

# Efficient Constrained Multiple Sequence Alignment with Performance Guarantee

Francis Y.L. Chin      N.L. Ho      T.W. Lam

Department of Computer Science and Information Systems,  
The University of Hong Kong, Hong Kong.  
{chin, nlho, twlam}@csis.hku.hk

## Abstract

The Constrained Multiple Sequence Alignment problem is to align a set of sequences 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 [14], 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.

## 1 Introduction

Multiple sequence alignment (MSA) is one of the problems in computational biology that have been studied extensively [1, 3, 5, 6, 8, 11, 13]. 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 optimal alignment is NP-hard, i.e., intractable<sup>1</sup> [3, 15]. In the literature, there are a number of MSA algorithms that attempt to approximate the optimal alignment, some of them can provide a worst-case approximation ratio [2, 4, 12], while some others work well in practice [7, 10]. 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 allows users to provide additional constraints.

Tang *et al.* [14] are 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

---

<sup>1</sup>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 the alignment cost of all pairs of sequences.

	Time	Space	Approximation Ratio
Tang <i>et al.</i> 's algorithm [14]	$O(\alpha kn^4)$	$O(\alpha n^4)$	–
Improved Tang <i>et al.</i> 's algorithm	$O(\alpha k^2 n^2)$	$O(\alpha n^2)$	–
Center-star	$O(\alpha C k^2 n^2)$	$O(\alpha k^2 n^2)$	$2 - \frac{2}{k}$

Figure 1: Performance of constrained multiple sequence alignment algorithms.

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 RNases 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.* [14] 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  $\alpha$  is the length of constrained sequence. For aligning  $k \geq 3$  sequences, they gave a heuristic algorithm (called progressive alignment algorithm) with time and space requirements being  $O(\alpha kn^4)$  and  $O(\alpha n^4)$ , respectively. When applied to align multiple RNase sequences, this algorithm produces satisfactory alignments. Yet the application of the algorithm is limited as the memory requirement is too big 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 Tang *et al.*'s multiple sequence 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  $(2 - \frac{2}{k})$ . This algorithm adopts the framework of Gusfield's (unconstrained) multiple sequence alignment algorithm [4]. The time and space complexities of the new algorithm are respectively  $O(\alpha C k^2 n^2)$  and  $O(\alpha k^2 n^2)$ , where  $C$  is the maximum number of occurrences of the constraint in individual sequences. The improved memory requirement allows us to handle sequences with thousands of characters on ordinary workstations. See Figure 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 15% to 30% better, while the memory requirement is at most one-hundredths of Tang *et al.*'s algorithm. Results are briefly summarized in Figure 2. More details will be given in Section 5.

The rest of this paper is organized as follows. Section 2 defines the constrained sequence alignment, and Section 3 presents the new optimal constrained pair-wise sequence alignment algorithm. Section 4 presents algorithms for *constrained multiple sequence alignment*. In particular, an approximation algorithm is given with an approximation ratio  $(2 - \frac{2}{k})$ . We

	Tang's Alg.			Center-Star Alg.		
	Score	Time	Space	Score	Time	Space
7 sequences max length 125 $\alpha = 3$	46319	127 sec	425 MB	40051	25 sec	4.2 MB
6 sequences max length 185 $\alpha = 3$	71208	381 sec	1192 MB	49875	77 sec	2.8 MB
6 sequences max length 186 $\alpha = 4$	63315	254 sec	654 MB	45241	82 sec	3.1 MB
5 sequences max length 327 $\alpha = 3$	Memory exhausted			57325	482 sec	6.2 MB

Figure 2: Alignment scores of CMSA algorithms

report empirical results of our developed CMSA tools in Section 5. Finally, we conclude this paper by giving some further research directions in CMSA.

## 2 Preliminaries

Let  $\Sigma$  be the set of characters (residues),  $S = \{S_1, S_2, \dots, S_k\}$  be a set of  $k$  sequences, with maximum length  $n$ , over  $\Sigma$ . Let  $S_i[x..y]$  denote the sub-string of  $S_i$  starting at the  $x$ -th character to the  $y$ -th character of  $S_i$ , where  $1 \leq x < y \leq n$ . In particular, let  $S_i[x]$  denote the  $x$ -th character in 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(residues), where  $x, y \in \Sigma \cup \{-\}$ , the pair-wise score of two length- $n'$  sequences  $S'_1$  and  $S'_2$  is defined as  $\sum_{1 \leq i \leq n'} \delta(S'_1[i], S'_2[i])$ . In the multiple sequence alignment (MSA) problem, we are given  $k$  sequences  $S = \{S_1, S_2, \dots, S_k\}$ , MSA is an *alignment matrix*  $A$ , with  $k$  rows and  $n' (\geq n)$  columns, such that removing space characters from the  $i$ -th row of  $A$  gives  $S_i$  for  $1 \leq i \leq k$ . The *sum-of-pair (SP)* score of a (MSA) is defined as the sum of the pair-wise scores of all pairs of the sequences,  $\sum_{1 \leq p < q \leq k} \sum_{1 \leq i \leq n'} \delta(A_{pi}, A_{qi})$  where each row of the alignment matrix is treated as a sequence, as  $S'_1$ ,  $A_p$  is the  $p$ -th row of the alignment matrix  $A$  and  $A_{pi}$  is the character at  $p$ -th row and  $i$ -th column of  $A$ . It is shown in [3, 15] that finding an alignment matrix with the minimum SP score is NP-complete.

In the *constrained multiple sequence alignment problem (CMSA)*, we are given, in addition to the inputs of the MSA problem, a constrained sequence  $P = (P[1], P[2], \dots, P[\alpha])$ , where  $P$  is a common subsequence of  $S_i \in \{S_1, S_2, \dots, S_k\}$ . 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  $\{c_1, c_2, \dots, c_\alpha\}$  where  $1 \leq c_1 < c_2 < \dots < c_1 < \dots < c_\alpha \leq n'$ , and for all  $1 \leq i \leq k$

and all  $1 \leq l \leq \alpha$ ,  $(A_{i c_l} = P[l])$ .

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

### 3 Constrained Pair-wise Sequence Alignment

#### 3.1 Problem definition

The *constrained pair-wise sequence alignment (CPSA)* problem is a special for CMSA problem with  $k = 2$ . Given two sequences  $S_1$  and  $S_2$ , a constrained sequence  $P$  ( $|P| = \alpha$ ) and a distance function  $\delta$ , the problem is to compute an optimal CPSA,  $\left[ \begin{array}{c} S'_1 \\ S'_2 \end{array} \right]$ , such that  $\sum_{1 \leq i \leq |S'_1|} \delta(S'_1[i], S'_2[i])$  is minimized subject to  $S'_1[c_\gamma] = S'_2[c_\gamma] = P[\gamma]$  for  $1 \leq \gamma \leq \alpha$ , and  $1 \leq c_1 < c_2 < \dots < c_\gamma < \dots < c_\alpha \leq n'$ . 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

We define  $D(i, j, \gamma)$  to be the optimal constrained pair-wise sequence alignment(CPSA) score of sequences  $S_1[1..i]$  and  $S_2[1..j]$  with constraint  $P[1..\gamma]$ , where  $1 \leq i \leq |S_1|, 1 \leq j \leq |S_2|, 1 \leq \gamma \leq \alpha$ . In particular,  $D(n_1, n_2, \alpha)$  gives 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  $|P| = \alpha$ .

**Theorem 3.1** For  $0 \leq i \leq n_1, 0 \leq j \leq n_2$  and  $0 \leq \gamma \leq \alpha$ ,

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

with boundary condition  $\begin{cases} D(0, 0, 0) = 0 \\ D(i, 0, \gamma) = \infty \text{ for } \gamma \geq 1, 0 \leq i \leq n_1 \\ D(0, j, \gamma) = \infty \text{ for } \gamma \geq 1, 0 \leq j \leq n_2 \end{cases}$

*Proof.* Consider an alignment of  $S_1[1..i]$  with an empty string, the only possible alignment is to align  $S_1[1..i]$  with the “space” characters. Thus,  $D(i, 0, 0) = D(i-1, 0, 0) + \delta(S_1[i], -)$  for  $i > 0$ . Obviously, an alignment cannot match with any character in  $P$ , so for  $\gamma > 0$ ,  $D(i, 0, \gamma) = \infty$ . Similarly,  $D(0, j, 0) = D(0, j-1, 0) + \delta(-, S_2[j])$  for  $j > 0$  and  $D(0, j, \gamma) = \infty$  for  $\gamma > 0$ . With  $\gamma, i$  and  $j > 0$ , and if  $S_1[i]$  and  $S_2[j]$  do not match with  $P[\gamma]$ ,  $D(i, j, \gamma)$  is the minimum of (i)  $D(i-1, j-1, \gamma) + \delta(S_1[i], S_2[j])$ , (ii)  $D(i-1, j, \gamma) + \delta(S_1[i], -)$ , or (iii)  $D(i, j-1, \gamma) + \delta(-, S_2[j])$ , corresponding to substitution of  $S_1[i]$  and  $S_2[j]$ , insertion of space characters to  $S_2$  or  $S_1$  with the assumption that constraint  $P[1..\gamma]$  is already met in the subsequences. If  $S_1[i] = S_2[j] = P[\gamma]$ , then we have to consider an additional situation where the last character in both sequence, are used to match with the constrained character  $P[\gamma]$ , i.e.  $D(i-1, j-1, \gamma) + \delta(S_1[i], S_2[j])$ . ■

**Theorem 3.2** *The optimal constrained pair-wise alignment can be computed in  $O(\alpha n_1 n_2)$  time, where  $|S_1| = n_1$ ,  $|S_2| = n_2$  and  $|P| = \alpha$ .*

*Proof.* Using the recurrence in Theorem 3.1, we can construct a 3-dimensional matrix  $D$  of size  $[(n_1 + 1) \times (n_2 + 1) \times (\alpha + 1)]$ , 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  $D[i, j - 1, \gamma]$ . Each  $D[i, j, \gamma]$  can be computed in constant time using dynamic programming similar to the unconstrained pair-wise alignment algorithm [5]. Since the size of matrix  $D$  is  $O(\alpha n_1 n_2)$ , the optimal CPSA score of two sequences of lengths  $n_1$  and  $n_2$  and constrained sequence  $P$  of length  $\alpha$  can be computed in  $O(\alpha n_1 n_2)$  time complexity. The CPSA can be obtained from matrix  $D$  by back-tracking the path from  $D[0, 0, 0]$  to  $D[n_1, n_2, \gamma]$ . If the path goes vertical (along the first dimension) then space character is inserted into  $S'_2$ . If the path goes horizontal (along the second dimension of  $D$ ), space character is inserted into  $S'_1$ . Finally the length of the path will be  $n'$ , that is the length of  $S'_1$  and  $S'_2$ . ■

## 4 Constrained Multiple Sequence Alignment

In Section 4.1, we reduce the time and space complexities of the progressive CMSA algorithm described in [14] from  $O(\alpha k n^4)$  to  $O(\alpha k^2 n^2)$ . We then show an algorithm computes the optimal CMSA based on the sum-of-pair score in Section 4.2. In Section 4.3, we apply the center-star approximation algorithm [4] to the CMSA problem, and show that this algorithm achieves an approximation ratio  $(2 - \frac{2}{k})$  in time complexity  $O(\alpha C k^2 n^2)$ , where  $C$  is the maximum number of occurrences of constraint  $P$  in  $S$ . Throughout this section, we assume that the scoring function  $\delta(x, y)$  follows the *triangular inequalities*, that is,  $\delta(x, y) \leq \delta(x, z) + \delta(z, y)$ , for any  $x, y, z \in \Sigma \cup \{-\}$ , and  $\delta(-, -) = 0$ .

### 4.1 Improved Progressive CMSA Algorithm

Tang *et al* [14] presented an  $O(\alpha k n^4)$  time and  $O(\alpha n^4)$  space progressive approximate algorithm for the CMSA problem for a set of  $k$  sequences of length at most  $n$  and a constrained sequence  $P$  of length  $\alpha$ . In Tang *et al.*'s algorithm, a  $k \times k$  distance matrix of  $k$  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*) based on the distance matrix of these sequences. Sequences are 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 Section 3.1. 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

Extending the optimal CPSA in Section 3.2 to  $k$  sequences involves constructing a matrix  $D$  of  $(k + 1)$  dimensions, which takes  $O(\alpha n^k)$  time and space. Specifically, let the multi-dimensional array  $D(i_1, i_2, \dots, i_k; \gamma)$  be the optimal CMSA score matrix for  $S' = \{S_1[1..i_1], S_2[1..i_2], \dots, S_k[1..i_k]\}$  with  $\{P[1], P[2], \dots, P[\gamma]\}$  aligned in  $\gamma$  columns. Then

the optimal alignment score for  $\{S_1 \dots S_k\}$  with respect to the constrained sequence  $P$  is given by  $D(n_1, n_2, \dots, n_k; \alpha)$ , where  $n_i = |S_i|$  for  $1 \leq i \leq k$ ,  $D(i_1, i_2, \dots, i_k, \gamma)$  can be computed with the following recurrence,

$$1) D(\{0\}^k; 0) = 0$$

$$2) D(i_1, i_2, \dots, i_k; \gamma) = \min \begin{cases} \min_{\epsilon \in \{0,1\}^k} D(i_1 - \epsilon_1, i_2 - \epsilon_2, \dots, i_k - \epsilon_k; \gamma) + \\ \delta(\epsilon_1 S_1[i_1], \epsilon_2 S_2[i_2], \dots, \epsilon_k S_k[i_k]) \\ D(i_1 - 1, i_2 - 1, \dots, i_k - 1; \gamma - 1) + \\ \delta(S_1[i_1], S_2[i_2], \dots, S_k[i_k]) \\ \text{(if } S_1[i_1] = S_2[i_2] = \dots = S_k[i_k] = P[\gamma]) \end{cases}$$

where  $\epsilon = 0$  or  $1$ ,  $\epsilon S_j[i_j]$  with  $\epsilon = 0$  represents a space character, and

$$\delta(x_1, \dots, x_k) = \sum_{1 \leq i < j \leq k} \delta(x_i, x_j)$$

The above recurrence relation given above computes the optimal CMSA for multiple sequences, a generalization of the dynamic programming approach in solving CPSA problem. Practically, optimal CMSA can be computed for less than 6 short sequences of length at most 200 [9]. Thus the following section gives an approximation algorithm for solving the CMSA problem.

### 4.3 The Center-Star Alignment Approximation for CMSA

For the un-constrained multiple sequence alignment, center-star algorithm[4] is an approximation algorithm with the approximation ratio of  $2 - \frac{2}{k}$ . This section outlines an approximation algorithm, based on the center-star approximation algorithm that yields an approximation ratio  $(2 - \frac{2}{k})$  for CMSA. For a set of  $k$  sequences,  $S = \{S_1 \dots S_k\}$ ,  $S_c$  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  $S_c$  to  $S_j$  where  $j \neq c$ .

The *star-sum score* of CMSA w.r.t. a center sequence  $S_c$  is the sum of pair-wise score between  $S_c$  and each  $S_j \in S$  with  $j \neq c$ . The center-star approximation algorithm is to find the CMSA and its center sequence  $S_c$  such that its star-sum score w.r.t.  $S_c$  is minimized.

#### Main algorithm of center-star alignment

1. For each  $S_i \in S$ , treat  $S_i$  as the center sequence  $S_c$  and for each list of positions  $(c_1, c_2, \dots, c_\alpha)$  in  $S_c$  to be aligned with  $P$ , i.e.  $P[\gamma] = S_c[c_\gamma]$ ,  $1 \leq \gamma \leq \alpha$ , align  $S_i$  with  $S_c$  at the positions specified by  $(c_1, c_2, \dots, c_\alpha)$  (see below).
2. Find the  $S_c$  and  $(c_1, c_2, \dots, c_\alpha)$  with the minimum star-sum.
3. Merge the  $(k-1)$  constrained pair-wise sequence alignments under positions  $(c_1, c_2, \dots, c_\alpha)$  into an constrained alignment matrix of size  $[k \times n']$ .

Without loss of generality, let us assume that the center sequence is  $S_1$ . Given  $c_1, c_2, \dots, c_\alpha$ , we perform the CPSA algorithm of  $S_1$  to  $S_2 \dots S_k$  under  $(c_1, c_2, \dots, c_\alpha)$ , using a slightly modified recurrence in Theorem 3.1, treating  $S_c$  as  $S_1$  and  $S_j$  as  $S_2$ .

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

Using this recurrence, the alignment path constructed from  $D(0, 0, 0)$  to  $D(n_c, n_j, \alpha)$  must pass through the points  $S(c_\gamma, j, \gamma)$  for  $1 \leq \gamma \leq \alpha$ ,  $0 \leq j \leq n_j$ , where  $n_c = |S_c|$  and  $n_j = |S_j|$ .

**Optimal center sequence and the constrained positions** Consider each sequence  $S_i$  and a list of positions of occurrence of  $P$  in  $S_i$ . The combination  $(S_i; c_1, c_2, \dots, c_\alpha)$  that gives the minimum sum of constrained pair-wise alignment scores with other sequences under the positions  $(c_1, c_2, \dots, c_\alpha)$  is selected as  $S_c$  and the list of positions to be aligned with  $P$ . The time for locating  $(S_c; c_1, c_2, \dots, c_\alpha)$  is then  $O(\alpha k C n^2)$  for  $k$  sequences with  $P$  appears in each sequence for at most  $C$  lists of positions.

**Merging constrained pair-wise alignments** Suppose the center sequence of  $S$  is  $S_1$  under a list of positions  $(c_1, c_2, \dots, c_\alpha)$ . There are  $(k - 1)$  constrained pair-wise alignments, each alignment has  $S_1$  aligned with  $S_j$ ,  $2 \leq j \leq k$ . Suppose  $|S_1| = n$ , and let  $A_{1,j}$  be the optimal constrained pair-wise alignment of  $S_1$  and  $S_j$  under  $(c_1, c_2, \dots, c_\alpha)$ . Define  $s_0$  and  $s_n$  be the longest spaces inserted before  $S_1[1]$  and after  $S_1[n]$  in all  $(k - 1)$   $A_{1,j}$ 's respectively. Similarly for  $1 \leq i \leq n - 1$ , let  $s_i$  be the longest spaces between  $S_1[i]$  and  $S_1[i + 1]$  in all  $S_1$ 's in  $(k - 1)$  constrained pair-wise alignments. Initially, set  $A$  to have only one row  $S'_1 = [s_0 + S_1[1] + s_1 + S_1[2] + \dots + s_{i-1} + S_1[i] + s_i + \dots + S_1[n] + s_n]$ ,  $|S'_1| = n'$ , where the  $+$  operation denotes the string concatenation. For each  $S_j$ ,  $2 \leq j \leq k$ , add  $S_j$  to  $A$  according to the optimal constrained pair-wise sequence alignment of  $S_1$  and  $S_j$ ,  $[\bar{S}_1, \bar{S}_j]$ , i.e. insert columns of spaces to  $[\bar{S}_1, \bar{S}_j]$  until  $\bar{S}_1$  is identical to  $S'_1$  in  $A$ , then add each  $S'_j$  (with spaces inserted) to  $A$ . After  $(k - 1)$   $S'_j$  are inserted to  $A$ , according to the way  $S'_j$  is inserted, for any pair of  $(S'_i, S'_j)$  in  $A$ , removing columns of spaces gives the constrained pair-wise alignment of  $S_1$  and  $S_j$  under  $(c_1, c_2, \dots, c_\alpha)$ .  $A$  is then the optimal constrained multiple sequence alignment of  $S = \{S_1, \dots, S_k\}$  under the star-sum score, because the pair-wise score of the rows  $S'_1$  and  $S'_j$  does not increase the optimal constrained pair-wise alignment score of  $S_1$  and  $S_j$  with respect to  $P$  under  $(c_1, c_2, \dots, c_\alpha)$ .

## Performance of Center-Star Algorithm in CMSA

We now show that the center-star approximation algorithm for CMSA has the approximation bound  $(2 - \frac{2}{k})$ . We define the distance  $D(S'_i, S'_j)$  of two aligned sequences  $S'_i$  and  $S'_j$  as the sum of pair-wise distance between the two characters at the same positions in  $S'_i$  and  $S'_j$ .

**Theorem 4.1** *Given  $S = \{S_1, \dots, S_k\}$  and a constrained sequence  $P$ . Suppose  $A^s$  is the alignment output by the center-star CMSA algorithm,  $A^*$  be the optimal constrained alignment with respect to  $P$ . Then,  $\frac{sp\_score(A^*)}{sp\_score(A^s)} \leq 2 - \frac{2}{k}$ .*

*Proof.* Since  $A^*$  is a CMSA for  $S$  and  $P$ ,  $sp\_score(A^*) = \sum_{1 \leq i < j \leq k} D(A_i^*, A_j^*)$  where  $D(A_i^*, A_j^*)$  specifies the pair-wise score of row  $A_i^*$  and row  $A_j^*$ . Thus,

$$\begin{aligned} sp\_score(A^*) &= \frac{1}{2} \sum_{1 \leq i, j \leq k, i \neq j} D(A_i^*, A_j^*) \\ &= \frac{1}{2} \sum_{1 \leq i \leq k} [\sum_{1 \leq j \leq k, i \neq j} D(A_i^*, A_j^*)] \\ &= \frac{1}{2} \sum_{1 \leq i \leq k} (\text{star-sum score of } A^* \text{ with } A_i^* \text{ as the center sequence}) \\ &\geq \frac{k}{2} \min_{1 \leq i \leq k} (\text{star-sum score of } A^* \text{ with } A_i^* \text{ as the center sequence}) \\ &\geq \frac{k}{2} (\text{star-sum score of } A^s \text{ with } A_c^s \text{ as the center sequence}) \end{aligned}$$

On the other hand, assume  $A_c$  is the center sequence,

$$\begin{aligned}
sp\_score(A^s) &= \frac{1}{2} \sum_{1 \leq i, j \leq k} D(A_i^s, A_j^s) \\
&\leq \frac{1}{2} \sum_{1 \leq i, j \leq k} D(A_i^s, A_c^s) + D(A_c^s, A_j^s) \\
&= \frac{2(k-1)}{2} \sum_{1 \leq i \leq k, i \neq c} D(A_i^s, A_c^s) \\
&= (k-1) \times (\text{star-sum score of } A^s \text{ with } A_c^s \text{ as the center sequence}).
\end{aligned}$$

Thus, the approximation ratio of center-star algorithm for CMSA is

$$\frac{sp\_score(A^s)}{sp\_score(A^*)} \leq \frac{2(k-1)}{k} = 2 - \frac{2}{k} \quad \blacksquare$$

## 5 Empirical Results

In this section, we present 4 real data sets taken from the NCBI. We run these data sets on our developed center-star approximation algorithm, the improved progressive alignment, as well as the original CMSA algorithm that appears in [14]. In Section 5.2, we justify  $C$ , the maximum number of occurrences of the constrained sequence in the sequences, is relatively small compared to  $O(n^2)$ .

### 5.1 Experiments on CMSA Algorithms

All our experiments are conducted on an Intel workstation of 2.0 MHz CPU with 4GB of main memory. Besides our improved progressive alignment algorithm and center-star approximation alignment, we also implemented the Tang *et al.*'s algorithm presented in [14].

For pair-wise alignment, we run an instance of aligning of 2 sequences of length 150, with a constraint length 3. The algorithm in [14] consumes up to 400 MB main memory and running time 127 seconds. For the first same problem instance, our pair-wise sequence alignment algorithm runs in a fraction of second with only 1.5 MB of main memory. This shows the practicality of our pair-wise constrained sequence alignment algorithm for longer sequences and longer constrained sequence.

Along with the data set used in [14] (data set 0), we obtained 3 other data sets; data set 1 and 2 contain 6 sequences of length about 180, and data set 3 contains long sequences with maximum length of 327. We performed the CMSA algorithms on data sets 0, 1 and 3 with  $P = \text{HKH}$  and  $P = \text{HKSH}$  for data set 2. Since we cast the CMSA as a minimization problem, we used a modified scoring function based on Pam70, with the mutation distances between two characters range from 0 to 24. We present the experimental results in two stages. For the first stage, we align data set 0 to 3 with Tang *et al.*'s progressive algorithm, our improved progressive algorithm and the center-star approximation. We measured the running time, space requirement and alignment score for each algorithm. In the second stage of the experiment, we took the first set of results, divided the alignment into  $(\alpha + 1)$  blocks of small sequences by the constraint characters and subsequently re-aligned individual blocks sequences without constraint using the web-tool ClustalW<sup>2</sup> as described in [14]. Only the alignment scores are recorded. We show the alignments with the least score for data set 2 in Figures 6. Due to the page limit, only data set 2 is shown, the alignment matrix is divided into blocks of 80 characters, the first 80 characters of all sequences are listed first before the subsequent blocks. Columns which are aligned with the constrained sequence are marked by an asterisk. We note that for data sets 1 to 3, the constrained sequence  $P$  is aligned at different set of positions in the center-star approximation algorithm and Tang

<sup>2</sup>The ClustalW web tool is provided by *European Bioinformatics Institute*, URL: <http://www.ebi.ac.uk/clustalw/>.

	Tang’s Alg.			Improved Tang’s Alg.		Center-Star Alg.		
	Time	Space	Score	Time	Space	Score	Time	Space
Data set 0	127 sec	425 MB	46319	< 1 sec	2.0 MB	25 sec	4.2 MB	40051
Data set 1	381 sec	1192 MB	71208	< 1 sec	2.6 MB	77 sec	2.8 MB	49875
Data set 2	254 sec	654 MB	63315	< 1 sec	2.7 MB	82 sec	3.1 MB	45241
Data set 3	–	–	60849	< 2 sec	6.2MB	482sec	6.2 MB	57325

Figure 3: Alignment scores of CMSA algorithms

	Tang’s Alg. Score	Center-Star Alg. Score
Data set 0	38668	38668
Data set 1	69368	47216
Data set 2	63315	31776
Data set 3	62966	59021

Figure 4: Alignment scores of the two algorithms after the refinement by ClustalW

*et la.*’s progressive CMSA algorithm [14]. Computationally, the center-star approximation algorithm for CMSA gives better alignment score.

## 5.2 Number of Constraint Occurrences

We refer to [14] for an application used for CMSA. In their experiment, 7 RNase sequences are aligned with the constraint that three active-site residues, {HKH}, should be aligned in the same columns. 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 the number of occurrences of the constraint {HKH} in each sequence is reasonably small, that makes the running time of our center-star algorithm faster than the algorithm in [14].

We collected all RNase sequences, total 2869 sequences from the NCBI<sup>3</sup> web-site. A total of 2266 seq500 residues. For each RNase, we measure the mean, mode and average number of occurrences of the constraint {HKH} which is reported in Figure 5.

<sup>3</sup>National Center of Biotechnology Information, URL: <http://www.ncbi.nlm.nih.gov>.

No. 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

Figure 5: Occurrences of constraint “HKH” in RNase sequences

MSA is usually done for a set of sequences of approximately the same length. As shown in the Figure 5, for alignment sequences with length below 500,  $C = 105$  on average. Even among the longer RNase sequences the average number of occurrences is only about 464. Thus, running time of our center-star algorithm has time complexity  $O(500\alpha k^2 n^2)$  which is much less expensive compared to running time of  $O(\alpha k n^4)$ .

## 6 Conclusion

For the traditional MSA tools, biologists have no absolute control over the alignment of sequences. They only have the liberty of choosing some high level alignment parameters such as gap penalty, scoring function. They could not incorporate their knowledge on the known functionalities or structures of the input sequences in the sequence alignment. This information is essential for accurate and biologically meaningful sequence alignment. Constrained sequence alignment provides with users the ability to differentiate important residues that need to be aligned together over other residues. Due to the large time and space requirement of CMSA algorithm in [14], many techniques used for the MSA could not work on CMSA 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  $O(\alpha n^4)$ . With this improvement, existing techniques that work for MSA can now be modified to solve the CMSA problem. We have demonstrated how the center star sequence approximation algorithm can be modified to solve CMSA problem. With the reduction on time and space complexities, it is hoped that the improved quality of sequence alignment can be treasured by biologists.

## References

- [1] A. PHILLIPS, D. JANIES, W. W. Multiple sequence alignment in phylogenetic analysis. *Molecular Phylogenetics and Evolution* 16-3 (Sep 2000), 317–330.
- [2] BAFNA, V., LAWLER, E. L., AND PEVZNER, P. A. Approximation algorithms for multiple sequence alignment. In *Theoretical Computer Science* (Aug 1997), vol. 182, issues 1-2, ACM Press, pp. 233–244.
- [3] BONIZZONI, P., AND VEDOVA, G. D. The complexity of multiple sequence alignment with  $SP$ -score that is a metric. *Theoretical Computer Science* 259, 1–2 (2001), 63–79.
- [4] GUSFIELD, D. Efficient methods for multiple sequence alignment with guaranteed error bounds. In *Bull. Math. Biol.*, 30 (1993), vol. 30, pp. 141–154.
- [5] GUSFIELD, D., Ed. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, 1997.
- [6] II, H. N. J. A. R. D. D. Strategies for multiple sequence alignment. *BioTechniques* 32 (Mar 2002), 572–591.
- [7] J.D., T., D.G., H., AND GIBSON, T. Clustalw: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific

- gap penalties and weight matrix choice. In *Nucleic Acids Research* (1994), vol. 22, pp. 4673–4680.
- [8] JIANG, T., ZHANG, M., AND XU, Y., Eds. *Chapter 4: Algorithmic Methods for Multiple Sequence Alignment, Current Topics in Computational Molecular Biology*. The MIT Press, 2002.
- [9] LIPMAN, D. J., F, S., ALTSCHUL, AND KECECIOGLU, J. D. A tool for multiple sequence alignment. In *Proceedings of the National Academy of Sciences of the United States of America* (Jun 1989), vol. 86-12, pp. 4412–4415.
- [10] NOTREDAME, C., HIGGINS, D., AND HERINGA, J. T-coffee: A novel method for multiple sequence alignments. *Journal of Molecular Biology* 302 (2000), 205–217.
- [11] PETER CLOTE, R. B. *Computational Molecular Biology: An Introduction*. John Wiley and Sons, Ltd, Aug 2000.
- [12] PEVZNER, P. A. Multiple alignment, communication cost, and graph matching. In *SIAM J. Applied Mathematics* (1992), vol. 52, pp. 1763–1779.
- [13] REINERT, K., LENHOF, H.-P., MUTZEL, P., MEHLHORN, K., AND KECECIOGLU, J. D. A branch-and-cut algorithm for multiple sequence alignment. In *Proceedings of the 1st Annual International Conference on Computational Molecular Biology (RECOMB)* (Santa Fe, NM, 1997), ACM Press, pp. 241–250.
- [14] TANG, C. Y., LU, C. L., CHANG, M. D.-T., TSAI, Y.-T., SUN, Y.-J., CHAO, K.-M., CHANG, J.-M., CHIOU, Y.-H., WU, C.-M., CHANG, H.-T., AND CHOU, W.-I. Constrained multiple sequence alignment tool development and its application to rna family alignment. In *Proceedings of the First IEEE Computer Society Bioinformatics Conference (CSB 2002)* (2002), pp. 127–137.
- [15] WANG, L., AND JIANG, T. On the complexity of multiple sequence alignment. In *Journal of Computational Biology* (1994), vol. 1, pp. 337–348.

	Data Set 0	Data Set 1
no. sequences	6	6
Max seq length	125	185
Constraint	HKH	HKH
Sequence ID	H-RNase3 H-RNase2 BP-RNaseA BS-RNase H-RNaseA H-RNase4 RC-RNase	gi/119124/sp/P12724/ecp_human gi/2500564/sp/P70709/ecp_rat gi/13400006/pdb/ldyt/ gi/20930966/ref/xp_142859.1/ gi/20873960/ref/xp_127690.1/ gi/20930966/ref/xp_142859.1/
	Data Set 2	Data Set 3
no. sequences	6	5
Max seq length	186	327
Constraint	HKSH	HKH
Sequence ID	gi/20930966/ref/XP_142859.1/ 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/10068295/gb_aae40716.1/ gi/17549935/ref/np510780.1/ gi/28509297/ref/xp282983.1/ gi/28499937/ref/xp204162.2/ gi/4902995/dbj/baa77929.1/

Data set 1

```

Seq1: -----MVIS---PGSLLLVFLLS--LDV--IPP-TLAQDNRYRKNFL---N---QH-YDAKP-TGRD
Seq2: -----KET-AAAKFE---R---QH-MDSSTSAASS
Seq3: -----MTMS---PCPLLLVFLG--LVV--IPP-TLAQNE-RYEKFL---R---QH-YDAKP-NGRD
Seq4: -----MVVD---LPRYLPLLLL---LEL--WEP-MYLLCS-QPKGLS---R---AHWFEIQH-VQTS
Seq5: -----MKPLVIKFAWPLPLLLLLLPPKLGNYWDFGEYELNP-EVRDFI---R---EYESTGPTKPTV
Seq6: MDDEWERPEQATSAAEHPHTAA---QAAYNLADKLG--LEVPSWNPTSSLRQ-KDRKLESNRPAPSQKFYTEP IHNST
      *
Seq1: YRYCESMMKK-RKLT--SPCK-EVNT-FIH-----DTKNNIKAICGENRPYGVNLR-I-SNSRFQ---ITTC
Seq2: SNYCNQMMKS-RNLTK-DRCK-PVNT-FVH-----ESLADVQAVCSQKNVACKNGQTN-CYQSYSTMSITDC
Seq3: DRYCESMMKE-RKLT--SPCK-DVNT-FIH-----GTKKNIRAICGKKGSPYGENFRI-SNSPFQ---ITTC
Seq4: RQPCNTAMRGVNNYT--QHCK-QINT-FLH-----ESFQNVAAATCSLHNITCKNGRKN-CHESAEPVKMTDC
Seq5: KRIIEMITIGDQPFNDYDYNTELRTKQIHYKGRCPYEHYIAGVPYELVKACDGEVQCKNGVKS-CRRSMNLIEGVRC
Seq6: YPRCDDPMLVNNRYR--PRCK-DIDT-FLH-----TSFANV-GVCGHPSGFCKEHSANCHNSSSQVPIIVC
      *
Seq1: KHKG-GSPKPPCQYKAF---K-DFR--YIVIACE----DG--W---PVHFDEFISM
Seq2: RETG-SSKYPNCAKYTTQANK-----HIIVACE----GNP-Y--VPVHFDAS-V--
Seq3: THSG-ASPRPPCGYRAF---K-DFR--YIVIACE----DG--W---PVHFDEFISP
Seq4: SHTG-GA-YPNCRYSSD---K-QYK--FFIVACEH-PKEDPPYQLVPVHLDKI-V--
Seq5: VLET-GQMTNCTY-----KTILMIGYPVVSQW--DEETKIF--IPDHYNMMLPK
Seq6: NLTPGRYTYQCRYQM---KGSVE--YYTVACKPRTPWDSPIYPVVPVHLHGT-F--

```

Data set 2

```

Seq1: MVPKLFTSQICLLLLLGLMGVEGSLHARPPQFTRAQWFAIQHISLNP-PRCTIAMRAINNYR--W--RC-KNQNTFLRRTTFANVV
Seq2: MGLKLESRLCLLLSLGLVMLAS--CQPP--TPSQWFEIQHIYNRAYPRCNDAMRHRNRF--G--HC-KDINTFLHTSFASVV
Seq3: -----RPPQFTRAQWFAIQHISLNP-PRCTIAMRAINNYR--W--RC-KNQNTFLRRTTFANVV
Seq4: MGSKTLKSQICLLLLLGLLMLVSCQAQTP--S--QWFEIQHIYNSAYPRCDDAMRVIHGYSGVYLQRQEK---Y-K-----
Seq5: MVVDL-PRYLPLLLLLELWEPMYLLCSQPKGLSRAHWFEIQHVQTSR-QPCNTAMRGVNNYT--Q--HC-KQINTFLHESFQNV
Seq6: MGSKTLKSQICLLLLLGLLMLVSCQAQTP--S--QWFEIQHIYNSAYPRCDDAMRVIHGYSGVYLQRQEK---Y-K-----
      *
Seq1: NVCGNQSIIRCPHNRNLTNNCHRSRFRVPLHCDLINPGAQNISNCRYADRPGRFYVACDNRDPRDPRYPVVPVHLDTTI
Seq2: GVCGRNIPCG-NRTYRNCHNSRYRVSITFCNLITTP-ARIYTYCRYQTTRSFKFYTVGCDPRTPRDPSMYPVVPVHLDRI
Seq3: NVCGNQSIIRCPHNRNLTNNCHRSRFRVPLHCDLINPGAQNISNCRYADRPGRFYVACDNRDPRDPRYPVVPVHLDTTI
Seq4: --C-----HD-S-S---SK--IPVIICDLITWSNQH-THCRYKTTVAMKSYTVACNPRTPRNSPRYPFVPCHLDTI
Seq5: TCSLHN%ITCKNGR--KNCHESAEPVKMTDCSHTG-GA--YPNCRYSSDKYKFFIVACEHPKEDPP-YQLVPVHLDKIV
Seq6: --C-----HD-S-S---SK--IPVIICDLITWSNQH-THCRYKTTVAMKSYTVACNPRTPRNSPRYPFVPCHLDTI

```