Algorithms for Molecular Biology BioMed Central

Research

Open Access

The approximability of the String Barcoding problem

Giuseppe Lancia* and Romeo Rizzi

Address: Dipartimento di Matematica ed Informatica, Universitá di Udine, Via delle Scienze 206, Udine, Italy

Email: Giuseppe Lancia* - lancia@dimi.uniud.it; Romeo Rizzi - rizzi@dimi.uniud.it

* Corresponding author

Published: 08 August 2006

Algorithms for Molecular Biology 2006, **1**:12 doi:10.1186/1748-7188-1-12

Received: 16 May 2006
Accepted: 08 August 2006

This article is available from: http://www.almob.org/content/1/1/12

Abstract

The String Barcoding (SBC) problem, introduced by Rash and Gusfield (RECOMB, 2002), consists in finding a minimum set of substrings that can be used to distinguish between all members of a set of given strings. In a computational biology context, the given strings represent a set of known viruses, while the substrings can be used as probes for an hybridization experiment via microarray. Eventually, one aims at the classification of new strings (unknown viruses) through the result of the hybridization experiment. In this paper we show that SBC is as hard to approximate as Set Cover. Furthermore, we show that the constrained version of SBC (with probes of bounded length) is also hard to approximate. These negative results are tight.

Background

The following setting was introduced by Rash and Gusfield in [1]: Given a set V of n strings $v_1,...,v_n$ (representing the genomes of n known viruses), and an extra string s (representing a virus in V, but not yet classified), we aim at recognizing s as one of the known viruses through an hybridization experiment. In the experiment, we utilize a set $\prod$ of k probes (DNA strings) and we will are able to determine which ones are contained in s (as substrings) and which are not. The result of the experiment is therefore a binary k-vector (called, in [1] a *barcode*) which can be seen as the signature of s with respect to the given probes. In order for the barcode to be able to discriminate between all the viruses, it must be true that, for each pair of viruses v_i, v_j, with $1 \le i < j \le n$, there exists at least one $\pi \in \prod$ which is a substring of either v_i or v_j but not of both. This amounts to saying that the barcodes of all v_i's must be distinct binary k-vectors. The cost of the hybridization experiment turns out to be proportional to k, and therefore the goal of the optimization problem, known as Minimum String Barcoding (SBC), is to find a feasible set $\prod$ of smallest possible cardinality. The problem has been popularized by Rash and Gusfield [1], who proposed an Integer Programming approach for its solution. In [2,3], DasGupta et al. describe a greedy algorithm for *robust* barcoding (i.e., where each pair of viruses must be distinguished by at least a given number l of probes), which scales well to whole-genome sequences. For real-life instances, this algorithm is more effective than alternative approaches [1,4] whose time complexity grows very quickly with the length of the input sequences.

In [1], Rash and Gusfield stated that a variant of SBC, in which the maximum length of each probe is bounded by a constant, and the alphabet size is at least 3, is NP-hard. As for the unconstrained case, where no bound is given on the length of each probe, they left as an open problem to determine whether this version of SBC is NP-complete or not. In this paper we prove that both SBC and unconstrained SBC are in fact NP-complete already for binary alphabets. We do so by actually linking the approximability of SBC (both constrained and unconstrained) to the approximability of the classical Set Cover problem. This way, a sharp $\log n$ bound on the best achievable approxi-

mation ratio is established for both versions of SBC. It must here be said that essentially the same result has independently been obtained, and already published, by Berman et al. [5]. The inapproximability result in [5] actually holds for a very general family of Minimum Test Collection problems which includes unconstrained SBC as a special case. However, our inapproximability result for constrained SBC is not covered by the general framework proposed in [5]. Note that the very nature of the hybridization experiment imposes that the used probes cannot be too long for technological and biological reasons (such as possible self-hybridization of the probes). Therefore, the bounded-length SBC problem is quite important in practice. In [5] the authors also obtain a $(1 + \log n)$-approximation algorithm for the general Minimum Test Collection problem. Their result is the first improvement over the $\log n^2 = 2 \log n$ approximation ratio that can essentially be achieved by a standard reduction of Minimum Test Collection to Set Cover followed by a run of the classical set covering greedy algorithm. Thanks to this positive result, all the bounds on the approximability ratios obtained either here or in [5] are tight also in terms of the multiplicative constant of the $\log n$ factor. This $(1 + \log n)$-approximation proposed in [5] is a greedy algorithm in which the choice of the test set to be added at each step is driven by a suitable entropy function. The analysis of the algorithm, also given in [5], is an elegant and non-trivial reinterpretation of the celebrated proof by Lovasz of the approximation ratio of the greedy algorithm for set cover.

The remainder of the paper is organized as follows. In next section, we introduce the Minimum Test Collection problem (MTC), a known NP-complete problem (see, e.g., Garey and Johnson [6]) for which set-cover-like inapproximability results are known [7]. We also introduce a restricted version of MTC and we show that the same inapproximability results hold for this restricted version as well. In the following section, we address the computational complexity of SBC and show that the approximation algorithm by Berman, DasGupta and Kao [5] delivers an essentially tight approximation ratio even for constrained SBC. More precisely, in the opening of the section we introduce formally the string barcoding problems studied and also point out that every SBC instance (either constrained or unconstrained) can be formulated as an MTC instance, which directly implies set-cover-like approximability results for SBC. We also observe here that the constrained SBC problem, when parameterized over the maximum probe length and the alphabet size, is in FPT and, in particular, it can be solved in linear time whenever these parameters are fixed (for a comprehensive treatment of FPT theory, see [8]). Next, we prove set-cover-like inapproximability results for SBC and for the maximum-length version of SBC via a common reduction from the restricted version of MTC introduced in the first section. (The NP-hardness of the maximum-length version of SBC had been already stated in [1], although without reporting the proof).

A starting problem: the Min Test Collection

In this section we introduce the Minimum Test Collection (MTC) problem, both in its general form and in a restricted version. We also report (and obtain) set-cover-like inapproximability results for MTC and its restricted version. Both the inapproximability of MTC and that of its restricted version will be used in later sections, when characterizing the approximability of the two variants of SBC.

The MTC problem, as defined in [6], is the following problem.

MTC INSTANCE

$D = \{d_1,...,d_p\}$: a set of (ground) elements.

$\mathcal{T} = \{T_1,...,T_q\}$: a set of subsets of D (representing tests that may *succeed* or *fail* on the elements. A test T succeeds on d if $d \in T$ and fails on d otherwise).

MTC PROBLEM

Find a minimum-size set $\mathcal{T}' \subseteq \mathcal{T}$ such that for any pair of elements $d, d' \in D$ there is at least one test $T \in \mathcal{T}'$ such that $|\{d, d'\} \cap T| = 1$ (i.e., the test fails on one element and succeeds on the other). A set that verifies this property is called a *testing set* of D; $\mathcal{T}'$ is a *minimum testing set* of D.

The MTC problem appears in many contexts. For example, the elements may represent a set of p diseases, and the T_i are diagnostic tests, that can verify the presence/absence of q symptoms. The goal is to minimize the number of symptoms whose presence/absence should be verified in order to correctly diagnose the disease. In [6], Garey and Johnson proved that MTC is NP-complete by reducing 3-dimensional Matching (3DM), which is NP-complete [9], to it. In [7] it was also proven by means of a reduction from Set Cover that no fully polynomial-time approximation scheme exists for MTC, unless P = NP. Later in this section we essentially employ this reduction. The same reduction had also been reconsidered in [10] where it was shown that MTC is not approximable within $(1 - \varepsilon) \log p$ for any $\varepsilon > 0$. We now introduce a special type of MTC instances, which we call *standard*. In this version of the problem, some particular tests must always be part of the problem instance.

In order to define these particular instances, assume the elements in D are ordered as $d_1,...,d_p$ and let $D_j = \{d_j,...,d_p\}$ for $j = 1,...,p$. A set of tests $\mathcal{T}$ is called *suffix-closed* if $D_j \cap$

$T \in \mathcal{T}$ for each $T \in \mathcal{T}$ and $j = 1,...,p$. A suffix-closed set of tests $\mathcal{T}$ is called *standard* if $D_j \in \mathcal{T}$ and $\{d_j\} \in \mathcal{T}$ for each $j = 1,...,p$. An instance $(D, \mathcal{T})$ of MTC is *standard* when $\mathcal{T}$ is standard. In other words, a standard instance of MTC consists of a finite set $D = \{d_1,...,d_p\}$ and a set of tests $\mathcal{T}$ which can be written as $\mathcal{T} = \mathcal{T}_D \cup \mathcal{T}_I \cup \mathcal{T}_A \cup \mathcal{T}_E$, where

$\mathcal{T}_D = \{S_1,...,S_{q'}\}$: a generic set of subsets of D;

$\mathcal{T}_I = \{S_{q'+1},...,S_{q'+p}\} = \{\{d_i\} \mid i = 1,...,p\}$;

$\mathcal{T}_A = \{S_{q'+p+1},...,S_{q'+2p}\} = \{D_j \mid 1 \le j \le p\}$;

$\mathcal{T}_E = \{S_{q'+2p+1},...,S_{p(q'+2)}\} = \{S \cap D_j \mid S \in \mathcal{T}_D, 2 \le j \le p\}$.

Note that $\mathcal{T}_D$, $\mathcal{T}_I$, $\mathcal{T}_A$ and $\mathcal{T}_E$ may have non-empty intersection. In other words, where we assume $\mathcal{T} = \{T_1,...,T_q\}$ with $q = p(q' + 2)$ and $T_i = S_i$ for $i = 1,2,...,q$, then it might be the case that $T_i = T_j$ with $i \neq j$.

We now prove the following result.

Theorem 1 *Minimum Test Collection (MTC) cannot be approximated within* $(1 - \varepsilon) \log p$ *for any* $\varepsilon > 0$ *even when restricted to standard instances.*

We prove the above theorem by a reduction from the Set Cover (SC) problem, which is defined ([11]) as follows.

SC INSTANCE

A finite set $S = \{s_1,...,s_m\}$ and a collection $\mathcal{C} = \{C_1,...,C_n\} \subseteq 2^S$ such that $S = \bigcup_{i=1}^{n} C_i$.

SC PROBLEM

Find a minimum-size collection $\mathcal{C}' \subseteq \mathcal{C}$ such that every element in S belongs to at least one subset in $\mathcal{C}'$, i.e.

$$S = \bigcup_{C \in \mathcal{C}'} C. \qquad (1)$$

We say that any $\mathcal{C}'$ satisfying (1) *covers* S, and we call such a set a *set cover* for S.

It is well known that SC cannot be approximated within $(1 - \varepsilon) \log m$ for any $\varepsilon > 0$ (see [12]).

Let $S = \{s_1,...,s_m\}$ and $\mathcal{C} = \{C_1,...,C_n\} \subseteq 2^S$ be an arbitrary instance of SC. We show how to obtain a standard instance of MTC representing the given instance of SC.

First, let $K := 2^k$ be the smallest power of 2 such that $K \ge m$. To each $j \in \{1, 2,..., K\}$, we associate a unique binary string $b(j)$ of length k. Let $R := \{r_1,...,r_K\}$, be a set of size K with $R \cap S = \emptyset$. The set of elements D is defined as $D = R \cup S$, with a particular order:

$$D = \{r_1, s_1, r_2, s_2, ..., r_m, s_m, r_{m+1}, r_{m+2} ..., r_K\}$$

(i.e., $D = \{d_1, ..., d_p\}$ with $p = m + K$). The set of tests $\mathcal{T}$ is constructed in the following way. First, for each $i = 1,...,k$, we call T_i the test containing all the r_j and s_j such that the bit in position i of the binary string $b(j)$ is set to 1. Then let $\mathcal{T} = \mathcal{T}_D \cup \mathcal{T}_I \cup \mathcal{T}_A \cup \mathcal{T}_E$ where

$\mathcal{T}_D = \mathcal{C} \cup \{T_i \mid i = 1,...,k\}$,

$\mathcal{T}_I = \{\{d_i\} \mid i = 1,...,p\}$,

$\mathcal{T}_A = \{D_j \mid 1 \le j \le p\}$,

$\mathcal{T}_E = \{T \cap D_j \mid T \in \mathcal{T}_D, 2 \le j \le p\}$.

The following two lemmas investigate the properties of the proposed reduction.

Lemma 1 *If S has a set cover* $\mathcal{C}' \subseteq \mathcal{C}$ *of size h, then D has a testing set* $\mathcal{T}' \subseteq \mathcal{T}$ *of size at most h + k.*

Proof: Let $\mathcal{C}' \subseteq \mathcal{C}$ be a set cover for S of size h. We claim that $\mathcal{T}' := \mathcal{C}' \cup \{T_i \mid i = 1,...,k\}$ is a testing set for D, which proves the lemma. Indeed, consider two elements s_i (or r_i) and s_j (or r_j). If $i \neq j$ then the binary strings associated to i and j differ in some position x, and hence T_x distinguishes between them. Otherwise, if $i = j$ and the two elements still differ, then we are talking about s_i and r_i, for some $i = 1,...,m$. Notice that s_i is contained in at least one set C in $\mathcal{C}'$ since $\mathcal{C}'$ covers S. Moreover, $r_i \notin C$ since $C \subseteq S$. It follows that there exists some set in $\mathcal{C}'$, and hence in $\mathcal{T}'$, which distinguishes between s_i and r_i. □

Lemma 2 *If D has a testing set* $\mathcal{T}' \subseteq \mathcal{T}$ *of size h, then S has a set cover* $\mathcal{C}' \subseteq \mathcal{C}$ *of size at most h.*

Proof: Let $\mathcal{T}' \subseteq \mathcal{T}$ be a testing set of D of size h. We propose a polynomial-time algorithm to produce a set $\mathcal{C}' \subseteq \mathcal{C}$ with $|\mathcal{C}'| \leq |\mathcal{T}'|$ such that $\mathcal{C}' \cup \{T_i \mid i = 1,...,k\}$ is also a testing set of D. At the end, we argue that such a $\mathcal{C}'$ must be a set cover of S.

Let $X = \mathcal{T}'$. Clearly, $X \cup \{T_i \mid i = 1,...,k\}$ distinguishes all the elements in D, and this invariant will be maintained throughout the algorithm. If $X \subseteq \mathcal{C}$, then we just let $\mathcal{C}' = X$, and stop. Otherwise, let $T \in X \setminus \mathcal{C}$. Notice that all pairs of elements which are not distinguished by $(X \setminus \{T\}) \cup \{T_i \mid i = 1,...,k\}$ necessarily belong to the set $P = \{\{s_i, r_i\} \mid i = 1,...,m\}$. Our plan is hence to replace T by any set in $\mathcal{C}$ which distinguishes all the pairs in P that are distinguished by T. It remains to show that such a set in $\mathcal{C}$ always exists. Indeed, if T is a test D_j with $j = 2i$ and $j \leq 2m$, then the ordering we have imposed among the elements of D implies that T distinguishes only the pair $\{s_i, r_i\}$ of P, so it can be replaced by any $C \in \mathcal{C}$ with $s_i \in C$; if T is a test D_j with j odd or $j > 2m$, then T distinguishes no pair in D, so that T can be dropped from X without the need for any replacement. If T is a test of the form $T_i \cap D_j$, then it again distinguishes at most one pair in P, and a similar reasoning holds. The same holds if $T \in T_I$, that is, $T = \{d\}$ for some $d \in D$. Finally, if T is a test $C \cap D_j$ for some $C \in \mathcal{C}$, then, clearly, it can be replaced with C. Hence, by substituting every test $T \in X \setminus \mathcal{C}$ by tests in $\mathcal{C}$ as shown, we obtain that $X \subseteq \mathcal{C}$, and we let $\mathcal{C}' = X$.

We now argue simply that, since $\mathcal{C}' \cup \{T_i \mid i = 1,...,k\}$ is a testing set of D, then $\mathcal{C}'$ is a set cover of S. Indeed, no pair in P is distinguished by a set T_i. Therefore, for each $j = 1,...,m$, the pair $\{r_j, s_j\}$ is distinguished by some test $\overline{T} \in \mathcal{C}'$. Moreover, since $r_j \notin T$ for any $T \in \mathcal{C}' \subseteq \mathcal{C}$, it must be that $s_j \in \overline{T}$. Therefore, each s_j is covered, and $\mathcal{C}'$ is a set cover of S. □

With Lemmas 1 and 2, we are now ready to prove Theorem 1.

Proof of Theorem 1: We first remark that SC is not approximable within $(1 - \varepsilon) \log m$ even when restricted to instances for which $opt = \omega(\log m)$. Indeed, just consider duplicating a generic instance of SC into $t := \lceil \log^2 m \rceil = \omega(\log m)$ identical and disjoint copies to obtain a new instance $(S^*, \mathcal{C}^*)$ with $|S^*| = tm$. Let opt denote the optimum value for the original instance $(S, \mathcal{C})$ and opt^* the optimum value for the instance $(S^*, \mathcal{C}^*)$. Then $opt^* = t\, opt \geq t = \omega(\log|S^*|)$. Notice also that a solution to the instance $(S^*, \mathcal{C}^*)$ of size at most $opt^*(1 - \varepsilon) \log|S^*|$ could be immediately translated into a solution to the instance $(S, \mathcal{C})$ of size at most

$$\frac{1}{t} opt^*(1-\varepsilon)\log|S^*| = opt(1-\varepsilon)\log|S^*| = (1-\varepsilon)(\log m + \log t)opt.$$

Here, $\varepsilon > 0$ and $\log t = o(\log m)$, in contrast with the inapproximability results explicitly derived in [12]. In the analysis to follow we therefore assume that $opt = \omega(\log m)$.

Denote now by opt and opt' the optimal solution values for the original problem (SC) and the transformed problem (MTC) respectively, and by apx and apx' the values of the respective approximated solutions that we can produce in polynomial time. By Lemma 1,

$$opt' \leq opt + k = opt + o(opt).$$

Then, if we assume that we can obtain an approximate solution

$$apx' \leq f(|D|)opt'$$

for the MTC problem, we can also guarantee that

$$apx' \leq f(|D|)(opt + o(opt)).$$

Since the proof of Lemma 2 is constructive, we obtain that

$$apx \leq apx' \leq f(|D|)(opt + o(opt)).$$

Notice that $p := |D| \leq 2m$. Consequently, since we know that SC is not approximable within $(1 - \varepsilon) \log m$ for any $\varepsilon > 0$, then we can conclude that MTC is not approximable within $(1 - \varepsilon) \log p$ for any $\varepsilon > 0$. □

The String Barcoding problems

The following is a formal definition of the String Barcoding problem (SBC):

SBC INSTANCE

An alphabet Σ (e.g., $\Sigma = \{A, C, G, T\}$) and a set $V = \{v_1,...,v_n\}$ of strings over Σ (representing virus genomes).

SBC PROBLEM

Find a minimum-size set $\prod$ of strings such that for any pair of strings $v, v' \in V$ there is at least one string $\pi \in \prod$ such that π is a substring of v or v', but not of both. A set

that verifies this property is called a *testing set* of V; $\prod$ is a *minimum testing set* of V.

Rash and Gusfield state in [1] that it is unknown whether the basic String Barcoding problem is NP-hard or not and they also state that a variant of SBC called Max-length String Barcoding (MLSBC) is NP-hard when the underlying alphabet contains at least three elements. In this variant, a constraint on the maximum length of the substrings in $\prod$ is specified in input. More formally, MLSBC is the following problem:

MLSBC INSTANCE
An alphabet Σ, a set $V = \{v_1,...,v_n\}$ of strings over Σ and a constant L.

MLSBC PROBLEM
Find a testing set $\prod$ of V such that the length of each string $\pi \in \prod$ is less than or equal to L, and $\prod$ has smallest possible cardinality among such testing sets.

The main point of this paper is to link the approximability of SBC (both constrained and unconstrained) to the approximability of the classical Set Cover problem. Indeed, both SBC and MLSBC can be naturally regarded as instances of MTC, for which, in turn, a natural reduction to Set Cover is well known. In the next section we provide reductions for the reverse direction. These reductions will characterize the approximability of SBC and MLSBC from a computational complexity point of view. To better appreciate some aspects of these reductions, we make the following remark.

Fact 1 *MLSBC can be solved in linear time whenever L and* $|\Sigma|$ *are bounded by a constant.*

Proof: Indeed, the number of strings π which may possibly end up in the testing set $\prod$ is bounded by

$$f(|\Sigma|,L) := \sum_{t=1}^{L} |\Sigma|^t = \frac{|\Sigma|^{L+1}-1}{|\Sigma|-1},$$

whence the number of possible solutions is bounded by $2^{f(|\Sigma|,L)}$. Thus we have a constant number of possible solutions, and each can be checked in linear time. □.

Inapproximability of SBC and MLSBC
In this subsection we prove the inapproximability of both SBC and MLSBC by means of a common reduction from the restricted form of MTC introduced in Section.

Theorem 2 *The String Barcoding (SBC) problem cannot be approximated within* $(1 - \varepsilon) \log n$ *for any* $\varepsilon > 0$. *This negative result holds already for binary alphabets.*

Theorem 3 *The Max-length String Barcoding problem cannot be approximated within* $(1 - \varepsilon) \log n$ *for any* $\varepsilon > 0$. *This negative result holds already for binary alphabets.*

Let $D = \{d_1,...,d_p\}$ and $\mathcal{T} = \{T_1,...,T_q\} = \mathcal{T}_D \cup \mathcal{T}_I \cup \mathcal{T}_A \cup \mathcal{T}_E$ be a standard instance of MTC, with

$\mathcal{T}_D = \{T_1,...,T_{q'}\}$,

$\mathcal{T}_I = \{T_{q'+1},...,T_{q'+p}\}$,

$\mathcal{T}_A = \{T_{q'+p+1},...,T_{q'+2p}\}$,

$\mathcal{T}_E = \{T_{q'+2p+1},...,T_{p(q'+2)}\}$.

Where Ω is a set of strings, $\circ\sigma\in\Omega(\sigma)$ denotes the string obtained as the concatenation of all the strings in Ω lined up in lexicographic order (as a matter of fact, for the purpose of our reduction to work, the strings in Ω could be concatenated in any order, but we prefer to refer to a specific order so that the instance generated through the proposed reduction is uniquely defined).

An instance of SBC (or of MLSBC) is obtained in the following way. First, let $k = \lceil \log_2 q \rceil$. Then, let $\Sigma = \{A, B\}$ and $\Sigma_+ = \{A, B, X\}$ (the dummy symbol X will be used as a separator, to divide the really interesting substrings, made only of As and Bs). We will often treat Σ and Σ_+ as alphabets, even if the *intermediate symbols* A, B, and X actually stand for binary strings according to the rules: $A \mapsto 10101$, $B \mapsto 11011$, and $X \mapsto 00000$. Thanks to these rules, any given string in Σ^* or Σ_+^* ultimately represents a unique binary string in $\Sigma = \{0,1\}^*$. Let Σ^l denote the set of all the strings of length l over the alphabet Σ. Finally, uniquely encode each different test $T \in \mathcal{T}$ by a string $f_T \in \Sigma^k$ (called the *signature* of T) and let $F = \{f_T \mid T \in \mathcal{T}\}$; certainly this is possible since $|\Sigma^k| = 2^k \geq q = |\mathcal{T}|$. Now, the instance of SBC is completed by constructing the set of strings $V = \{v_j \mid j = 1,...,p\}$ such that each string $v_j \in V$ contains all the strings in Σ^{2k-1} plus the signatures $f \in F$ of those tests $T \in \mathcal{T}$ that succeed on d_j (that is, such that $d_j \in T$). More formally, the codification of an element $d_j \in D$ is the string

$$v_j = X^{2k+j} \bigcirc_{\sigma\in\Sigma^{2k-1}} (\sigma X^{2k+j}) \bigcirc_{T \ni d_j} (f_T f_T X^{2k+j})$$

seen as a binary string. Notice that the role of X is to separate the substrings, and that a different number of X characters is used in each string v in order to uniquely identify

it when dealing with one of its substrings which includes a whole block of *X*'s. The MLSBC instance is the same as the SBC instance plus the bound $L = 10\,k$.

The number and size of the strings constructed above, and hence the above described transformation from an MTC instance to either an SBC instance or an MLSBC instance, is polynomial. With the next two lemmas we show that this is an objective-function preserving reduction from MTC to either SBC or MLSBC whence Theorems 2 and 3 follow.

Lemma 3 *If D has a testing set $\mathcal{T}' \subseteq \mathcal{T}$ of size h, then V has a testing set $\prod$ of size at most h. Furthermore, $|\pi| \leq L$ for every $\pi \in \prod$.*

Proof: Consider the set of strings $\prod = \{f_T f_T \mid f_T$ is the signature of $T \in \mathcal{T}'\}$. Clearly, $|\prod| \leq |\mathcal{T}'|$ and we aim at showing that $\prod$ is a testing set for *V*. More precisely, we claim that the binary string $f_T f_T$ is a substring of the binary string v_j if and only if $d_j \in T$. Indeed, when $d_j \in T$, it follows immediately from the construction of v_j that $f_T f_T$ is a substring of v_j. As for the converse, when $f_T f_T$ is a substring of $v_{j'}$ then the shift of any of its occurrences within v_j is necessarily a multiple of 5, and hence $f_T f_T$ is actually a substring of v_j also when $f_T f_T$ and v_j are regarded as strings over Σ_+. It follows that $d_j \in T$. Notice moreover that $|\pi| = 10\,k \leq L$ for every $\pi \in \prod$. □

Lemma 4 *If V has a testing set of size h, then D has a testing set $\mathcal{T}' \subseteq \mathcal{T}$ of size at most h.*

Proof: We want to show that, given a testing set $\prod$ for *V*, there exists a testing set $\mathcal{T}' \subseteq \mathcal{T}$ for *D* with $|\mathcal{T}'| \leq |\prod|$. We actually commit ourselves to show that for every binary string $\pi \in \prod$ we can find a $T_\pi \in \mathcal{T}$ such that, for each $j = 1,\ldots,p$, the string π occurs as a substring of v_j if and only if $d_j \in T_\pi$. In following this plan of action, for each $\pi \in \prod$, we can clearly assume that π is a substring of some $v_j \in V$ but not all. Thus, if π contains a substring of the form $10^\gamma 1$ for some $\gamma > 1$, then γ is a multiple of 5, that is, $\gamma = 5t$, and, actually, $t = 2k + j$ with $1 \leq j \leq p$, in which case we can take $T_\pi := \{d_j\}$. This works since v_j is the only string in *V* of which π is a substring. Similarly, in case the string π contains no symbol 1 except in the first (or except in the last) $x \leq 2$ positions, and where $t = \lceil(|\pi| - x)/5\rceil$ (here we are assuming that the symbol in position *x* is forced to be a 1 if $x > 0$), then $t = 2k + j$ with $1 < j \leq p$, in which case we can take $T_\pi := D_j$. This works since v_i contains π as a substring if and only if $i \geq j$. Furthermore, in case 00 is not a substring of π, and since π is a substring of some $v_j \in V$ but not all, then $10\,k - 8 \leq |\pi| \leq 10\,k + 2$.

Actually, where π' is the longest substring of π which both begins and ends with 1, then $10\,k - 8 \leq |\pi'| \leq 10\,k$, and π' is a substring of $f_{\tilde{T}} f_{\tilde{T}}$ for precisely one $\tilde{T} \in \mathcal{T}$ – and in this case $T_\pi := \tilde{T}$ works. We are left with the case $\pi = 0^a 1\alpha 1 0^b$ with α containing no 00 substring and where one of *a* or *b* may possibly be 0 but $M := \max\{a, b\} \geq 2$. Assume w.l.o.g. that $a = M$. Again, let $t = \lceil M/5\rceil$. Clearly, we can assume $t \leq 2k + p$. If $t \leq 2k + 1$, then we can also assume that $1\alpha 1 0^b$ is a substring of $f_{\tilde{T}} f_{\tilde{T}} 0^b$ for precisely one $\tilde{T} \in \mathcal{T}$ – in this case $T_\pi := \tilde{T}$ works since the set of those strings in *V* having π as a substring is precisely $\{v_j \mid d_j \in \tilde{T}\}$. We hence turn to consider $t = 2k + j$ with $1 < j \leq p$. We can also assume that $|\alpha| \leq 10\,k - 2$. Let *z* be an indicator variable whose value is 1 if $b \neq 0$ and 0 otherwise. If $|1\alpha 1| + z < 5\,k - 3$ then consider $T_\pi := D_j$, which works since the set of those strings in *V* having π as a substring is precisely $\{v_i \mid i \geq j\} = \{v_i \mid d_i \in D_j\}$. (Actually, for the sake of precision, it can be observed that whenever $|1\alpha 1| \geq 10\,k - 5$, the string $001\alpha 100$ will be a substring of all v_i, or none at all). If $|1\alpha 1| + z \geq 5\,k - 3$ then $1\alpha 10$ is a substring of $f_{\tilde{T}} f_{\tilde{T}} 0$ for precisely one $\tilde{T} \in \mathcal{T}$ and $T_\pi := \tilde{T} \cap D_j$ works since the set of those strings in *V* having π as a substring is precisely $\{v_i \mid i \geq j, d_i \in \tilde{T}\} = \{v_i \mid d_i \in D_j \cap \tilde{T}\}$. □

Authors' contributions

All authors equally contributed to this paper. All authors read and approved the final manuscript.

Acknowledgements

We thank two anonymous referees for their careful reading of the paper. In particular, the first referee is acknowledged for pointing out to us the important reference [5], and the second referee for his detailed list of suggestions which greatly helped in improving the presentation. Part of this work was supported through MIUR grants P.R.I.N. and the F.I.R.B. project "Bioinformatica per la Genomica e la Proteomica".

References

1. Rash S, Gusfield D: **String Barcoding: Uncovering Optimal Virus Signatures.** In *Proceedings of the Annual International Conference on on Computational Molecular Biology (RECOMB)* ACM press; 2002:254-261.

2. DasGupta B, Konwar KM, Mandoiu II, Shvartsman A: **Highly scalable algorithms for robust string barcoding.** *Int J of Bioinf Res and Appls* 2005, **1(2):**145-161.
3. DasGupta B, Konwar KM, Mandoiu II, Shvartsman A: **DNA-BAR: distinguisher selection for DNA barcoding.** *Bioinf* 2005, **21(16):**3424-3426.
4. Borneman J, Chrobak M, Della Vedova G, Figueroa A, Jiang T: **Probe selection algorithms with applications in the analysis of microbial communities.** *Bioinf* 2001, **17(Suppl 1):**39-48.
5. Berman P, DasGupta B, Kao MY: **Tight approximability results for test set problems in bioinformatics.** *J of Comp and Sys Sc* 2004, **71(2):**145-162. [Also in *Proc. Workshop on Algorithm Theory*, Lec Notes in Comp Sc, Springer, 3111:39–50, 2004]
6. Garey MR, Johnson DS: *Computers and Intractability: A Guide to the Theory of NP-Completeness* San Francisco: W. H. Freeman and Co; 1979.
7. Moret BME, Shapiro HD: **On minimizing a set of tests.** *SIAM J on Sc and Stat Comp* 1985, **6:**983-1003.
8. Downey RG, Fellows MR: *Parametrized Complexity* Berlin: Springer-Verlag; 1998.
9. Karp RM: **Reducibility among combinatorial problems.** *Compl and Comp Computations* 1972.
10. De Bontridder KMJ, Halldórsson BV, Halldórsson MM, Hurkens CAJ, Lenstra JK, Ravi R, Stougie L: **Approximation algorithms for the test cover problem.** *Math Prog B* 2003, **1–3:**477-491.
11. Cormen TH, Leiserson CE, Rivest RL: *Introduction to Algorithms* Boston: MIT press; 2001.
12. Feige U: **A threshold of for approximating set cover.** *J ACM* 1998, **45:**634-652.