Journal of Bioinformatics and Computational Biology
Vol. 3, No. 1 (2005) 1–18
© Imperial College Press

EFFICIENT CONSTRAINED MULTIPLE SEQUENCE ALIGNMENT WITH PERFORMANCE GUARANTEE

FRANCIS Y. L. CHIN*, N. L. HO and T. W. LAM†
Department of Computer Science, The University of Hong Kong
Pokfulum Road, Hong Kong
{*chin,nlho,twlam@cs.hku.hk*}

PRUDENCE W. H. WONG
Department of Computer Science, The University of Liverpool
Liverpool L69 78F, United Kingdom
pwong@csc.liv.ac.uk

Received 12 February 2004
Revised 22 April 2004
Accepted 22 April 2004

The constrained multiple sequence alignment problem is to align a set of sequences of maximum length n subject to a given constrained sequence, which arises from some knowledge of the structure of the sequences. This paper presents new algorithms for this problem, which are more efficient in terms of time and space (memory) than the previous algorithms,[15] and with a worst-case guarantee on the quality of the alignment. Saving the space requirement by a quadratic factor is particularly significant as the previous $O(n^4)$-space algorithm has limited application due to its huge memory requirement. Experiments on real data sets confirm that our new algorithms show improvements in both alignment quality and resource requirements.

Keywords: Multiple sequence alignment; approximation algorithm.

1. Introduction

Multiple sequence alignment (MSA) is one of the problems in computational biology that have been studied extensively.[2,3,6,8,10,13,14] Roughly speaking, given a set of $k \geq 2$ sequences, the MSA problem is to align similar subsequences in the same region. From the computational point of view, the optimal alignment of two sequences can be found in $O(n^2)$ time, where n is the length of the longer sequence. Yet, for three or more sequences, it has been proved that finding the

*This research was supported in part by Hong Kong RGC Grant HKU-7135/04E.
†This research was supported in part by Hong Kong RGC Grant HKU-7042/02E.

optimal alignment is NP-hard,[2,17] i.e., intractable.[a] In the literature, there are a number of MSA algorithms that attempt to approximate the optimal alignments, some of them can provide a worst-case approximation ratio,[1,5,12] while some others work well in practice.[11,16] Notice that with all these algorithms, users (biologists) can only control the alignment results by adjusting parameters like the scoring function and gap penalty. In other words, users could not incorporate their knowledge of the functionalities or structures of the input sequences, which is indeed very useful for accurate and biologically meaningful alignment. This naturally triggers the studies of sequence alignment that allow users to provide additional constraints.

Tang *et al.*[15] were the first to investigate the MSA problem with an additional input of a constrained sequence, which imposes a structure on the alignment by requiring every character in the constrained sequence to appear in an entire column in the alignment of the multiple sequences. As an example, Tang *et al.* considered the alignment of *RNase* sequences. Such sequences are all known to contain three active-site residues `His(H)`, `Lyn(K)`, `His(H)` that are essential for RNA degrading. Therefore, one would expect that in an alignment of RNase sequences, each of these three residues should be aligned in the same column, i.e., an alignment satisfying the constrained sequence "`HKH`".

Tang *et al.*[15] presented the first algorithm for finding an optimal constrained sequence alignment for two sequences; both the time and space (memory) requirements of the algorithm are $O(\alpha n^4)$, where α is the length of the constrained sequence. For aligning $k \geq 3$ sequences, they gave a heuristic algorithm (called progressive alignment algorithm) with time and space requirements being $O(\alpha k n^4)$ and $O(\alpha n^4)$, respectively. When applied to aligning multiple RNase sequences, this algorithm produces satisfactory alignments. Yet the application of the algorithm is limited as the memory requirement is too large and it runs too long. For example, for aligning sequences of length 250 with a constraint of length 3, the memory requirement already exceeds 15 Gigabytes. Nowadays ordinary workstations are equipped with at most 4 Gigabytes.

This paper attempts to improve the results of Tang *et al.* from a theoretical as well as a practical point of view. For pair-wise alignment, we give a new algorithm for finding the optimal constrained alignment that uses $O(\alpha n^2)$ time and $O(\alpha n^2)$ space. Based on this result, we can immediately improve the time and space complexities of the progressive alignment algorithm by a quadratic factor. Furthermore, we give an algorithm, called center-star, for constrained multiple sequence alignment with worst-case performance guarantee; more precisely, for aligning k sequences, the new algorithm can produce an alignment that approximates the optimal alignment within a factor of $\left(2 - \frac{2}{k}\right)$. This algorithm adopts the framework of Gusfield's (unconstrained) multiple sequence alignment algorithm.[5] The time and

[a] There are several possible ways to define the optimal alignment. In this paper we adopt the widely-used *Sum-of-Pair* (*SP*) score, which asks for an alignment that minimizes the sum of the alignment cost of all pairs of sequences.

Table 1. Performance of constrained multiple sequence alignment approximation algorithms on k sequences of maximum length n, with a constrained sequence P of length α. C is the total number of occurrences of the constrained sequence P in all sequences.

	Time Complexity	Space Complexity	Approximation Ratio
Tang *et al.*'s algorithm[15]	$O(\alpha k n^4)$	$O(\alpha n^4)$	—
Improved Tang *et al.*'s algorithm (this paper)	$O(\alpha k^2 n^2)$	$O(\alpha n^2)$	—
Center-star (this paper)	$O(Ckn^2)$	$O(n^2)$	$2 - \frac{2}{k}$

Table 2. Alignment scores of CMSA algorithms.

	Tang's Algorithm			Center-star Algorithm		
	Score	Time (second)	Space (MB)	Score	Time (second)	Space (MB)
7 sequences max length 125 $\alpha = 3$	46319	127	425	40051	12	1.9
6 sequences max length 185 $\alpha = 3$	71208	381	1192	49875	46	2.0
6 sequences max length 186 $\alpha = 4$	63315	254	654	45241	54	2.0
5 sequences max length 327 $\alpha = 3$	— Memory exhausted —			57325	208	2.3

space complexities of the new algorithm are respectively $O(Ckn^2)$ and $O(n^2)$, where C is the total number of occurrences of the constrained sequence in all sequences. The improved memory requirement allows us to handle sequences with thousands of characters on ordinary workstations. See Table 1 for a summary of these results.

We have implemented all the algorithms mentioned above and tested them with several real data sets. In all data sets, the center-star algorithm shows improvement in all aspects. In particular, the quality of the alignment is 13% to 30% better, while the memory requirement is at most one-percent of Tang *et al.*'s algorithm. Results are briefly summarized in Table 2. More details will be given in Sec. 6.

The rest of this paper is organized as follows. Section 2 defines the constrained sequence alignment problem, and Sec. 3 presents a new algorithm that computes the optimal constrained pair-wise sequence alignment. Section 4 presents algorithms for *constrained multiple sequence alignment* (CMSA). In Sec. 5, an approximation algorithm is given with an approximation ratio $\left(2 - \frac{2}{k}\right)$. We report empirical results of our developed CMSA tools in Sec. 6. Finally, in Sec. 7, we conclude this paper by giving some further research directions in CMSA.

2. Preliminaries

Let Σ be the set of characters (residues), $S = \{S_1, S_2, \ldots, S_k\}$ be a set of k sequences, with maximum length n, over Σ. Let $S_i[x \ldots y]$ denote the substring of S_i from the xth character to the yth character of S_i, where $1 \leq x \leq y \leq n$. In particular, let $S_i[x]$ denote the xth character in the sequence S_i.

We define the *pair-wise sequence alignment* of two sequences S_1 and S_2 as two equal-length sequences S'_1 and S'_2 such that $|S'_1| = |S'_2| = n'$, and removing all space characters "$-$" from S'_1 and S'_2 gives S_1 and S_2, respectively. For a given distance function $\delta(x, y)$ which measures the *mutation distance* between two characters, where $x, y \in \Sigma \cup \{-\}$, the *pair-wise score* of two length-n' sequences S'_1 and S'_2 is defined as $\sum_{1 \leq x \leq n'} \delta(S'_1[x], S'_2[x])$. In the multiple sequence alignment (MSA) problem, we are given k sequences $S = \{S_1, S_2, \ldots, S_k\}$, an MSA is an *alignment matrix* A, with k rows and $n'(\geq n)$ columns, such that removing space characters from the ith row of A gives S_i for $1 \leq i \leq k$. We denote by $A[x, y]$ the character at the xth row and yth column of A. The *sum-of-pair* (*SP*) score of an MSA A is defined as the sum of the pair-wise scores of all pairs of the sequences, i.e. $\sum_{1 \leq p < q \leq k} \sum_{1 \leq y \leq n'} \delta(A[p, y], A[q, y])$. It has been shown that finding the alignment matrix with the minimum sum-of-pair alignment score is NP-complete.[2,17]

In the *constrained multiple sequence alignment problem* (*CMSA*), we are given, in addition to the inputs of the MSA problem, a constrained sequence $P = (P[1], P[2], \ldots, P[\alpha])$, where P is a common subsequence of all $S_i \in \{S_1, S_2, \ldots, S_k\}$. The solution of a CMSA problem is a *constrained alignment matrix* A which is an alignment matrix such that each character in P appears in an entire column of A and also in the same order, i.e. there exists a list of integers $\{r_1, r_2, \ldots, r_\alpha\}$ where $1 \leq r_1 < r_2 < \cdots < r_\alpha \leq n'$, such that for all $1 \leq i \leq k$ and all $1 \leq \gamma \leq \alpha$, we have $A[i, r_\gamma] = P[\gamma]$.

Consider a constrained alignment matrix A for $S = \{S_1, S_2, \ldots, S_k\}$ and the constrained sequence P. Denote by *sp_score*(A) the SP score of the constrained alignment matrix A. Let A^*_S be the optimal constrained alignment matrix for S and A'_S be the constrained alignment matrix derived by some approximation algorithm. The approximation algorithm is said to have an approximation ratio ϕ if $\frac{sp\_score(A'_S)}{sp\_score(A^*_S)} \leq \phi$ for all S and P.

3. Constrained Pair-Wise Sequence Alignment (CPSA)

3.1. *Problem definition*

The *constrained pair-wise sequence alignment* (*CPSA*) problem is a special case for CMSA problem with $k = 2$. Given two sequences S_1 and S_2, a constrained sequence P (with length α) and a distance function δ, the problem is to compute an optimal CPSA, $\binom{S'_1}{S'_2}$, such that $|S'_1| = |S'_2| = |n'|$, and $\sum_{1 \leq i \leq n'} \delta(S'_1[i], S'_2[i])$ is minimized subject to $P[\gamma] = S'_1[r_\gamma] = S'_2[r_\gamma]$ for some r_γ, $1 \leq \gamma \leq \alpha$ and $1 \leq r_1 < r_2 < \cdots < r_\alpha \leq n'$. Note that removing all spaces in S'_1 and S'_2 gives S_1 and S_2 respectively.

3.2. *Optimal constrained pair-wise sequence alignment*

Tang *et al.*[15] presented an algorithm to compute the optimal constrained pair-wise alignment with time and space complexities $O(\alpha n^4)$ and $O(\alpha n^4)$, respectively. This algorithm first computes the $O(n^4)$ pair-wise alignment scores of all substrings in S_1 and all substrings in S_2 and then further determines the best positions such that the constrained characters are aligned. The overall time complexity is $O(\alpha n^4)$. To improve the time complexity, our algorithm takes into consideration the constrained alignment as we compute the alignment score. This approach makes it not necessary to consider all the pair-wise alignments between every pair of substrings of S_1 and S_2, and thus, facilitates the reduction in the time complexity.

Below we show how to compute the sum-of-pair score of the optimal CPSA by dynamic programming and how to obtain the alignment by backtracking through the path of computation of the score. Recall that for any sequence S, $S[x..y]$ denotes the substring of S starting at the xth character and ending at the yth character of S. The dynamic programming computes the optimal CPSA incrementally by considering the pair-wise alignments of $S_1[1\ldots1]$ with $S_2[1\ldots1], \ldots, S_1[1\ldots i]$ with $S_2[1\ldots j]$ and so on, for $1 \le i \le |S_1|$ and $1 \le j \le |S_2|$.

We denote by $D(i,j,\gamma)$ the optimal constrained pair-wise sequence alignment score of sequences $S_1[1..i]$ and $S_2[1..j]$ with constrained characters $P[1..\gamma]$ matched. In particular, $D(n_1,n_2,\alpha)$ is the optimal CPSA score of S_1 and S_2 with respect to the constrained sequence P, where $|S_1| = n_1$, $|S_2| = n_2$ and $|P| = \alpha$. The following theorem gives a recurrence formula for $D(i,j,\gamma)$ in terms of $D(i',j',\gamma')$ where $i' \le i$, $j' \le j$, and $\gamma' \le \gamma$.

Theorem 1. *For any* $0 \le i \le n_1, 0 \le j \le n_2$ *and* $0 \le \gamma \le \alpha$,

$$D(i,j,\gamma) = \min \begin{cases} D(i-1,j-1,\gamma-1) + \delta(S_1[i],S_2[j]) & \text{if } S_1[i] = S_2[j] = P[\gamma], \\ D(i-1,j-1,\gamma) + \delta(S_1[i],S_2[j]) & \text{if } i,j > 0, \\ D(i-1,j,\gamma) + \delta(S_1[i],-) & \text{if } i > 0, \\ D(i,j-1,\gamma) + \delta(-,S_2[j]) & \text{if } j > 0, \end{cases}$$

with boundary conditions $D(0,0,0) = 0, D(i,0,\gamma) = \infty$ *for* $\gamma \ge 1, 0 \le i \le n_1$, *and* $D(0,j,\gamma) = \infty$ *for* $\gamma \ge 1, 0 \le j \le n_2$.

Proof. There are four cases to align $\{S_1[1..i], S_2[1..j]\}$ with respect to $P[1..\gamma]$,

- If $S_1[i] = S_2[j] = P[\gamma]$, we can align $S_1[i]$ and $S_2[j]$ while aligning $\{S_1[1..(i-1)], S_2[1..(j-1)]\}$ with $P[1..(\gamma-1)]$ matched. Then, the score is $D(i-1,j-1,\gamma-1) + \delta(S[i],S[j])$.
- If $i,j > 0$, we can align $S_1[i]$ and $S_2[j]$ while aligning $\{S_1[1..(i-1)], S_2[1..(j-1)]\}$ with $P[1..\gamma]$ matched. Then the score is $D(i-1,j-1,\gamma) + \delta(S[i],S[j])$.
- If $i > 0$, we can align $S_1[i]$ with a space while aligning $\{S_1[1..(i-1)], S_2[1..j]\}$ with $P[1..\gamma]$ matched. Then the score is $D(i-1,j,\gamma) + \delta(S[i],-)$.
- If $j > 0$, we can align $S_2[j]$ with a space while aligning $\{S_1[1..i], S_2[1..(j-1)]\}$ with $P[1..\gamma]$ matched. Then the score is $D(i,j-1,\gamma) + \delta(-,S[j])$.

The alignment to be chosen is the one such that the new score is minimum. Therefore, $D(i, j, \gamma)$ can be computed by taking the minimum of the above four values.

□

Based on Theorem 1, we can compute the sum-of-pair score of the optimal CPSA using dynamic programming (see Algorithm 1). After filling in the three dimensional table $D(i, j, \gamma)$, we can obtain the CPSA by backtracking through the computation path from $D(n_1, n_2, \alpha)$ to $D(0, 0, 0)$. Let S'_1 and S'_2 be the aligned sequence for S_1 and S_2, respectively. Initially, set S'_1 and S'_2 to two empty strings, and start backtracking from $D(n_1, n_2, \alpha)$. If $D(i, j, \gamma)$ is computed from $D(i-1, j-1, \gamma)$ or $D(i-1, j-1, \gamma-1)$, prepend $S_1[i]$ and $S_2[j]$ to S'_1 and S'_2, respectively. If $D(i, j, \gamma)$ is computed from $D(i-1, j, \gamma)$, prepend $S_1[i]$ and a space to S'_1 and S'_2, respectively; similarly, if $D(i, j, \gamma)$ is computed from $D(i, j-1, \gamma)$, prepend a space and $S_2[j]$ to S'_1 and S'_2, respectively. Repeat backtracking until reaching $D(0, 0, 0)$, and $\binom{S'_1}{S'_2}$ is the optimal constrained sequence alignment of $\{S_1, S_2\}$ with P matched.

Theorem 2. *The optimal constrained pair-wise alignment can be computed in both $O(\alpha n_1 n_2)$ time and space, where $|S_1| = n_1$, $|S_2| = n_2$ and $|P| = \alpha$.*

Proof. The 3-dimensional table D is of size $((n_1 + 1) \times (n_2 + 1) \times (\alpha + 1))$. The computation of each entry $D(i, j, \gamma)$ needs only the values of $D(i-1, j-1, \gamma-1)$, $D(i-1, j-1, \gamma)$, $D(i-1, j, \gamma)$ and $D(i, j-1, \gamma)$, thus, each $D(i, j, \gamma)$ can be computed in constant time. Therefore, the optimal CPSA score of two sequences of lengths n_1 and n_2 and constrained sequence P of length α can be computed in $O(\alpha n_1 n_2)$ time. On the other hand, in each step of the backtracking from $D(n_1, n_2, \alpha)$ to $D(0, 0, 0)$, at least one of the indices i, j or γ is decreased by one. Thus, there are at most $(\alpha + n_1 + n_2)$ steps. Notice that each step takes constant time. Therefore, the backtracking takes $O(\alpha + n_1 + n_2)$ time. Thus, the whole algorithm takes $O(\alpha n_1 n_2)$ time. Table D has $O(\alpha n_1 n_2)$ entries; each entry $D(i, j, \gamma)$ requires constant amount

```
begin
  // Initialization:
  Initialize $D(0,0,0)$, $D(i,0,\gamma)$, and $D(0,j,\gamma)$, for $0 \le i \le n_1$, $0 \le j \le n_2$,
  and $1 \le \gamma \le \alpha$, according to Theorem 1.
  // Dynamic Programming:
  for $\gamma = 0$ to $\alpha$ do
    for $i = 0$ to $n_1$ do
      for $j = 0$ to $n_2$ do
        If $D(i,j,\gamma)$ is not initialized, compute $D(i,j,\gamma)$ according to
        Theorem 1 in terms of $D(i-1,j-1,\gamma-1)$, $D(i-1,j-1,\gamma)$,
        $D(i-1,j,\gamma)$ and $D(i,j-1,\gamma)$.
end
```

Algorithm 1: Dynamic programming algorithm for the optimal CPSA score.

of space (for storing the alignment score and the direction of the computation path). Therefore, the algorithm requires $O(\alpha n_1 n_2)$ space. Thus, the theorem follows. □

Remarks: It is easy to see that computation of scores for row i (column j, respectively) only depends on that for row $i-1$ (column $j-1$, respectively), hence, the optimal score can be computed in $O(\alpha \min(n_1, n_2))$ space. By exploiting the idea of Hirschberg's algorithm,[7] we can recover the alignment in $O(\alpha \min(n_1, n_2))$ space while maintaining the time complexity.

4. Constrained Multiple Sequence Alignment (CMSA)[15]

In this section, we study the constrained multiple sequence alignment problem. In Sec. 4.1, we reduce the time and space complexities of the progressive CMSA heuristic algorithm[15] presented by Tang *et al.* from $O(\alpha k n^4)$ to $O(\alpha k^2 n^2)$. In Sec. 4.2, we show an algorithm that computes the optimal CMSA in $O(\alpha n^k)$ time using the sum-of-pair score.

4.1. *Improved progressive CMSA algorithm*

Tang *et al.*[15] presented an $O(\alpha k n^4)$ time and $O(\alpha n^4)$ space progressive heuristic algorithm for the CMSA problem for a set of k sequences of length at most n and a constrained sequence P of length α. In Tang *et al.*'s algorithm, a $k \times k$ distance matrix of the k sequences is constructed, where the (i, j) entry represents the pair-wise sequence alignment score of S_i and S_j (note that this alignment score does not consider the constrained sequence P). A minimum spanning tree (MST) is then constructed using the *Kruskal algorithm*[4] based on the distance matrix of these sequences. Sequences are then progressively aligned using the CPSA algorithm in the order of the construction of MST. This algorithm performs exactly $(k-1)$ constrained pair-wise sequence alignments. By using the constrained pair-wise alignment algorithm described in Sec. 3.2, the time and space complexities can be improved from $O(\alpha k n^4)$ and $O(\alpha n^4)$ to $O(\alpha k^2 n^2)$ and $O(\alpha n^2)$, respectively.

4.2. *An algorithm for the optimal CMSA*

In this section, we extend the optimal CPSA algorithm described in Sec. 3.2 to k sequences. This involves the construction of a $(k+1)$-dimensional matrix D, which takes $O(\alpha n^k)$ time and space. More precisely, let the multi-dimensional array $D(i_1, i_2, \ldots, i_k; \gamma)$ be the optimal CMSA score matrix for $\{S_1[1..i_1], S_2[1..i_2], \ldots, S_k[1..i_k]\}$ with $P[1..\gamma]$ aligned in γ columns. Then the optimal alignment score for $\{S_1 \ldots S_k\}$ with respect to the constrained sequence P is given by $D(n_1, n_2, \ldots, n_k; \alpha)$, where $n_i = |S_i|$ for $1 \leq i \leq k$. $D(i_1, i_2, \ldots, i_k, \gamma)$ can be computed by the following recurrence,

(i) $D(\{0\}^k; 0) = 0, D(i_1, i_2, \ldots, i_k; \gamma) = \infty$ if $\gamma \geq 1$ and some but not all of i_j's equal zero;

(ii) $D(i_1, i_2, \ldots, i_k; \gamma) =$

$$\min \begin{cases} D(i_1 - 1, i_2 - 1, \ldots, i_k - 1; \gamma - 1) + \delta(S_1[i_1], S_2[i_2], \ldots, S_k[i_k]) \\ \quad \text{if } S_1[i_1] = S_2[i_2] = \cdots = S_k[i_k] = P[\gamma], \\ \\ \min\limits_{\epsilon \in \{0,1\}^k - \{0\}^k} \begin{Bmatrix} D(i_1 - \epsilon_1, i_2 - \epsilon_2, \ldots, i_k - \epsilon_k; \gamma) \\ + \, \delta(\epsilon_1 S_1[i_1], \epsilon_2 S_2[i_2], \ldots, \epsilon_k S_k[i_k]) \end{Bmatrix} \end{cases}$$

where $\epsilon_j = 0$ or 1, $\epsilon_j S_j[i_j]$ with $\epsilon_j = 0$ representing a space character, and $\delta(x_1, \ldots, x_k) = \sum_{1 \leq i < j \leq k} \delta(x_i, x_j)$.

Based on the above recurrence, we have a dynamic programming formulation that computes the optimal CMSA for multiple sequences, which is a generalization of the dynamic programming formulation for the CPSA problem. Practically, Lipman *et al.*[9] noted that the optimal CMSA can be computed for less than 6 short sequences of length at most 200. In the following section, we present an approximation algorithm for the CMSA problem.

5. An Approximation Algorithm for CMSA

Gusfield[5] showed that the center-star algorithm (the idea of center-star originates from Wong[18]) has an approximation ratio $\left(2 - \frac{2}{k}\right)$ for the unconstrained multiple sequence alignment problem. Based on the center-star approximation algorithm,we derive an approximation algorithm for CMSA that yields an approximation ratio $\left(2 - \frac{2}{k}\right)$. This algorithm runs in $O(Ckn^2)$ time, where C is the total number of occurrences of the constrained sequence P in all sequences. Throughout this section, we assume that the distance function $\delta(x, y)$ follows the *triangle inequality*, i.e. $\delta(x, y) \leq \delta(x, z) + \delta(z, y)$, for any $x, y, z \in \Sigma \cup \{-\}$, and $\delta(-, -) = 0$.

5.1. *The center-star alignment approximation for CMSA*

For a set of k sequences $S = \{S_1 \cdots S_k\}$, the *center sequence* $S_c \in S$ is the sequence such that the sum of constrained pair-wise alignment scores to the other $(k - 1)$ sequences is minimized, with the additional constraint that P must appear in the same list of positions of S_c in every constrained pair-wise alignment of S_c with S_j, where $j \neq c$. The *star-sum score* of a CMSA with respect to a center sequence S_c is the sum of pair-wise score of S_c with all $S_j \in S - \{S_c\}$. The constrained center-star approximation algorithm is to find the CMSA and its center sequence S_c such that the star-sum score with respect to S_c is minimized. The algorithm can be summarized below,

(i) For each S_i in S, treat S_i as the center sequence and for each list of positions $(r_1, r_2, \ldots, r_\alpha)$ that S_i is aligned with P, i.e. $P[\gamma] = S_i[r_\gamma] \; \forall \, 1 \leq \gamma \leq \alpha$, align all other S_j with S_i at the positions specified by $(r_1, r_2, \ldots, r_\alpha)$.

(ii) Find the S_c and $(r_1, r_2, \ldots, r_\alpha)$ with the minimum star-sum score.

(iii) Merge the $(k-1)$ constrained pair-wise sequence alignments between S_c and other S_j under the positions $(r_1, \dots, r_\alpha)$ into a constrained alignment matrix.

We elaborate on Steps (i) to (iii) of the above algorithm in the discussion below.

(i) *Aligning a candidate center sequence with another sequence under a list of constrained positions* $(r_1, \dots, r_\alpha)$ — WLOG, assume that the candidate center sequence is S_1. Given $r_1, r_2, \dots, r_\alpha$, we perform the CPSA algorithm on S_1 with $S_2 \cdots S_k$ under $(r_1, r_2, \dots, r_\alpha)$, using a slightly modified recurrence in Theorem 1 treating S_c as S_1 and S_p as S_2 $(2 \leq p \leq k)$. Below, $D(i, j, \gamma)$ denotes the optimal constrained pair-wise sequence alignment score of sequences $S_1[1..i]$ and $S_2[1..j]$ with constrained characters $P[1..\gamma]$ matched in positions $r_1, \dots, r_\gamma$, respectively.

$$D(i,j,\gamma) = \min \begin{cases} D(i-1, j-1, \gamma-1) + \delta(S_1[i], S_2[j]) & \\ \qquad \text{if } r_\gamma = I, S_1[i] = S_2[j] = P[\gamma], & \\ D(i-1, j-1, \gamma) + \delta(S_1[i], S_2[j]) & \text{if } i, j > 0, \\ \qquad r_\gamma = I \Rightarrow r_\gamma = i & \\ D(i-1, j, \gamma) + \delta(S_1[i], -) & \text{if } i > 0, \\ D(i, j-1, \gamma) + \delta(-, S_2[j]) & \text{if } j > 0, \end{cases} \qquad (1)$$

with boundary conditions $D(0,0,0) = 0, D(i,0,\gamma) = \infty$ for $\gamma \geq 1, 0 \leq i \leq n_1$, and $D(0,j,\gamma) = \infty$ for $\gamma \geq 1, 0 \leq j \leq n_2$.

Suppose $n_1 = |S_1|$ and $n_2 = |S_2|$. Notice that using the above recurrence, for any $1 \leq \gamma \leq \alpha$, the computation path from $D(n_1, n_2, \alpha)$ to $D(0,0,0)$ must pass through some points $D(r_\gamma, j, \gamma)$ with $1 \leq j \leq n_2$. This implies that the alignment occurs in S_1 at the positions $(r_1, r_2, \dots, r_\alpha)$.

(ii) *Finding the optimal center sequence and the constrained positions* — Consider each sequence S_i and a list of positions of occurrence $(r_1, r_2, \dots, r_\alpha)$ of P in S_i where $1 \leq i \leq k$. The combination $(S_i; r_1, r_2, \dots, r_\alpha)$ that gives the minimum sum of constrained pair-wise alignment scores with other sequences under the positions $(r_1, r_2, \dots, r_\alpha)$ is selected as the center sequence S_c and the list of positions to be aligned with P.

(iii) *Merging the* $(k-1)$ *constrained pair-wise alignments* — Based on the optimal center sequence, we construct a CMSA, denoted by A. Suppose the center sequence of S is S_1 under a list of positions $(r_1, r_2, \dots, r_\alpha)$. There are $(k-1)$ constrained pair-wise alignments, one for S_1 aligning with each S_j, for $2 \leq j \leq k$. Suppose $|S_1| = n$, and let A_j be the optimal constrained pair-wise alignment of S_1 and S_j under $(r_1, r_2, \dots, r_\alpha)$. Define s_0 and s_n be the longest sequences of spaces inserted before $S_1[1]$ and after $S_1[n]$ in all $(k-1)$ A_j's, respectively. Similarly for $1 \leq i \leq n-1$, let s_i be the longest sequence of spaces between $S_1[i]$ and $S_1[i+1]$ in all $(k-1)$ A_j's. Initially, set A to contain a single row $S'_1 = [s_0 \oplus S_1[1] \oplus s_1 \oplus S_1[2] \oplus \cdots \oplus s_{i-1} \oplus S_1[i] \oplus s_i \oplus \cdots \oplus S_1[n] \oplus s_n]$, where the $\oplus$ operator denotes the string concatenation operation. Note that $|S'_1| = |S_1| + \sum_{0 \leq i \leq n} |s_i|$.

For each S_j with $2 \leq j \leq k$, add S_j to A according to the optimal constrained pair-wise sequence alignment $\binom{\bar{S}_1}{\bar{S}_j}$ of S_1 and S_j, i.e. insert columns of spaces to $\binom{\bar{S}_1}{\bar{S}_j}$ until $\bar{S}_1$ is identical to S'_1.

Notice that the insertion of spaces during the construction of A does not change the pair-wise alignment score of S_1 with each of the other sequences with respect to P under the positions $(r_1, r_2, \ldots, r_\alpha)$. Therefore, A is the constrained multiple sequence alignment for $S = \{S_1, \ldots, S_k\}$ and P with the minimum star-sum score.

5.2. *Complexities of the center-star algorithm for CMSA*

Based on the recurrence equation presented in Eq. (1), the optimal constrained pair-wise alignment of S_1 with S_2, under a set of α positions $\{r_1, \ldots, r_\alpha\}$ matching P takes time and space $O(\alpha n^2)$. This can be further reduced to $O(n^2)$ time and space.

Lemma 1. *Consider two sequences S_1 and S_2, and a constrained sequence P with $|S_1| = n_1$, $|S_2| = n_2$ and $|P| = \alpha$. Given a list of positions $r = (r_1, \ldots, r_\alpha)$ such that $S_1[r_\gamma]$ is to match $P[\gamma]$, for all $1 \leq \gamma \leq \alpha$, it suffices to compute only $O(n_1 n_2)$ entries in the matrix $D(i, j, \gamma)$ to obtain $D(n_1, n_2, \alpha)$.*

Proof. Based on Eq. (1), we have the following three observations about the computation of $D(i, j, \gamma)$. Consider any $0 \leq \gamma \leq \alpha$, and any $1 \leq j \leq n_2$. (i) When $i < r_\gamma$, $D(i, j, \gamma) = \infty$. (ii) When $i = r_\gamma$, $D(i, j, \gamma)$ is computed from $D(i-1, j-1, \gamma-1)$. (iii) When $i > r_\gamma$, $D(i, j, \gamma)$ is computed from $D(i', j', \gamma)$ for some $i' \leq i$ and $j' \leq j$. As a result, we can see that for any $0 \leq \gamma \leq \alpha$, we only need to compute the entries $D(i, j, \gamma)$ for all $r_\gamma \leq i \leq r_{\gamma+1} - 1$ and all $1 \leq j \leq n_2$. (We set $r_0 = 0$ and $r_{\alpha+1} = n_1 + 1$ for the boundary case.) Therefore, the total number of entries that we need to compute equals to $\sum_{0 \leq \gamma \leq \alpha+1} (r_{\gamma+1} - r_\gamma)\, n_2 = O(n_1 n_2)$. □

Based on Lemma 1, we can compute the optimal constrained alignment for S_1 and S_2 with P matched in a particular sequence of positions as shown in Algorithm 2. In Step (i), we need to compute $(k-1)$ pair-wise alignments for each combination $(S; r_1, \ldots, r_\alpha)$ and there are C such combinations. Therefore, by Lemma 1, Step (i) can be computed in $O(Ckn^2)$ time. To find the best combination, we only need to keep track of the current best one as we consider a new combination. Therefore, Steps (i) and (ii) together takes $O(Ckn^2)$ time and $O(n^2)$ space. For Step (iii), notice that the resulting sequence S'_1 has length at most kn. If we store the whole resulting alignment matrix, we need $O(k^2 n)$ space and the merging would take $O(k^2 n)$ time. However, we can simply store the locations of the non-space characters of the sequences in the alignment matrix. Then the merging can be done incrementally by adding each sequence in turn using $O(n)$ time, i.e. in total the merging takes $O(kn)$ time. The space requirement can also be reduced to $O(kn)$. As a result, the overall time and space complexities of the center-star algorithm is $O(Ckn^2)$ and $O(n^2)$, respectively.

```
begin
  // Initialization:
  Initialize $D(0,0,0)$, $D(i,0,\gamma)$, and $D(0,j,\gamma)$, for $0 \le i \le n_1$, $0 \le j \le n_2$,
  and $1 \le \gamma \le \alpha$, according to Equation 1.
  // Dynamic Programming: set $r_0 = 0, r_{\alpha+1} = n_1$
  for $\gamma = 0$ to $\alpha$ do
    for $i = r_\gamma$ to $r_{\gamma+1}$ do
      for $j = 0$ to $n_2$ do
        If $D(i,j,\gamma)$ is not initialized, compute $D(i,j,\gamma)$ according to
        Equation 1 in terms of $D(i-1,j-1,\gamma-1)$, $D(i-1,j-1,\gamma)$,
        $D(i-1,j,\gamma)$ and $D(i,j-1,\gamma)$.
end
```

Algorithm 2: The dynamic programming of Center-star alignment (Step (i)).

5.3. *Performance of the center-star algorithm for CMSA*

The following theorem shows that the center-star approximation algorithm for CMSA has an approximation ratio $\left(2 - \frac{2}{k}\right)$. Define the distance $\Delta(S'_i, S'_j)$ of two aligned sequences S'_i and S'_j as the sum of pair-wise distances between the two characters at the same positions in S'_i and S'_j, i.e., $\Delta(S'_i, S'_j) = \sum_{1 \le p \le |S'_i|} \delta(S'_i[p], S'_j[p])$.

Theorem 3. *Given $S = \{S_1, \ldots, S_k\}$ and a constrained sequence P. Suppose A^s is the alignment output by the constrained center-star algorithm, and A^* be the optimal constrained alignment with respective to P. Then, $\frac{sp\_score(A^s)}{sp\_score(A^*)} \le 2 - \frac{2}{k}$.*

Proof. For any alignment matrix A, let A_i be the ith row of A, for $1 \le i \le k$, and $ss\_score_i(A)$ be the star-sum score of A with A_i as the center sequence. Let A_c be the optimal center sequence of A. Since A^* is a CMSA for S and P, $sp\_score(A^*) = \sum_{1 \le i < j \le k} \Delta(A^*_i, A^*_j)$. Then,

$$\begin{aligned}
sp\_score(A^*) &= \frac{1}{2} \sum_{1 \le i,j \le k, i \ne j} \Delta(A^*_i, A^*_j) \\
&= \frac{1}{2} \sum_{1 \le i \le k} \left(\sum_{1 \le j \le k, i \ne j} \Delta(A^*_i, A^*_j) \right) \\
&= \frac{1}{2} \sum_{1 \le i \le k} (ss\_score_i(A^*)) \\
&\ge \frac{k}{2} \min_{1 \le i \le k} (ss\_score_i(A^*)) \\
&\ge \frac{k}{2} (ss\_score_c(A^s)).
\end{aligned}$$

On the other hand,

$$\begin{aligned}
sp\_score(A^s) &= \frac{1}{2} \sum_{1\le i,j\le k, i\ne j} \Delta\big(A_i^s, A_j^s\big) \\
&\le \frac{1}{2} \sum_{1\le i,j\le k, i\ne j, i\ne c, j\ne c} \big(\Delta(A_i^s, A_c^s) + \Delta(A_c^s, A_j^s)\big) \\
&\quad + \frac{1}{2}\left(\sum_{1\le j\le k, j\ne c} \Delta(A_c^s, A_j^s) + \sum_{1\le i\le k, i\ne c} \Delta(A_i^s, A_c^s)\right) \\
&= \frac{2(k-1)}{2} \sum_{1\le i\le k, i\ne c} \Delta(A_i^s, A_c^s) \\
&= (k-1)(ss\_score_c(A^s)).
\end{aligned}$$

Note that the inequality above (i.e., lines 1 and 2) is due to the triangle inequality. Therefore, $\frac{sp\_score(A^s)}{sp\_score(A^*)} \le \frac{2(k-1)}{k} = 2 - \frac{2}{k}$, and the theorem follows. □

6. Empirical Results

In Sec. 6.1, we evaluate the performance of our CPSA and CMSA algorithms using four data sets of RNase sequences taken from the NCBI.[b] In Sec. 6.2, we show that, in practice, C, the total number of occurrences of the constrained sequence as a subsequence in all sequences, is much smaller than n^2.

6.1. *Experiments on CPSA and CMSA algorithms*

All our experiments are conducted on an Intel workstation with 2.0 GHz CPU and 4 GB main memory. First, we use CPSA to align two RNase sequences (with lengths about 150) with three constrained characters ($\alpha = 3$), using our CPSA algorithm (Sec. 3) and Tang *et al.*'s constrained pair-wise sequence algorithm.[15] Tang *et al.*'s CPSA algorithm took 127 seconds and 400 MB memory. For the same problem instance, our CPSA algorithm ran within a fraction of second with only 1.5 MB of memory space. This shows the practicality of our CPSA algorithm especially for long sequences and long constrained sequence.

To evaluate the performance of different CMSA algorithms, we implemented the original Tang *et al.*'s progressive CMSA algorithm,[15] the improved progressive CMSA algorithm, and the constrained center-star algorithm. We ran these CMSA algorithms on four sets of RNase sequences, summarized in Table 3. In data set 0, we used the same set of 7 RNase sequences used by Tang *et al.*[15] We obtained 3 other data sets of RNase sequences from the NCBI; data sets 1 and 2 contain 6 RNase sequences of lengths about 180, and data set 3 contains 5 long RNase sequences (with maximum length 327). Since we cast the CMSA as a minimization

[b] *National Center of Biotechnology Information*, URL: `http://www.ncbi.nlm.nih.gov`.

Table 3. Sequences used in data sets 0 to 3.

<table>
<tr><th></th><th>Data Set 0</th><th>Data Set 1</th><th>Data Set 2</th><th>Data Set 3</th></tr>
<tr><td>Num Seq</td><td>7</td><td>6</td><td>6</td><td>5</td></tr>
<tr><td>Max seq len</td><td>125</td><td>185</td><td>186</td><td>328</td></tr>
<tr><td>Constrained Sequence</td><td>HKH</td><td>HKH</td><td>HKSH</td><td>HKH</td></tr>
<tr><td>Sequence ID</td><td>H-RNase3
H-RNase2
BP-RNaseA
BS-RNase
H-RNaseA
H-RNase4
RC-RNase</td><td>gi|119124|sp|P12724|ecp_human
gi|2500564|sp|P70709|ecp_rat
gi|13400006|pdb|1dyt|
gi|20930966|ref|xp_142859.1|
gi|20873960|ref|xp_127690.1|
gi|20930966|ref|xp_142859.1|</td><td>gi|20930966|ref|XP_142859.1|
gi|119124|sp|P12724|ECP_HUMAN
gi|2500564|sp|P70709|ECP_RAT
gi|13400006|pdb
gi|20930966|ref|XP_142859.1|
gi|20873960|ref|XP_127690.1|</td><td>gi|10068295|gb|AAE40716.1|
gi|17549935|ref|NP_510780.1|
gi|28509297|ref|XP_282983.1|
gi|28499937|ref|XP_204162.2|
gi|4902995|dbj|BAA77929.1|</td></tr>
</table>

Table 4. Alignment scores of CMSA algorithms.

	Tang's Alg.			Improved Tang's Alg.			Center-star Alg.		
	Time (sec)	Space (MB)	Score	Time (sec)	Space (MB)	Score	Time (sec)	Space (MB)	score
Data set 0	127	425	46319	<1	2.0	46319	12	1.9	40051
Data set 1	381	1192	71208	<1	2.6	71208	46	2.0	49875
Data set 2	254	654	63315	<1	2.7	63315	54	2.0	45241
Data set 3	— Memory exhausted —			<2	6.2	60849	208s	2.3	57325

Table 5. Alignment scores of the two algorithms after the refinement by ClustalW.

	Tang's Algorithm	Center-star Algorithm
Data set 0	38668	38668
Data set 1	69368	47216
Data set 2	63315	31776
Data set 3	62966	59021

problem, we used a modified scoring function based on `Pam70`. These four data sets were aligned using the CMSA algorithms, measuring the running time, memory requirement and the modified minimizing-`Pam70` alignment score. These alignments were then post-processed using the web-tool ClustalW[c] as described by Tang *et al.*[15]

Table 4 summarizes the performance of the three different CMSA algorithms on these data sets; while the performance of the algorithms after the ClustalW refinement on these data sets is presented in Table 5. The alignments of data sets

[c] The ClustalW web tool is provided by *European Bioinformatics Institute*, URL: `http://www.ebi.ac.uk/clustalw/`.

Data set 0

```
                        *                            *
Seq1: -K--E-TA-A-AK-FERQHMDSSTSAASSSNYCNQMMKSRNLTKDRCKPVNTFVHESLADVQAVCSQKNVAC-KN-GQT-N-CYQSYSTM
Seq2: -RPPQFTR-A-QW-FAIQHI-S-LNPPR----CTIAMRAINNYRWRCKNQNTFLRTTFANVVNVCGNQSIRC-PH-NRTLNNCHRSRFRV
Seq3: MK--P-PQFTWAQWFETQHINM-TSQQCN-N-AMQVI-N-NFQR-RCKNQNTFLRTTFANVVNVCGNPNITCPSNRSRN-N-CHHSGVQV
Seq4: -K--E-SA-A-AK-FERQHIDSSTSSVSSSNYCNEMMTSRNLTQDRCKPVNTFVHESLADVQAVCSQKNVAC-KN-GQT-N-CYQSYSAM
Seq5: -K--E-SR-A-KK-FQRQHMDSDSSPSSSSTYCNQMMRRRNMTQGRCKPVNTFVHEPLVDVQNVCFQEKVTC-KN-GQG-N-CYKSNSSM
Seq6: -M--Q-DG-MYQR-FLRQHVHPEETGGSDR-YCNLMMQRRKMTLYHCKRFNTFIHEDIWNIRSICSTTNIQC-KN-GKM-N-CHEG--VV
Seq7: ----Q-NW-A-T--FQQKHI-INTPIINC-N--TIMDNNIYIVGGQCKRVNTFIISSATTVKAICT--GVIN-MN-VLS-T--TR-FQ-L

                                                          *
Seq1: SITDC--R-ETGSSKY-PNCAYKTTQANKHIIVACEG-NP------Y-V-PVHFDA-S--V
Seq2: PLLHCDLI-NPGAQNI-SNCRYADRPGRRFYVVACDNRDPRDSPR-YPVVPVHLDT-T--I
Seq3: PLIHC--NLTTPSPQNISNCRYAQTPANMFYIVACDNRDPRRDPPQYPVVPVHLD--R--I
Seq4: SITDC--R-ETGNSKY-PNCAYQTTQAEKHIIVACEG-NP------Y-V-PVHYDA-S--V
Seq5: HITDC--R-LTNGSRY-PNCAYRTSPKERHIIVACEG-SP------Y-V-PVHFDA-S--V
Seq6: KVTDC--R-DTGSSRA-PNCRYRAIASTRRVVIACEG-NP------Q-V-PVHFDG-----
Seq7: N-T-C--T-RTSITPR-P-CPYSSRTETNYICVKCE--NQ------Y---PVHFAGIGRCP
```

Data set 1

```
Seq1: ------------------MVIS---PGSLLLVFLLS--LDV--IPP-TLAQDNYRYKNFL---N---QH-YDAKP-TGRDYRYCESMMKK
Seq2: --------------------------------------------------KET-AAAKFE---R---QH-MDSSTSAASSSNYCNQMMKS
Seq3: ------------------MTMS---PCPLLLVFVLG--LVV--IPP-TLAQNE-RYEKFL---R---QH-YDAKP-NGRDDRYCESMMKE
Seq4: ------------------MVVD---LPRYLPLLLL---LEL--WEP-MYLLCS-QPKGLS---R---AHWFEIQH-VQTSRQPCNTAMRG
Seq5: ---------------MKPLVIKFAWPLPLLLLLLLPPKLQGNYWDFGEYELNP-EVRDFI---R---EYESTGPTKPPTVKRIIEMITIG
Seq6: MDDEWERPEQATSAAEHPHTAA---QAAYNLADKLG--LEVPSWNPTTSSLRQ-KDRKLESNPRPAPSQKFYTEPIHNSTYPRCDDPMLV

                         *
Seq1: -RKLT--SPCK-EVNT-FIH-------------DTKNNIKAICGENGRPYGVNLRI-SNSRFQ---ITTCKHKG-GSPKPPCQYKAF---
Seq2: -RNLTK-DRCK-PVNT-FVH-------------ESLADVQAVCSQKNVACKNGQTN-CYQSYSTMSITDCRETG-SSKYPNCAYKTTQAN
Seq3: -RKLT--SPCK-DVNT-FIH-------------GTKKNIRAICGKKGSPYGENFRI-SNSPFQ---ITTCTHSG-ASPRPPCGYRAF---
Seq4: VNNYT--QHCK-QINT-FLH-------------ESFQNVAATCSLHNITCKNGRKN-CHESAEPVKMTDCSHTG-GA-YPNCRYSSD---
Seq5: DQPFNDYDYCNTELRTKQIHYKGRCYPEHYIAGVPYGELVKACDGEEVQCKNGVKS-CRRSMNLIEGVRCVLET-GQQMTNCTY------
Seq6: VNRYR--PRCK-DIDT-FLH-------------TSFANV-GVCGHPSGFCKEHKSANCHNSSSQVPIIVCNLTTPGRTYTQCRYQM----

      *                            *
Seq1: K-DFR--YIVIACE-----DG--W---PVHFDESFISM
Seq2: K------HIIVACE-----GNP-Y--VPVHFDAS-V--
Seq3: K-DFR--YIVIACE-----DG--W---PVHFDESFISP
Seq4: K-QYK--FFIVACEH-PKKEDPPYQLVPVHLDKI-V--
Seq5: KTILMIGYPVVSCQW--DEETKIF--IPDHIYNMSLPK
Seq6: KGSVE--YYTVACKPRTPWDSPIYPVVPVHLHGT-F--
```

Data set 2

```
                                               *                            *
Seq1: MVPKLFTSQICLLLLLGLMGVEGSLHARPPQFTRAQWFAIQHISLNP-PRCTIAMRAINNYR--W--RC-KNQNTFLRTTFANVVNVCGN
Seq2: MGLKLLESRLCLLLSLGLVLMLAS--CQPP--TPSQWFEIQHIYNRAYPRCNDAMRHRNRFT--G--HC-KDINTFLHTSFASVVGVCGN
Seq3: ---------------------------RPPQFTRAQWFAIQHISLNP-PRCTIAMRAINNYR--W--RC-KNQNTFLRTTFANVVNVCGN
Seq4: MGSKTLKSQLCLLLLLGLLLMLVSCQAQTP--S--QWFEIQHIYNSAYPRCDDAMRVIHGYSGVYLQRQEK----Y-K---------C--
Seq5: MVVDL-PRYLPLLLLLELWEPMYLLCSQPKGLSRAHWFEIQHVQTSR-QPCNTAMRGVNNYT--Q--HC-KQINTFLHESFQNVATCSLH
Seq6: MGSKTLKSQLCLLLLLGLLLMLVSCQAQTP--S--QWFEIQHIYNSAYPRCDDAMRVIHGYSGVYLQRQEK----Y-K---------C--

                      *                                                     *
Seq1: QSIRCPHNRTLNNCHRSRFRVPLLHCDLINPGAQNISNCRYADRPGRRFYVVACDNRDPRDSPRYPVVPVHLDTTI
Seq2: RNIPCG-NRTYRNCHNSRYRVSITFCNLTTP-ARIYTQCRYQTTRSRKFYTVGCDPRTPRDSPMYPVVPVHLDRIF
Seq3: QSIRCPHNRTLNNCHRSRFRVPLLHCDLINPGAQNISNCRYADRPGRRFYVVACDNRDPRDSPRYPVVPVHLDTTI
Seq4: ------HD-S-S----SK--IPVIICDLITWSNQH-THCRYKTTVAMKSYTVACNPRTPRNSPRYPFVPCHLDGTI
Seq5: N-ITCKNGR--KNCHESAEPVKMTDCSHTG-GA--YPNCRYSSDKQYKFFIVACEHPKKEDPP-YQLVPVHLDKIV
Seq6: ------HD-S-S----SK--IPVIICDLITWSNQH-THCRYKTTVAMKSYTVACNPRTPRNSPRYPFVPCHLDGTI
```

Data set 3

```
Seq1: -MP--INIISDSVYVVNAVLALETAGNFKQSSPVSEILMKIQNCILMREHPFYIQHIRAHTSLPGPMVKGNAIADSATRDM---VFL---
Seq2: MLRWLVALLSHSCFVSKGGGMFYAVRKGRQTGVYRTWAE-CQQ-QVNRFPSASFKKFAT--EKEAWAFVGAGPPDGQQSAP---AET--H
Seq3: -MR--VNGRNLTNLRFADDIVLIANHPNTASKMLQELVQKCSE-VGLEINTGKTKVLRNRFADPSKVYFGSPSPTTQLDDVDEYIYLGRQ
Seq4: -MP--INIISDSVYVVNAVLALETAGNFKQSSPVSEILMKIQNCILMHEHPFYIQHIRAHTSLPGPMVKGNAIADSATRDM---VFL---
Seq5: -M---V-LIS----LLNPETQ-NRSQNMSQNNPLRALLDK-QD-ILL------LDGAMA-TELEA---RGCNLAD-SLWS--AWAA----

                 *                                        *
Seq1: SQSSIESAKKFH-QLYYVPASTL--RQ-KFKLTRK--EARDIVLQCG-----KCVE--------FVNA-PSVG-VNPRG-LRPLD-VWQM
Seq2: GASAVAQENASH-RE-EPETDVLCCNACKRRYEQS--TNEEHTVRRA-----KHDE--------EQST-PVVS-EAKFSYMGEFAVVYTD
Seq3: INAQNNLMPEIH-R--RRRAA----WA-AFNGIKN--TTDSITDK-------K----------------IRAN-LFDSI-VLPAL-TYGS
Seq4: SQSSIESAKNFH-QLYHVPASTL--RQ-KFKLTRK--EARNIVLQCG-----KCVE--------FVNA-PSVG-VNPRG-LRPLD-VWQM
Seq5: VENP-ELIREVHLDYYRAGAQCA--ITASYQATPAGFAARGLDEAQSKALIGKSVELARKAREAYLAENPQAGTLLVAGSVRPYG-AYLT

                            *
Seq1: D--GMHIP-SFG-KLQ-YV---H-----------------------VSIDTSSGVLHASPLTGEKAVH-VIS-HCLE-----AWAA----
Seq2: GCCSGNGRNRARAGIGVYWGPGH------------------------------------PLN-------ISE-R-LP-----GRQT----
Seq3: E--AWTFTKALSERVR-IT---H-ASLERRLVG--ITLTQQRERDLHREDIRTMSLVRDPLN-----F-VKK-RKLGWAGHVARRK----
Seq4: D--VTHIP-SFG-KLQ-YV---H-----------------------VSIDTSSGVLHASPLTGEKAVH-VIS-HCLE-----AWAA----
Seq5: D--GSEYRGDYHCTVEAFQAF-HRPRVEALLVAGADLLACETLPNFSEIEALAELLTAYPRARAWFSFTLRDSEHLS-DGTPLRDVVALL

Seq1: WGKPLVLKTDNGPAYTSSKFSQFCKQMQVKHITGLPYNPQG---QGIIERAHHT------LKQYL-QKQKGGIEAMTPKMALSLTIFTLNF---
Seq2: -N--------------------------------------------------------------------------------------------
Seq3: DGRWTTLMTEWRPYGWKRPVGRPPMRWTDSLRKEITTRDAD---GEVITPWSTI------AKDRK-EWLAVIRRNTTNS---------------
Seq4: WGKPLVLKTDNGPAYTSSKFSQFCKQM-------------------------------------------------------------------
Seq5: AGYPQVVALGINCIALENTTAALQHLHGLTVLPLVVYPNSGEHYDAVSKTWHHHGEHCAQLADYLPQWQAAGARLIGGCCRTTPADIAALKARS
```

Fig. 1. Alignments of data sets 0 and 3.

0 to 3, using the constrained center-star algorithm, are shown in Fig. 1. Due to the page limit, the alignment matrices are divided into blocks of 90 characters, the first 90 characters of each sequence are listed first before the subsequent blocks. Columns that match the constrained characters are marked by an asterisk (*).

Comparing the alignment matrices produced by center-star approximation algorithm and Tang *et al.*'s progressive CMSA algorithm[15] in data sets $1, 2$ and 3, we note that the constrained characters $P[1]..P[\alpha]$ are aligned at different columns of the alignment matrices. Computationally, the center-star approximation algorithm for CMSA produces alignments with better SP alignment scores (see Table 4).

6.2. *Number of constraint occurrences*

We refer to the paper by Tang *et al.*[15] for an application used for CMSA. In their experiments, seven RNase sequences were aligned so that the three active-site residues, "HKH", were in the same columns in the alignment matrix. This motivates the CMSA problem as all RNase sequences contain the active-site residues "HKH" that are essential for the main functionality of the RNase degrading to RNA. In our experiment, we show that C, the total number of occurrences of the constraint "HKH" in each sequence is reasonably small. Since the value of C is relatively small, the running time of our center-star algorithm is more efficient than the $O(\alpha kn^4)$-time algorithm described by Tang *et al.*[15] There are totally 2351 sequences from the NCBI. In these 2351 RNase sequences, a total of 1954 sequences, or 83%, contain less than 500 residues. For each RNase, we measured the mean, mode and average number of occurrences of the constraint "HKH" and the result is reported in Table 6.

CMSA is usually done for a set of sequences of approximately the same length. As shown in Table 6, for CMSA of k sequences with lengths below 500, we have $C \leq 105k$ on average and $C \leq 241k$ for 90% of these RNase sequences. Even among the longer RNase sequences, the average number of occurrences is only about 464. Thus, the running time of our center-star algorithm is $O((500k)kn^2)$ on average which is much shorter than the running time $O(\alpha kn^4)$ of Tang *et al.*'s algorithm.[15]

Table 6. Occurrences of constraint "HKH" in RNase sequences from NCBI.

No. of samples	1954	2351
Percentage	83.11	100.0
Max sequence length	500	3989
Median occurrences	42	56
Average occurrences	105	464
80 Percentile	124	413
90 Percentile	241	1248

7. Conclusion

Using traditional MSA tools, biologists have limited control over the output of the sequence alignment. They can only choose high level alignment parameters such as gap penalty, scoring function, etc. As such, they are unable to incorporate their knowledge about the sequences, such as known functionalities and structures of the input sequences for use by the sequence alignment tool. This information is essential for accurate and biologically meaningful sequence alignment. Constrained sequence alignment provides users with the ability to differentiate important residues that need to be aligned together over other residues. This problem was first studied by Tang *et al.*[15] However, many existing techniques developed for MSA in the literature do not work for the CMSA problem due to the time and space complexities of $O(\alpha n^4)$. In this paper, we reduce the time and space complexities of solving the optimal pair-wise constrained alignment from $O(\alpha n^4)$ to $O(\alpha n^2)$. With this improvement, existing techniques for MSA can now be modified to solve the CMSA problem. We have demonstrated how the center-star sequence approximation algorithm can be applied to solve the CMSA problem. With the reduction in time and space complexities, it is hoped that the improved quality of sequence alignment can help biologists. It is worth-mentioning that the constrained center-star alignment problem can be shown to be NP-hard, exploiting approximation algorithms of this problem for the constrained multiple sequence alignment problem would be of theoretic and practical interest.

Acknowledgement

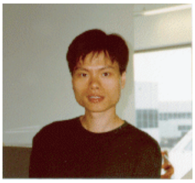
We thank the reviewer for pointing out that the space of our pair-wise constrained alignment can be improved to $O(\alpha n)$.

References

1. V. Bafna, E.L. Lawler, and P.A. Pevzner. Approximation algorithms for multiple sequence alignment. In *Theoretical Computer Science*, volume 182, issues 1-2, pages 233–244. ACM Press, Aug 1997.
2. P. Bonizzoni and G.D. Vedova. The complexity of multiple sequence alignment with *SP*-score that is a metric. *Theoretical Computer Science*, 259(1–2):63–79, 2001.
3. P. Clote and R. Backofen. *Computational Molecular Biology: An Introduction.* John Wiley and Sons, Ltd, Aug 2000.
4. T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Chapter 23: Minimum Spanning Trees, Introduction to Algorithms, Second Edition.* The MIT Press, 2001.
5. D. Gusfield. Efficient methods for multiple sequence alignment with guaranteed error bounds. In *Bull. Math. Biol., 30*, volume 30, pages 141–154, 1993.
6. D. Gusfield. *Algorithms on Strings, Trees and Sequences.* Cambridge U. Press, 1997.
7. D.S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Commun. ACM*, 18(6):341–343, 1975.
8. T. Jiang, M. Zhang, and Y. Xu, editors. *Chapter 4: Algorithmic Methods for Multiple Sequence Alignment, Current Topics in Computational Molecular Biology.* The MIT Press, 2002.

9. D.J. Lipman, S.F. Altschul, and J.D. Kececioglu. A tool for multiple sequence alignment. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 86-12, pages 4412–4415, Jun 1989.
10. H.B. Jr Nicholas, A.J. Ropelewski, and D.W. Deerfield II. Strategies for multiple sequence alignment. *BioTechniques*, 32:572–591, Mar 2002.
11. C. Notredame, D. Higgins, and J. Heringa. T-Coffee: A novel method for multiple sequence alignments. In *Journal of Molecular Biology*, volume 302, page pp20, 2000.
12. P.A. Pevzner. Multiple alignment, communication cost, and graph matching. In *SIAM J. Applied Mathematics*, volume 52, pages 1763–1779, 1992.
13. A. Phillips, D. Janies, and W. Wheeler. Multiple sequence alignment in phylogenetic analysis. *Molecular Phylogenetics and Evolution*, 16-3:317–330, Sep 2000.
14. K. Reinert, H.P. Lenhof, P. Mutzel, K. Mehlhorn, and J.D. Kececioglu. A branch-and-cut algorithm for multiple sequence alignment. In *Proceedings of the 1st Annual International Conference on Computational Molecular Biology (RECOMB)*, pages 241–250, Santa Fe, NM, 1997. ACM Press.
15. 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, and W.I. Chou. Constrained multiple sequence alignment tool development and its application to RNase family alignment. In *Proceedings of the First IEEE Computer Society Bioinformatics Conference (CSB 2002). The full version appears in Journal of Bioinformatics and Computational Biology, 1(2):267–287.*, pages 127–137, 2002.
16. J.D. Thompson, D.G. Higgins, and T.J. Gibson. ClustalW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. In *Nucleic Acids Research*, volume 22, pages 4673–4680, 1994.
17. L. Wang and T. Jiang. On the complexity of multiple sequence alignment. In *Journal of Computational Biology*, volume 1, pages 337–348, 1994.
18. R.T. Wong. Worst-case analysis of network design problem heuristics. *SIAM J. Alg. Disc. Meth.*, 1(1):51–63, 1980.

Francis Chin is the Chair of Computer Science, Taikoo Professor of Engineering and Associate Dean of the Graduate School at The University of Hong Kong (HKU). Francis received his B.Sc. from the University of Toronto, Canada, in 1972, and M.S., M.A. and Ph.D. from Princeton University, in 1974, 1975, and 1976, respectively. He became an IEEE Fellow in 1996. Before joining HKU, he taught at the University of Maryland, Baltimore County; University of California, San Diego; University of Alberta; and the Chinese University of Hong Kong. He has also visited the City University of Hong Kong and the University of Texas at Dallas. He was the Head of the Computer Science at HKU for nearly 15 years until the end of 1999. His research interests include Bioinformatics and Design and Analysis of Algorithms.



Tak-Wah Lam is an Associate Professor at the Department of Computer Science of the University of Hong Kong. He received his PhD degree from the University of Washington in 1988. His current research interests include online algorithms and computational biology.

Ngai Lam Ho is an M.Phil. student at Department of Computer Science of University of Hong Kong and is expected to graduate in late 2004. He received his B.Sc. Computer Science (Hon.) degree from National University of Singapore in 2001. His research interests include computational biology, algorithms and data structures.

Prudence Wong is a Lecturer at the Department of Computer Science of the University of Liverpool. She received her PhD degree from the University of Hong Kong in 2003. Before joining the University of Liverpool in 2004, she was a Postdoctoral Fellow at the University of Hong Kong. Her research interests include on-line algorithms and computational biology.