Received: 19 November 2020 | Revised: 7 January 2021 | Accepted: 29 January 2021

DOI: 10.1096/fba.2020-00118

METHODS

FASEB BioAdvances WILEY

The multiple alignments of very short sequences

Kristóf Takács[1] | Vince Grolmusz[1,2]

[1]PIT Bioinformatics Group, Eötvös University, Budapest, Hungary

[2]Uratim Ltd, Budapest, Hungary

Correspondence
Vince Grolmusz, PIT Bioinformatics Group, Eötvös University, Budapest, Hungary.
Email: grolmusz@pitgroup.org

Funding information
European Union and the State of Hungary

Abstract

The multiple sequence alignment (MSA) is an increasingly important task in bioinformatics as we have to deal with the constantly increasing gene- and protein sequence databases. MSA is applied in phylogenetic analysis, in discovering conservative protein domains, in the assignment of secondary and tertiary structural features in proteins, or in the metagenomic sample analysis and gene discovery. Usually, the focus is on the MSA of long sequences, since in the practice these tasks appear most frequently. However, the strict analysis of the optimal MSA of short sequences is an area of negligence, and findings there may contribute to better and faster algorithms for the multiple alignment of long sequences. In the present contribution, we are examining length-1 sequences using arbitrary metric and length-2 sequences using unit metric, and we show that the optimum of the MSA problem can be achieved by the trivial alignment in both cases.

1 | INTRODUCTION

Multiple sequence alignment (MSA) is one of the central problems in classical bioinformatics. While the exact optimal global and local alignments of two sequences can be computed in quadratic time with the Neddleman-Wunsch[1] and the Smith-Waterman[2,3] algorithms, respectively, the exact multiple alignment generally is proven to be an NP-hard problem,[4] and therefore, it is very unlikely to be computable in n^α-time algorithm, for any constant α, where n denotes the input size.

Heuristic MSA algorithms include the different versions of CLUSTAL,[5-11] MSACompro,[12] PRALINE,[13] TCS,[14] PASTASpark,[15] and numerous others.

Multiple sequence alignment algorithms have a range of applications in bioinformatics, for example, in HMM profile building in the famous HMMER suite of programs,[16-19] in identifying conservative protein- and genome-sequences, in phylogenetic tree building and analysis,[20-23] motif discovery and gene identification in metagenomic samples,[19] secondary protein structure prediction,[24,25] and solvent accessibility computation.[26]

In general, the MSA problem is computationally hard (i.e., NP-hard).[4] An interesting question is when does the problem become a hard instance when the parameters are modified?

It is known that for a constant number of sequences, the MSA problem is solvable in polynomial time by dynamic programing algorithms (Needleman-Wunsch and Smith-Waterman generalizations). Therefore, the MSA problem is "easy", that is, polynomial time computable, if the number of the sequences is small (i.e., it is a constant). To the best of our knowledge, no one examined the complexity of MSA when the number of the sequences is not small, but their length is.

One can assume that for length-1 (and perhaps even for length-2) sequences, it may not be that hard to find an optimal alignment. Furthermore, if an optimal alignment for short sequences can be determined in polynomial time, then it could also help to develop faster or more accurate heuristic algorithms. In this work, some new results regarding the alignment of short sequences are presented.

1.1 | Definitions and notations

Definition 1 *Let $\Sigma = \{a_1, \ldots, a_n\}$ be a finite alphabet; a string over Σ is called a sequence. The pair of sequences s_1', s_2' is an alignment of sequences s_1 and s_2 if for $i = 1, 2$: s_i' is obtained from s_i by inserting gaps (spaces, denoted by –) into or at either end of s_i and after that, s_1' and s_2' have the same length. It is assumed that "—" is not an element of alphabet Σ.*

The alignment of Definition 1 consists of two sequences of the same length. Consequently, every character of s_1' is uniquely corresponded to a character of s_2', simply by locating at the same position.

Let ℓ be the common length of s_1' and s_2'. The *cost* of this alignment is

$$\text{cost}\left(s_1', s_2'\right) = \sum_{i=1}^{\ell} d\left(s_1'(i), s_2'(i)\right), \tag{1}$$

where d is a *score scheme* over $\Sigma \cup \{-\}$, and $s_j'(i)$ is the ith character of s_j'. The score scheme is usually required to be a metric on the set $\Sigma \cup \{-\}$, that is, it needs to satisfy $\text{d}(u, v) = 0 \Leftrightarrow u = v$; $\text{d}(u, v) = \text{d}(v, u)$; and the triangle inequality: $\text{d}(u, w) \leq \text{d}(u, v) + \text{d}(v, w)$, $\forall u, v, w \in \Sigma \cup \{-\}$. A frequently used score scheme is the *unit metric*, where $\text{d}(u, v) = 0$ if $u = v$ and 1 otherwise. We call an alignment *optimal* for two sequences if its cost is minimal among every possible alignments.

The definition of aligning two sequences can easily be generalized for more strings: let $k \geq 2$ be a positive integer, and suppose that we want to align the sequences $s_1, s_2, \ldots, s_k$. Let us insert gaps into or at either end of strings $s_1, s_2, \ldots, s_k$, so that they have the same length ℓ, and in the proper order, write the k sequences $s_1', s_2', \ldots, s_k'$, each of length ℓ, under one another. This table can be considered a matrix of size $k \times \ell$, and it is called a *multiple alignment* of sequences $s_1, s_2, \ldots, s_k$. Different *scoring methods* can be applied for multiple alignments, perhaps the most often used one is the *sum of pairs* method, where the cost is the sum of the costs of the alignments of the $\binom{k}{2}$ pairs from the aligned sequences. More exactly, if $s_1, \ldots, s_k$ are sequences to be aligned, then their sum of pair cost[27] is

$$\sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \text{cost}\left(s_i', s_j'\right). \tag{2}$$

Examples. (i) Let $S := \{CCG, GCG, CGC\}$. The following set of sequences is a multiple alignment $\mathscr{A}$ of S:

C	C	G	–
G	C	G	–
–	C	G	C

Using the unit metric and computing the costs of the columns, $\text{cost}(\mathscr{A}) = 3 + 0 + 0 + 2 = 5$.

(ii) Let Σ now contain only two characters (C and G) with the following metric:

	C	G	–
C	0	2	1
G	2	0	1
–	1	1	0

Let $S := \{CG, GC, GG\}$. A multiple alignment $\mathscr{A}$ of S: $\mathscr{A} =$

–	C	G
G	C	–
G	–	G

Using the given metric, $\text{cost}(\mathscr{A})$ is equal to $2 + 2 + 2 = 6$.

2 | MULTIPLE SEQUENCE ALIGNMENT FOR LENGTH-1 SEQUENCES

In this section, we focus on aligning length-1 sequences (equivalently, characters of Σ). An important earlier result needs to be quoted here[28]:

Theorem 2 (Lemma 3) *Let U be a subset of a set S of sequences over Σ, such that U contains only identical sequences, and let $\mathscr{A}$ be an optimal alignment of S. Let $\mathscr{A}_U$ denote the restriction of $\mathscr{A}$ to the rows of U. Then*

$$\text{cost}\left(\mathscr{A}_U\right) = \sum_{u_i \in U} \sum_{\substack{u_j \in U \\ i<j}} d\left(u_i, u_j\right) = 0.$$

An important corollary of this theorem is the following one: it is enough to examine the sets of pairwise different sequences because in each optimal alignment, every instance of a given sequence is aligned identically.

The next definition will be used frequently throughout this work:

Definition 3 *Let S be a set of sequences that have the same length. $\mathscr{A}$ is called the trivial alignment of S if $\mathscr{A}$ is constructed by writing every sequence under each another, without using any gaps.*

2.1 | Multiple sequence alignment for length-1 sequences using unit metric

The main result of this subsection is the next theorem:

Theorem 4 *Using unit metric, there cannot be a multiple sequence alignment for length-1 sequences that has cost less than the cost of their trivial alignment. Additionally, if we align k pairwise different length-1 sequences, then the cost of an optimal alignment is* $\binom{k}{2}$.

Proof By Theorem 2, we may assume that the characters to be aligned are pairwise different. It is easy to see that the trivial alignment of k different characters has a cost of $\genfrac{}{}{0pt}{}{k}{2}$: there are $\binom{k}{2}$ pairs among these characters and in every pair, there are two different sequences, so the cost of an aligned pair is always 1.

Let us suppose that this alignment is not optimal, then the length of every aligned sequence must be at least 2 in an optimal alignment. If this common length of aligned sequences is $\ell \geq 2$, then the general structure of the $n \times \ell$ matrix of this multiple alignment is as follows: $\forall i: 1 \leq i \leq \ell$, there are k_i characters in the ith column, where $\sum_{i=1}^{\ell} k_i = k$, and they are placed so that in each row, there is only one character and $\ell - 1$ gaps (see Table 1).

Obviously, the cost of the first column is

$$\binom{k_1}{2} + \left(k - k_1\right) k_1$$

since there are k_1 different characters with cost of $\binom{k_1}{2}$, and besides that, all of the $\left(k - k_1\right)$ gaps increase the cost by one with every alphabetical character. A similar statement is true for every column, so the cost of this alignment is:

$$\sum_{i=1}^{\ell} \binom{k_i}{2} + (k-k_i)k_i = k\sum_{i=1}^{\ell} k_i - \frac{1}{2}\left(\sum_{i=1}^{\ell} k_i + \sum_{i=1}^{\ell} k_i^2\right) = k^2 - \frac{1}{2}\left(k + \sum_{i=1}^{\ell} k_i^2\right)$$

Consequently, the cost above is minimized, when $\sum_{i=1}^{\ell} k_i^2$ is maximized. Since

$$k^2 = \left(\sum_{i=1}^{\ell} k_i\right)^2 = \sum_{i=1}^{\ell} k_i^2 + 2\sum_{i=1}^{\ell} \sum_{\substack{j=1 \\ i<j}}^{\ell} k_i k_j,$$

holds, it is clear that $\sum_{i=1}^{\ell} k_i^2 = k^2$, and the cost of this alignment cannot be less than $\left(k^2 - k\right)/2 = \binom{k}{2}$, that is, the cost of the trivial alignment.

Note: From the proof, it is also clear (by minimizing $\sum_{i=1}^{\ell} k_i^2$) that a multiple alignment for k different length-1 sequences cannot have a higher cost than $k^2 - k/2 - k^2/(2\ell)$, if the length of aligned sequences is ℓ. Since $\ell \leq k$, the cost can be at most $k^2 - k$ and this limit can be reached if there is only one character in every column and in every row, then the cost is $k(k-1) = k^2 - k$.

3 | MULTIPLE SEQUENCE ALIGNMENT FOR LENGTH-1 SEQUENCES USING ARBITRARY METRIC

In this subsection, it will be shown that for length-1 sequences, we can use any metric as a score scheme, and the MSA problem still remains as easy as in the case of the unit metric.

Theorem 5 *Using arbitrary metric, the minimum cost of the multiple sequence alignment for length-1 sequences is attained by the trivial alignment, and if k different sequences are aligned,* then the optimal *cost is equal to.*

$$C = \sum_{i=1}^{k} \sum_{\substack{j=2 \\ i<j}}^{k} d(a_i, a_j).$$

TABLE 1 A multiple alignment for length-1 sequences on ℓ columns

a_1	–	…	–
…	…	…	…
a_{k_1}	–	–	–
–	a_{k_1+1}	…	–
…	…	…	…
–	a_{k_2}	…	–
…	…	…	…
–	–	…	$a_{k_{\ell-1}+1}$
…	…	…	…
–	–	…	a_{k_ℓ}

Proof Because of Theorem 2, it can be assumed again that every sequence has exactly one instance in the set S of sequences to be aligned. If we consider the trivial alignment of the S, it is easy to see that its cost is equal to C. Induction for the number of the columns in a MSA will be used to show that no alignment can have lower cost than C.

Let be assumed that the trivial alignment is not optimal, and let $\mathscr{A}$ denote an optimal alignment. If $\mathscr{A}$ is not the trivial alignment, then $\mathscr{A}$ has ℓ columns where $\ell \geq 2$. It can be shown that $\mathscr{A}$ cannot have exactly two columns, because in this case, the trivial alignment would have a lower cost than $\mathscr{A}$ has.

Let us assume to the contrary that $\mathscr{A}$ has exactly two columns; so there are k_1 sequences in the first column and k_2 in the second column, where $k_1 + k_2 = k$ and there is exactly one character in each row (since our sequences to be aligned have length equal to 1, see Table 2).

We assume, without loss of generality, that the sequences in the first column are $a_1, a_2, \ldots, a_{k_1}$ and every other sequences are placed in the second column. If the cost of the first column of $\mathscr{A}$ is denoted by cost (1), then

$$\mathrm{cost}(1) = \sum_{i=1}^{k_1} \sum_{\substack{j=2 \\ i<j}}^{k_1} d(a_i, a_j) + k_2 \sum_{i=1}^{k_1} d(a_i, -).$$

Similarly, the cost of the second column is

$$\mathrm{cost}(2) = \sum_{i=k_1+1}^{k} \sum_{\substack{j=k_1+2 \\ i<j}}^{k} d(a_i, a_j) + k_1 \sum_{j=k_1+1}^{k} d(a_j, -).$$

and $\mathrm{cost}(\mathscr{A}) = \mathrm{cost}(1) + \mathrm{cost}(2)$.

A lower bound for cost $(\mathscr{A})$ can be determined by pairing the $\mathrm{d}\left(a_i, -\right)$ summands in cost (1) to the summands of same form in cost (2) and using the triangle inequality. For example, for a fix i $(1 \leq i \leq k_1)$ and $\forall j\colon k_1 + 1 \leq j \leq k$, it is true that $\mathrm{d}\left(a_i, -\right) + \mathrm{d}\left(a_j, -\right) \geq \mathrm{d}\left(a_i, a_j\right)$, so

$$k_2 d(a_i, -) + \sum_{j=k_1+1}^{k} d(a_j, -) \geq \sum_{j=k_1+1}^{k} d(a_i, a_j).$$

It is useful to notice that the summands on the right side of this inequality are exactly those ones that are not included in cost (1) when we consider summands of the form of $d\left(a_i, a_j\right)$ for this fix i.

By considering this inequality for every $1 \leq i \leq k_1$, the following lower bound can be given:

$$k_2 \sum_{i=1}^{k_1} d(a_i, -) + k_1 \sum_{j=k_1+1}^{k} d(a_j, -) \geq \sum_{i=1}^{k_1} \sum_{j=k_1+1}^{k} d(a_i, a_j),$$

This implies that

$$\mathrm{cos}t\,(\mathscr{A}) \geq \sum_{i=1}^{k_1} \sum_{\substack{j=2 \\ i<j}}^{k_1} d\left(a_i, a_j\right) + \sum_{i=k_1+1}^{k} \sum_{\substack{j=k_1+2 \\ i<j}}^{k} d\left(a_i, a_j\right) + \sum_{i=1}^{k_1} \sum_{j=k_1+1}^{k} d\left(a_i, a_j\right) = \sum_{i=1}^{k} \sum_{\substack{j=2 \\ i<j}}^{k} d\left(a_i, a_j\right) = C.$$

It is assumed that the trivial alignment with cost C is not optimal; therefore, $\mathscr{A}$ cannot be an optimal alignment of S. By this contradiction, it is proved that an optimal alignment of S cannot have exactly 2 columns.

Using induction, we assume that it is shown $\forall i\colon 2 \leq i < \ell$ that an optimal alignment cannot have exactly i columns, and let $\mathscr{A}$ be an optimal alignment with ℓ columns. Considering the cost of the first two columns of $\mathscr{A}$, there are k_1 sequences in the first column and k_2 sequences in the second one. It is enough to prove that by merging these two columns, the cost of the new alignment is lower than the cost of $\mathscr{A}$. The cost of these columns (see Table 3) in $\mathscr{A}$ is equal to

$$\sum_{i=1}^{k_1} \sum_{\substack{j=2 \\ i<j}}^{k_1} d\left(a_i, a_j\right) + \left(k - k_1\right) \sum_{i=1}^{k_1} d\left(a_i, -\right) + \sum_{i=k_1+1}^{k_2} \sum_{\substack{j=k_1+1 \\ i<j}}^{k_2} d\left(a_i, a_j\right) + \left(k - k_2\right) \sum_{i=k_1+1}^{k_2} d\left(a_i, -\right).$$

TABLE 2 A multiple alignment for k length-1 sequences in two columns

a_1	–
a_2	–
—	…
a_{k_1}	–
–	a_{k_1+1}
–	a_{k_1+2}
…	…
–	a_k

TABLE 3 The first two columns of $\mathscr{A}$

a_1	–
a_2	–
…	…
a_{k_1}	–
–	a_{k_1+1}
–	a_{k_1+2}
…	…
–	$a_{k_1+k_2}$
–	–
…	…
–	–

Let us focus on the first $k' = k_1 + k_2$ characters of these columns. It is an alignment of $\{a_1, a_2, \ldots, a_{k'}\}$ on two columns and it was shown that if these sequences are aligned trivially instead of using two columns, then the cost of the alignment cannot be higher. It means the following:

$$\sum_{i=1}^{k_1} \sum_{\substack{j=2 \\ i<j}}^{k_1} d(a_i, a_j) + k_2 \sum_{i=1}^{k_1} d(a_i, -) + \sum_{i=k_1+1}^{k'} \sum_{\substack{j=k_1+1 \\ i<j}}^{k'} d(a_i, a_j)$$

$$+k_1 \sum_{i=k_1+1}^{k'} d(a_i, -) + (k-k) \sum_{i=1}^{k_1} d(a_i, -)$$

$$+(k-k') \sum_{i=k_1+1}^{k'} d(a_i, -) \geq \sum_{i=1}^{k'} \sum_{\substack{j=2 \\ i<j}}^{k'} d(a_i, a_j) + (k-k') \sum_{i=1}^{k'} d(a_i, -)$$

On the left side of this inequality, there is the cost of the first two columns of $\mathscr{A}$, while on the right side, there is the cost of the column that is constructed by merging the first two columns of $\mathscr{A}$. Therefore, a lower bound for *cost* ($\mathscr{A}$) is given by an alignment that has $l-1$ columns, implying that $\mathscr{A}$ is not optimal W.

4 | MULTIPLE SEQUENCE ALIGNMENT FOR LENGTH-2 SEQUENCES

In this section, it will be shown that using the unit metric, a set of length-2 sequences cannot be aligned with less cost than their trivial alignment; however, this statement does not hold for using arbitrary metric.

Theorem 6 *Using the unit metric, no multiple sequence alignment for length-2 sequences has less cost than their trivial alignment. If we align k different sequences* $\left(s_1 = a_{i_1}a_{i_{k+1}}, s_2 = a_{i_2}a_{i_{k+2}}, \ldots, s_k = a_{i_k}a_{i_{2k}}\right)$, *then the cost of the optimal alignment is.*

$$\sum_{j=1}^{k} \sum_{\substack{\ell=2 \\ j<\ell}}^{k} d\left(a_{i_j}, a_{i_\ell}\right) + \sum_{j=k+1}^{2k} \sum_{\substack{\ell=k+2 \\ j<\ell}}^{2k} d\left(a_{i_j}, a_{i_\ell}\right).$$

Proof Let S denote the set of sequences that need to be aligned. It is clear that the trivial alignment of S has the cost written above, so this lower bound is accessible. In other words, it is enough to prove that for any S, a non-trivial alignment cannot have less cost than the trivial one.

Let $\mathscr{A}$ be an alignment of S on ℓ columns where $\ell \geq 3$. Let the rows of $\mathscr{A}$ be permuted, so that those aligned sequences, where the indices of the two non-gap characters are the same, are placed under each other, forming a block of sequences. This operation does not change the cost of $\mathscr{A}$. In every row of $\mathscr{A}$, there are exactly two characters and $\ell - 2$ gaps, so there can be $\binom{\ell}{2}$ types of aligned sequences in $\mathscr{A}$, considering only the positions of the non-gap characters in a row. This implies that there will be $\binom{\ell}{2}$ (not necessarily non-empty) blocks after permuting the rows of $\mathscr{A}$ (e.g., if $\ell = 4$, then there are $\binom{4}{2}=6$ blocks after the permutation of the rows, see Table 4).

After making this block setting, it is clear that there are six types of aligned character pairs in $\mathscr{A}$:

1. first characters of some sequences aligned with other sequences' first characters;
2. first characters of some sequences aligned with other sequences' second characters;
3. first characters of some sequences aligned with gaps;
4. second characters of some sequences aligned with other sequences' second characters;
5. second characters of some sequences aligned with gaps;
6. gaps aligned with gaps.

TABLE 4 The structure of $\mathscr{A}$ after permuting its rows and making its block setting with $\ell = 4$. Number 1 denotes the first characters, and number 2 the second letters. During the proof, an upper bound is given for the cost of aligning letters with the same order that are not aligned in $\mathscr{A}$ by using character-gap alignment costs that are included in cost ($\mathscr{A}$)

1	2	–	–
…	…	…	…
1	2	–	–
1	–	2	–
…	…	…	…
1	–	2	–
1	–	–	2
…	…	…	…
1	–	–	2
–	1	2	–
…	…	…	…
–	1	2	–
–	1	–	2
…	…	…	…
–	1	–	2
–	–	1	2
…	…	…	…
–	–	1	2

TABLE 5 The block setting of $\mathscr{A}$ if $\ell = 4$, denoting only that an element is the first/second character of its aligned sequence or a gap. For example, the first element of the first row in the block setting and the second element of the fourth row (which are denoting the first characters of some sequences) are not aligned in $\mathscr{A}$, so the cost of their alignment with each other, which is a part of cost ($\mathscr{T}$) but not a part of cost ($\mathscr{A}$), must be estimated from above with a part of cost ($\mathscr{A}$) . Namely, with the cost of aligning the block setting's first element of the first row with the gaps in the first element of the fourth row

1	2	–	–
1	–	2	–
1	–	–	2
–	1	2	–
–	1	–	2
–	–	1	2

In the trivial alignment $\mathscr{T}$, there are only pairs of types (i) and (iv); moreover, *every* sequence's first character is aligned with each another in $\mathscr{T}$ (and it holds similarly for *every* second character of the sequences of S). Nevertheless, in a nontrivial alignment $\mathscr{A}$, there are aligned sequences whose first or second characters are not aligned with each other in $\mathscr{A}$. This implies that it is enough to give an upper bound for the cost of these characters in $\mathscr{T}$ that are aligned with each other in $\mathscr{T}$ but are not aligned with each other in $\mathscr{A}$, using parts of cost($\mathscr{A}$) for this bound. (Because every part of cost ($\mathscr{A}$) is non-negative, if a bijection can be given between the letter-letter alignments in $\mathscr{T}$ that are not aligned in $\mathscr{A}$ and some other alignments of characters of $\mathscr{A}$ (not excluded character-gap alignments), so that the latter alignments have always at least as much cost as the former ones, then it means that cost ($\mathscr{A}$) $\geq$ cost($\mathscr{T}$).)

If d denotes the unit metric, then the following inequality holds for every pair of sets P, R on arbitrary alphabet (where P and R can contain a letter more than once):

$$\sum_{a_{i_j}\in P}\sum_{a_{i_l}\in R} d\left(a_{i_j}, a_{i_l}\right) \leq |P| \sum_{a_{i_l}\in R} d\left(a_{i_l}, -\right) = |P|\,|R|.$$

Using this inequality, a bijection mentioned above can be given: first, let be considered two sequences whose first characters (a_i and a_j) are not aligned in $\mathscr{A}$ (it can be assumed that a_j has bigger column index). This implies that the element that is in the intersection of the row of a_j and the column of a_i must be a gap. $d\left(a_i, a_j\right) \leq d\left(a_i, -\right)$, so the cost of the alignment of a_i and a_j in $\mathscr{T}$ can be estimated by the cost of the alignment of two characters in $\mathscr{A}$.

Similarly, if two sequences are considered whose second characters (a_i and a_j) are not aligned in $\mathscr{A}$, then (assuming that a_j has bigger column index) the element in the intersection of the row of a_i and the column of a_j must be a gap. The same estimation can be given like before, meaning that the cost of the alignment of a_i and a_j in $\mathscr{T}$ is less or equal to the cost of a character-gap alignment in $\mathscr{A}$.

Considering the block setting (Table 4) of $\mathscr{A}$, let B_i and B_j be the two blocks whose sequences' first characters are not aligned in $\mathscr{A}$. Assuming that the first characters of sequences in B_j have bigger column index, there must be $\left|B_j\right|$ gaps in the intersection of the column of the first characters of sequences in B_i and the rows of B_j. If we denote the first letters of the sequences of B_i $\left(B_j\right)$ by a_{b_i} $\left(a_{b_j}\right)$, then (because of the statements of the latter two paragraphs) the following holds:

$$\sum_{b_i\in B_i}\sum_{b_j\in B_j} d\left(a_{b_i}, a_{b_j}\right) \leq \left|B_j\right| \sum_{b_i\in B_i} d\left(a_{b_i}, -\right) = \left|B_i\right| \left|B_j\right|$$

Besides that, a similar result can be established if we consider two blocks whose sequences' second characters are not aligned, using the gaps of the block that has the column with smaller column index (see Table 5). By these estimations, it is clear that this assignment between the character–character alignments in $\mathscr{T}$, which are not present in $\mathscr{A}$, and character-gap alignments in $\mathscr{A}$ lead to a result that the latter costs in $\mathscr{A}$ cannot be less than the corresponding costs in $\mathscr{T}$. We also need to show that this assignment is a bijection, that is, there are no character-gap alignments that are used more than one time.

A set of gaps in the block setting are considered in an estimation if and only if some characters in the block that are containing these gaps and some characters from another block that are aligned in the same column must be aligned in $\mathscr{T}$ but they are not aligned in $\mathscr{A}$. This implies that these gaps are not used in estimations like above more times than the alignment of this gap set with the rest of the given column. Therefore, the former assignment is a bijection, implying that cost ($\mathscr{A}$) $\geq$ cost ($\mathscr{T}$).W.

Remark In the proof, only the following property of the unit metric has been used: $\forall a_i, a_j \in \Sigma: d\left(a_i, a_j\right) \leq d\left(a_i, -\right)$. It follows that Theorem 6 remains valid for any metric, satisfying this property.

As the next example shows, the trivial alignment will not always be optimal for length-2 sequences if an arbitrary metric is used. Let Σ contain two characters (C and G) with the same metric on Σ as in the Example (*ii*) at the end of the Introduction. Let S be also the same as in Example (*ii*): $S = \{CG, GC, GG\}$. The trivial alignment of S has a cost of 8, but as Table 6 shows, there is an alignment of S that has cost only of 6.

Remark In the previous section, it was shown that we can easily determine the minimum cost of a set to be aligned if it includes only length-1 sequences; moreover, we also can construct an optimal alignment in the

TABLE 6 The trivial and an optimal alignment of S

C	G	–	C	G
G	C	G	C	–
G	G	G	–	G

TABLE 7 The trivial and an optimal alignment of S

C	C	G	C	C	G	–
G	C	G	G	C	G	–
C	G	C	–	C	G	C

most trivial way using any metric. We have also seen that for length-2 sequences, the trivial alignment is optimal if the unit metric is used but it is not optimal for arbitrary metric. Besides that, it is also known that the trivial alignment is not always optimal for length-3 sequences even using unit metric.

As in Example (i) at the end of the Introduction, let S be as follows:

$$S = \{CCG, GCG, CGC\}.$$

Using the unit metric, the cost of the trivial alignment is 6, but it is not optimal: as we have seen, there is a non-trivial alignment $\mathcal{A}$ of S so that cost ($\mathcal{A}$) is only 5 (see Table 7).

5 | CONCLUSIONS

In this work, it was shown that the MSA problem is “easy” for length-1 sequences and also for length-2 sequences in special cases. While the MSA problem is well-examined for a small number of long sequences, it is a pioneering work covering the specialties of a large number of very short sequences.

Since we know that the general problem is **NP**-hard,[4] it is still an interesting question that for how long sequences the MSA problem starts to become to be difficult? It is another open problem that in the case of length-2 sequences, how can those metrics be characterized for which trivial alignment is always optimal for arbitrary alphabet?

ACKNOWLEDGMENT

KT was partially funded by the VEKOP-2.3.2-16-2017-00014 program, supported by the European Union and the State of Hungary, co-financed by the European Regional Development Fund, and by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002). VG was partially funded by the NKFI-127909 and the 2017-1.3.1-VKE-2017-00013 and the VEKOP-2.3.2-16-2017-00014 grants of the National Research, Development and Innovation Office of Hungary.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

KT proved the theorems, wrote the first version of this manuscript, and prepared figures. VG initiated the study, finalized the manuscript, and secured funding.

ORCID

Vince Grolmusz https://orcid.org/0000-0001-9456-8876

REFERENCES

1. Needleman SB, Wunsch CD. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol*. 1970;480(3):443-453.
2. Smith TF, Waterman MS. Identification of common molecular subsequences. *J Mol Biol*. 1981;1470(1):195-197.
3. Ivan G, Banky D, Grolmusz V. Fast and exact sequence alignment with the Smith-Waterman algorithm: the SwissAlign webserver. *Gene Rep*. 2016;4:26-28.
4. Elias I. Settling the intractability of multiple alignment. *J Computational Biol*. 2006;13:1323-1339.
5. Higgins DG, Sharp PM. Clustal: a package for performing multiple sequence alignment on a microcomputer. *Gene*. 1988;73:237-244.
6. Higgins DG, Bleasby AJ, Fuchs R. Clustal v: improved software for multiple sequence alignment. *Computer Appl Biosci*. 1992;8:189-191.
7. Higgins DG. Clustal v: multiple alignment of dna and protein sequences. *Methods Molecular Biol (Clifton, N.J.)*. 1994;25:307-318.
8. Higgins DG, Thompson JD, Gibson TJ. Using clustal for multiple sequence alignments. *Methods Enzymol*. 1996;266:383-402.
9. Thompson JD, Gibson TJ, Plewniak F, Jeanmougin F, Higgins DG. The CLUSTAL_X windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools. *Nucleic Acids Res*. 1997;25:4876-4882.
10. Sievers F, Higgins DG. Clustal omega, accurate alignment of very large numbers of sequences. *Methods Molecular Biol (Clifton. N.J.)*. 2014;1079:105-116.
11. Sievers F, Higgins DG. Clustal omega for making accurate alignments of many protein sequences. *Protein Sci*, 2018;27:135-145.
12. Deng X, Cheng J. Msacompro: improving multiple protein sequence alignment by predicted structural features. *Methods Molecular Biol (Clifton, N.J.)*. 2014;1079:273-283.
13. Bawono P, Heringa J. Praline: a versatile multiple sequence alignment toolkit. *Methods Molecular Biol (Clifton. N.J.)*. 2014;1079:245-262.
14. Chang J-M, Di Tommaso P, Notredame C. Tcs: a new multiple sequence alignment reliability measure to estimate alignment accuracy and improve phylogenetic tree reconstruction. *Mol Biol Evol*. 2014;31:1625-1637.
15. Abuin JM, Pena TF, Pichel JC. PASTASpark: multiple sequence alignment meets big data. *Bioinformatics (Oxford, England)*. 2017;33:2948-2950.
16. Eddy SR. A new generation of homology search tools based on probabilistic inference. *Genome Inform*, 2009;230(1):205-211.
17. Eddy SR. Accelerated profile HMM searches. *PLoS Comput Biol*, 2011;70(10):e1002195.

18. Szalkai B, Scheer I, Nagy K, Vertessy BG, Grolmusz V. The metagenomic telescope. *PLoS One*. 2014;9:e101605.
19. Szalkai B, Grolmusz V. MetaHMM: A webserver for identifying novel genes with specified functions in metagenomic samples. *Genomics*, 2019;111(4):883–885. ISSN 1089–8646. https://doi.org/10.1016/j.ygeno.2018.05.016.
20. Feng DF, Doolittle RF. Progressive alignment and phylogenetic tree construction of protein sequences. *Methods Enzymol*. 1990;183:375-387.
21. Reizer A, Reizer J. Progressive multiple alignment of protein sequences and the construction of phylogenetic trees. *Methods Molecular Biol (Clifton. N.J.)*. 1994;25:319-325.
22. Metcalf V, Brennan S, George P. Using serum albumin to infer vertebrate phylogenies. *Appl Bioinform*. 2003;2:S97-107.
23. Hagopian R, Davidson JR, Datta RS, Samad B, Jarvis GR, Sjolander K. SATCHMO-JS: a webserver for simultaneous protein multiple sequence alignment and phylogenetic tree construction. *Nucleic Acids Res*. 2010;38:W29-W34.
24. Cuff JA, Barton GJ. Application of multiple sequence alignment profiles to improve protein secondary structure prediction. *Proteins*. 2000;40:502-511.
25. Al-Lazikani B, Sheinerman FB, Honig B. Combining multiple structure and sequence alignments to improve sequence detection and alignment: application to the sh2 domains of janus kinases. *Proc Natl Acad Sci USA*. 2001;98:14796-14801.
26. Garg A, Kaur H, Raghava GPS. Real value prediction of solvent accessibility in proteins using multiple sequence alignment and secondary structure. *Proteins*. 2005;61:318-324.
27. Wang L, Jiang T. On the complexity of multiple sequence alignment. *J Comput Biol*. 1994;10(4):337-348.
28. Bonizzoni P, Vedova GD. The complexity of multiple sequence alignment with SP-score that is a metric. *Theoret Comput Sci*. 2001;2590(1–2):63-79.

How to cite this article: Takács K, Grolmusz V. The multiple alignments of very short sequences. *FASEB BioAdvances*. 2021;3:523–530. https://doi.org/10.1096/fba.2020-00118