Efficient Constrained Multiple Sequence Alignment with Performance Guarantee

Francis Y.L. Chin* N.L. Ho T.W. Lam† Prudence W.H. Wong M.Y. Chan
Department of Computer Science and Information Systems,
The University of Hong Kong, Hong Kong.
$\{$*chin,nlho,twlam,whwong,mychan*$\}$*@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 [2, 5, 8, 6, 10, 12, 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[1] [2, 16]. 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, 4, 11], while some others work well in practice [9, 15]. 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] 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.* [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 α 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 $O(\alpha k n^4)$ 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.*

*This research was supported in part by Hong Kong RGC Grant HKU7019/00E
†This research was supported in part by Hong Kong RGC Grant 10204076
[1] 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.

	Time	Space	Approx. Ratio
Tang *et al.* 's algorithm [14]	$O(\alpha k n^4)$	$O(\alpha n^4)$	–
Improved Tang *et al.* 's algorithm (this paper)	$O(\alpha k^2 n^2)$	$O(\alpha n^2)$	–
Center-star (this paper)	$O(\alpha C k^2 n^2)$	$O(\alpha k^2 n^2)$	$2 - \frac{2}{k}$

Figure 1. Performance of constrained multiple sequence alignment approximation algorithms.

No. sequences		7	6	6	5
Max length		125	185	186	327
Constraint (α)		3	3	4	3
Tang's	score	46319	71208	63315	***
Ctr-star	score	40051	49874	45241	57325
Tang's	time (sec)	127	381	254	***
Ctr-star	time (sec)	25	77	82	482
Tang's	space (MB)	425	1192	654	***
Ctr-star	space (MB)	4.2	2.8	3.1	6.2

*** – Memory exhausted

Figure 2. Alignment scores of CMSA algorithms

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 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 Σ be the set of characters (residues), $S = \{S_1, S_2, ..., S_k\}$ be a set of k sequences, with maximum length n, over Σ. 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 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 (residues), where $x, y \in \Sigma \cup \{-\}$, the *pair-wise score* of two length-n' sequences S'_1 and S'_2 is defined as $\sum_{1 \leq x \leq n'} \delta(S'_1[x], S'_2[x])$. 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 i-th row of A gives S_i for $1 \leq i \leq k$. The *sum-of-pair (SP)* score of an MSA is defined as the sum of the pair-wise scores of all pairs of the sequences, $\sum_{1 \leq p < q \leq k} \sum_{1 \leq x \leq n'} \delta(A_{p,x}, A_{q,x})$ where each row of the alignment matrix is treated as a sequence, A_p is the p-th row of the alignment matrix A and $A_{p,x}$ is the character at the p-th row and the x-th column of A. It is shown in [2, 16] that finding an alignment matrix with the minimum sum-of-pair alignment 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 $P = (P[1], P[2], \ldots, P[\alpha])$, where P is a common subsequence of $S_i \in \{S_1, S_2, \ldots, S_k\}$. 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 $\{c_1, c_2, \ldots, c_\alpha\}$ where $1 \leq c_1 < c_2 < \ldots < c_l < \ldots < c_\alpha \leq n'$, and for all $1 \leq i \leq k$ and all $1 \leq l \leq \alpha$, we have $A_{ic_l} = P[l]$.

Let A be a constrained alignment matrix for $S = \{S_1, S_2, \ldots, S_k\}$ and the constrained sequence P. Define $sp\_score(A)$ as 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 and only if for all S and P,

$$\frac{sp\_score(A_s')}{sp\_score(A_s^*)} \leq \phi.$$

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 $P[\gamma] = S_1'[c_\gamma] = S_2'[c_\gamma]$ for $1 \leq \gamma \leq \alpha$, and $1 \leq c_1 < c_2 < \ldots < c_\alpha \leq n'$. 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

The optimal constrained pair-wise algorithm presented in [14] has 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 x-th character and ending at the y-th character of S. The dynamic programming computes the optimal CPSA incrementally by considering the pair-wise alignment of $S_1[1..1]$ with $S_2[1..1]$, ..., $S_1[1..i]$ with $S_2[1..j]$ and so on, for $1 \leq i \leq |S_1|$ and $1 \leq j \leq |S_2|$.

We define $D(i, j, \gamma)$ to be the optimal constrained pairwise 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 $D(i', j', \gamma')$ with smaller i', j' and γ' values.

Theorem 3.1 *For any* $0 \leq i \leq n_1, 0 \leq j \leq n_2$ *and* $0 \leq \gamma \leq \alpha$, $D(i, j, \gamma) =$

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

with boundary conditions

$$\begin{array}{rcl} D(0,0,0) & = & 0, \\ D(i,0,\gamma) & = & \infty \quad \textit{for } \gamma \geq 1,\ 0 \leq i \leq n_1, \textit{ and} \\ D(0,j,\gamma) & = & \infty \quad \textit{for } \gamma \geq 1,\ 0 \leq j \leq n_2. \end{array}$$

Proof. To align $\{S_1[1..i], S_2[1..j]\}$ with $P[1..\gamma]$ in γ columns, there are four possible cases.

- If $S_1[i] = S_2[j] = P[\gamma]$, we can align $S_1[i]$ and $S_2[j]$ with $P[\gamma]$ while aligning $\{S_1[1..(i-1)], S_2[1..(j-1)]\}$ with $P[1..(\gamma-1)]$. 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 $P[1..\gamma]$. 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 $P[1..\gamma]$. Then the score is $D(i-1, j, \gamma) + \delta(S[i], -)$.
- Similarly, if $j > 0$, we can align $S_2[j]$ with a space while aligning $\{S_1[1..i)], S_2[1..(j-1)]\}$ with $P[1..\gamma]$. 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 the minimum. Therefore, $D(i, j, \gamma)$ can be computed by taking the minimum of the above four values. □

Algorithm 1 The dynamic programming algorithm for the optimal CPSA score.

1. Initialize $D(0,0,0) = 0$, $D(i,0,\gamma) = \infty$, and $D(0,j,\gamma) = \infty$ for $0 \leq i \leq n_1$, $0 \leq j \leq n_2$, and $1 \leq \gamma \leq \alpha$.

2. For γ = 0 to α 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 3.1 in terms of $D(i-1,j-1,\gamma-1)$, $D(i-1,j-1,\gamma)$, $D(i-1,j,\gamma)$ and $D(i,j-1,\gamma)$.

Based on Theorem 3.1, 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. 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.

Theorem 3.2 *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 indexes i, j or γ decreases 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 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.

□

4. Constrained Multiple Sequence Alignment

In this section, we study the constrained multiple sequence alignment problem. In Section 4.1, we reduce the time and space complexities of the progressive CMSA heuristic algorithm presented in [14] from $O(\alpha k n^4)$ to $O(\alpha k^2 n^2)$. In Section 4.2, we show an algorithm that computes the optimal CMSA in $O(\alpha n^k)$ time based on the sum-of-pair score. In Section 4.3, we apply the center-star approximation algorithm [4] to the CMSA problem, and show that the constrained version of the center-star 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 distance function $\delta(x,y)$ follows the *triangular inequality*, i.e. $\delta(x,y) \leq \delta(x,z) + \delta(z,y)$, for any $x,y,z \in \Sigma \bigcup \{-\}$, 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 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* [3] 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 Section 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 Section 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) $D(\{0\}^k; 0) = 0$

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

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

where $\epsilon_j = 0$ *or* 1, $\epsilon_j S_j[i_j]$ with $\epsilon_j = 0$ represents 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 that computes the optimal CMSA for multiple sequences, which is a generalization of the dynamic programming for the CPSA problem. Practically, optimal CMSA can be computed for less than 6 short sequences of length at most 200 [7]. In the following section, we present another approximation algorithm for the CMSA problem.

4.3. The Center-star Alignment Approximation for CMSA

For the unconstrained multiple sequence alignment problem, Gusfield [4] presented the center-star algorithm which is an approximation algorithm with an approximation ratio $(2 - \frac{2}{k})$. Based on the center-star approximation algorithm, we derive an approximation algorithm for CMSA that yields an approximation ratio $(2 - \frac{2}{k})$.

For a set of k sequences $S = \{S_1 \ldots S_k\}$, 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 constrained version of the center-star approximation algorithm

i) For each S_i in S, treat S_i as the center sequence S_c and for each list of positions $(c_1, c_2, \ldots, c_\alpha)$ that S_c is aligned with P, i.e. $P[\gamma] = S_c[c_\gamma]\ \forall 1 \leq \gamma \leq \alpha$, align all other S_j with S_c at the positions specified by $(c_1, c_2, \ldots, c_\alpha)$.

ii) Find the S_c and $(c_1, c_2, \ldots, c_\alpha)$ 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 $(c_1, c_2, \ldots, c_\alpha)$ into a constrained alignment matrix.

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

Aligning a sequence and the center sequence with a list of constrained positions Without loss of generality, assume that the center sequence is S_1. Given $c_1, c_2, \ldots, c_\alpha$, we perform the CPSA algorithm of S_1 to $S_2 \ldots S_k$ under $(c_1, c_2, \ldots, c_\alpha)$, using a slightly modified recurrence in Theorem 3.1, treating S_c as S_1 and S_i as S_2 ($2 \leq i \leq k$).

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

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 $D(c_\gamma, j, \gamma)$ with $1 \leq j \leq n_2$. This implies that the alignment occurs in S_1 at the positions $(c_1, c_2, \ldots, c_\alpha)$.

Optimal center sequence & the constrained positions Consider each sequence S_i and a list of positions of occurrence $(c_1, c_2, \ldots, c_\alpha)$ of P in S_i where $1 \leq i \leq k$. The combination $(S_i; c_1, c_2, \ldots, c_\alpha)$ that gives the minimum sum of constrained pair-wise alignment scores with other sequences under the positions $(c_1, c_2, \ldots, c_\alpha)$ is selected as the center sequence S_c and the list of positions to be aligned with P. The time for locating $(S_c; c_1, c_2, \ldots, c_\alpha)$ is then $O(\alpha k C n^2)$ for k sequences, where C is the maximum number of sets of positions that P appears in each sequence.

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 $(c_1, c_2, \ldots, c_\alpha)$. There are $(k-1)$ constrained pair-wise alignments, one for S_1 aligning with S_j, for $2 \leq j \leq k$. Suppose $|S_1| = n$, and let $A_{1,j}$ be the optimal constrained pair-wise alignment of S_1 and S_j under $(c_1, c_2, \ldots, c_\alpha)$. Define s_0 and s_n be the longest sequences of spaces inserted before $S_1[1]$ and after $S_1[n]$ in all $(k-1)$ $A_{1,j}$'s, respectively. Similarly for $1 \le i \le n-1$, let s_i be the longest sequence of spaces between $S_1[i]$ and $S_1[i+1]$ in all $(k-1)$ $A_{1,j}$'s. Initially, set A to contain a single row $S'_1 = [s_0 \oplus S_1[1] \oplus s_1 \oplus S_1[2] \oplus \ldots \oplus s_{i-1} \oplus S_1[i] \oplus s_i \oplus \ldots \oplus S_1[n] \oplus s_n]$, where the $\oplus$ operator denotes the string concatenation operation. Notice that $|S'_1| = |S_1| + \sum_{0 \le i \le n} |s_i|$.

For each S_j with $2 \le j \le k$, add S_j to A according to the optimal constrained pair-wise sequence alignment of S_1 and S_j, $(\frac{\bar{S_1}}{\bar{S_j}})$, i.e. insert columns of spaces to $(\frac{\bar{S_1}}{\bar{S_j}})$ 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 $(c_1, c_2, \ldots, c_\alpha)$. Therefore, A is the constrained multiple sequence alignment for $S = \{S_1, \ldots, S_k\}$ and P with the minimum star-sum score.

Performance of the constrained version of the center-star algorithm The following theorem shows that the center-star approximation algorithm for CMSA has an approximation ratio $(2 - \frac{2}{k})$. 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 4.1 *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, 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 $ss\_score_i(A)$ be the star-sum score of A with the row A_i as the center sequence. Let A^s_c be the optimal center sequence of A^s. 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,

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

On the other hand,

$$\begin{aligned} & sp\_score(A^s) \\ = \;& \tfrac{1}{2} \textstyle\sum_{1 \le i,j \le k} \Delta(A^s_i, A^s_j) \\ \le \;& \tfrac{1}{2} \textstyle\sum_{1 \le i,j \le k} (\Delta(A^s_i, A^s_c) + \Delta(A^s_c, A^s_j)) \\ = \;& \tfrac{2(k-1)}{2} \textstyle\sum_{1 \le i \le k, i \ne c} \Delta(A^s_i, A^s_c) \\ = \;& (k-1)(ss\_score_c(A^s)). \end{aligned}$$

Note that the inequality above (i.e., from line 2 to line 3) is due to the triangular inequality. Therefore, $\frac{sp\_score(A^s)}{sp\_score(A^*)} \le \frac{2(k-1)}{k} = 2 - \frac{2}{k}$, and the theorem follows. □

5. Empirical Results

In Section 5.1, we first evaluate the performance of our CPSA algorithm. We then evaluate the performance of (i) constrained center-star approximation algorithm, (ii) the improved progressive alignment, and (iii) the original CMSA algorithm presented in [14], on four sets of RNase sequences taken from the NCBI[2]. In Section 5.2, we show that, in practice, C, the maximum number of occurrences of the constrained sequence as a sub-sequence in the input 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 GHz CPU with 4GB of 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 (Section 3.1) and the constrained pair-wise sequence algorithm in [14] with 3 constrained characters. The CPSA algorithm presented in [14] took 127 seconds and 400 MB memory. For the same problem instance, our CPSA algorithm runs 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 [14] and our constrained center-star and the improved progressive CMSA algorithms. We ran these CMSA algorithms on four sets of RNase sequences. In data set 0, we used the same set of 7 RNase sequences used in [14]. 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 consists of 5 long RNase sequences (with maximum length 327). Since we cast the CMSA as a minimization 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[3] as described in [14]. The performance of these three CMSA algorithms is shown in Figure 3. The alignments of data

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

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

	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

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

sets 0 to 3 using the constrained center-star algorithm are shown in Figures 6 to 9. Due to the page limit, the alignment matrices are divided into blocks of 40 characters, the first 40 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 la.*'s progressive CMSA algorithm [14] 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 sequence alignments with better SP alignment scores (see Figure 3).

5.2 Number of Constraint Occurrences

We refer to [14] for an application used for CMSA. In their experiments, 7 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 the number of occurrences (C) 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 $O(\alpha k n^4)$ algorithm described in [14]. There are total 2869 sequences from the NCBI. In these 2869 RNase sequences, a total of 2266 sequences, or 78%, contained 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 Figure 5.

CMSA is usually done for a set of sequences of approximately the same length. As shown in the Figure 5, for CMSA of sequences with length below 500, $C = 105$ on average and $C \leq 241$ for 90% of these sequences. 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 $O(500\alpha k^2 n^2)$ on average which is much shorter compared to the running time of $O(\alpha k n^4)$ in [14].

6. 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 [14]. 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.

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] 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.

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

Figure 3. Alignment scores of CMSA algorithms

[4] D. Gusfield. Efficient methods for multiple sequence alignment with guaranteed error bounds. In *Bull. Math. Biol., 30*, volume 30, pages 141–154, 1993.

[5] D. Gusfield, editor. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, 1997.

[6] 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.

[7] 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.

[8] H. J. Nicholas, A. Ropelewski, and D. D. II. Strategies for multiple sequence alignment. *BioTechniques*, 32:572–591, Mar 2002.

[9] 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.

[10] R. B. Peter Clote. *Computational Molecular Biology: An Introduction*. John Wiley and Sons, Ltd, Aug 2000.

[11] P. A. Pevzner. Multiple alignment, communication cost, and graph matching. In *SIAM J. Applied Mathematics*, volume 52, pages 1763–1779, 1992.

[12] A. Phillips, D. Janies, and W. Wheeler. Multiple sequence alignment in phylogenetic analysis. *Molecular Phylogenetics and Evolution*, 16-3:317–330, Sep 2000.

[13] 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 the1st Annual International Conference on Computational Molecular Biology (RECOMB)*, pages 241–250, Santa Fe, NM, 1997. ACM Press.

[14] 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)*, pages 127–137, 2002.

[15] J. Thompson, D. Higgins, and T. 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.

[16] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. In *Journel of Computational Biology*, volume 1, pages 337–348, 1994.

Data set 0
Number of sequences : 7
Max sequence length : 125
Constrained sequence : `HKH`
Sequence ID

```
     Seq1: H-RNase3
     Seq2: H-RNase2
     Seq3: BP-RNaseA
     Seq4: BS-RNase
     Seq5: H-RNaseA
     Seq6: H-RNase4
     Seq7: RC-RNase

                  *
-K--E-TA-A-AK-FERQHMDSSTSAASSSNYCNQMMKSR
-RPPQFTR-A-QW-FAIQHI-S-LNPPR----CTIAMRAI
MK--P-PQFTWAQWFETQHINM-TSQQCN-N-AMQVI-N-
-K--E-SA-A-AK-FERQHIDSSTSSVSSSNYCNEMMTSR
-K--E-SR-A-KK-FQRQHMDSDSSPSSSSTYCNQMMRRR
-M--Q-DG-MYQR-FLRQHVHPEETGGSDR-YCNLMMQRR
----Q-NW-A-T--FQQKHI-INTPIINC-N--TIMDNNI

       *
NLTKDRCKPVNTFVHESLADVQAVCSQKNVAC-KN-GQT-
NNYRWRCKNQNTFLRTTFANVVNVCGNQSIRC-PH-NRTL
NFQR-RCKNQNTFLRTTFANVVNVCGNPNITCPSNRSRN-
NLTQDRCKPVNTFVHESLADVQAVCSQKNVAC-KN-GQT-
NMTQGRCKPVNTFVHEPLVDVQNVCFQEKVTC-KN-GQG-
KMTLYHCKRFNTFIHEDIWNIRSICSTTNIQC-KN-GKM-
YIVGGQCKRVNTFIISSATTVKAICT--GVIN-MN-VLS-


N-CYQSYSTMSITDC--R-ETGSSKY-PNCAYKTTQANKH
NNCHRSRFRVPLLHCDLI-NPGAQNI-SNCRYADRPGRRF
N-CHHSGVQVPLIHC--NLTTPSPQNISNCRYAQTPANMF
N-CYQSYSAMSITDC--R-ETGNSKY-PNCAYQTTQAEKH
N-CYKSNSSMHITDC--R-LTNGSRY-PNCAYRTSPKERH
N-CHEG--VVKVTDC--R-DTGSSRA-PNCRYRAIASTRR
T--TR-FQ-LN-T-C--T-RTSITPR-P-CPYSSRTETNY

                      *
IIVACEG-NP------Y-V-PVHFDA-S--V
YVVACDNRDPRDSPR-YPVVPVHLDT-T--I
YIVACDNRDPRRDPPQYPVVPVHLD--R--I
IIVACEG-NP------Y-V-PVHYDA-S--V
IIVACEG-SP------Y-V-PVHFDA-S--V
VVIACEG-NP------Q-V-PVHFDG-----
ICVKCE--NQ------Y---PVHFAGIGRCP
```

Figure 6. Alignment of 7 sequences in data set 0, with $P =$ HKH

Data set 1
Number of sequences : 6
Max sequence length : 185
Constrained sequence : `HKH`
Sequence ID

```
Seq1: gi|119124|sp|P12724|ecp_human
Seq2: gi|2500564|sp|P70709|ecp_rat
Seq3: gi|13400006|pdb|1dyt|
Seq4: gi|20930966|ref|xp_142859.1|
Seq5: gi|20873960|ref|xp_127690.1|
Seq6: gi|20930966|ref|xp_142859.1|
```

```
Seq1: ------------------MVIS---PGSLLLVFLLS--LD
Seq2: ----------------------------------------
Seq3: ------------------MTMS---PCPLLLVFVLG--LV
Seq4: ------------------MVVD---LPRYLPLLLL---LE
Seq5: ---------------MKPLVIKFAWPLPLLLLLLLPPKLQ
Seq6: MDDEWERPEQATSAAEHPHTAA---QAAYNLADKLG--LE

Seq1: V--IPP-TLAQDNYRYKNFL---N---QH-YDAKP-TGRD
Seq2: ----------KET-AAAKFE---R---QH-MDSSTSAASS
Seq3: V--IPP-TLAQNE-RYEKFL---R---QH-YDAKP-NGRD
Seq4: L--WEP-MYLLCS-QPKGLS---R---AHWFEIQH-VQTS
Seq5: GNYWDFGEYELNP-EVRDFI---R---EYESTGPTKPPTV
Seq6: VPSWNPTTSSLRQ-KDRKLESNPRPAPSQKFYTEPIHNST

                                   *
Seq1: YRYCESMMKK-RKLT--SPCK-EVNT-FIH----------
Seq2: SNYCNQMMKS-RNLTK-DRCK-PVNT-FVH----------
Seq3: DRYCESMMKE-RKLT--SPCK-DVNT-FIH----------
Seq4: RQPCNTAMRGVNNYT--QHCK-QINT-FLH----------
Seq5: KRIIEMITIGDQPFNDYDYCNTELRTKQIHYKGRCYPEHY
Seq6: YPRCDDPMLVVNRYR--PRCK-DIDT-FLH----------

Seq1: ---DTKNNIKAICGENGRPYGVNLRI-SNSRFQ---ITTC
Seq2: ---ESLADVQAVCSQKNVACKNGQTN-CYQSYSTMSITDC
Seq3: ---GTKKNIRAICGKKGSPYGENFRI-SNSPFQ---ITTC
Seq4: ---ESFQNVAATCSLHNITCKNGRKN-CHESAEPVKMTDC
Seq5: IAGVPYGELVKACDGEEVQCKNGVKS-CRRSMNLIEGVRC
Seq6: ---TSFANV-GVCGHPSGFCKEHKSANCHNSSSQVPIIVC

                          *
Seq1: KHKG-GSPKPPCQYKAF---K-DFR--YIVIACE-----D
Seq2: RETG-SSKYPNCAYKTTQANK------HIIVACE-----G
Seq3: THSG-ASPRPPCGYRAF---K-DFR--YIVIACE-----D
Seq4: SHTG-GA-YPNCRYSSD---K-QYK--FFIVACEH-PKKE
Seq5: VLET-GQQMTNCTY------KTILMIGYPVVSCQW--DEE
Seq6: NLTTPGRTYTQCRYQM----KGSVE--YYTVACKPRTPWD

               *
Seq1: G--W---PVHFDESFISM
Seq2: NP-Y--VPVHFDAS-V--
Seq3: G--W---PVHFDESFISP
Seq4: DPPYQLVPVHLDKI-V--
Seq5: TKIF--IPDHIYNMSLPK
Seq6: SPIYPVVPVHLHGT-F--
```

Figure 7. Alignment of 6 sequences in data set 1, with P =HKH

Data set 2
Number of sequences : 6
Max sequence length : 186
Constrained sequence : `HKSH`
Sequence ID

```
Seq1: gi|20930966|ref|XP_142859.1|
Seq2: gi|119124|sp|P12724|ECP_HUMAN
Seq3: gi|2500564|sp|P70709|ECP_RAT
Seq4: gi|13400006|pdb
Seq5: gi|20930966|ref|XP_142859.1|
Seq6: gi|20873960|ref|XP_127690.1|
```

```
Seq1: MVPKLFTSQICLLLLLGLMGVEGSLHARPPQFTRAQWFAI
Seq2: MGLKLLESRLCLLLSLGLVLMLAS--CQPP--TPSQWFEI
Seq3: ---------------------------RPPQFTRAQWFAI
Seq4: MGSKTLKSQLCLLLLLGLLLMLVSCQAQTP--S--QWFEI
Seq5: MVVDL-PRYLPLLLLLELWEPMYLLCSQPKGLSRAHWFEI
Seq6: MGSKTLKSQLCLLLLLGLLLMLVSCQAQTP--S--QWFEI

       *                             *
Seq1: QHISLNP-PRCTIAMRAINNYR--W--RC-KNQNTFLRTT
Seq2: QHIYNRAYPRCNDAMRHRNRFT--G--HC-KDINTFLHTS
Seq3: QHISLNP-PRCTIAMRAINNYR--W--RC-KNQNTFLRTT
Seq4: QHIYNSAYPRCDDAMRVIHGYSGVYLQRQEK----Y-K--
Seq5: QHVQTSR-QPCNTAMRGVNNYT--Q--HC-KQINTFLHES
Seq6: QHIYNSAYPRCDDAMRVIHGYSGVYLQRQEK----Y-K--

                                *
Seq1: FANVVNVCGNQSIRCPHNRTLNNCHRSRFRVPLLHCDLIN
Seq2: FASVVGVCGNRNIPCG-NRTYRNCHNSRYRVSITFCNLTT
Seq3: FANVVNVCGNQSIRCPHNRTLNNCHRSRFRVPLLHCDLIN
Seq4: -------C--------HD-S-S----SK--IPVIICDLIT
Seq5: FQNVATCSLHN-ITCKNGR--KNCHESAEPVKMTDCSHTG
Seq6: -------C--------HD-S-S----SK--IPVIICDLIT

Seq1: PGAQNISNCRYADRPGRRFYVVACDNRDPRDSPRYPVVPV
Seq2: P-ARIYTQCRYQTTRSRKFYTVGCDPRTPRDSPMYPVVPV
Seq3: PGAQNISNCRYADRPGRRFYVVACDNRDPRDSPRYPVVPV
Seq4: WSNQH-THCRYKTTVAMKSYTVACNPRTPRNSPRYPFVPC
Seq5: -GA--YPNCRYSSDKQYKFFIVACEHPKKEDPP-YQLVPV
Seq6: WSNQH-THCRYKTTVAMKSYTVACNPRTPRNSPRYPFVPC

      *
Seq1: HLDTTI
Seq2: HLDRIF
Seq3: HLDTTI
Seq4: HLDGTI
Seq5: HLDKIV
Seq6: HLDGTI
```

Figure 8. Alignment of 6 sequences in data set 1, with P =HKSH

Data set 3
Number of sequences : 6
Max sequence length : 185
Constrained sequence : HKH
Sequence ID

```
   Seq1: gi|10068295|gb|AAE40716.1|
   Seq2: gi|17549935|ref|NP_510780.1|
   Seq3: gi|28509297|ref|XP_282983.1|
   Seq4: gi|28499937|ref|XP_204162.2|
   Seq5: gi|4902995|dbj|BAA77929.1|
```

```
Seq1: -MP--INIISDSVYVVNAVLALETAGNFKQSSPVSEILMK
Seq2: MLRWLVALLSHSCFVSKGGGMFYAVRKGRQTGVYRTWAE-
Seq3: -MR--VNGRNLTNLRFADDIVLIANHPNTASKMLQELVQK
Seq4: -MP--INIISDSVYVVNAVLALETAGNFKQSSPVSEILMK
Seq5: -M---V-LIS----LLNPETQ-NRSQNMSQNNPLRALLDK


Seq1: IQNCILMREHPFYIQHIRAHTSLPGPMVKGNAIADSATRD
Seq2: CQQ-QVNRFPSASFKKFAT--EKEAWAFVGAGPPDGQQSA
Seq3: CSE-VGLEINTGKTKVLRNRFADPSKVYFGSPSPTTQLDD
Seq4: IQNCILMHEHPFYIQHIRAHTSLPGPMVKGNAIADSATRD
Seq5: -QD-ILL------LDGAMA-TELEA---RGCNLAD-SLWS

                           *
Seq1: M---VFL---SQSSIESAKKFH-QLYYVPASTL--RQ-KF
Seq2: P---AET--HGASAVAQENASH-RE-EPETDVLCCNACKR
Seq3: VDEYIYLGRQINAQNNLMPEIH-R--RRRAA----WA-AF
Seq4: M---VFL---SQSSIESAKNFH-QLYHVPASTL--RQ-KF
Seq5: A---KVL---VENP-ELIREVHLDYYRAGAQCA--ITASY

                            *
Seq1: KLTRK--EARDIVLQCG-----KCVE--------FVNA-P
Seq2: RYEQS--TNEEHTVRRA-----KHDE--------EQST-P
Seq3: NGIKN--TTDSITDK-------K----------------I
Seq4: KLTRK--EARNIVLQCG-----KCVE--------FVNA-P
Seq5: QATPAGFAARGLDEAQSKALIGKSVELARKAREAYLAENP


Seq1: SVG-VNPRG-LRPLD-VWQMD--GMHIP-SFG-KLQ-YV-
Seq2: VVS-EAKFSYMGEFAVVYTDGCCSGNGRNRARAGIGVYWG
Seq3: RAN-LFDSI-VLPAL-TYGSE--AWTFTKALSERVR-IT-
Seq4: SVG-VNPRG-LRPLD-VWQMD--VTHIP-SFG-KLQ-YV-
Seq5: QAGTLLVAGSVRPYG-AYLTD--GSEYRGDYHCTVEAFQA

        *
Seq1: --H-----------------------VSIDTSSGVLHASP
Seq2: PGH------------------------------------P
Seq3: --H-ASLERRLVG--ITLTQQRERDLHREDIRTMSLVRDP
Seq4: --H-----------------------VSIDTSSGVLHASP
Seq5: F-HRPRVEALLVAGADLLACETLPNFSEIEALAELLTAYP


Seq1: LTGEKAVH-VIS-HCLE-----AWAA----WGKPLVLKTD
Seq2: LN-------ISE-R-LP-----GRQT-----N--------
Seq3: LN-----F-VKK-RKLGWAGHVARRK----DGRWTTLMTE
Seq4: LTGEKAVH-VIS-HCLE-----AWAA----WGKPLVLKTD
Seq5: RARAWFSFTLRDSEHLS-DGTPLRDVVALLAGYPQVVALG
```

```
Seq1: NGPAYTSSKFSQFCKQMQVKHITGLPYNPQG---QGIIER
Seq2: ----------------------------------------
Seq3: WRPYGWKRPVGRPPMRWTDSLRKEITTRDAD---GEVITP
Seq4: NGPAYTSSKFSQFCKQM-----------------------
Seq5: INCIALENTTAALQHLHGLTVLPLVVYPNSGEHYDAVSKT


Seq1: AHHT------LKQYL-QKQKGGIEAMTPKMALSLTIFTLN
Seq2: ----------------------------------------
Seq3: WSTI------AKDRK-EWLAVIRRNTTNS-----------
Seq4: ----------------------------------------
Seq5: WHHHGEHCAQLADYLPQWQAAGARLIGGCCRTTPADIAAL


Seq1: F---
Seq2: ----
Seq3: ----
Seq4: ----
Seq5: KARS
```

Figure 9. Alignment of 5 sequences in data set 3, with $P =$HKH