag{4.21}$$
$$\Phi_{\mathrm{cert}}(D) \geq \mathbb{E}_{\xi\in\Sigma^L}\Phi_{\mathrm{cert}}(D,\xi) \qquad \forall\ L\,. \tag{4.22}$$

This solves the issue of performing the limit $n \to \infty$, and reduces to a problem which, for a fixed dictionary, is of finite size, namely L (although, in fact, the optimal value of L as a function of D is unbounded). From this moment on, we can thus concentrate on a single block.

It is rather clear that we need L to be not below the scale of the value $m^*(D)$ realising the maximum for our quantity ϕ (or, if this is realised on more than one integer, the largest of these values), otherwise we would just miss the totality of the occurrences which give the leading term of the complexity. Surprisingly, it will turn out that the optimal value of L is barely above m^*.

4.2 The lattice of possible algorithms Under our complexity paradigm, and for a fixed text ξ of length L, we can visualise any algorithm as a recipe for performing a directed walk on the lattice of subsets I of $\{1,\dots,L\}$.

Any algorithm starts at the bottom of the lattice, i.e. the node associated to the empty subset. The algorithm evaluates a function of D, in order to choose a position $1 \leq i_1 \leq L$ to access. Then, it reads ξ_{i_1}, thus 'moving' to the set $\{i_1\}$, and it evaluates a function of D and ξ_{i_1}, in order to choose a position $i_2 \neq i_1$. Then, it reads ξ_{i_2}, thus moving to $\{i_1, i_2\}$, and chooses a position $i_3 \neq i_1, i_2$ through a function of D and $\{\xi_{i_1}, \xi_{i_2}\}$, and so on, until a certificate is reached (see Figure 3). We say that the algorithm, at a certain moment of its execution, is at the node $I = \{i_1, \dots, i_t\}$ of the lattice, if, so far, it has accessed the text characters at positions $\{i_1, \dots, i_t\}$ (in any of the possible orders). The nodes are associated to unordered sets of positions, instead that of ordered lists, because there is no reason why the function of D and $\{\xi_{i_1}, \dots, \xi_{i_t}\}$ shall depend on the order in which the positions $\{i_1, \dots, i_t\}$ have been accessed: any such dependence would only make the algorithm less performing.

A set I may or may not be a *certificate* for the given text and dictionary, i.e. it may or not contain enough information to determine the set $S(D, \xi)$ of occurrences. Clearly, if I is a certificate and $J \supseteq I$, also J is a certificate, so that, for fixed ξ and D, the certificates are an up-set in the lattice. Following Yao, we call *negative certificate* a set certifying that $S(D, \xi) = \varnothing$.

Let us call $\chi_{D,\xi}(I)$ the boolean function valued 1 if the knowledge of the text at positions I does *not* provide a certificate, and 0 otherwise (i.e., we have $\chi_{D,\xi} = 1$ on the bottom part of the lattice, and $\chi_{D,\xi} = 0$ on the top part).

If we neglect (for the moment) the walks which stop at a certificate, and we consider the algorithm on a given text ξ, we have an upper bound on the number of distinct nodes at time t, which is $\binom{L}{t}$. Some walks may stop before time t because they reach a certificate. In order to preserve the stochasticity of the process, we can imagine that the algorithm continues (arbitrarily) even when a certificate is reached. As a result, at time t, the walk determined by the algorithm may be at any set I among the possible $\binom{L}{t}$ ones, with a probability $P_{D,\xi}(I)$ (we consider probabilities because the optimal algorithm may be a probabilistic one), these sets I may or not be a certificate.

The complexity $\Phi_{\mathcal{A}}(D, \xi)$ of the algorithm $\mathcal{A}$ is, by definition, the average depth of the nodes at which a certificate is first reached, weighted with their branch probabilities.

If this average is evaluated for an algorithm which is *optimal* at given D and L, and averaged over all text ξ, it would correspond exactly to $\Phi(D)$ for texts of length L. Thus, and by using (4.21), any lower bound constructed by analysing this probabilistic process provides a lower bound to the complexity. However, we have a very mild control on the probability distribution $P_{D,\xi}(I)$ associated to the optimal algorithm (because constructing explicitly the optimal algorithm is a formidable task). For this reason we rather concentrate on the simpler (and smaller) quantity $\Phi_{\text{cert}}(D)$, introduced in Section 1.3 (in fact, we will rather analyse the related quantity $\Phi_{\text{cert}}(\boldsymbol{r}) = \mathbb{E}_D \Phi_{\text{cert}}(D)$), as we now describe.

Say that I is a set of size t, and introduce the quantity

$$f_{\boldsymbol{r}}(I) := \mathbb{E}_D \Big(1 - \prod_{\xi} \chi_{D,\xi}(I) \Big) . \tag{4.23}$$

Informally, this is the average over all dictionaries of content $\boldsymbol{r}$, of the probability that the set I is a certificate for some text ξ (i.e., for some t-uple $(\xi_{i_1}, \dots, \xi_{i_t}) \in \Sigma^t$).

Call $1 - p_t(\boldsymbol{r})$ the fraction of dictionaries of content $\boldsymbol{r}$ which have, for some text, some certificate of size at most t. By the union bound, this quantity is bounded from above by the average, over the dictionaries, of the number of certificates of size at most t (i.e., if there is a certificate J of cardinality t', which is minimal w.r.t. inclusion, it gets weighted by $\binom{L-t'}{t-t'}$, i.e. the number of sets $I' \supseteq J$ of cardinality t). But this latter quantity is, by definition, $\sum_I f_{\boldsymbol{r}}(I)$, from which we get

$$1 - p_t(\boldsymbol{r}) \leq \sum_I f_{\boldsymbol{r}}(I) . \tag{4.24}$$

If, for the given text, a dictionary has no certificate of length at most t, its complexity is at least $t+1$. If it has one or more such certificates, its complexity is bounded from below at least by the forementioned obvious bound $\lceil (L - m_{\min} + 1)/m_{\min} \rceil$, which, for large L, is essentially $L/m_{\min}$. For this reason, and by using (4.21), we can conclude that

$$\Phi_{\text{cert}}(\boldsymbol{r}) \geq \ln s \, \max_t \Big(\frac{t \, p_t}{L} + \frac{1 - p_t}{m_{\min}} \Big) . \tag{4.25}$$

It will be our care to consider values of t larger than $L/m_{\min}$ (otherwise we would make statements which are weaker than the trivial bound!). In this case we can rewrite the expression above as

$$\begin{aligned} &\max_{t \geq \frac{L}{m_{\min}}} \Big(\frac{(t+1) \, p_t}{L} + \frac{1 - p_t}{m_{\min}} \Big) \\ &\qquad = \frac{1}{m_{\min}} + \frac{1}{L} \max_{t \geq \frac{L}{m_{\min}}} \Big(p_t \Big(t + 1 - \frac{L}{m_{\min}} \Big) \Big) \end{aligned} \tag{4.26}$$

which is at sight a monotone increasing function of t and p_t (when considered as independent variables). For this reason, the bound (4.24) provides an actual bound

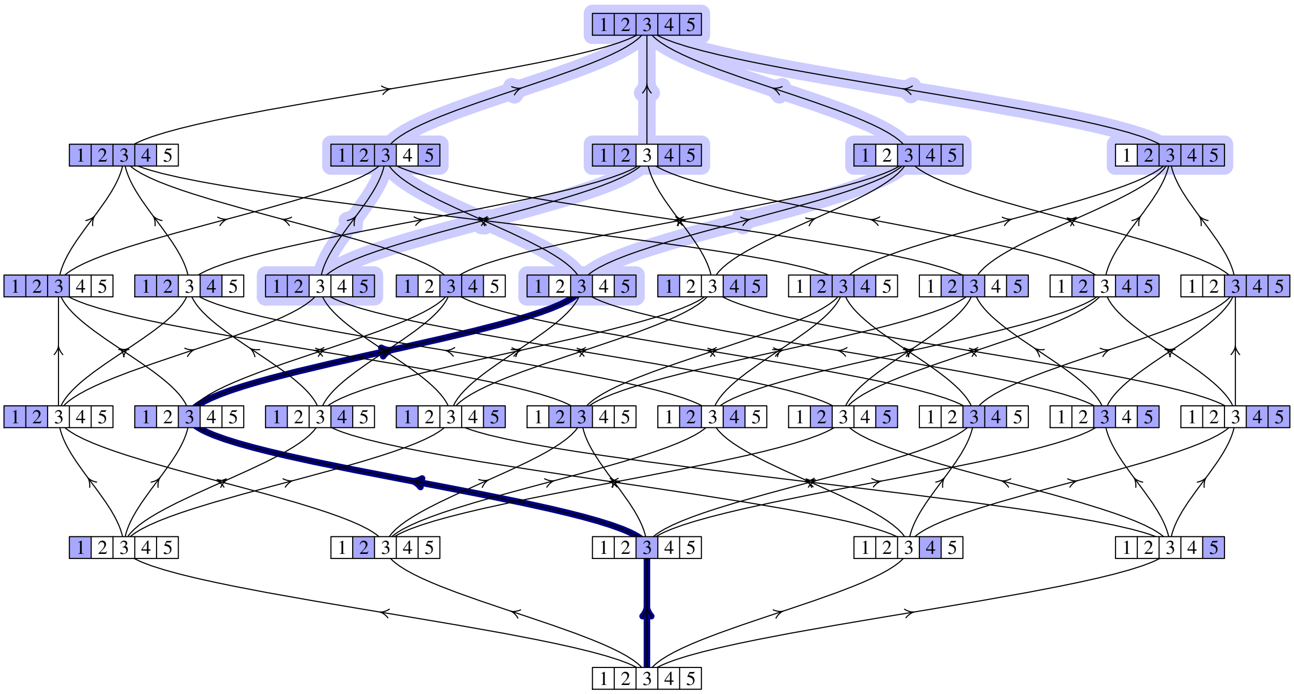

Figure 3: The lattice of subsets of $\{1, \dots, 5\}$. The shadowing denotes an example of up-set, corresponding, for a certain pair (D, ξ), to certificate nodes. The thick path is a typical run of an algorithm, which halts whenever it reaches a shadowed node, in the example at $t = 3$.

to the complexity, namely

$$\Phi_{\text{cert}}(\boldsymbol{r}) \geq \ln s \max_{t \geq L/m_{\min}} \left(\frac{t\,\tilde{p}_t}{L} + \frac{1-\tilde{p}_t}{m_{\min}} \right); \tag{4.27}$$

$$\tilde{p}_t(\boldsymbol{r}) = 1 - \sum_I f_{\boldsymbol{r}}(I)\,. \tag{4.28}$$

Of course, t and $\tilde{p}_t$ are not independent variables, and in fact $\tilde{p}_t$ is a monotonically *decreasing* function of t. This is what makes the optimisation problem non-trivial. In the remainder of this section we will estimate the quantity $\tilde{p}_t$, at values t and L, choosen as a function of $\boldsymbol{r}$, which are near to the optimal value, and thus, once plugged in (4.27), produce a bound of the desired functional form.

4.3 Reduction to a problem on surjections Let us consider an algorithm, together with its lattice of executions. Consider one given word $w \in D$, of length m, and one given set $I = \{i_1, \dots, i_t\}$ of size $t < m$, appearing as a node of depth t in the lattice. Suppose that the algorithm has read the values $\xi_{i_1}, \dots, \xi_{i_t}$ in the text.

We shall estimate the probability that this branch is not certifying that w does not occur anywhere within the block. As all certificates are negative if $|I| < m$, this implies in particular that I is not a certificate, thus, in such a case, we know that the algorithm shall continue, and this branch will contribute to a lower bound to the complexity (both ordinary and certificate) of the given algorithm.

The word w has $d := L-m+1$ potential occurrence positions within the block. The occurrence at position $j+1$ may still occur if $\xi_{i_a} = w_{i_a - j}$ for all $a \leq t$ such that $1 \leq i_a - j \leq m$. Thus, if we average naïvely over all words of length m, this gives a probability at least s^{-t} (it may be larger if some of the positions $i_a - j$ are out of range). Annoyingly, the probabilities for different values of j are correlated. Nonetheless, a lemma proven by Yao in [14] (called the *Counting Lemma*) states that, among the d possible positions, there exists a set J, of cardinality at least $\lceil d/t^2 \rceil$, for which the values $\{i_a - j\}_{a \leq t,\, j \in J}$ are all distinct, and thus, when averaging over one word, the probabilities decorrelate. For the positions $j \notin J$, we are neglecting the possibility that they are enforcing I not to be a certificate, and thus we are under-estimating the depth of certificate nodes, which gives a valid lower bound.

In the case of multiple words the things are not harder. When averaging over all dictionaries of a given content, with repetitions allowed, the probabilities associated to different words just decorrelate, so that we do not need to generalise the counting lemma of Yao in passing from his analysis of single-word dictionaries to our context of arbitrary dictionaries.

If, as we do here, we insist on considering only negative certificates, we can drop from the dictionaries all the words of length smaller than t, i.e. consider the content $\boldsymbol{r}'$ defined as $r'_m = r_m$ if $m > t$ and $r'_m = 0$ otherwise (of course, this makes an even smaller set X w.r.t. (4.20)). The overall number of pairs of word/position which are certified to be independent for a given set I of length t is thus at least a certain quantity $c_0\, s^t$, regardless from the precise set I, where c_0 is defined as

$$c_0 = c_0(\boldsymbol{r}', L, t) = s^{-t} \sum_{m>t} r_m \left\lceil \frac{L+1-m}{t^2} \right\rceil , \tag{4.29}$$

where we make a crucial use of Yao's Counting Lemma. Given that we aim at producing a bound in terms of the function $\phi(\boldsymbol{r})$, only the value $m^* := \mathrm{argmax}_m \frac{\ln s\, m\, r_m}{m}$ is relevant. So we expect that the following analysis will show that the optimal value of t in (4.27) is smaller than m^*, and that at leading order we would obtain the same estimate of the complexity if, instead of c_0, we use the smaller value

$$c = s^{-t} r_{m^*} \left\lceil \frac{L+1-m^*}{t^2} \right\rceil . \tag{4.30}$$

We are now ready to bound the quantity $f_{\boldsymbol{r}}(I)$ in (4.23), of crucial importance in (4.27), by a function of t and c (or c_0). For any dictionary D, and at the given value of t, we have a list $\mathcal{X}$, of length $X = c\, s^t$, of t-uples in Σ^t corresponding to the entries of any given word w, at its positions $\{i_a - j\}_{a=1,\dots,t}$, for a pair (w, j) in the outcome of Yao's Counting Lemma. We also have the list $\mathcal{Y} = \Sigma^t$, thus of length $Y = s^t$, of all possible t-uples, i.e., of all possible texts restricted to the positions of I. For a pair (w, j) in $\mathcal{X}$, if $i_t - |w| \leq j \leq i_1 - 1$, call $\psi(w, j) = (w_{i_1 - j}, \dots, w_{i_t - j})$. If instead we have some $(i_a - j)$ out of range, define the corresponding entry of $\psi(w, j)$ by taking one character at random in the alphabet. The map ψ is thus from $\mathcal{X}$ to $\mathcal{Y}$, and, because of the forementioned decorrelation, ψ is distributed uniformly over all possible Y^X maps, and $f_{\boldsymbol{r}}(I)$ is the probability of the event that ψ is not a surjection.

It is well-known (see e.g. [6, pag. 107]) that the fraction of random maps from a set of cardinality X to a set of cardinality Y which are surjections is

$$\pi_{X,Y} = Y^{-X} Y! \left\{ {X \atop Y} \right\} \tag{4.31}$$

where $\left\{ {n \atop r} \right\}$ are the Stirling numbers of second kind. This quantity is used in the relation

$$f_{\boldsymbol{r}}(I) \leq 1 - \pi_{c\, s^t, s^t} , \tag{4.32}$$

valid for all sets I of size t. We have an inequality because we have produced lower-bound estimates on various quantities (for example, the actual list associated to I may be longer than $c\, s^t$, or there could be entries $(i_a - j)$'s out of range, which give rise to more pairs (w, j) to be checked), and, importantly, $\pi_{X,Y}$ is a monotone function of its first argument. As a result,

$$\tilde{p}_t(\boldsymbol{r}) = 1 - \sum_I f_{\boldsymbol{r}}(I) \geq 1 - \binom{L}{t} (1 - \pi_{c\, s^t, s^t}) . \tag{4.33}$$

4.4 Estimates At this point we have completed our abstract reasonings, and we only need to combine the inequalities (4.27), (4.30), (4.31) and (4.33), and to estimate them effectively. We shall start with the forementioned fraction of surjections. We will use the fact that for $c > 1$ there exists a function $\tilde{c}(c) = c - \frac{1}{2} c\, e^{-c} + \dots$ such that $\pi_{cn,n} \geq \exp[-n e^{-\tilde{c}}]$ for all n. The precise function is

$$\begin{aligned} \tilde{c}(c) = -\ln \big[& c - c\ln(c) - \ln(-1 + e^{c + W(-c\, e^{-c})}) \\ & + c \ln(c + W(-c\, e^{-c})) \big] \end{aligned} \tag{4.34}$$

where $W(w)$, the Lambert W-function, is the principal solution for w in $z = we^w$. This is discussed, e.g., in [3, Lemma 6]). As a matter of fact, investigating (4.34) gives the two-sided bound

$$1 - e^{-c+1} \leq \frac{\tilde{c}(c)}{c} \leq 1 , \qquad c \geq 1 \tag{4.35}$$

or also the simpler but weaker

$$1 - \varepsilon_1 \leq \frac{\tilde{c}(c)}{c} \leq 1 , \qquad c \geq 1 + \varepsilon_2 \tag{4.36}$$

for some pairs $(\varepsilon_1, \varepsilon_2)$, including, for example, $(\varepsilon_1, \varepsilon_2) = (1/240, 4)$. We shall now analyse the resulting expression (4.27) for the lower bound on Φ. From this point on, for brevity we shall use m for m^*, the argmax of the function $\phi(\boldsymbol{r})$, and r for r_{m^*}. We shall also call for short $Q = \ln(s\, m^*\, r_{m^*}) = \ln(s\, m\, r)$, so that $\phi(\boldsymbol{r}) = Q/m$. We thus have, neglecting round-offs,

$$c = \frac{r(L+1-m)}{t^2\, s^t} ; \quad \tilde{p}_t = 1 - \binom{L}{t} (1 - \pi_{c\, s^t, s^t}) . \tag{4.37}$$

Monotonicity in $\tilde{p}$ gives that, under the condition $\tilde{p}_t \leq 1 - (2s\, \kappa_s \ln s)^{-1}$, for some given constant κ_s, we could get the desired summand $1/(2s\,\kappa_s\,m_{\min})$, on the LHS of (1.7), from the second term alone of (4.27). So, in order to conclude we could just replace (4.27) by the simpler

$$\frac{1}{\ln s}\frac{L\,\phi(\boldsymbol{r})}{\kappa_s t} \leq \tilde{p}_t \leq 1-\frac{1}{2s\,\kappa_s \ln s}\,. \tag{4.38}$$

In our parametrisation a condition holds for all possible dictionary contents $\boldsymbol{r}$ if it holds for $m_{\min} \geq 2$, $m \geq m_{\min}$ and $1 \leq r \leq s^m$ (recall that $r_m \leq s^m$), and this latter condition can be rephrased as $\ln(sm) \leq Q \leq \ln(sm) + m\ln s$. Define $\Omega_s \subseteq (\mathbb{R}^+)^2$ as $\Omega_s = \{(m,Q) \mid m \geq 2\,,\, \ln(sm) \leq Q \leq \ln(sm)+m\ln s\}$. Then (4.38) can be rephrased as follows: for any $s \geq 2$, there exists a positive real value κ_s such that, for all $(m,Q) \in \Omega_s$, there exist values t and L satisfying the forementioned constraints, with c and $\tilde{p}_t$ as in (4.37).

This problem is, in principle, just a matter of elementary calculus. The only difficulty is the fact that the involved expressions are quite cumbersome. The use of the union bound in deriving (4.28) is a very crude approximation, which does not even make clear the fact that p_t is a positive quantity (as its estimate $\tilde{p}_t \leq p_t$) may be negative). We shall start by investigating under which conditions we have, say, $\tilde{p}_t \geq 1-\delta$, for some $0<\delta<1$. This condition reads

$$\binom{L}{t}(1-\pi_{cs^t,s^t}) \leq \delta \tag{4.39}$$

which, by Stirling approximation, can be strengthened into

$$1-\pi_{cs^t,s^t} \leq \delta\left(\frac{Le}{t}\right)^{-t} \tag{4.40}$$

By using $\pi_{cs^t,s^t} \geq \exp(-s^t e^{-\tilde{c}})$, and $1-e^{-x} \leq x$, this can be strengthened into

$$s^t e^{-\tilde{c}} \leq \delta\left(\frac{Le}{t}\right)^{-t}\,, \tag{4.41}$$

that is, we shall verify the condition

$$\tilde{c} \geq -\ln\delta + t\ln\left(\frac{Les}{t}\right)\,, \tag{4.42}$$

or, by using the bound (4.36), its strengthening

$$(1-\varepsilon_1)c \geq -\ln\delta + t\ln\left(\frac{Les}{t}\right)\,;\quad c \geq 1+\varepsilon_2\,. \tag{4.43}$$

At this point, we shall find values of L, t and δ satisfying both (4.43) and the two sides of (4.38) (but in the simplifying situation in which we use $\tilde{p}_t \geq 1-\delta$ instead of (4.37)).

Instead of solving a maximisation problem, we provide an *ansatz* for values (L,t,δ) which are almost optimal for the given (s,m,r). To start with, we shall choose to reparametrise the dependence from L and t into a dependence from two more convenient new variables, a and τ, defined by

$$L = \frac{Q+a}{Q}m\,; \qquad t = \tau\frac{Q+a}{\ln s}\,. \tag{4.44}$$

We have a range $a,\tau \geq 0$, accounting for $t \geq 0$ and $L \geq m$. The left-most inequality in (4.38), which implicitly gives a bound on $\phi(\boldsymbol{r})$ in terms of $\tilde{p}_t$, L, t and s, considerably simplifies under this reparametrisation, and gives

$$\begin{aligned}\frac{(\ln s)\,t\,\tilde{p}_t}{L}\frac{\kappa m}{Q} &\geq \frac{(\ln s)\,t\,(1-\delta)}{L}\frac{\kappa m}{Q} \\ &= \frac{(\ln s)\,\frac{(Q+a)\tau}{\ln s}\,(1-\delta)}{\frac{Q+a}{Q}m}\frac{\kappa m}{Q} = \tau\kappa\,(1-\delta) \geq 1\,,\end{aligned}$$

while right-most inequality in (4.38), which is already rather simple in form, reads

$$\kappa \geq \frac{1}{2\delta s\ln s} \tag{4.45}$$

from which we deduce that the following choice for κ_s is legitimate

$$\kappa_s = \min_{(\tau,\delta)\in\overline{\Omega}_s} \max\left(\frac{1}{(1-\delta)\tau}, \frac{1}{2\delta s\ln s}\right) \tag{4.46}$$

where $\overline{\Omega}_s \subseteq [0,1]^2$ is the region of parameters (τ,δ) such that (4.43) is verified (not to be confused with Ω_s, which is the domain in (m,Q) in which (4.43) shall be verified). The reparametrisation is convenient also for the quantities entering (4.43), and it gives

$$c = \frac{e^{Q(1-\tau)}}{Q(Q+a)^2\tau^2}\frac{ae^{-\tau a}(\ln s)^2}{s}\,; \tag{4.47}$$

$$t\ln\left(\frac{Les}{t}\right) = (Q+a)\tau\frac{\ln\frac{me}{Q\tau\ln s}}{\ln s}\,. \tag{4.48}$$

The structure of equations (4.43) and (4.47) shows the emergence of a natural barrier to obtain an arbitrarily large lower-bound, as it should be the case. We can heuristically observe that, in the limit of large Q, the exponential factor $e^{Q(1-\tau)}$ in c dominates over the algebraic powers of Q, thus satisfying the inequality, *provided that* $\tau<1$. This in turns gives a maximal value of t, which, from (4.27), gives a maximal value for the lower bound that can be attained with the present methods.

Note that we have

$$\begin{aligned}\ln\frac{me}{Q\tau\ln s} &= \ln m - \ln\tau - \ln Q + \ln\frac{e}{\ln s}\\ &\le Q - \ln\tau - \ln Q + \lambda_s\,,\end{aligned}\tag{4.49}$$

with $\lambda_s = 1 - \ln s - \ln\ln s$, because $r \ge 1$. As a result we have

$$\begin{aligned}t\ln\left(\frac{Les}{t}\right) &= (Q+a)\tau\frac{\ln\frac{me}{Q\tau\ln s}}{\ln s}\\ &\le (Q+a)\tau\frac{Q-\ln\tau-\ln Q+\lambda_s}{\ln s}\,.\end{aligned}\tag{4.50}$$

Substituting (4.47) and (4.50) into (4.43) gives

$$\begin{aligned}\frac{e^{Q(1-\tau)}}{Q(Q+a)^2\tau^2}\frac{ae^{-\tau a}(\ln s)^2}{s} &\ge \max\Bigg[1+\varepsilon_2,\\ \frac{1}{1-\varepsilon_1}&\left(-\ln\delta+(Q+a)\tau\frac{Q-\ln\tau-\ln Q+\lambda_s}{\ln s}\right)\Bigg]\,;\end{aligned}\tag{4.51}$$

This shall be verified for all $(m,Q)\in\Omega_s$, with values ε_1 and ε_2 fixed, e.g., to $\varepsilon_1 = 1/240$ and $\varepsilon_2 = 4$, which are a legitimate pair. However, the expressions depend on Q but *not* on m (this has come as a result of the smarter parametrisation in a and τ, and of some manipulations of the bounds). As a consequence, instead of optimising for $(m,Q)\in\Omega_s$, we shall just optimise for $Q\in[\ln(2s),+\infty[$.

At this point it is already clear that we will succeed in finding a finite suitable value of κ_s satisfying (4.46). Indeed, consider the case in which $\delta(1-\delta)=\Theta(1)$ and $\tau=\Theta(\varepsilon)$. We have $\kappa_s=\Theta(\varepsilon^{-1})<\infty$, and, in (4.51),

$$\begin{aligned}\Theta(\varepsilon^{-2})\tfrac{e^Q}{Q(Q+a)^2}\tfrac{a(\ln s)^2}{s} &\ge \max\Bigg[\Theta(1),\\ \Bigg(\Theta(1)+(Q+a)&\left(\Theta(\varepsilon)\tfrac{Q-\ln Q+\lambda_s}{\ln s}+\Theta(\varepsilon\ln\varepsilon)\tfrac{1}{\ln s}\right)\Bigg)\Bigg]\end{aligned}$$

for $Q\in[\ln(2s),+\infty[$, which, at sight, is satisfied for ε small enough, even with an arbitrary choice of a finite. So, now it is just a matter of providing explicit values, and trying to optimise them to some extent.

Let us fix tentatively $a=Q/2$, and reparametrise τ as $\tau = T\ln s/\sqrt{s}$. This gives in (4.51) (substitute $\varepsilon_1 = 1/240$, $\varepsilon_2 = 4$, and use $Q\ge\ln(2s)$)

$$\begin{aligned}\frac{2e^{Q(1-T\frac{3\ln s}{2\sqrt{s}})}}{(3TQ)^2} &\ge \max\Bigg[5,\\ \frac{240}{239}&\left(-\ln\delta+\frac{3TQ(Q-\ln T)}{2\sqrt{s}}\right)\Bigg]\,.\end{aligned}\tag{4.52}$$

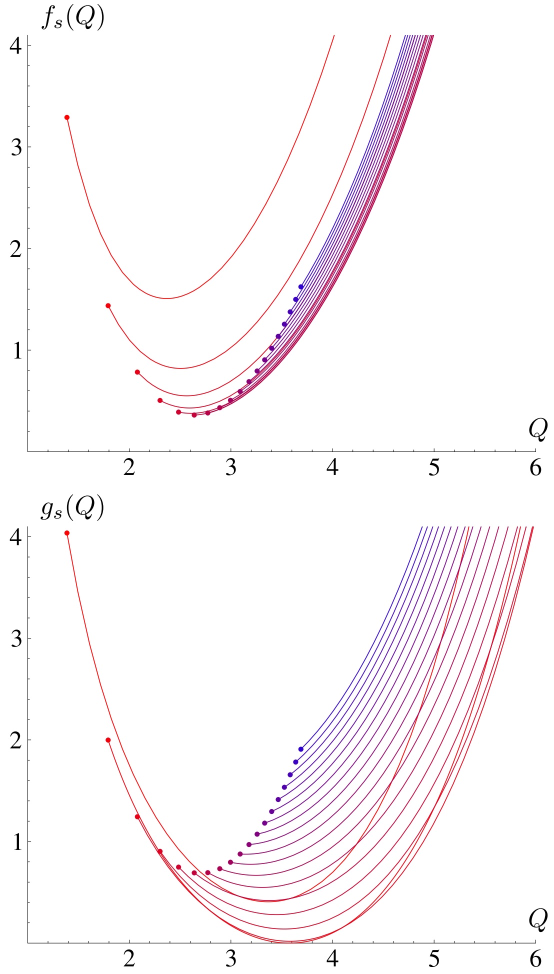

Figure 4: Curves $f_s(Q)$ and $g_s(Q)$, for $s=2,\dots,20$ (red to blue), at the values $(T,\delta)=(0.21,0.03)$. The fact that all these curves remain above the real axis implies that $(T\frac{\ln s}{\sqrt{s}},\delta)\in\overline{\Omega}_s$ for all $s\ge 2$.

It turns out that the inequality above is always satisfied for $s\in\{2,3,4,\dots\}$, $Q\ge\ln(2s)$, $T\le 0.212011$ and $\delta\ge 0.03613$, the worst case being $s=3$ and $Q=3.57575\dots$, as indeed, for fixed values of T and δ, we have a family (in s) of inequalities of the form $a_s(Q)\ge\max\big(b_s(Q),c_s(Q)\big)$, which is satified if all the functions $f_s(Q):=a_s(Q)-b_s(Q)$ and $g_s(Q):=a_s(Q)-c_s(Q)$ are positive-valued in their ranges. The behaviour for Q or s large is easily bounded, while the one near the worst-case region $s,Q=\mathcal{O}(1)$, is illustrated in Figure 4. This determines that $(T\frac{\ln s}{\sqrt{s}},\delta)\in\overline{\Omega}_s$ for all $s\ge 2$. Thus, we can substitute the corresponding values of (τ,δ) in (4.46), where it turns out that the max in (4.46) is always attained on the left-most quantity.

Using a round-off upper bound for the resulting constant $\frac{1}{(1-\delta)T}=4.89354<5$ finally gives the valid choice

$$\kappa_s = 5\frac{\sqrt{s}}{\ln s}\,. \tag{4.53}$$

This proves Proposition 4.1.

The combination of Propositions 3.1 and 4.1 clearly provides Theorem 1.1.

We recall that δ has an interesting interpretation: for each content $\boldsymbol{r}$ there exists at least a fraction $1-\delta$ (i.e., roughly the 96.3%) of the dictionaries of content $\boldsymbol{r}$ such that the complexity of MPMP is at least $\frac{1}{\kappa_s(1-\delta)}\phi(\boldsymbol{r})$, i.e. has the same functional dependence of the upper bound to the worst-case dictionary. The procedure above also makes clear that the fraction $1-\delta$ can be chosen arbitrarily near to 1, at the price of having a larger factor in place of κ_s.

References

[1] A. V. Aho and M. J. Corasick, *Efficient string matching,* Commun. ACM **18** (1975) 333-340.

[2] A. Apostolico and Z. Galil, *Pattern Matching Algorithms,* Oxford Univ. Press, 1997.

[3] F. Bassino and C. Nicaud, *Enumeration and Random Generation of Accessible Automata,* Theor. Comp. Science **381** (2007) 86-104.

[4] R. S. Boyer and J. S. Moore, *A Fast String Searching Algorithm,* Commum. ACM **20** (1977) 762-772.

[5] M. Crochemore, C. Hancart and T. Lecroq, *Algorithms on Strings,* Cambridge Univ. Press, 2014.

[6] Ph. Flajolet and R. Sedgewick, *Analytic Combinatorics,* Cambridge Univ. Press, 2009.

[7] K. Fredriksson and S. Grabowski, *Average-optimal string matching,* J. Discr. Algo. **7** (2009) 579-594.

[8] K. Fredriksson and G. Navarro, *Average complexity of exact and approximate multiple string matching,* Theor. Comput. Sci. **321** (2004) 283-290.

[9] D. Gusfield, *Algorithms on Strings, Trees and Sequences,* Cambridge Univ. Press, 1997.

[10] Ph. Jacquet and W. Szpankowski, *Analytic Pattern Matching: from DNA to Twitter,* Cambridge Univ. Press, 2015.

[11] D. Knuth, J. H. Morris and V. Pratt, *Fast pattern matching in strings,* SIAM J. Comput. **6** (1977) 323-350.

[12] M. Lothaire, *Applied Combinatorics on Words*, vol. 105 of *Encyclopedia of Mathematics and its Applications*, Cambridge Univ. Press, 2005.

[13] R. L. Rivest, *On the Worst-Case Behavior of String-Searching Algorithms,* SIAM J. Comput. **6** (1977) 669-674.

[14] A. C.-C. Yao, *The Complexity of Pattern Matching for a Random String,* SIAM J. Comput. **8** (1979) 368-387.

A A bound on harmonic sums

Define

$$S(x) := \sum_{m\geq x} \frac{1}{m^2}\,. \tag{A.1}$$

The main aim of this section is to determine the following fact.

LEMMA A.1. *The function*

$$S_k^+(x) := \frac{k}{x}\Big(\zeta(2) - \sum_{m=1}^{k-1}\frac{1}{m^2}\Big) \tag{A.2}$$

is an upper bound to $S(x)$ in the domain $x \geq k$:

$$S(x) \leq S_k^+(x) \qquad \forall\ x-k \in \mathbb{N}\,. \tag{A.3}$$

Clearly, we have in particular $S_k^+(k) = S(k)$. The lemma follows as an immediate corollary of the following proposition providing the induction step.

PROPOSITION A.1. *If*

$$\sum_{m\geq x}\frac{1}{m^2} \leq \frac{A}{x} \tag{A.4}$$

then

$$\sum_{m\geq x+1}\frac{1}{m^2} \leq \frac{A}{x+1}\,. \tag{A.5}$$

This proposition, in turns, will be proven by using standard facts on the monotone transport of measures, which we remind here.

DEFINITION 1. *Let $\mu(k)$, $\mu'(k)$ be two normalised probability measures on $\mathbb{N}$. We say that μ' is the* monotone transport *of μ if there exists a matrix B, lower-triangular and left-stochastic, such that $\mu'(h) = \sum_k B_{hk}\,\mu(k)$.*

Then we have

PROPOSITION A.2. *Let $f(k)$ a real-valued function on $\mathbb{N}$. Define*

$$\langle f\rangle_\mu := \sum_k f(k)\mu(k)\,. \tag{A.6}$$

If f is a monotone decreasing function, and μ' is the monotone transport of μ, then $\langle f\rangle_{\mu'} \leq \langle f\rangle_\mu$.

Proof. The matrix B in Definition 1 has the property $B_{hk} \in \mathbb{R}^+$, $B_{hk} = 0$ if $k > h$, and $\sum_h B_{hk} = 1$ for all k. Thus we have, on one side,

$$\langle f\rangle_\mu = \sum_k f(k)\mu(k) = \sum_{k\leq h} f(k)B_{hk}\,\mu(k) \tag{A.7}$$

and on the other side

$$\langle f\rangle_{\mu'} = \sum_h f(h)\mu'(h) = \sum_{k\leq h} f(h)B_{hk}\,\mu(k) \tag{A.8}$$

so that

$$\langle f\rangle_{\mu} - \langle f\rangle_{\mu'} = \sum_{k\leq h} \big(f(k)-f(h)\big)B_{hk}\,\mu(k) \tag{A.9}$$

As B_{hk}, $\mu(k)$ and $f(k)-f(h)$ are separately non-negative, the latter because of the monotonicity of f, all the terms in the sum are non-negative, and the statement follows. □

We can now pass to the proof of the main statement. *Proof of Proposition A.1.* Rewrite (A.4) as

$$A \geq \frac{\sum_{m\geq x} \frac{1}{m(m+1)} \frac{m+1}{m}}{\sum_{m\geq x} \frac{1}{m(m+1)}} \tag{A.10}$$

If we define $\mu_x(m)$ as the probability measure on the integers $\mu_x(m) = \frac{x}{m(m+1)} 1_{m\geq x}$, we thus have

$$A \geq \mathbb{E}_{\mu_x}\Big(\frac{m+1}{m}\Big). \tag{A.11}$$

The measure μ_{x+1} is such that $\mu_{x+1}(m) > \mu_x(m)$ if $m > x$ and $\mu_{x+1}(m) = 0 < \mu_x(m)$ if $m = x$. As a result, μ_{x+1} is the monotone transportation of $\mu(x)$. The matrix B is easily calculated[8], although irrelevant at our purposes. This fact, together with the monotonicity of the function $\frac{m+1}{m}$, by Proposition A.2 imply

$$\mathbb{E}_{\mu_x}\Big(\frac{m+1}{m}\Big) \geq \mathbb{E}_{\mu_{x+1}}\Big(\frac{m+1}{m}\Big) \tag{A.12}$$

and thus Proposition A.1. □

[8] $B_{mm} = 1$ for $m > x$, $B_{mx} = \frac{x+1}{m(m+1)}$ for $m > x$, all other entries are zero.