Sequence Alignment with Weighted Constraints

Yung-Hsing Peng, Chang-Biau Yang†, Kuo-Tsung Tseng and Kuo-Si Huang
Department of Computer Science and Engineering
National Sun Yat-sen University, Kaohsiung, Taiwan
†cbyang@cse.nsysu.edu.tw

Abstract

Given two sequences S_1, S_2 and a constrained sequence C, the longest common subsequence of S_1, S_2 with restriction to C is defined as the constrained longest common subsequence (CLCS) of S_1, S_2 and C. At the same time, the best alignment of S_1, S_2 with restriction to C is defined as the constrained pairwise sequence alignment (CPSA) of S_1, S_2 and C. Previous algorithms have shown that both CLCS and CPSA can be solved in $O(rnm)$ time using similar dynamic programming formulas, where $r = |C|$, $n = |S_1|$ and $m = |S_2|$. In 2004, Arslan first extended the definition of CLCS to a more flexible version, where the number of ignored constraints is allowed to a degree d and time complexity increases to $O(drnm)$. In this paper, we extend the definition of CPSA to another version, called weighted CPSA (WCPSA), and show that WCPSA can not only be solved in $O(rnm)$ time but also allow ignoring constraints by setting proper weights. In addition, we also show that some constraint-related problems can be immediately solved by adopting WCPSA.

Key words: algorithm, longest common subsequence, sequence alignment, dynamic programming

1 Introduction

In recent years, the concept of the constrained longest common subsequence (CLCS) problem was first delivered by Tsai, who also proposed an $O(rn^2m^2)$ algorithm for solving this problem[10]. As a tool for measuring the similarity of two sequences, the CLCS problem is more flexible than the traditional LCS (longest common subsequence) problem[5, 12] since the output of the former problem must contain a user-defined constrained sequence. Therefore, CLCS was immediately extended to the constrained pairwise sequence alignment (CPSA)[9], which is a new concept that allows biologists to define some single residues or nucleotides as constraints which must be covered in alignment. In addition, Tang *et al.* also used CPSA as a kernel to devise a progressive algorithm for finding constrained alignment in multiple sequences, called constrained multiple sequence alignment (CMSA)[9]. However, Tang's algorithm for solving CPSA also requires $O(rn^2m^2)$ time and space, thus it is not feasible for long sequences. Afterwards, four groups of researchers proposed improved algorithms independently[7, 13, 2, 3, 1]. In their articles, an $O(rnm)$ time algorithm was proposed for solving CPSA, which greatly improved Tsai and Tang's results by using different dynamic programming. Later, Tsai *et al.* first extended the definition of a single constraint from a sequence of characters to a sequence of strings[11], which makes CPSA more feasible for real applications. Finally, by applying Hirschberg's algorithm[4], Lu and Huang reduced the required space from $O(rnm)$ to $O(rn)$[6], which makes CPSA more applicable to long sequences.

However, all algorithms mentioned above are all non-ignoring, that is, the output sequence (or alignment) is forced to cover all constraints. This may be too rigorous in some circumstances. For example, given two sequences S_1="cccccggaga", S_2="aggaaccccc" and C="ag", we have CLCS="aga". However, if the constraint "g" in C can be ignored, we have CLCS="ggaa", which is longer than "aga". Furthermore, if both "a" and "g" can be ignored, then the CLCS would be "ccccc", which happens to be the longest common subsequence of S_1 and S_2. Therefore, we will have longer CLCS if some constraints can be ignored properly. As a result, Arslan *et al.* first extended CLCS to another flexible version, where the number of ignored constraints is allowed to a degree d and the time complexity for solving this new extended problem is $O(drnm)$[1]. By the same token, we may yield better result in CPSA

if we apply Arslan's idea to alignment. However, with the cost of $O(n^4)$ time and $O(n^3)$ space in the worst case, Arslan's extension may be inapplicable to long sequences.

In this paper, we propose another extension for CPSA, called *weighted CPSA* (WCPSA), and show that WCPSA can still be solved in $O(rnm)$ time. In addition, we will also show that based on WCPSA, some constraint-related problems can immediately be solved by simply setting proper parameters.

2 Sequence Alignment with Weighted Constraints

Suppose we are given three sequences S_1, S_2 and C, where $|S_1| = n$, $|S_2| = m$ and $|C| = r$. Let $S[i]$ denote the ith symbol in the sequence S. Let $\sigma[k]$ and $\delta[k]$ denote the gain and penalty weights when $C[k]$ is included and excluded in the final output, respectively. Let p, q, g denote the scores of matching, mismatching, and gap penalty, respectively. The WCPSA problem is to find the constrained alignment with the maximum score based on above scoring scheme. By extending the formula of CPSA, we can solve WCPSA efficiently. Let $R(k, i, j)$ denote the score of optimal WCPSA of $S_1[1] \cdots S_1[i]$, $S_2[1] \cdots S_2[j]$, and $C[1] \cdots C[k]$, where $1 \leq i \leq n$, $1 \leq j \leq m$, $0 \leq k \leq r$. The dynamic programming for solving WCPSA can be formulated as Figure 2.

In above formula, $\sigma[k]$ represents the gain if $C[k]$ is covered, while $\delta[k]$ is the penalty when $C[k]$ is ignored. Except for the last case which ignores $C[k]$, our formula of WCPSA is almost the same with the formula of CPSA[2, 3]. If $C[k]$ is ignored, then the optimal solution relies on $R(k-1, i, j)$. Therefore, considering the penalty, we have $R(k, i, j) = R(k-1, i, j) - \delta[k]$ if $C[k]$ is ignored. Hence, we can conclude that the last case in our formula can ignore $C[k]$, in other words, our formula is valid for solving WCPSA.

However, note that the optimal solution in WCPSA may not cover the entire constrained sequence, we still need to prove that CPSA is a special scheme of WCPSA. The main idea of our proof is to select a proper value for σ, which guarantees that the optimal alignment in WCPSA is the alignment that fulfills the definition of CPSA.

Theorem 1. *The CPSA problem is a special scheme of the WCPSA problem.*

Proof: Let $u = max\{r, n, m\}$ and $v = max\{g, p, q\}$ in CPSA. Let $\delta[i]$ be zero and $\sigma[i] = x$, for $1 \leq i \leq r$. Assume there exists an alignment covering α constraints, β matches, γ mismatches and ω gaps, with its alignment score $\beta p - \gamma q - \omega g$ and WCPSA score $\alpha x + \beta p - \gamma q - \omega g$. For any alignment, we have $\beta + \gamma + \omega \leq 2u$. Since $v = max\{g, p, q\}$, we have $\beta p + \gamma q + \omega g \leq \beta v + \gamma v + \omega v \leq 2uv$. Therefore, we conclude that $-2uv \leq \beta p - \gamma q - \omega g \leq 2uv$. Based on this conclusion, by setting x to $4uv + 1$, we can guarantee that any alignment which maximizes the WCPSA score must first maximize α (number of covered constraints), and then maximize $\beta p - \gamma q - \omega g$ (alignment score). Therefore, we can obtain α in constant time by dividing $R(r, n, m)$ with $4uv + 1$ and round off the answer. By the above conclusion, we can set each σ to $4uv + 1$ before performing WCPSA, and check if α equals to $|C|$ after obtaining $R(r, n, m)$. If α equals to $|C|$, then the solution in WCPSA is the same as that in original CPSA, since the solution covers all constraints and its alignment score is optimal. At the same time, it is clear that there is no solution for CPSA if α is less than $|C|$. Consequently, the CPSA problem is a special scheme of the WCPSA problem. □

In addition, for solving CLCS, the proper value of x in WCPSA can be smaller. In CLCS, we have $g = q = 0$, $p = 1$. Again, by setting $\delta[i]$ to zero and $\sigma[i]$ to x, for $1 \leq i \leq r$, we know that any solution in WCPSA will have its score equals to $\alpha x + \beta$. In this case, we have $0 \leq \beta \leq u$. Therefore, the proper value for x is $u + 1$, which is smaller than $4uv+1$ if we want to solve CLCS by using WCPSA.

According to the formula, both time and space complexities of our algorithm to solve WCPSA is $O(rnm)$. However, the space complexity can be further reduced to $O(rn)$ by applying Hirschberg's algorithm[4]. Therefore, WCPSA is an extension for both CLCS and CPSA, with the same time complexity and space complexity. The main difference between Arslan's extension and WCPSA depends on the scoring scheme. In Arslan's extension, the scoring scheme cannot be written in a form linear to the number of covered constraints. However, from the proof of Theorem 1, we can clearly see that the scoring scheme in WCPSA is linear to α, which is a crucial property that enables WCPSA to be solved in less time and space.

$$R(k,i,j) = \max \begin{cases} R(k,i-1,j) - g & \text{if } S_1[i] \neq S_2[j], \\ R(k,i,j-1) - g & \text{if } S_1[i] \neq S_2[j], \\ R(k,i-1,j-1) + p & \text{if } S_1[i] = S_2[j], \\ R(k,i-1,j-1) - q & \text{if } S_1[i] \neq S_2[j], \\ R(k-1,i-1,j-1) + p + \sigma[k] & \text{if } S_1[i] = S_2[j] = C[k], \\ R(k-1,i,j) - \delta[k]. \end{cases} \tag{1}$$

with boundary conditions:

$$\begin{aligned} &R(0,0,0) = 0, \\ &R(0,0,j) = -j \times g, \text{ for } 1 \leq j \leq m \\ &R(0,i,0) = -i \times g, \text{ for } 1 \leq i \leq n \\ &R(k,0,j) = -\sum_{t=1}^{k} \delta[t] - j \times g, \text{ for } 1 \leq k \leq r,\ 0 \leq j \leq m \\ &R(k,i,0) = -\sum_{t=1}^{k} \delta[t] - i \times g, \text{ for } 1 \leq k \leq r,\ 0 \leq i \leq n \\ &R(k,i,j) = -\infty, \text{ for any } k < 0 \text{ or } i < 0 \text{ or } j < 0. \end{aligned} \tag{2}$$

Figure 1: The dynamic programming formula of WCPSA

3 Applications

In general, WCPSA is a tool which is more flexible than CPSA when we want to measure the similarity of two sequences with an additional constraint sequence. Since ignoring is allowed, now the length of C can be greater than n or m. This makes WCPSA more feasible for some applications. In addition, with proper setting to its parameters, WCPSA can still be used to solve some questions other than CPSA. Here we will give two applications to show the flexibility of WCPSA.

The first application is among all longest common subsequences in S_1 and S_2, finding the one that can covers the most constraints. Note that this problem is quite different from CLCS, because this time the restriction is placed on the length of LCS. Obviously, as the restriction varies, CLCS and CPSA is no more sufficient for solving such kind of problem. However, by setting $p = u + 1$, $g = q = 0$, $\sigma[i] = 1$ and $\delta[i] = 0$ for $1 \leq i \leq r$, the score in WCPSA can be written as $\alpha + \beta(u+1)$. Using the same concept in our proof, we can then solve this problem in $O(rnm)$ time by using WCPSA, which is more efficient than adopting Arslan's extension with $d = |C|$.

The second application is to design hierarchical constraints, which means the constraints can be classified into several levels according to their importance. By doing so, we can specify which constraints should be covered first, in case ignoring is unavoidable. Take two-level constraints in LCS for example, we have two kinds of constraints in group A and B, respectively. Suppose that group A is prior to group B, then the scoring scheme can be written as $\alpha_A(u+1)^2 + \alpha_B(u+1) + \beta$, where α_A and α_B denote the numbers of covered constraints in A and B, respectively. Since we know that α_A, α_B and β are all nonnegative integers which never exceed $u + 1$, it is clear that these three numbers can be computed by simply doing some constant operations to $R(r,n,m)$. In addition, under this scoring scheme, we know that any solution must first maximize α_A, then maximize α_B. Therefore, the two-level constraints problem in LCS, which cannot be solved by directly using Arslan's extension, can now be achieved by adopting WCPSA. Nevertheless, if we extend this idea to h-level constraints, the proper weight for the highest level constraints exceeds $(u+1)^h$. This means we have to spend $O(h)$ time for doing one mathematic operation to such a huge number. Therefore, the time complexity for solving h-level constraints should be $O(hrnm)$, which will be $O(r^2nm)$ in the worst case. However, in the hierarchical system, the number of levels are usually limited and we have $h \ll max\{r,n,m\}$, which means the time complexity of h-level constraints in WCPSA can in general be kept in $O(rnm)$. Therefore, the idea of hierarchical constraints can still be implemented by setting proper weights for constraints in WCPSA.

In addition to the applications above, the constraints and sequences in WCPSA can also be extended from letters to strings. In this way, WCPSA will make more sense to applications such as article comparison with specified keywords, or sequence alignment with specified functional sites [11]. Furthermore, the parameters p and q in WCPSA can be replaced with another score ma-

trix such as PAM250[8], which makes WCPSA more compatible with protein sequence alignment.

4 Conclusion

Compared with Arslan's $O(drnm)$ time extension for CLCS, WCPSA is another extension for CPSA. However, we have shown that WCPSA can be solved more efficiently in $O(rnm)$ time by using dynamic programming. With the aid of WCPSA, some constraint-related problems can also be solved by merely setting proper parameters. In general, WCPSA is more flexible than Arslan's extension and the traditional CPSA, thus can further be used as another kernel for designing algorithms for multiple sequence alignment with constraints.

References

[1] A. N. Arslan and Ö̈mer Eğecioğlu, "Algorithms for the constrained longest common subsequence problems," *Proceedings of the Prague Stringology Conference '04*, Prague, Czech Republic, pp. 24–32, 2004.

[2] F. Y. L. Chin, N. L. Ho, T. W. Lam, P. W. H. Wong, and M. Y. Chan, "Efficient constrained multiple sequence alignment with performance guarantee," *IEEE Computer Society Bioinformatics Conference (CSB'03)*, CA, USA, pp. 337–346, 2003.

[3] F. Y. L. Chin, A. D. Santis, A. L. Ferrara, N. L. Ho, and S. K. Kim, "A simple algorithm for the constrained sequence problems," *Information Processing Letters*, Vol. 90, pp. 175–179, 2004.

[4] D. S. Hirschberg, "A linear space algorithm for computing maximal common subsequences," *Communications of the ACM*, Vol. 18, pp. 341–343, 1975.

[5] J. W. Hunt and T. G. Szymanski, "A fast algorithm for computing longest common subsequences," *Communications of the ACM*, Vol. 20(5), pp. 350–353, 1977.

[6] C.-L. Lu and Y.-P. Huang, "A memory-efficient algorithm for multiple sequence alignment with constraints," *Bioinformatics*, Vol. 21(1), pp. 20–30, 2005.

[7] C.-L. Peng, "An approach for solving the constrained longest common subsequence problem." http://etd.lib.nsysu.edu.tw/ETD-db/ETD-search/search, Master's Thesis, Department of Computer Science and Engineering, National Sun Yat-Sen University, Kaohsiung, Taiwan, 2003.

[8] R. M. Schwartz and M. O. Dayhoff., *Matrices for detecting distant relationships.* In M. Dayhoff, editor, Atlas of Protein Sequence and Structure, volume 5, pages 353-358. National Biomedical Research Foundation, Washington, DC, 1979.

[9] 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," *Journal of Bioinformatics and Computational Biology*, Vol. 1, pp. 267–287, 2003.

[10] Y.-T. Tsai, "The constrained common sequence problem," *Information Processing Letters*, Vol. 88, pp. 173–176, 2003.

[11] Y.-T. Tsai, Y.-P. Huang, C.-T. Yu, and C.-L. Lu, "Music: a tool for multiple sequence alignment with constraints," *Bioinformatics*, Vol. 20(14), pp. 2309–2311, 2004.

[12] C.-B. Yang and R. C. T. Lee, "Systolic algorithms for the longest common subsequence problem," *Journal of the Chinese Institute of Engineers*, Vol. 10(6), pp. 691–699, 1987.

[13] D. C. T. Yu, "Efficient algorithms for constrained sequence alignment problems." http://ethesys.lib.pu.edu.tw/ETD-db/ETD-search/search, Master's Thesis, Department of Computer Science and Information Management, Providence University, Taichung, Taiwan, 2003.