Available online at www.sciencedirect.com
ScienceDirect
Discrete Applied Mathematics 155 (2007) 2471–2486
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Constrained sequence alignment: A general model and the hardness results

Yun-Sheng Chung[a], Chin Lung Lu[b], Chuan Yi Tang[a,*]

[a] *Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan 300, ROC*
[b] *Department of Biological Science and Technology, National Chiao Tung University, Hsinchu, Taiwan 300, ROC*

Received 12 June 2006; received in revised form 28 April 2007; accepted 16 June 2007
Available online 13 August 2007

Abstract

Imposing constraints is a way to incorporate information into the sequence alignment procedure. In this paper, a general model for constrained alignment is proposed so that analyses admitted are more flexible and that different pattern definitions can be treated in a simple unified way. We give a polynomial time algorithm for pairwise constrained alignment for the generalized formulation, and prove the inapproximability of the problem when the number of sequences can be arbitrary. In addition, previous works deal only with the case that the patterns in the constraint have to occur in the output alignment in the same order as that specified by the input. It is of both theoretical and practical interest to investigate the case when the order is no longer limited. We show that the problem is not approximable even when the number of sequences is two. We also give the NPO-completeness results for the problems with bounds imposed on the objective function value.

Keywords: Constrained sequence alignment; Nonapproximability; NPO-completeness

1. Introduction

Sequence alignment is one of the most fundamental problems in computational biology. Due to its importance, it has been extensively studied in the past (see, e.g., [13]). As biological knowledge and predictions grow, it is often desirable if one can incorporate more information into the alignment procedure in hope that the alignment result can be more biologically meaningful and reasonable. In particular, when the input sequences share some properties, one then expects that the resulting alignment should not violate these properties. Such kind of preservation is in its nature a satisfaction of properly defined constraints. Due to this need, Tang and coauthors [20] defined the constrained sequence alignment problem (CSA for short). They considered the alignment of RNase sequences which share a conserved sequence of residues H, K, H. It is expected that the alignment result should have these conserved residues aligned together so that the semantic meaning is not violated. A constraint is then defined in [20] as a sequence of characters, say $c_1c_2\cdots c_m$. The problem requires that, in the output alignment, there exist m columns such that the ith of these columns contains solely of c_i. The goal is to find the best such alignment.

* Corresponding author. Tel.: +886 3 5715131x31077; fax: +886 3 5723694.
E-mail addresses: yschung@algorithm.cs.nthu.edu.tw (Y.-S. Chung), cllu@mail.nctu.edu.tw (C.L. Lu), cytang@cs.nthu.edu.tw (C.Y. Tang).

0166-218X/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2007.06.016

Later, Chin and coauthors [6] proposed an efficient algorithm for the pairwise version, along with a 2-approximation algorithm for the multiple alignment version with scoring function satisfying triangle inequality. In [21], Tsai and coauthors generalized the definition of a constraint from a sequence of characters to a sequence of strings (patterns), and the occurrences of each pattern in the sequences need not be identical to the pattern specified in the input. Instead, the Hamming distances between each pattern and its occurrences need only to be within a threshold. This formulation enables the user to align sequences so that known motifs specified in the constraint are required to be aligned together. Lu and Huang [16] then reduced the memory requirement of the algorithm in [21], which significantly improves the applicability of the tool. Arslan introduced the regular expression constrained sequence alignment (RECSA) problem [2], in which a constraint is a regular expression, and a feasible solution is an alignment containing a run of contiguous columns corresponding to two substrings, one for each input sequence, such that both substrings match the regular expression constraint. Chung and coauthors [7] followed the formulation of Arslan and gave more time and space efficient algorithms for RECSA.

The formulation in this paper is generalized from the aforementioned works (see Section 2 for an example). To facilitate further discussion we say that in all these works a constraint is a sequence of patterns; a pattern is a simple character in [20,6], a substring allowing mismatches in [21,16] and a regular expression in [2,7]. Each pattern is described by its sequence content in these works. In this paper, we characterize each pattern directly by the set of the positions of its candidate occurrences in the input sequences. The input sequences are in effect annotated with the substrings representing the candidate occurrences of each pattern. This generalizes the previous formulations since, given patterns defined in [20,6,21,16,2,7], the occurrences can be easily annotated in the sequences by well-established pattern matchers (e.g., the UNIX utility `grep` for regular expression patterns), thereby transforming the original patterns into an instance of our formulation. The formulation here also enables one to adopt mixed pattern definitions simultaneously in the analysis, for example some patterns can be specified by regular expressions and the others by strings with mismatch tolerance. Another immediate and desirable consequence is that, it is now easy to use string edit distance instead of Hamming distance as in [21,16] to define pattern occurrences. We note that annotating substrings on input sequences directly to represent patterns rather than specifying the forms of the patterns had appeared in the context of annotated sequence comparison to support patterns of various types (see [9, Sections 3.2, 3.3, Chapter 6]).

Myers and coauthors [18] had adopted a different definition of constraints, and applications admitted are different from those in [20,6,21,2,7]. In [18], a constraint is an order relation of two characters on two different sequences in the input set of sequences. The order has to be satisfied in the resulting multiple alignment. To keep two specific characters lined up in a column in the multiple alignment, one can easily use two constraints simultaneously; e.g., $i_s \preccurlyeq j_\ell$ and $i_s \succcurlyeq j_\ell$, which means that the ith character in the sth sequence should appear no later than the jth character in the ℓth sequence in the alignment, and vice versa. This kind of constraint is particularly useful when one knows exactly which characters are to be aligned together. However, one may not have this strong information. If one has the knowledge only about the form or candidate occurrences of the patterns shared by all sequences rather than the exact positions of them, one needs the capability of choosing from the candidate occurrence set of a pattern on each of the sequences which candidate is to be aligned with those on the other sequences such that the overall score can be optimized. This is not supported in the formulation in [18].

Although there have been web-based tools "`MuSiC`" [21] and "`MuSiC-ME`" [16] for CSA which are shown to be useful supplements to classic sequence alignment tools, the actual hardness of CSA remains not investigated, except for those inherited directly from the classic MSA. As mentioned before, a 2-approximation algorithm for the multiple CSA is proposed in [6], but its running time is not bounded by a polynomial, and hence the result does not imply the approximability of CSA. Indeed, the past development of CSA had been founded on applications and a lack of theoretical discussion is observable.

For the problem formulation generalized in this paper, we propose a polynomial time algorithm in the pairwise case. It is straightforward to apply a progressive strategy as in [21,16] to extend the pairwise algorithm to the case of multiple sequences, but with no performance guarantee. We shall prove that the problem is not approximable within any function computable in polynomial time if the number of sequences can be arbitrary. On the other hand, previous works require that the order of the appearances of the patterns in the output alignment must be the same as that in the input constraint. Hence the constraint is a *sequence* of patterns. It is theoretically interesting to investigate the problem without this limitation. That is, how is it if the constraint is in the sense a *set* of patterns. It is noted that from the biological perspective, the order of regulatory elements are sometimes important for determining gene expression [15,10]. However, the necessity for functional motifs to keep in some fixed order is not always clear. In light of this, the

order limitation should not always be imposed on the problem. As we shall see, the problem turns out to be hard even when the number of input sequences is only two; in this paper it is proved to be not approximable within any function computable in polynomial time.

The remainder of this paper is organized as follows. In Section 2, we give some definitions and problem formulations. In Section 3, an algorithm for the pairwise CSA in generalized formulation is given, along with the proofs of inapproximability when the number of input sequences can be arbitrary and of NPO-completeness for the bounded version. Section 4 establishes symmetric hardness results for the problem without the order limitation in the pairwise case. Finally, in Section 5, we give conclusions and discussions.

2. Preliminaries

In this section, we formulate the notion of constraints and constrained alignments; we begin with some standard definitions. Let Σ be the alphabet where the characters in sequences are drawn from. For a sequence S, let $S[i]$ be the ith character on S, and denote the string $S[i]S[i+1]..S[j]$ as $S[i..j]$. The length of S is denoted as $|S|$. Throughout this paper, we denote the number of input sequences as σ. An (unconstrained) alignment of strings $S_1, \ldots, S_\sigma$ is a set of strings $\bar{S}_1, \ldots, \bar{S}_\sigma$, where $\bar{S}_i$ is obtained by inserting some space characters "-" into S_i, such that for all j, $\bar{S}_i[j] \neq$ - for some i, and $|\bar{S}_i| = |\bar{S}_j|$ for all i, j.

An ordered constraint $\mathscr{P}$ is a sequence of patterns, where each pattern is characterized by σ sets of index pairs to indicate the start and end indices of the candidate occurrences of the pattern in the sequences. The set of index pairs of candidate occurrences for pattern k on sequence i will be denoted as $occur_i(k)$, with each element $[x_1, x_2] \in occur_i(k)$ indicating that $S_i[x_1..x_2]$ is a candidate occurrence of pattern k. Constraint $\mathscr{P}$ is formally specified as the list $\langle occur_i(k) : 1 \leqslant i \leqslant \sigma, 1 \leqslant k \leqslant m \rangle$, where m is the number of patterns. A constrained alignment $\mathscr{A}$ for sequences $S_1, \ldots, S_\sigma$ satisfying $\mathscr{P}$ is a set of equal-length sequences $\bar{S}_1, \ldots, \bar{S}_\sigma$, with some additional properties:

(1) There exist exactly m positions $j_1 < \cdots < j_m$ in $\mathscr{A}$ such that for each i, $1 \leqslant i \leqslant \sigma$, each $\bar{S}_i[j_k]$ corresponds to a candidate occurrence of pattern k on S_i. That is, there is an index pair $[x_{i,k}, x'_{i,k}] \in occur_i(k)$ such that $\bar{S}_i[j_k]$ corresponds to $S_i[x_{i,k}..x'_{i,k}]$, which we denote as $\bar{S}_i[j_k] = [x_{i,k}, x'_{i,k}]$ directly. This condition can then be restated as $\bar{S}_i[j_k] \in occur_i(k)$ for all pattern k and sequence S_i. Note that j_k corresponds to pattern k, hence $j_k < j_{k+1}$ implies that the order of the patterns appeared in the constrained alignment is the same as specified in the constraint.
(2) It is required that $x'_{i,k} < x_{i,k+1}$ for all $1 \leqslant i \leqslant \sigma$ and $1 \leqslant k < m$. This, together with the next condition, ensures that the constrained alignment preserves the order of the characters in the original sequences, as an unconstrained alignment does.
(3) Regions in $\mathscr{A}$ excluding those positions j_k are standard unconstrained alignments. Specifically, $\{\bar{S}_i[1..j_1 - 1] : 1 \leqslant i \leqslant \sigma\}$ forms an unconstrained alignment of $\{S_i[1..x_{i,1} - 1] : 1 \leqslant i \leqslant \sigma\}$, $\{\bar{S}_i[j_1 + 1..j_2 - 1] : 1 \leqslant i \leqslant \sigma\}$ forms an unconstrained alignment of $\{S_i[x'_{i,1} + 1..x_{i,2} - 1] : 1 \leqslant i \leqslant \sigma\}$, and so on.

We illustrate these notions in the following example. Let input sequences S_1 and S_2 be

index	12345678	9012345678	901
S_1	agcatgcctg	acctcgtcgc	t
S_2	agcagaccaa	tcgtcgcgca	t

Let there be two patterns in the constraint, with sets of candidate occurrences $occur_1(1) = \{[3, 6], [7, 10], [8, 10]\}$, $occur_1(2) = \{[15, 19]\}$, $occur_2(1) = \{[3, 5], [19, 21]\}$, and $occur_2(2) = \{[12, 16]\}$. Then a feasible constrained alignment is

index	12	3	456789012	3	45678
$\bar{S}_1$	ag	$[3,6]$	-cctgacct	$[15,19]$	--c-t
$\bar{S}_2$	ag	$[3,5]$	acc--a-at	$[12,16]$	cgcat

(1)

This constrained alignment cannot be found under the formulation of [21,16] since the substrings in S_1 and S_2 corresponding to the first pattern are of different lengths, while as a consequence of adopting Hamming distance, such substrings must be of identical length in [21,16].

Intuitively, $\mathscr{A}$ is regarded as satisfying $\mathscr{P}$ since for each pattern k we can find a position j_k in $\mathscr{A}$ consisting solely of the pattern. The substrings $S_i[x_{i,k} \,..\, x'_{i,k}]$ responsible for this witness are then said to satisfy pattern k. This formulation follows the logic in the previous works [20,6,21,16,2,7] that, for each pattern, a legal occurrence on each sequence is chosen to support the feasibility of a constrained alignment, while those occurrences not chosen are treated as normal characters. Unlike the previous studies, we do not specify how the characters in the substrings $S_i[x_{i,k} \,..\, x'_{i,k}]$ satisfying pattern k are aligned. Instead we simply put $\bar{S}_i[j_k] = [x_{i,k}, x'_{i,k}]$. Aligning these substrings is not always meaningful since the patterns may correspond to entities irrelevant to the character composition, e.g., local features of secondary structure. Whenever desirable, one may as well expand these $\bar{S}_i[j_k]$ into alignments of the corresponding substrings without difficulty.

As in [9], the usefulness of treating patterns and characters in different levels is embodied in the ability to adopt different cost functions for them. In general, for a pattern, we may have different confidence on different candidate occurrence's being a true occurrence of the pattern. In particular, patterns are not necessarily recognized by character compositions, hence such confidence need not be based on character compositions either. For example, protein secondary structure prediction tools such as the well-known PSIPRED [17] give a putative secondary structure (helix, strand or coil) for each residue associated with a confidence value for the prediction. Hence in case that patterns correspond to secondary structures with candidate occurrences found by such a tool, our cost function should be able to differentiate the candidates according to the confidence. In this light, when we are evaluating the cost of matching two pattern occurrences, their positions and which sequences they lie on should be considered. Furthermore, a substring on an input sequence may simultaneously be candidate occurrences of two distinct patterns. For example, if there are two patterns `cgt` and `cgc` in the constraint, then a substring `cgt` of a sequence may be regarded as a candidate occurrence for each of both patterns, and we may regard it as being more likely to be the first and in turn may favor this possibility over the other. In terms of the cost function this means that the cost function should depend on which pattern a candidate occurrence is now expected to satisfy. Under the above considerations and following our notation like $\bar{S}_i[j_k] = [x_1, x_2]$, we use $\delta([x_1, x_2], [y_1, y_2])$ to denote the cost of matching the two candidate occurrences, which is meaningful only if we have specified i_1, i_2 and k such that $[x_1, x_2] \in occur_{i_1}(k)$, $[y_1, y_2] \in occur_{i_2}(k)$ and that these two occurrences are expected to satisfy pattern k (since, say, $[x_1, x_2]$ may be a member of $occur_{i_1}(k')$ for some other pattern k' as well).[1] In a constrained alignment $\{\bar{S}_i : 1 \leqslant i \leqslant \sigma\}$, for each pattern k there is a column j_k such that each $\bar{S}_i[j_k]$ corresponds to an occurrence satisfying the pattern. Hence the meaning of writing "$\delta(\bar{S}_i[j_k], \bar{S}_{i'}[j_k])$" is clear. For $\delta(a, b)$ where $a, b \in \Sigma \cup \{-\}$, typically we care only about what symbols a and b are and neglect information about their positions or which sequences they lie on.

Given the above discussion, the score of a constrained alignment $\mathscr{A} = \{\bar{S}_i : 1 \leqslant i \leqslant \sigma\}$ can be expressed as the simple form

$$score(\mathscr{A}) = \sum_{1 \leqslant i < i' \leqslant \sigma} \sum_{j=1}^{|\bar{S}_1|} \delta(\bar{S}_i[j], \bar{S}_{i'}[j]). \tag{2}$$

Consider the constrained alignment shown in (1). If we define the cost of matching two pattern occurrences as the score of the optimal unconstrained alignment of the two corresponding substrings, and for $a, b \in \Sigma \cup \{-\}$ we define $\delta(a, b) = 0$ if $a = b$ while $\delta(a, b) = 1$ otherwise, then the alignment in (1) has a score of 9, which is optimal with respect to δ. In particular, $\delta(\bar{S}_1[3], \bar{S}_2[3]) = 1$ and $\delta(\bar{S}_1[13], \bar{S}_2[13]) = 0$, corresponding, respectively, to the alignments

```
catg      cgtcg
ca-g  and cgtcg
```

The (generalized) CSA problem can then be stated as follows.

Constrained Sequence Alignment (CSA). *Given strings $S_1, \ldots, S_\sigma$, a constraint $\mathscr{P}$, and cost function δ, find a constrained alignment $\mathscr{A}$ satisfying $\mathscr{P}$ such that $score(\mathscr{A})$ is minimized.*

In Section 3, we study this problem.

[1] We may have introduced auxiliary symbols for each pattern occurrence that carry information about the start and end positions, the pattern to be satisfied, and which sequence the occurrence lies on. Then δ when applied on such symbols can be interpreted properly. For notational brevity, however, we choose to use $\delta([x_1, x_2], [y_1, y_2])$ directly.

Now we consider the unordered case. An unordered constraint $\mathscr{P}_u$ is also specified by $\langle occur_i(k) : 1 \leqslant i \leqslant \sigma$ and $1 \leqslant k \leqslant m\rangle$. An alignment $\mathscr{A}$ satisfying $\mathscr{P}_u$ is similarly defined, and we can also find m columns $j_1 < \cdots < j_m$, but now each j_k need not correspond to pattern k. What is required is that each pattern k is satisfied in exactly one column j_{ℓ_k}. Referring to the example given before, if we reverse patterns 1 and 2, that is, if we let $occur_1(2)=\{[3, 6], [7, 10], [8, 10]\}$, $occur_1(1) = \{[15, 19]\}$, $occur_2(2) = \{[3, 5], [19, 21]\}$, and $occur_2(1) = \{[12, 16]\}$, then (1) is a feasible solution for the unordered version, with column 3 in the alignment corresponding to pattern 2 and column 13 to pattern 1. Provided the knowledge of which column satisfies each pattern, we can evaluate an alignment satisfying $\mathscr{P}_u$ using Eq. (2). Then (1), with the new occurrence sets defined here and the function δ defined before, is also an optimal solution for the unordered version. The problem in question can be stated as follows.

Sequence Alignment with Unordered Constraint (SAUC). *Given sequences $S_1, \ldots, S_\sigma$, an unordered constraint $\mathscr{P}_u$, and cost function δ, find a constrained alignment $\mathscr{A}$ satisfying $\mathscr{P}_u$ such that score($\mathscr{A}$) is minimized.*

Section 4 will be devoted to this problem.

For the convenience of the reader, we give the definitions regarding the class NPO, AP-reduction, and approximation algorithms in the Appendix. They are adapted from [8,3].

3. Constrained sequence alignment

In this section, we study CSA with ordered constraints. First, we propose a polynomial time algorithm for the pairwise case, and then prove the inapproximability of the general version. A bounded version is shown to be complete in the class NPO.

3.1. An algorithm for the pairwise case

Let $\langle (S_1, S_2), \delta, \mathscr{P}\rangle$ be an instance of CSA. To compute an optimal solution for the instance, let $T[j_1, j_2; k]$ be the optimal score of constrained alignment of $S_1[1\,..\,j_1]$ and $S_2[1\,..\,j_2]$ satisfying the first k patterns. In addition, we take the convention that $\min \emptyset = \infty$. Then $T[j_1, j_2; k]$ has the following recurrence:

$$T[j_1, j_2; k] = \min \begin{cases} T[j_1 - 1, j_2; k] + \delta(S_1[j_1], -) & \text{if } j_1 > 0, \\ T[j_1, j_2 - 1; k] + \delta(-, S_2[j_2]) & \text{if } j_2 > 0, \\ T[j_1 - 1, j_2 - 1; k] + \delta(S_1[j_1], S_2[j_2]) & \text{if } j_1, j_2 > 0, \\ \min_{\substack{[x, j_1]\in occur_1(k) \\ [y, j_2]\in occur_2(k)}} \{T[x - 1, y - 1; k - 1] + \delta([x, j_1], [y, j_2])\} & \text{if } k > 0. \end{cases}$$

This is a simple generalization of the recurrence presented in [6]. Note that in the fourth item $[x, j_1]$ and $[y, j_2]$ are expected to satisfy pattern k, so that $\delta([x, j_1], [y, j_2])$ should have an unambiguous interpretation. The initial values are

$$T[0, 0; 0] = 0,$$

$$T[j_1, 0; k] = T[0, j_2; k] = \infty \quad \text{if } k > 0.$$

Also T is defined over $j_1 \in [0, |S_1|]$, $j_2 \in [0, |S_2|]$ and $k \in [0, m]$. The correctness can be easily seen by observing that, in an optimal alignment $\{\bar{S}_1, \bar{S}_2\}$ of $S_1[1\,..\,j_1]$ and $S_2[1\,..\,j_2]$ satisfying the first k patterns, there are two possibilities for the last column ℓ in the alignment: either $\bar{S}_i[\ell] \in occur_i(k)$ for $i = 1, 2$ or not. In the first case, let $\bar{S}_1[\ell] = [x, x']$ and $\bar{S}_2[\ell] = [y, y']$. It must be that $x' = j_1$ and $y' = j_2$ for the alignment to be feasible. The score of the alignment can then be decomposed into two parts: an optimal alignment of $S_1[1\,..\,x - 1]$ and $S_2[1\,..\,y - 1]$ satisfying the first $k - 1$ patterns whose cost is $T[x - 1, y - 1; k - 1]$, and the matching of occurrences $S_1[x\,..\,j_1]$ and $S_2[y\,..\,j_2]$ for the satisfaction of pattern k, with a cost of $\delta([x, j_1], [y, j_2])$. The fourth item clearly handles this case. If, on the other hand, the last column in the alignment is not responsible for the satisfaction of pattern k, then we can have $\bar{S}_1[\ell] = S_1[j_1]$ and $\bar{S}_2[\ell] = S_2[j_2]$, $\bar{S}_1[\ell] = S_1[j_1]$ and $\bar{S}_2[\ell] = -$, or $\bar{S}_1[\ell] = -$ and $\bar{S}_2[\ell] = S_2[j_2]$. If the first possibility is the case, then $\bar{S}_1[1\,..\,\ell - 1]$ and $\bar{S}_2[1\,..\,\ell - 1]$ must be an optimal alignment of $S_1[1\,..\,j_1 - 1]$ and $S_2[1\,..\,j_2 - 1]$ satisfying patterns 1 to k with score $T[j_1 - 1, j_2 - 1; k]$, while $\delta(\bar{S}_1[\ell], \bar{S}_2[\ell]) = \delta(S_1[j_1], S_2[j_2])$. This is handled by the third item in the recurrence. The other cases are similar.

The optimal score of constrained alignment satisfying $\mathscr{P}$ is then given by $T[|S_1|, |S_2|; m]$. It is also easy to reconstruct the corresponding alignment by doing some additional bookkeeping. To concentrate on the problem structure itself, we assume here that function δ can be computed in constant time, since different definitions can lead to different directions of optimization, which would complicate the presentation of the algorithm. If each $occur_i(k)$ is sorted by the end indices of its member occurrences, then for fixed j_1 and j_2, all those pairs $[x, j_1] \in occur_1(k)$ or $[y, j_2] \in occur_2(k)$ can be found when needed during the computation in time linear to the number of such intervals. The sorting itself is easy, taking $O(mn^2)$ time in total. The time taken by the fourth item in the recurrence is then

$$\begin{aligned}
&\sum_{k=1}^{m}\sum_{j_1=1}^{|S_1|}\sum_{j_2=1}^{|S_2|} |\{x : [x, j_1] \in occur_1(k)\}| \times |\{y : [y, j_2] \in occur_2(k)\}| \\
&\quad = \sum_{k=1}^{m}\left(\sum_{j_1=1}^{|S_1|} |\{x : [x, j_1] \in occur_1(k)\}| \times \sum_{j_2=1}^{|S_2|} |\{y : [y, j_2] \in occur_2(k)\}|\right) \\
&\quad = \sum_{k=1}^{m}(|occur_1(k)| \times |occur_2(k)|).
\end{aligned}$$

For brevity let $M = \sum_{k=1}^{m}(|occur_1(k)| \times |occur_2(k)|)$. The total time needed to compute table T is then $O(M + mn^2)$, where n is the length of the longer input sequence. We summarize the results stated above as the following theorem.

Theorem 1. *Given sequences S_1 and S_2, along with a constraint $\mathscr{P}$ and cost function δ, where $\mathscr{P} = \langle occur_i(k) : i=1, 2, k \in [1, m]\rangle$, an optimal constrained alignment of S_1 and S_2 satisfying the constraint can be found in $O(M+mn^2)$ time, where $M = \sum_{k=1}^{m}(|occur_1(k)| \times |occur_2(k)|)$.*

If $|occur_i(k)| = O(n)$ for $i = 1, 2$ and $k \in [1, m]$, then $M = O(mn^2)$ and the algorithm takes $O(mn^2)$ time. It is also straightforward to adopt affine gap penalty in the alignment, which is commonly used in biological applications. By applying the progressive framework in [21,16], one can use the recurrences given here as a core to solve the case when the number of sequences is more than two, but with no performance guarantee. The recurrences can also be easily modified to give optimal solutions when more than two sequences exist, but can only be practical when the number of sequences is small.

We sketch how the above algorithm can be conveniently applied to the formulation in [21,16]. Given patterns (strings) $P_1, \ldots, P_m$ over Σ, which constitute a constraint in [21,16], we find all substrings in S_1 and S_2 whose Hamming distances from each P_k are within some specified threshold. Each such set of occurrences for pattern P_k on string S_i is named $occur_i(k)$. This can be done in $O(\gamma n)$ time in a straightforward manner, where $\gamma = \sum_{k=1}^{m}|P_k|$. Note that $\gamma = O(n)$ or there can be no feasible constrained alignment, and that as a consequence of adopting Hamming distance, all occurrences in $occur_1(k) \cup occur_2(k)$ must be of identical length. Also, since $\delta([x, x+\ell], [y, y+\ell])$ is defined to be $\sum_{j=0}^{\ell}\delta(S_1[x+j], S_2[y+j])$ in [21,16], for a fixed k it is easy to compute in $O(n^2)$ time all $\delta([x, x+\ell], [y, y+\ell])$, where $[x, x+\ell] \in occur_1(k)$, $[y, y+\ell] \in occur_2(k)$ and both are expected to satisfy pattern k, utilizing that given $\sum_{j=0}^{\ell}\delta(S_1[x+j], S_2[y+j])$ it takes $O(1)$ time to compute $\sum_{j=0}^{\ell}\delta(S_1[x+1+j], S_2[y+1+j])$. Now our algorithm can be applied, taking $O(mn^2)$ time, since $|occur_i(k)|=O(n)$ and $M=O(mn^2)$ in this case. This overall time complexity is the same as that in [21,16], with a much simpler presentation due to our general model.

3.2. Inapproximability of the general case

Now we turn to examine the hardness of CSA for the general cases. We shall prove that CSA is not approximable within any function computable in polynomial time if the number of input sequences can be general. The proof involves a reduction from the weighted 3-satisfiability problem (W3SAT) [3,8], which is among the best known NPO-complete[2] problems.

[2] The problem defined in [19] is slightly different from that in [8,3] in that, the former always regards the trivial truth assignment as a feasible solution; this version is treated in [3] as exp-APX-complete.

Weighted 3-Satisfiability (W3SAT)

Instance: Set U of variables, collection C of clauses over U such that each clause $c \in C$ has $|c| = 3$, and weight function $w : U \to \mathbb{N}$.

Solution: A truth assignment, i.e., function $\tau : U \to \{\text{true}, \text{false}\}$, satisfying C.

Measure: $\sum_{v:\tau(v)=\text{true}} w(v)$.

Goal: min.

To fit the standard definition of approximation algorithms (as given in the Appendix), we restrict the range of δ to nonnegative integers so that the problem becomes a member of NPO. This restricted version of CSA will be referred to as *restricted CSA*, whose definition is given as follows.

Restricted Constrained Sequence Alignment (RCSA)

Instance: Sequences $S_1, \ldots, S_\sigma$, nonnegative integer function δ, and a constraint $\mathscr{P}$.

Solution: A constrained alignment $\mathscr{A}$ satisfying $\mathscr{P}$ with $score(\mathscr{A}) > 0$.

Measure: $score(\mathscr{A})$.

Goal: min.

We denote the set of all instances of W3SAT by I_{W3SAT}, and that of RCSA by I_{RCSA}. In what follows, we construct a transformation $f_\alpha : I_{\text{W3SAT}} \to I_{\text{RCSA}}$, such that if $\langle U, C, w\rangle \in I_{\text{W3SAT}}$ is satisfiable, then $score(\mathscr{A}) \leqslant |U| w_{\max}$, while if not, we have $score(\mathscr{A}) \geqslant \alpha |U| w_{\max}$, where $\mathscr{A}$ is an optimal solution to $f_\alpha(\langle U, C, w\rangle)$ and $w_{\max} = \max_{v\in U}\{w(v)\}$. We can then utilize the gap between these two values to prohibit the existence of a polynomial time approximation algorithm for RCSA, assuming P $\neq$ NP. For any fixed $\alpha > 1$, there is a corresponding mapping f_α from I_{W3SAT} to I_{RCSA}.

Given $\langle U, C, w\rangle \in I_{\text{W3SAT}}$, we shall construct $\langle (S_1, \ldots, S_\sigma), \delta, \mathscr{P}\rangle \in I_{\text{RCSA}}$, where $\sigma = |C| + 1$ and m, the number of patterns, is $|U|$. Assume without loss of generality that any $c_i \in C$ does not contain both v_k and $\neg v_k$ for $v_k \in U$, since such a clause is always true and its removal will not affect the satisfiability of the whole formula. Let $\mathtt{a}$ be the only character in alphabet Σ under consideration. For all $1 \leqslant i < \sigma$ and $1 \leqslant k \leqslant m$, construct sequence S_i as $\mathtt{a}^{2m}$ (i.e., the concatenation of $2m$ $\mathtt{a}$'s), and $occur_i(k)$ as

$$occur_i(k) = \begin{cases} \{[k,k],[k,k+m],[k+m,k+m]\} & \text{if } v_k \in c_i \text{ or } \neg v_k \in c_i, \\ \{[k,k],[k+m,k+m]\} & \text{otherwise.} \end{cases}$$

It can be noticed that, since each S_i has a length of $2m$ while each $[k, k+m]$ is an occurrence of length $m+1$, in a feasible constrained alignment there can be at most one pattern satisfaction of the form $[k, k+m]$. If $[k, k+m]$ does appear in $\bar{S}_i$, it must appear in column k, and in this case $\bar{S}_i[j]$ corresponds to a pattern for each j. In addition, $occur_i(k) \cap occur_i(k') = \emptyset$ for $k \neq k'$, hence any $[x_1, x_2] \in occur_i(k)$ can only be expected to satisfy pattern k. Sequence S_σ is constructed to be $\mathtt{a}^m$, and we let $occur_\sigma(k) = \{[k,k]\}$. In any feasible solution, $\bar{S}_\sigma[j]$ must correspond to pattern j for each j. Since $|S_\sigma| = m$ while $|S_i| = 2m$ for $i < \sigma$, provided that the cost of matching a space character against any nonspace character is sufficiently high, in any $\bar{S}_i$, $i < \sigma$, it is much preferred to have some pattern k satisfied by an occurrence of the form $[k, k+m]$ if $[k, k+m] \in occur_i(k)$ exists, otherwise the alignment will have a poor measure. As an illustration of the above construction, formula $(v_1 \vee v_2 \vee v_2) \wedge (v_1 \vee \neg v_2 \vee \neg v_2) \wedge (\neg v_1 \vee v_2 \vee v_2) \wedge (\neg v_1 \vee \neg v_2 \vee v_3)$will be transformed into sequences $S_1 = S_2 = S_3 = S_4 = \mathtt{a}^6$ and $S_5 = \mathtt{a}^3$, with occurrence sets $occur_1(1) = occur_2(1) = occur_3(1) = occur_4(1) = \{[1,1],[1,4],[4,4]\}$, $occur_5(1) = \{[1,1]\}$, $occur_1(2) = occur_2(2) = occur_3(2) = occur_4(2) = \{[2,2],[2,5],[5,5]\}$, $occur_5(2) = \{[2,2]\}$, $occur_1(3) = occur_2(3) = occur_3(3) = \{[3,3],[6,6]\}$, $occur_4(3) = \{[3,3],[3,6],[6,6]\}$, and $occur_5(3) = \{[3,3]\}$.

Note that for $i < \sigma$, we have only three kinds of intervals in $occur_i(k)$; they are designed as indicators for the states of clauses' satisfaction. More specifically, our purpose is that, in an optimal constrained alignment of $f_\alpha(\langle U, C, w\rangle)$, $\bar{S}_i[j] = [j, j]$ indicates that variables v_1 through v_j are set to values unable to satisfy clause c_i, $\bar{S}_i[j] = [j, j+m]$ indicates that v_j is the first variable satisfying c_i, and $\bar{S}_i[j] = [j+m, j+m]$ indicates that c_i has already been satisfied by some v_k, $k < j$. If both v_k and $\neg v_k$ are not in c_i, the value of v_k can never affect the satisfaction of c_i, and hence $occur_i(k)$ is constructed to contain only $[k,k]$ and $[k+m, k+m]$. On the other hand, as we shall see, S_σ is responsible for the score of an optimal solution to $f_\alpha(\langle U, C, w\rangle)$ for $\langle U, C, w\rangle$ satisfiable to be some value of particular use, namely, to be the same as the value of the optimal solution to $\langle U, C, w\rangle$. This will be used to prove the NPO-completeness of the bounded version of RCSA, rather than the inapproximability of RCSA itself.

We need to ensure the consistency of the values of variables, e.g., if $v_k \in c_i$ and $\neg v_k \in c_{i'}$, then v_k cannot be used to satisfy both c_i and $c_{i'}$. This is done by the construction of the cost function δ; inconsistent cases will incur a high

cost, hence prohibited in an optimal solution for satisfiable instances. By our discussion in the preceding paragraph, for the value of v_k to be consistent, we design symmetric function δ as follows. For $i, i' < \sigma$, $[k, k] \in occur_i(k)$ and $[k, k+m] \in occur_{i'}(k)$,

$$\delta([k,k],[k,k+m]) = \begin{cases} \lceil\alpha\rceil m w_{\max} & \text{if } v_k \text{ in } c_i \text{ and } c_{i'}, \text{ or } \neg v_k \text{ in } c_i \text{ and } c_{i'}, \\ 0 & \text{otherwise.} \end{cases}$$

For $[x_1, x_2] \in occur_i(k)$ and $[y_1, y_2] \in occur_{i'}(k)$, $[x_1, x_2] = [y_1, y_2]$ being $[k, k]$ or $[k, k+m]$, and $i, i' < \sigma$,

$$\delta([x_1,x_2],[y_1,y_2]) = \begin{cases} \lceil\alpha\rceil m w_{\max} & \text{if } v_k \in c_i \text{ and } \neg v_k \in c_{i'}, \text{ or } \neg v_k \in c_i \text{ and } v_k \in c_{i'}, \\ 0 & \text{otherwise.} \end{cases}$$

For any $[x_1, x_2] \in occur_i(k)$, $[k+m, k+m] \in occur_{i'}(k)$ and $i, i' < \sigma$, we let $\delta([x_1, x_2], [k+m, k+m]) = 0$. Finally, if $[k, k] \in occur_\sigma(k)$ and $[x_1, x_2] \in occur_i(k)$,

$$\delta([k,k],[x_1,x_2]) = \begin{cases} w(v_k) & \text{if } [x_1,x_2] = [k,k+m],\ v_k \in c_i \text{ and } v_k \notin c_{i'} \text{ for all } i' < i, \\ 0 & \text{otherwise.} \end{cases}$$

Following the above ideas, for a satisfiable instance, we expect to see in an optimal constrained alignment that each $\bar{S}_i$ contains exactly one symbol representing an interval of the form $[k, k+m]$, reflecting the fact that each clause is satisfied, and that there can be only one "first" variable satisfying a clause. For this aim, for $a, b \in \Sigma \cup \{-\}$ we define $\delta(a, b)$ to be a constant of value $\lceil\alpha\rceil m w_{\max}$. In this manner, there will be no characters or spaces in each $\bar{S}_i$ in an optimal constrained alignment for a satisfiable instance. As we shall see, this, along with the definition of feasibility of a constrained alignment, then ensures the existence of a unique pattern occurrence of the form $[k, k+m]$ in each $\bar{S}_i$. The construction is now complete; we let $f_\alpha(\langle U, C, w\rangle) = \langle (S_1, \ldots, S_\sigma), \delta, \mathscr{P}\rangle$, where $\mathscr{P} = \langle occur_i(k) : 1 \leqslant i \leqslant \sigma, 1 \leqslant k \leqslant m\rangle$.

We can now establish the following property. For $\langle U, C, w\rangle \in I_{\text{W3SAT}}$, we denote the set of all solutions to $\langle U, C, w\rangle$ as $sol_{\text{W3SAT}}(\langle U, C, w\rangle)$. A similar notation is used for instances in I_{RCSA}.

Lemma 2. *Given $\langle U, C, w\rangle \in I_{\text{W3SAT}}$, if there is a satisfying truth assignment μ for $\langle U, C, w\rangle$, then there exists a feasible constrained alignment $\mathscr{A}$ for $f_\alpha(\langle U, C, w\rangle)$ with $score(\mathscr{A}) \leqslant \sum_{v_k:\mu(v_k)=\text{true}} w(v_k)$.*

Proof. Construct $\mathscr{A} = \{\bar{S}_1, \ldots, \bar{S}_\sigma\}$ as follows. For all $i < \sigma$ let $j_i = \min\{j : v_j \text{ satisfies } c_i\}$. Then one of v_{j_i} and $\neg v_{j_i}$ is in c_i, hence $[j_i, j_i + m] \in occur_i(j_i)$. For $i < \sigma$, we set $\bar{S}_i[j_i] = [j_i, j_i + m]$, $\bar{S}_i[j] = [j, j]$ for all $j < j_i$, and $\bar{S}_i[j] = [j + m, j + m]$ for all $j > j_i$, where $m = |U|$. Also let $\bar{S}_\sigma[j] = [j, j]$ for all $1 \leqslant j \leqslant m$. Clearly $|\bar{S}_i| = m$ for all i. The feasibility of $\mathscr{A}$ is easy to see.

For any $1 \leqslant i < i' < \sigma$ and $1 \leqslant j \leqslant m$, we claim that $\delta(\bar{S}_i[j], \bar{S}_{i'}[j]) = 0$. First, this holds if $\bar{S}_i[j]$ or $\bar{S}_{i'}[j]$ corresponds to $[j + m, j + m]$. If both $\bar{S}_i[j]$ and $\bar{S}_{i'}[j]$ are $[j, j]$, then by construction of $\bar{S}_i$ and $\bar{S}_{i'}$, v_j is prior to the first variable satisfying c_i or $c_{i'}$. It follows that the sign (i.e., negated or not) of v_j in c_i and $c_{i'}$ must be the same, since if not, one of c_i and $c_{i'}$ is satisfied. The claim then holds in this case by the definition of δ. Now consider the case when $\bar{S}_i[j] = [j, j+m]$, and $\bar{S}_{i'}[j]$ is $[j, j]$ or $[j, j+m]$. If $\bar{S}_{i'} = [j, j]$, then v_j satisfies c_i but not $c_{i'}$, which suggests that none of v_j and $\neg v_j$ is in $c_{i'}$, or that the sign of v_j in c_i is different from that in $c_{i'}$; both of these lead to $\delta(\bar{S}_i[j], \bar{S}_{i'}[j]) = 0$. If $\bar{S}_{i'} = [j, j+m]$, then both c_i and $c_{i'}$ are satisfied by v_j, and again $\delta(\bar{S}_i[j], \bar{S}_{i'}[j]) = 0$ by construction.

On the other hand, we have

$$\sum_{j=1}^{m} \sum_{1 \leqslant i < \sigma} \delta(\bar{S}_\sigma[j], \bar{S}_i[j]) = \sum_{j=1}^{m} w(v_j) \times \psi(j),$$

where $\psi(j) = 1$ if for some i, $\bar{S}_i[j] = [j, j+m]$ and $v_j \in c_i$, and $\psi(j) = 0$ otherwise. By construction of $\bar{S}_i$, $\psi(j) = 1$ only if $\mu(v_j) = \text{true}$. Hence,

$$score(\mathscr{A}) = \sum_{j=1}^{m} w(v_j) \times \psi(j) \leqslant \sum_{v_k:\mu(v_k)=\text{true}} w(v_k),$$

and the lemma follows. □

To complete our reduction, we define function g to transform a feasible constrained alignment $\mathscr{A} = \{\bar{S}_1, \ldots, \bar{S}_\sigma\}$ of $f_\alpha(\langle U, C, w\rangle)$ to a truth assignment τ for U. For $v_j \in U$, let

$$\tau(v_j) = \begin{cases} \text{true} & \text{if for some } i,\ \bar{S}_i[j] = [j, j+m] \text{ and } v_j \in c_i, \\ \text{false} & \text{otherwise.} \end{cases}$$

Then let $g(\langle U, C, w\rangle, \mathscr{A}) = \tau$. Our purpose of defining g is the following lemma.

Lemma 3. *Let* $\langle U, C, w\rangle \in I_{\text{W3SAT}}$, *and* $\mathscr{A}$ *be a feasible solution of* $f_\alpha(\langle U, C, w\rangle)$, *where* $\alpha > 1$. *Let* $\tau = g(\langle U, C, w\rangle, \mathscr{A})$. *If* $score(\mathscr{A}) < \lceil\alpha\rceil m w_{\max}$, *then* τ *satisfies* C, *and* $\sum_{v_k:\tau(v_k)=\text{true}} w(v_k) = score(\mathscr{A})$.

Proof. Let $\mathscr{A} = \{\bar{S}_1, \ldots, \bar{S}_\sigma\}$. As noted before, since $\delta(a, b) = \lceil\alpha\rceil m w_{\max}$ for $a, b \in \Sigma \cup \{-\}$, $score(\mathscr{A}) < \lceil\alpha\rceil m w_{\max}$ implies that each $\bar{S}_i[j]$ corresponds to a pattern, and $|\bar{S}_i| = m$ for all i. Then for each $i < \sigma$, $\bar{S}_i[j] = [j, j+m]$ for exactly one j. If $\bar{S}_i[j] = [j, j+m]$ and $v_j \in c_i$, then $\tau(v_j) = \text{true}$, and c_i is satisfied. If $\bar{S}_i[j] = [j, j+m]$ and $\neg v_j \in c_i$, then for all other i' such that $\bar{S}_{i'}[j] = [j, j+m]$, it must also be that $\neg v_j \in c_{i'}$, since if not, then $\delta(\bar{S}_i[j], \bar{S}_{i'}[j]) = \lceil\alpha\rceil m w_{\max}$, which cannot hold by premise. We then have $\tau(v_j) = \text{false}$ in this case, and c_i is satisfied. It follows that all c_i's are satisfied.

By the definition of δ and $score(\mathscr{A}) < \lceil\alpha\rceil m w_{\max}$, if $\delta(\bar{S}_i[j], \bar{S}_{i'}[j]) > 0$ then i or i', say i', must be σ, and $\bar{S}_i[j] = [j, j+m]$ with $v_j \in c_i$, hence $\delta(\bar{S}_i[j], \bar{S}_{i'}[j]) = w(v_j)$ and $\tau(v_j) = \text{true}$. Define function ψ as in the proof of the previous lemma. The equality of $score(\mathscr{A})$ and $\sum_{v_k:\tau(v_k)=\text{true}} w(v_k)$ then follows by observing that ψ and τ are equivalent, and that $score(\mathscr{A}) = \sum_{j=1}^{m} w(v_j) \times \psi(j)$. □

The following corollary is used in the proof of NPO-completeness of the bounded version later. For $\langle U, C, w\rangle \in I_{\text{W3SAT}}$, let $opt(\langle U, C, w\rangle)$ be the score of an optimal solution to $\langle U, C, w\rangle$. The same notation is used for instances of other problems.

Corollary 4. *Given a satisfiable instance* $\langle U, C, w\rangle$ *of* W3SAT, *we have* $opt(\langle U, C, w\rangle) = opt(f_\alpha(\langle U, C, w\rangle))$ *for any* $\alpha > 1$.

Proof. By Lemma 2, we have $opt(f_\alpha(\langle U, C, w\rangle)) \leqslant opt(\langle U, C, w\rangle)$. On the other hand, Lemma 3 suggests that $opt(f_\alpha(\langle U, C, w\rangle)) \geqslant opt(\langle U, C, w\rangle)$. □

Our main result regarding the hardness of RCSA is stated as follows. For an instance of RCSA, let m, n and σ denote the number of patterns, length of the longest input sequence, and number of input sequences of the instance, respectively.

Theorem 5. RCSA *is not approximable within* $r(m, n, \sigma)$, *where* $r(m, n, \sigma) > 1$ *is any function computable in polynomial time*, *unless* P = NP.

Proof. Assume for contradiction that *ALG* is a polynomial time $r(m, n, \sigma)$-approximation algorithm for RCSA. We show that we can use *ALG* and f_α to decide the satisfiability of any given 3-CNF boolean formula, with set C of clauses over set U of variables, in polynomial time.

Let w be any weight function defined on U. Then $\langle U, C, w\rangle \in I_{\text{W3SAT}}$, and let $m = |U|$, $n = 2m$ and $\sigma = |C| + 1$. Compute $\alpha = 2r(m, n, \sigma)$. By Lemmas 2 and 3, if $\langle U, C, w\rangle$ is satisfiable, then $f_\alpha(\langle U, C, w\rangle)$ has a feasible solution with score at most $m w_{\max}$, while if not, any feasible solution of $f_\alpha(\langle U, C, w\rangle)$ has a score at least $\lceil\alpha\rceil m w_{\max} > r(m, n, \sigma) m w_{\max}$. Consequently, $\langle U, C, w\rangle$ is satisfiable if and only if $score(ALG(f_\alpha(\langle U, C, w\rangle))) \leqslant r(m, n, \sigma) m w_{\max}$. In addition, $r(m, n, \sigma)$, and hence α, is computable in polynomial time, and given α, the computation of function f_α takes polynomial time. In this way, the satisfiability of any 3-CNF boolean formula can be decided in polynomial time, which cannot hold unless P = NP. □

It is clear that RCSA is a special case of CSA. We then have the following conclusion.

Corollary 6. CSA *is not approximable within* $r(m, n, \sigma)$, *where* $r(m, n, \sigma) > 1$ *is any function computable in polynomial time*, *unless* P = NP.

The classic MSA problem is MAX SNP-hard when the score function is given as a part of an instance [14], which implies that it cannot have a polynomial time approximation scheme (PTAS) unless P = NP [1]. Good constant ratio approximation algorithms have been proposed for MSA with the distance function satisfying triangle inequality [12,4], which is NP-hard [22,5]. It is interesting to see that, when constraints defined here are imposed, sequence alignment becomes not approximable in general.

3.3. A bounded version

In this section we study a bounded version of RCSA, whose definition is given as follows.

Restricted Constrained Sequence Alignment with Bound (RCSAB)

Instance: Sequences $S_1, \ldots, S_\sigma$, nonnegative integer function δ, a constraint $\mathscr{P}$, and a positive integer B.

Solution: A constrained alignment $\mathscr{A} = (\bar{S}_1, \ldots, \bar{S}_\sigma)$ satisfying $\mathscr{P}$ with $0 < score(\mathscr{A}) < B$.

Measure: $score(\mathscr{A})$.

Goal: min.

Observe that, given an instance of RCSAB, the test of whether the instance has any feasible solution is just the decision version of RCSA, which is NP-hard. The following result is thus immediate.

Proposition 7. *Given an instance of* RCSAB, *the test of whether it has a feasible solution cannot be done in polynomial time*, *unless* P = NP.

In what follows we show that RCSAB is a member of the family of NPO-complete problems by giving an AP-reduction [3] from W3SAT to RCSAB. For this purpose, we define function $h : I_{\mathrm{W3SAT}} \times (1, \infty) \to I_{\mathrm{RCSAB}}$ to be $h(\langle U, C, w\rangle, \varepsilon) = \langle f_2(\langle U, C, w\rangle), B\rangle$, where $B = 2|U|w_{\max}$. Also, for a feasible solution $\mathscr{A}$ to $h(\langle U, C, w\rangle, \varepsilon)$, where $\langle U, C, w\rangle \in I_{\mathrm{W3SAT}}$, let $h'(\langle U, C, w\rangle, \mathscr{A}, \varepsilon) = g(\mathscr{A})$. In the following theorem, we show that $(h, h', 1)$ is a valid AP-reduction, and the NPO-completeness of RCSAB then follows. Since functions h and h' are defined to be independent of ε, in the proof we shall omit it for notational brevity.

Theorem 8. RCSAB *is* NPO-*complete*.

Proof. We check the five items of Definition 20 in the Appendix (or [3, Definition 8.3]) one by one. First, given $\langle U, C, w\rangle \in I_{\mathrm{W3SAT}}$, it is clear that $h(\langle U, C, w\rangle)$ is an instance of RCSAB. Second, if $\langle U, C, w\rangle \in I_{\mathrm{W3SAT}}$ is satisfiable, then by Corollary 4, we have $opt(h(\langle U, C, w\rangle)) = opt(\langle U, C, w\rangle) \leqslant m w_{\max} < 2m w_{\max} = B$, hence $h(\langle U, C, w\rangle)$ also has some feasible solution. Third, let $\mathscr{A}$ be a feasible solution to $h(\langle U, C, w\rangle)$. Then $score(\mathscr{A}) < 2m w_{\max}$, and by Lemma 3, $h'(\langle U, C, w\rangle, \mathscr{A}) = g(\mathscr{A})$ is a feasible solution to $\langle U, C, w\rangle$. Fourth, it is clear that both h and h' are computable in polynomial time. Finally, since $opt(\langle U, C, w\rangle)$ is equal to $opt(h(\langle U, C, w\rangle))$, if $score(\mathscr{A})$ is within ε times $opt(h(\langle U, C, w\rangle))$, we must have by Lemma 3 that $h'(\langle U, C, w\rangle, \mathscr{A})$ has a score within $\varepsilon = 1 + 1 \cdot (\varepsilon - 1)$ times $opt(\langle U, C, w\rangle)$. Hence $(h, h', 1)$ is a valid AP-reduction from W3SAT to RCSAB, and the theorem follows. □

Clearly, testing whether a given instance of W3SAT has some feasible solution amounts to testing whether the given boolean formula is satisfiable, which cannot be done efficiently unless P = NP. Since AP-reduction preserves feasibility of the involved instance, and since W3SAT is in NPO, any NPO-complete problem cannot admit an efficient algorithm for testing whether a given instance has feasible solutions, unless P = NP. Theorem 8 therefore justifies again Proposition 7.

To complete our discussion, we investigate whether RCSA, the unbounded problem, admits an efficient feasibility test algorithm. Given an instance of RCSA, it has feasible solutions if and only if in each S_i there exist nonoverlapping $[x_1, x'_1], \ldots, [x_m, x'_m]$ such that $[x_k, x'_k] \in occur_i(k)$ and $x'_k < x_{k+1}$. This test can be done in polynomial time in a greedy manner by favoring occurrences with smaller ending indices. Interestingly, this implies that, unlike its bounded counterpart, RCSA is not complete in class NPO.

4. Sequence alignment with unordered constraint

In this section we establish the hardness result of SAUC. As before, we force the problem to be a member of NPO by imposing a restriction on the range of the function δ. The function *score* is defined in Eq. (2). To distinguish from the original formulation, this version will be referred to as restricted SAUC (RSAUC).

Restricted Sequence Alignment with Unordered Constraint

Instance: Sequences $S_1, \ldots, S_\sigma$, nonnegative integer function δ, and an unordered constraint $\mathscr{P}_u$.

Solution: A constrained alignment $\mathscr{A} = \{\bar{S}_1, \ldots, \bar{S}_\sigma\}$ satisfying $\mathscr{P}_u$ with $score(\mathscr{A}) > 0$.

Measure: $score(\mathscr{A})$.

Goal: min.

In this section we consider the pairwise case when $\sigma = 2$, whose hardness implies that of the general case. We shall give a reduction from the shortest Hamiltonian path problem [23] to pairwise RSAUC.

Shortest Hamiltonian Path (SHP)

Instance: A weighted graph $G = (V, E)$ with weight function $w : E \to \mathbb{N}$. Let $V = \{v_1, \ldots, v_N\}$.

Solution: A simple path in G, i.e., a sequence $\langle v_{\pi(1)}, v_{\pi(2)}, \ldots, v_{\pi(N)}\rangle$ of vertices, where π is a permutation on $\{1, \ldots, N\}$, $\pi(1) = 1$, and $\pi(N) = N$, such that for any $1 \leqslant i < N$, $(v_{\pi(i)}, v_{\pi(i+1)}) \in E$.

Measure: The length of the path, i.e., $\sum_{i=1}^{N-1} w(v_{\pi(i)}, v_{\pi(i+1)})$.

Goal: min.

For a path P, let $len(P)$ be its length. Without loss of generality, it is assumed that for all $e \in E$, $w(e) > 0$, since if not, one can add a constant to $w(e)$ for all $e \in E$ without affecting the shortest Hamiltonian path. The decision of whether the set of feasible solutions is empty is the Hamiltonian path between two points problem, abbreviated simply as HP2, whose hardness is well known [11].

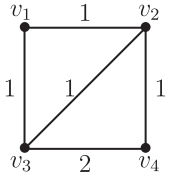
Lemma 9 (*Garey and Johnson [11]*). HP2 *is* NP*-complete*.

In what follows, we give a reduction from SHP to RSAUC, which will be used to prove that RSAUC is not approximable and that a bounded version of RSAUC is complete in the class of NPO. The reduction is done by two functions, as in the reduction from W3SAT to RCSA. For notational convenience, the two functions are denoted as before, namely, f_α and g. For an instance $\langle G, w\rangle$ of SHP, construct $f_\alpha(\langle G, w\rangle) = \langle (S_1, S_2), \delta, \mathscr{P}_u\rangle$ as follows. Let $G = (V, E)$ and $V = \{v_1, \ldots, v_N\}$. Let $n = N^2$, and construct $S_1 = \mathtt{a}^n$ and $S_2 = \mathtt{b}^n$, where the alphabet Σ is set to be $\{\mathtt{a}, \mathtt{b}\}$.

Let $m = N$. Construct $\mathscr{P}_u = \langle occur_i(k) : i = 1, 2 \text{ and } 1 \leqslant k \leqslant m\rangle$ as follows. For $i = 1, 2$ and for all $1 \leqslant k \leqslant m$,

$$occur_i(k) = \begin{cases} \bigcup_{p:(v_1,v_p)\in E}\{[1, N + p - 1]\} & \text{if } k = 1, \\ \bigcup_{q=1}^{N-2}\bigcup_{p:(v_k,v_p)\in E}\{[qN + k, (q + 1)N + p - 1]\} & \text{if } 1 < k < m, \\ \{[N^2, N^2]\} & \text{if } k = m. \end{cases}$$

For example, consider the following graph:



This graph will be transformed into sequences $S_1 = \mathtt{a}^{16}$ and $S_2 = \mathtt{b}^{16}$, and occurrence sets $occur_i(1) = \{[1, 5], [1, 6]\}$, $occur_i(2) = \{[6, 8], [6, 10], [6, 11], [10, 12], [10, 14], [10, 15]\}$, $occur_i(3) = \{[7, 8], [7, 9], [7, 11], [11, 12], [11, 13], [11, 15]\}$, and $occur_i(4) = \{[16, 16]\}$, for $i = 1, 2$.

The index pairs in $occur_i(k)$ are used to represent edges in E. For an integer x, let $\overline{x}$ denote $x \bmod N$. Then index pair $[x_1, x_2] \in occur_i(k)$ for $k < m$ corresponds to edge $(v_{\overline{x_1}}, v_{\overline{x_2}+1})$. Under proper arrangement, the index pairs are concatenated to encode paths from v_1 to v_N. Note that $occur_1(k) = occur_2(k)$ for all k; in an optimal constrained alignment for a positive instance, all the occurrences used to satisfy the constraint are intended to be identical on the two sequences.

Some paths from v_1 to v_N, when represented by occurrences in $occur_i(k)$, cause overlaps and hence result in infeasible constrained alignments. Other paths can be encoded in valid constrained alignments. Among these, we are interested in Hamiltonian ones. They are distinguished from others by alignment scores; those not Hamiltonian will incur a high cost. This will be achieved by setting $\delta(a, b) = \lceil\alpha\rceil N w_{\max}$ for any $a, b \in \Sigma \cup \{-\}$, where $w_{\max} = \max_{e\in E}\{w(e)\}$.

For occurrences $[x_1, x_2] \in occur_1(k)$ and $[y_1, y_2] \in occur_2(k)$, implying that both are expected to satisfy pattern k by the construction of $occur_i(k)$,

$$\delta([x_1, x_2], [y_1, y_2]) = \begin{cases} w(v_{\overline{x_1}}, v_{\overline{x_2}+1}) & \text{if } [x_1, x_2] = [y_1, y_2] \text{ and } k < m, \\ 0 & \text{otherwise.} \end{cases}$$

Then we let $f_\alpha(G, w) = \langle (S_1, S_2), \delta, \mathscr{P}_u \rangle$, which completes the construction.

Now we construct function g as follows. Given a feasible solution $\mathscr{A} = \{\bar{S}_1, \bar{S}_2\}$ for $\langle (S_1, S_2), \delta, \mathscr{P}_u \rangle \in I_{\text{RSAUC}}$, let indices $j_1 < \cdots < j_m$ be such that $\bar{S}_1[j_k] = [y_k, z_k]$. Then, we construct $g(\mathscr{A})$ as $\langle v_1, v_{\overline{y_2}}, \ldots, v_{\overline{y_{m-1}}}, v_N \rangle$.

For the example shown previously, the optimal alignment $\mathscr{A}^*$ of the instance is

[1, 6][7, 9][10, 15][16, 16],

[1, 6][7, 9][10, 15][16, 16]

with score 3, and $g(\mathscr{A}^*) = \langle v_1, v_3, v_2, v_4 \rangle$ is the shortest Hamiltonian path.

Lemma 10. *Given $\langle G, w \rangle \in I_{\text{SHP}}$, let $\mathscr{A} = \{\bar{S}_1, \bar{S}_2\}$ be a feasible solution for $f_\alpha(\langle G, w \rangle)$ such that $score(\mathscr{A}) < \alpha N w_{\max}$, where $\alpha \geqslant 1$. Then $g(\mathscr{A})$ is a feasible solution for $\langle G, w \rangle$, and $len(g(\mathscr{A})) = score(\mathscr{A})$.*

Proof. Let j_k, y_k and z_k be as defined above. As noted before, by the construction of $occur_i(k)$, we have $(v_{\overline{y_k}}, v_{\overline{z_k}+1}) \in E$ for all $k < m$. Since $score(\mathscr{A}) < \alpha N w_{\max}$, all positions in $\bar{S}_1$ and $\bar{S}_2$ must correspond to pattern occurrences. Hence $z_k + 1 = y_{k+1}$ for all $k < m$. Consequently, $\overline{z_k}+1 = \overline{y_{k+1}}$ and hence $(v_{\overline{y_k}}, v_{\overline{y_{k+1}}}) \in E$ for all $k \leqslant m-2$, and $(v_{\overline{y_{m-1}}}, v_N) \in E$. Since $[y_k, z_k] \in occur_i(\overline{y_k})$ for $k < m$, it is implied by the satisfaction of all m patterns that $\overline{y_k} \neq \overline{y_{k'}}$ for $k \neq k'$. It then follows that $\langle v_1, v_{\overline{y_2}}, \ldots, v_{\overline{y_{m-1}}}, v_N \rangle$ is a Hamiltonian path in G.

Let $\bar{S}_2[j_k] = [y'_k, z'_k] \in occur_2(\overline{y'_k})$. The same as above, we have $z'_k + 1 = y'_{k+1}$ for $k < m$ and $\overline{z'_k} + 1 = \overline{y'_{k+1}}$ for all $k \leqslant m-2$. Since $\bar{S}_1[j_k]$ and $\bar{S}_2[j_k]$ must satisfy the same pattern, we have $\overline{y_k} = \overline{y'_k}$ for all k, hence $\overline{z_k} = \overline{z'_k}$ for $k \leqslant m-2$. Also, $[y_m, z_m] = [y'_m, z'_m] = [N^2, N^2]$ and $z_{m-1} = z'_{m-1}$. We claim that actually $[y'_k, z'_k] = [y_k, z_k]$ for all k. Suppose not, and let k^* be the smallest $k < m$ such that inequality holds. Then $y'_{k^*} = z'_{k^*-1} + 1 = z_{k^*-1} + 1 = y_{k^*}$ and hence $z'_{k^*} \neq z_{k^*}$. By the construction of the occurrence sets, since both $[y_{k^*}, z_{k^*}]$ and $[y_{k^*}, z'_{k^*}]$ are in $occur_i(\overline{y_{k^*}})$, we must have $\overline{z'_{k^*}} \neq \overline{z_{k^*}}$, a contradiction. The claim follows.

From the above, we have $\delta(\bar{S}_1[j_k], \bar{S}_2[j_k])$ equal to $w(v_{\overline{y_k}}, v_{\overline{y_{k+1}}})$ for $k \leqslant m-2$, equal to $w(v_{\overline{y_{m-1}}}, v_N)$ for $k = m-1$, and equal to 0 if $k = m$, and hence

$$score(\mathscr{A}) = \sum_{j=1}^{m} \delta(\bar{S}_1[j], \bar{S}_2[j]) = \sum_{k=1}^{m-2} w(v_{\overline{y_k}}, v_{\overline{y_{k+1}}}) + w(v_{\overline{y_{m-1}}}, v_N) = len(g(\mathscr{A})).$$

This completes the proof. □

Lemma 11. *Given $\langle G, w \rangle \in I_{\text{SHP}}$ and $\alpha \geqslant 1$, if $sol_{\text{SHP}}(\langle G, w \rangle) \neq \emptyset$, then $opt(f_\alpha(\langle G, w \rangle)) = opt(\langle G, w \rangle)$.*

Proof. Let $\langle v_{\pi(1)}, \ldots, v_{\pi(N)} \rangle$ be an optimal path for $\langle G, w \rangle$, where $\pi(1) = 1$ and $\pi(N) = N$. Let $\mathscr{A} = \{\bar{S}_1, \bar{S}_2\}$ be an optimal solution for $f_\alpha(\langle G, w \rangle)$. Construct $\bar{S}_i{}'$ for $i = 1, 2$ to be such that $|\bar{S}_i{}'| = m$, $\bar{S}_i{}'[k] = [(k-1)N + \pi(k), kN + \pi(k+1) - 1] \in occur_i(\pi(k))$ for $k < m$, and $\bar{S}_i{}'[m] = [N^2, N^2] \in occur_i(\pi(N))$. Let $\mathscr{A}' = \{\bar{S}_1{}', \bar{S}_2{}'\}$, and then we have $score(\mathscr{A}') = \sum_{i=1}^{N-1} w(v_{\pi(i)}, v_{\pi(i+1)})$. Hence $score(\mathscr{A}) \leqslant score(\mathscr{A}') = opt(\langle G, w \rangle)$. By Lemma 10, $score(\mathscr{A}) \geqslant opt(\langle G, w \rangle)$, which completes the proof. □

We can now establish the main result for RSAUC.

Theorem 12. RSAUC *is not approximable within $r(m, n, \sigma)$, for any function $r(m, n, \sigma) \geqslant 1$ computable in polynomial time, even when $\sigma = 2$, unless* P = NP.

Proof. Suppose otherwise. Then there exists a polynomial time algorithm *ALG* that can approximate pairwise RSAUC within some $r(m, n, \sigma)$ computable in polynomial time. Then for any instance $\langle (S_1, S_2), \delta, \mathscr{P}_u \rangle \in I_{\text{RSAUC}}$ whose set of feasible solutions is not empty,

$$score(ALG(\langle (S_1, S_2), \delta, \mathscr{P}_u \rangle)) \leqslant r(m, n, 2) \times opt(\langle (S_1, S_2), \delta, \mathscr{P}_u \rangle),$$

where $n = \max\{|S_1|, |S_2|\}$ and m is the number of patterns defined in $\mathscr{P}_u$.

Let $\langle G, w \rangle$ be an instance of SHP with no Hamiltonian path on G from v_1 to v_N, where N is the number of nodes in G. Let $\mathscr{A} = ALG(f_\beta(\langle G, w \rangle))$, where $\beta = 2r(N, N^2, 2)$. By Lemma 10, if $score(\mathscr{A}) < \beta N w_{\max}$, then $g(\mathscr{A}) \in sol_{\text{SHP}}(\langle G, w \rangle)$. It follows that $score(\mathscr{A}) \geqslant \beta N w_{\max}$, since $sol_{\text{SHP}}(\langle G, w \rangle) = \emptyset$.

Conversely, let $\langle G, w \rangle$ be an instance of SHP with Hamiltonian path on G from v_1 to v_N. Again, let $\mathscr{A} = ALG(f_\beta(\langle G, w \rangle))$. By Lemma 11, $opt(f_\beta(\langle G, w \rangle)) = opt(\langle G, w \rangle)$, and we have $score(\mathscr{A}) \leqslant r(N, N^2, 2) opt(\langle G, w \rangle) < \beta N w_{\max}$.

Hence $ALG(f_\beta(\langle G, w \rangle))$ has score less than $\beta N w_{\max}$ if and only if $\langle G, w \rangle$ has feasible solutions. Since *ALG* and $r(m, n, \sigma)$ are computable in polynomial time, it follows that the determination of whether there is a Hamiltonian path from v_1 to v_N in G can be done in polynomial time. This contradicts Lemma 9 if P $\neq$ NP, hence such an approximation algorithm for RSAUC cannot exist unless P = NP. □

The hardness of general SAUC is then immediate.

Corollary 13. SAUC *cannot be approximated within* $r(m, n, \sigma)$ *for any function* $r(m, n, \sigma)$ *computable in polynomial time*, *unless* P = NP.

Interestingly, unlike CSA, even approximating the pairwise case of SAUC is hard.

In the following, we show that, just as the bounded version of RCSA, the bounded version of RSAUC is also a member of the family of NPO-complete problems.

RSAUC with Bound, RSAUCB

Instance: Two sequences S_1 and S_2, nonnegative integer function δ, an unordered constraint $\mathscr{P}_u$, and a positive integer B.

Solution: A constrained alignment $\mathscr{A} = \{\bar{S}_1, \bar{S}_2\}$ satisfying $\mathscr{P}_u$ with $0 < score(\mathscr{A}) < B$.

Measure: $score(\mathscr{A})$.

Goal: min.

Wu et al. showed that the longest Hamiltonian path problem is NPO-complete [23]; MAX-W3SAT is strictly reduced to the problem. If W3SAT is used instead, then SHP can be shown in the same way to be NPO-complete.

Lemma 14. *The shortest Hamiltonian path problem is* NPO-*complete*.

As before, we specify an AP-reduction from SHP to RSAUCB; the same notations are used for convenience. Define function $h : I_{\text{SHP}} \times (1, \infty) \to I_{\text{RSAUCB}}$ by $h(\langle G, w \rangle, \varepsilon) = \langle f_1(\langle G, w \rangle), B \rangle$, where $B = N w_{\max}$. Also, for a feasible solution $\mathscr{A}$ to $h(\langle G, w \rangle, \varepsilon)$, define $h'(\langle G, w \rangle, \mathscr{A}, \varepsilon) = g(\mathscr{A})$. We then have that $(h, h', 1)$ is a valid AP-reduction, as shown in the following theorem. As before, since our definition of h and h' are actually independent of ε, for notational brevity we shall omit it in the proof.

Theorem 15. RSAUCB *is* NPO-*complete*.

Proof. Let $\langle G, w \rangle \in I_{\text{SHP}}$. It is clear that $h(\langle G, w \rangle)$ can be computed in polynomial time and is an instance of RSAUCB. If $\langle G, w \rangle$ has feasible solutions, then by Lemma 11, $opt(h(\langle G, w \rangle)) = opt(\langle G, w \rangle) \leqslant (N - 1) w_{\max} < N w_{\max} = B$, and hence $h(\langle G, w \rangle)$ also has some feasible solution. Let $\mathscr{A}$ be a feasible solution to $h(\langle G, w \rangle)$. Since $score(\mathscr{A}) < N w_{\max}$, by Lemma 10, we have that $h'(\langle G, w \rangle, \mathscr{A})$, computable in polynomial time, is a feasible solution to $\langle G, w \rangle$. By Lemma 11, $opt(\langle G, w \rangle) = opt(h(\langle G, w \rangle))$, while by Lemma 10, we have $len(g(\mathscr{A})) = score(\mathscr{A})$, hence if any solution $\mathscr{A}$ to $h(\langle G, w \rangle)$ has performance ratio ε, $h'(\langle G, w \rangle, \mathscr{A})$ must also have performance ratio $\varepsilon = 1 + 1 \cdot (\varepsilon - 1)$. Hence $(h, h', 1)$ constitutes an AP-reduction from SHP to RSAUCB. □

The following result is then immediate by Theorem 15.

Proposition 16. *Given an instance of* RSAUCB, *the test of whether it has some feasible solution cannot be done in polynomial time*, *unless* P = NP.

5. Conclusion and discussion

In this paper, we proposed a general model for constrained sequence alignment, and demonstrated the brevity and uniformity of viewing previous formulations in this framework. The hardness results of CSA and SAUC for both pairwise and general cases were proposed. CSA is polynomial time solvable for the pairwise case and not approximable within any polynomial time computable function in general. SAUC is much harder in that it cannot be approximated even in the pairwise case. The bounded versions of CSA and SAUC are shown to be NPO-complete.

For the original formulation, due to the lack of a polynomial time approximation algorithm, it would usually be more feasible to adopt a progressive procedure, as in [20,21,16]. In fact, that the time complexity of the 2-approximation algorithm proposed in [6] is not bounded by a polynomial may be a hint for the hardness of approximation of the original formulations. The results in this paper ensure the inapproximability for the general model, and somehow support the guess of the inapproximability of the original problem. It is expected that the techniques used in this paper help to gain insights into the structure of the original formulation and may be modified to prove the hardness of the original formulation.

Certainly, the hardness of the original formulation remains to be of theoretical interest. Algorithmically, there are also some generalizations worth doing; for example, how to find a good constrained alignment if each satisfaction of the patterns need not be supported by all sequences, or if a proportion of the patterns need not be satisfied. These constitute future directions of research on this topic.

Acknowledgment

The authors would like to thank the anonymous reviewers for their useful comments which help to improve the clarity of this paper.

Appendix A. The class NPO and approximation algorithms

In this appendix, we give definitions regarding the class NPO and approximation algorithms. They are adapted from [8,3].

Definition 17. An *NP* optimization problem A is a fourtuple $(I_A, sol, m, goal)$ such that

(1) I_A is the set of the instances of A and it is recognizable in polynomial time.
(2) Given an instance x of I_A, $sol(x)$ denotes the set of feasible solutions of x. These solutions are short, that is, a polynomial p exists such that, for any $y \in sol(x)$, $|y| \leqslant p(|x|)$. Moreover, it is decidable in polynomial time whether, for any x and for any y such that $|y| \leqslant p(|x|)$, $y \in sol(x)$.
(3) Given an instance x and a feasible solution y of x, $m(x, y)$ denotes the positive integer measure of y. The function m is computable in polynomial time and is also called the objective function.
(4) $goal \in \{\max, \min\}$.

The class NPO is the set of all NP optimization problems. The goal of an NPO problem with respect to an instance$x$ is to find an optimum solution, that is, a feasible solution y such that

$$m(x, y) = goal\{m(x, y') : y' \in sol(x)\}.$$

Let *opt* denote the function mapping an instance x to the measure of an optimum solution.

Definition 18. Let A be an NPO problem. Given an instance x and a feasible solution y of x, we define the performance ratio of y with respect to x as

$$R(x, y) = \max\left\{\frac{m(x, y)}{opt(x)}, \frac{opt(x)}{m(x, y)}\right\}.$$

Definition 19. Let A be an NPO problem and let T be an algorithm that, for any instance x of A, returns a feasible solution $T(x)$ of x. Given an arbitrary function $r : \mathbb{N} \to (1, \infty)$, we say that T is an $r(n)$-approximate algorithm for A if, for any instance x, the performance ratio of the feasible solution $T(x)$ with respect to x verifies the following inequality:

$$R(x, T(x)) \leqslant r(|x|).$$

If an NPO problem admits an $r(n)$-approximate polynomial-time algorithm we say that it is approximable within $r(n)$.

Definition 20. Let A and B be two NPO problems. A is said to be AP-reducible to B, in symbols $A \leqslant_{\mathrm{AP}} B$, if two functions f and g and a positive constant $\alpha \geqslant 1$ exist such that:

(1) For any instance $x \in I_A$ and for any rational $r > 1$, $f(x, r) \in I_B$.
(2) For any instance $x \in I_A$ and for any rational $r > 1$, if $sol_A(x) \neq \emptyset$ then $sol_B(f(x, r)) \neq \emptyset$.
(3) For any instance $x \in I_A$, for any rational $r > 1$, and for any $y \in sol_B(f(x, r))$, $g(x, y, r) \in sol_A(x)$.
(4) Each of f and g is computable by an algorithm, whose running time is polynomial for any fixed rational r.
(5) For any $x \in I_A$, for any rational $r > 1$, and for any $y \in sol_B(f(x, r))$,

$$R_B(f(x, r), y) \leqslant r \quad \text{implies } R_A(x, g(x, y, r)) \leqslant 1 + \alpha(r - 1).$$

Definition 21. A problem $A \in$ NPO is *NPO-complete* if, for any $B \in$ NPO, $B \leqslant_{\mathrm{AP}} A$.

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy, Proof verification and the hardness of approximation problems, J. ACM 45 (1998) 501–555.
[2] A.N. Arslan, Regular expression constrained sequence alignment, in: A. Apostolico, M. Crochemore, K. Park (Eds.), Proceedings of the 16th Annual Symposium on Combinatorial Pattern Matching (CPM 2005), Lecture Notes in Computer Science, vol. 3537, Springer, Berlin, 2005, pp. 322–333.
[3] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, M. Protasi, Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties, Springer, Berlin, 1999.
[4] V. Bafna, E.L. Lawler, P.A Pevzner, Approximation algorithms for multiple sequence alignment, Theoret. Comput. Sci. 182 (1997) 233–244.
[5] P. Bonizzoni, G.D. Vedova, The complexity of multiple sequence alignment with SP-score that is a metric, Theoret. Comput. Sci. 259 (2001) 63–79.
[6] F.Y.L. Chin, N.L. Ho, T.W. Lam, P.W.H. Wong, Efficient constrained multiple sequence alignment with performance guarantee, J. Bioinform. Comput. Biol. 3 (2005) 1–18.
[7] Y.-S. Chung, C.L. Lu, C.Y. Tang, Efficient algorithms for regular expression constrained sequence alignment, in: M. Lewenstein, G. Valiente (Eds.), Proceedings of the 17th Annual Symposium on Combinatorial Pattern Matching (CPM 2006), Lecture Notes in Computer Science, vol. 4009, Springer, Berlin, 2006, pp. 389–400.
[8] P. Crescenzi, V. Kann, A compendium of NP optimization problems, Available at: ⟨http://www.nada.kth.se/∼viggo/wwwcompendium/wwwcompendium.html⟩, 2000.
[9] P.A. Evans, Algorithms and complexity for annotated sequence analysis, Ph.D. Thesis, Department of Computer Science, University of Victoria, Canada, 1999.
[10] S. Fesselea, H. Maierb, C. Zischeka, P.J. Nelsona, T. Werner, Regulatory context is a crucial part of gene function, Trends Genet. 18 (2002) 60–63.
[11] M.R. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman & Company, New York, 1979.
[12] D. Gusfield, Efficient methods for multiple sequence alignment with guaranteed error bounds, Bull. Math. Biol. 30 (1993) 141–154.
[13] T. Jiang, Y. Xu, M.Q. Zhang (Eds.), Current Topics in Computational Molecular Biology, MIT Press, Cambridge, MA, 2002.
[14] W. Just, Computational complexity of multiple sequence alignment with SP-score, J. Comput. Biol. 8 (2001) 615–623.
[15] A. Kel, O. Kel-Margoulis, V. Babenko, E. Wingender, Recognition of NFATp/AP-1 composite elements within genes induced upon the activation of immune cells, J. Mol. Biol. 288 (1999) 353–376.
[16] C.L. Lu, Y.P. Huang, A memory-efficient algorithm for multiple sequence alignment with constraints, Bioinformatics 21 (2005) 20–30.
[17] L.J. McGuffin, K. Bryson, D.T. Jones, The PSIPRED protein structure prediction server, Bioinformatics 16 (2000) 404–405.
[18] G. Myers, S. Selznick, Z. Zhang, W. Miller, Progressive multiple alignment with constraints, J. Comput. Biol. 3 (1996) 563–572.
[19] P. Orponen, H. Mannila, On approximation preserving reductions: complete problems and robust measures, Technical Report C-1987-28, Department of Computer Science, University of Helsinki, 1987.

[20] C.Y. Tang, C.L. Lu, M.D.-T. Chang, Y.-T. Tsai, Y.-J. Sun, K.-M. Chao, J.-M. Chang, Y.-H. Chiou, C.-M. Wu, H.-T. Chang, W.-I. Chou, Constrained sequence alignment tool development and its application to RNase family alignment, J. Bioinform. Comput. Biol. 1 (2003) 267–287.
[21] Y.-T. Tsai, Y.P. Huang, C.T. Yu, C.L. Lu, MuSiC: a tool for multiple sequence alignment with constraints, Bioinformatics 20 (2004) 2309–2311.
[22] L. Wang, T. Jiang, On the complexity of multiple sequence alignment, J. Comput. Biol. 1 (1994) 337–348.
[23] Q.S. Wu, K.-M. Chao, R.C.T. Lee, The NPO-completeness of the longest Hamiltonian cycle problem, Inform. Process. Lett. 65 (1998) 119–123.