One-Annotated Constrained Sequence Alignment

YUN-SHENG CHUNG, CHUAN YI TANG
Department of Computer Science
National Tsing Hua University
101, Section 2 Kuang Fu Road, Hsinchu, Taiwan 300
REPUBLIC OF CHINA

Abstract: The constrained multiple sequence alignment (CMSA) problem is to align a set of strings such that the given patterns (the constraint) appear in the same positions in a specified order in each of the strings in the resulting alignment. The best previous result for the pair-wise version takes $O(mn^2)$ time and space [2, 10], where m is the number of patterns (defined later) and n is the maximum string lengths. In this paper, we deal with the pair-wise case when the positions of occurrences of the patterns in one of the strings are given. This version arises in applications naturally but is not discussed previously [8, 2, 10]. In this paper, we present an algorithm taking $O(n^2)$ time and $O(n + r)$ space for this version, where r is the number of occurrences of all the patterns. This result in turn improves the 2-approximation algorithm proposed in [2] for CMSA from $O(Ck^2mn^2)$ time and $O(k^2mn^2)$ space to $O(Ck^2n^2)$ time and $O(kn)$ space for the original problem, where k is the number of sequences and C is the maximum number of valid "constrained lists" (defined later).

Key-Words: biological sequence comparison, constrained sequence alignment, computational biology

1 Introduction

Sequence alignment is essential in computational biology. Classical methods consider the alignment of residues on the sequences. This problem has been extensively studied (see, for example, [6]). As biological knowledge, predictions and hypotheses grow rapidly, it turns out to be important and desirable to incorporate biological information into alignment procedures, instead of merely comparing residues or corresponding quantities (e.g. polarity).

Constrained sequence alignment is proposed by Tang *et al.* [8]. They considered RNases for an example. RNases contain three conserved His12 (H), Lys41 (K) and His119 (H) active site residues as their major structural feature [8]. Thus if we are aligning RNase sequences, it is demanding that each of these conserved residues appears in the same columns in the resulting alignment.

In a follow-up work by Chin *et al.* [2], a more efficient algorithm for the pair-wise constrained alignment is proposed. The algorithm runs in $O(mn^2)$ time and space. For CMSA, they give a 2-approximation algorithm which takes $O(Cmk^2n^2)$ time and $O(mk^2n^2)$ space, where m is the number of patterns in the constraint, n is the maximum string length, and k is the number of input strings. C is defined in Sec. 4.

Tsai *et al.* [10] generalized the definition of patterns, from single characters to substrings. Their algorithm is equally efficient as that in [2].

In this paper, we focus on the pair-wise version when the positions of occurrences of the given patterns on one of the strings are known. Biologists are generally able to annotate important regions on a protein sequence known to have some function. Important regions can have different degrees: necessary, relevant, irrelevant, etc. This suggests that an effective tool must be able to consider both necessary and relevant regions. Previous results either focuses only on necessary regions [2, 8, 9, 10], or only on relevant regions [3]. Also, it is important to allow gaps when recognizing an occurrence of a motif; this is not considered in previous constrained alignment results [8, 2, 10].

The remainder of this paper is organized as follows. Section 2 formally defines the problem and some other terms. Section 3 gives the main algorithm. Section 4 shows how the results in Sec. 3 can be utilized to improve it. Finally, some concluding remarks are given.

2 Preliminaries

Let Σ be the alphabet where the residues of sequences are drawn. For a sequence X, let $X(i)$ be the ith character on X, and denote string $X(i)X(i+1)\ldots X(j)$ as $X[i..j]$.

An alignment of strings $S_1, \ldots, S_k$ is a set of strings $S_1', \ldots, S_k'$, where S_i' is obtained by inserting some spaces "–" into S_i for all $i = 1, \ldots, k$, and $|S_i'| = |S_j'|$ for all $1 \le i, j \le k$. Define $\#spaces(c; S_i')$ to be the number of "–"s in S_i' at or before position c.

Let δ be the cost function. Let $S_1', \ldots, S_k'$ be an alignment. The score of this alignment is given by $\sum_{l \le k} \sum_{i<j} \delta(S_i'(l), S_j'(l))$, which is the commonly used sum-of-pairs score.

Now we define some terms used in this paper.

Definition 1. *A pattern P is a language $P(1)P(2)\ldots P(|P|)$, where $P(l) \subset \Sigma$ for $1 \le l \le |P|$. We say that P occurs at position i in string X if $X[i + l] \in P[l]$ for $0 \le l \le |P| - 1$. If P occurs in X at positions $i_1 < \ldots < i_c$, then i_x is denoted as $start_x(X; P)$ (note the subscript). Also let $end_x(X; P) = i_x + |P| - 1$. We also say that $[start_x(X; P), end_x(X; P)]$ is the occurrence interval for P in X.*

This definition is beneficial for dealing with degenerate forms. For a pattern P_l, define $cost(P_l)$ to be the cost of matching two occurrences of P_l. We refer to this kind of pattern as type-1 pattern.

We now give a counterpart to the above definition for patterns.

Definition 2. *A pattern P is a string $P(1)P(2)\ldots P(|P|)$, where $P(l) \in \Sigma$ for $1 \le l \le |P|$. We say that P occurs at interval $[i, j]$ in string X if $d_E(X[i..j], P) \le t$, where t is a threshold value and d_E is the weighted edit distance. Suppose that P occurs at intervals $[i_1, j_1], [i_2, j_2], \ldots, [i_c, j_c]$, where $i_1 \le i_2 \le \ldots \le i_c$. Then we denote i_x as $start_x(X; P)$ and j_x as $end_x(X; P)$.*

We will refer to this kind of pattern as type-2 pattern. The definition for patterns in [10] uses Hamming distance and does not allow gaps in occurrences.

Definition 3. *A constraint is a sequence (rather than merely a set) of patterns $\langle P_1, \ldots, P_m \rangle$.*

Definition 4. *An alignment $S_1', \ldots, S_k'$ satisfies constraint $P_1, \ldots, P_m$ if there exists intervals $[c_1, c_1'], \ldots, [c_m, c_m']$ where $c_1 < \ldots < c_m$ such that for all $1 \le i \le k$ and all $1 \le l \le m$, P_l occurs at interval $[c_l, c_l']$ in S_i' and all $[c_l, c_l']$'s are disjoint (referred to as the non-overlapping condition). The interval $[c_l - \#spaces(c_l; S_i'), c_l' - \#spaces(c_l; S_i')]$ is said to be a "correct" interval of occurrence of P_l in S_i.*

We can now give the definition of the problem.

Definition 5. (Constrained Sequence Alignment Problem) *Given a set of sequences and a constraint,* *find an alignment satisfying the constraint with the minimum score.*

Since the multiple sequence alignment problem is NP-hard [1], it is clear that CMSA is also NP-hard. Tang *et al.* [8] and Tsai *et al.* [10] solved this problem by a progressive method, and Chin *et al.* adopted an approximation algorithm [2]. The method proposed in this paper improves Chin *et al.*'s 2-approximaiton algorithm.

The (pair-wise) case considered in this paper is stated as follows (referred to as "one-annotated case"):

Given two strings S_1 and S_2 and a constraint $\langle P_1, \ldots, P_m\rangle$, along with m "correct" intervals $[c_1, c_1']$, ..., $[c_l, c_l']$ *of occurrences* (*one for each pattern*) *in S_1 for each P_l such that the non-overlapping condition* (*see Def.* 4) *is satisfied. Find an alignment of S_1 and S_2 satisfying the constraint such that the score is minimized.*

Note that this version can be solved by the $O(mn^2)$ time and space algorithm in [2] (with some modifications). However, it is natural to ask whether the additional information given in this version (i.e., the positions of occurrences in one of the strings) can be utilized to improve the time and space bounds. The answer turns out to be positive, and is explored in the next section.

We now formalize a useful property used later. It is mentioned in [7] using the notion of grid graphs. Define *score*(X, Y) to be the score of the optimal alignment of strings X and Y.

Lemma 6. *If* $\min_{1\le j'\le j} score(X, Y[j'..j]) = score(X, Y[j^*..j])$, *then for all* $y > j$, $\min_{1\le y'\le y} score(X, Y[y'..y]) = \min_{j^*\le y'\le y} score(X, Y[y'..y])$, *and for all* $y < j$, $\min_{1\le y'\le y} score(X, Y[y'..y]) = \min_{1\le y'\le j^*} score(X, Y[y'..y])$.

3 Algorithm for One-Annotated Case

Suppose that we are given two strings S_1 and S_2, a constraint $\langle P_1, \ldots, P_m\rangle$, and m "correct" occurring intervals $[start(S_1; P_l), end(S_1; P_l)]$ in S_1, $1 \le l \le m$, one for each pattern P_l. Let $n = \max\{|S_1|, |S_2|\}$.

To deal with this problem, we first find the positions of occurrences in S_2 for each of the patterns. First consider type-1 patterns. For each pattern P_l in the constraint, we use $|\Sigma|$ bits to represent $P_l(i)$, e.g., if $\Sigma = \{A, T, C, G\}$ and $P_l(i) = \{A, C\}$, then $P_l(i)$ is represented as (1, 0, 1, 0). All the patterns can be converted in $O(L|\Sigma|)$ time and space, where $L = \sum_{1\le i\le m} |P_i|$. It is then easy to determine all positions of occurrences for all patterns in $O(Ln)$ time. The overall space needed is $O(L|\Sigma| + r)$, where r is the total number of occurrences of all patterns in S_2. We keep the list $occur(S_2; P_l) = \langle[start_1(S_2; P_l), end_1(S_2; P_l)], \ldots, [start_{c(S_2; P_l)}(S_2; P_l), end_{c(S_2; P_l)}(S_2; P_l)]\rangle$, where $c(S_2; P_l)$ is the number of occurrences of P_l in S_2.

For type-2 patterns, we use dynamic programming to compute weighted edit distance. By Lemma 6, along with some tie-breaking rule, it can be assured that if $end_x > end_y$, then $start_x \ge start_y$. Also keep $occur(S_2; P_l)$. This processing takes $O(Ln)$ time and $O(n + r)$ space, where $r = \sum_l c(S_2; P_l)$.

Recall that the "correct" intervals $[start(S_1; P_l), end(S_1; P_l)]$ for S_1 is taken as input for $1 \le l \le m$. To find the optimal constrained alignment of S_1 and S_2, we compute table T using the following recurrences:

(1) $T[0, 0] = 0$. $T[0, j] = T[0, j-1] + \delta(-, S_1[j])$ if $0 < j < start(S_1, 1)$.
$T[i, 0] = T[i-1, 0] + \delta(S_2[i], -)$ if $i > 0$.

(2) If $j \notin [start(S_1; P_l), end(S_1; P_l)]$ for all l,
$T[i, j] = \min\{T[i-1, j] + \delta(S_2[i], -), T[i-1, j-1] + \delta(S_2[i], S_1[j]), T[i, j-1] + \delta(-, S_1[j])\}$.

(3) If $j = start(S_1; P_l)$ for some l,
$T[i, j] = T[i-1, j-1]$, if $i = start_x(S_2; P_l)$ for some $1 \le x \le c(S_2; P_l)$;

$T[i, j]$ can be left undefined otherwise.

(4) If $j = end(S_1; P_l)$ for some l,

$T[0, j] = \infty$.

$T[i, j] = \min\{T[start_x(S_2; P_l), start(S_1; P_l)] + cost(P_l), T[i-1, j] + \delta(S_2[i], -)\}$, if $i = end_x(S_2; l)$ for some $1 \leq x \leq c(S_2; P_l)$;

$T[i, j] = T[i-1, j] + \delta(S_2[i], -)$, otherwise.

(5) If $j \in (start(S_1; P_l), end(S_1; P_l))$ for some l, then $T[i, j]$ can be left undefined.

Observe that T has a property that if $j \in [end(S_1; P_l), end(S_1; P_l+1))$ for some $l < m$, or if $j \geq end(S_1; P_l)$ for $l = m$, then $T[i, j]$ is the minimum score of aligning $S_2[1..i]$ and $S_1[1..j]$ satisfying the (sub-)constraint $\langle P_1, P_2, \ldots, P_k\rangle$. It follows that $T[|S_2|, |S_1|]$ is the optimal score of constrained alignment of S_1 and S_2 satisfying the constraint $\langle P_1, \ldots, P_m\rangle$.

By the sorted-property of the list $occur(S_2; P_l)$, each determination of what interval j belongs takes only $O(1)$ time.

With the above recurrences, we can now easily compute the optimal constrained alignment of S_1 and S_2. It is clear that $O(n^2)$ time suffices. If some care is taken, Hirschberg's well-known divide-and-conquer technique [5] can be modified to apply here (divide the columns). This reduces the required space to $O(n)$.

In total, it takes $O(n^2 + Ln)$ time and $O(n + L|\Sigma|)$ space for type-1 constraints, and the same time and $O(n + r)$ space for type-2 constraints. Let's take a closer look at these bounds. First, $L = O(n)$ by the non-overlapping condition. Also, assume that $|\Sigma|$ is a constant. In addition, it is clear that r is bounded above by $O(mn)$. Hence the time and space needed are $O(n^2)$ and $O(n + r)$, respectively. Note that if constraints are defined as in [2], $r = O(n)$ and the space complexity is $O(n)$.

To incorporate the analysis for important regions that are considered not necessary, a weighting scheme can be used, as in [3]. Specifically, for $x, y \in \Sigma \cup \{-\}$, $\delta(x, y)$ can be substituted by $\delta(x, y) \times w_1(x) \times w_2(y)$, where w_1 and w_2 are weights associated with the importance of characters in the first and second string, respectively.

4 A More Efficient Approximation Algorithm for CMSA

Chin *et al.* [2] extended Gusfield's well-known center-star approximation algorithm for the multiple sequence alignment problem [4], giving a $(2-2/k)$ approximation ratio for CMSA. It turns out that the algorithm developed in the previous section improves the performance of the 2-approximation algorithm in [2]. We now explore this.

Suppose we are given k strings $S_1, \ldots, S_k$, along with a constraint $\langle P_1, \ldots, P_m\rangle$. Let $n = \max_i |S_i|$. The algorithm is outlined below.

{Occurrence-finding Phase}

Find all intervals of occurrences of patterns $P_1, \ldots, P_m$ on all strings.

{Best Center Sequence Selection Phase}

Looping over all k sequences, consider each the center sequence once. Suppose that S_i is now the center sequence. Then for each list of occurrence intervals $[start_{c1}(S_i; P_1), end_{c1}(S_i; P_1)], \ldots, [start_{cm}(S_i; P_m), end_{cm}(S_i; P_m)]$ ($(c_1, \ldots, c_m)$ is referred to as "constrained list" later) of patterns $P_1, \ldots, P_m$ on S_i, $1 \leq c_l \leq c(S_i; P_l)$, S_i is pair-wise aligned with all strings in $\{S_1, \ldots, S_k\} - \{S_i\}$ with respect to this constrained list. The sum of these k–1 scores is kept for later comparison. Note that once the constrained list is fixed, the alignment can be done by the algorithm in the previous section.

We then determine for what choices of S_i and constrained list, the sum of pair-wise scores is minimized. This S_i is referred to as the best center sequence and $(c_1, \ldots, c_m)$ as the best constrained list.

{Merging Phase}

The $k-1$ pair-wise alignments of the best center sequence to all other sequences are merged, under the best constrained list, in the same way as in [2].

The total time and space taken are $O(Ck^2n^2)$ and $O(kn + R)$, respectively, where R is the total number of occurrences of all patterns in all strings. Details are omitted due to page limit. If the definition for patterns in [2] is adopted, our algorithm takes $O(Ck^2n^2)$ time and $O(kn)$ space, as opposed to $O(Cmk^2n^2)$ time and $O(mk^2n^2)$ space for the algorithm in [2].

5 Conclusion

In this paper, we study pair-wise constrained sequence alignment where occurrences of patterns in one of the strings are known. This is useful when determining whether a sequence has some properties. We propose an algorithm taking $O(n^2)$ time and $O(n + r)$ space. The analysis admitted here is more general than previous constrained sequence alignment results [2, 8, 10]. Also, the algorithm improves previous approximation algorithm for CMSA [2] both in generality and efficiency. The next step is to combine other kind of annotations or constraints into the analysis, yielding a yet more realistic tool.

References:

[1] P. Bonizzoni and G. D. Vedova, The Complexity of Multiple Sequence Alignment with SP-score that is a metric, *Theoretical Computer Science*, Vol. 259, No. 1-2, 2001, pp. 63-79.

[2] F. Y. L. Chin, N. L. Ho, T. W. Lam, P. W. H. Wong and M. Y. Chan, Efficient Constrained Sequence Alignment with Performance Guarantee, *Proceedings of the Computational Systems Bioinformatics* (CSB'03), IEEE, 2003, pp. 337-346.

[3] P. A. Evans, Algorithms and Complexity for Annotated Sequence Analysis, Ph. D Thesis, Department of Computer Science, University of Victoria, Canada, 1999.

[4] D. Gusfield, Efficient Methods for Multiple Sequence Alignment with Guaranteed Error Bounds, *Bulletin of Mathematical Biology*, Vol. 30, 1993, pp. 141-154.

[5] D. S. Hirschberg, A Linear Space Algorithm for Computing Maximal Common Subsequences, *Communications of the ACM*, Vol. 18, No. 6, 1975, pp. 341-343.

[6] T. Jiang, Y. Xu and M. Q. Zhang, editors, *Current Topics in Computational Molecular Biology*, The MIT Press, 2002.

[7] J. P. Schmidt, All Highest Scoring Paths in Weighted Grid Graphs and Their Application to Finding All Approximate Repeats in Strings, *SIAM Journal on Computing*, Vol. 27, No. 4, 1998, pp. 972-992.

[8] C. Y. Tang, C. L. Lu, M. D. T. Chang, Y. J. Sun, Y. T. Tsai, J. M. Chang, Y. H. Chiou, C. M Wu, H. T. Chang, W. I. Chou and S. C. Chiang, Constrained Sequence Alignment Tool Development and its Application to RNase Family Alignment, *Journal of Bioinformatics and Computational Biology*, Vol. 1, 2003, pp. 267-287.

[9] W. R. Taylor, Motif-biased Protein Sequence Alignment, *Journal of Computational Biology*, Vol. 1, 1994, pp. 297-310.

[10] Y. T. Tsai, C. L. Lu, C. T. Yu and Y. P. Huang, MuSiC: A Tool for Multiple Sequence Alignment with Constraints, *Bioinformatics*, to appear.