# An Efficient Dynamic Programming Algorithm for SEQ-IC-SEQ-IC-LCS Problem

Daxin Zhu[1, a] and Xiaodong Wang[2,b*]

[1] Quanzhou Normal University, Quanzhou 362000, China

[2] Fujian University of Technology, Fuzhou 350108, China

[a] dex@qztc.edu.cn, [b] wangxd@fzu.edu.cn

**Keywords:** Generalized LCS, subsequences, dynamic programming, algorithm.

**Abstract.** In this paper, we consider a generalized longest common subsequence problem, in which two constraining sequences of length $s$ and $t$ must be included as the subsequences of the two main sequences and the length of the result must be maximal. For the two input sequences $X$ and $Y$ of lengths $n$ and $m$, and the given two constraining sequences of length $s$ and $t$, we present an $O(nmst)$ time dynamic programming algorithm for solving the new generalized longest common subsequence problem. The correctness of the new algorithm is proved.

## Introduction

The longest common subsequence (LCS) problem is a well-known measurement for computing the similarity of two strings. It can be widely applied in diverse areas, such as file comparison, pattern matching and computational biology[3, 4, 8, 9].

Given two sequences $X$ and $Y$, the longest common subsequence (LCS) problem is to find a subsequence of $X$ and $Y$ whose length is the longest among all common subsequences of the two given sequences.

For some biological applications some constraints must be applied to the LCS problem. These kinds of variant of the LCS problem are called the constrained LCS (CLCS) problem. Recently, Chen and Chao[1] proposed the more generalized forms of the CLCS problem, the generalized constrained longest common subsequence (GC-LCS) problem. For the two input sequences $X$ and $Y$ of lengths $n$ and $m$ ,respectively, and a constraint string $P$ of length $r$, the GC-LCS problem is a set of four problems which are to find the LCS of $X$ and $Y$ including/excluding $P$ as a subsequence/substring, respectively.

In this paper, we consider a more general constrained longest common subsequence problem called SEQ-IC-SEQ-IC-LCS, in which two constraining sequences of length $s$ and $t$ must be included as the subsequences of the two main sequences and the length of the result must be maximal. We will present the first efficient dynamic programming algorithm for solving this problem.

The organization of the paper is as follows.

In the following 3 sections, we describe our presented dynamic programming algorithm for the SEQ-IC-SEQ-IC-LCS problem.

In section 2 the preliminary knowledge for presenting our algorithm for the SEQ-IC-SEQ-IC-LCS problem is discussed. In section 3 we give a new dynamic programming solution for the SEQ-IC-SEQ-IC-LCS problem with time complexity $O(nmst)$, where $n$ and $m$ are the lengths of the two given input strings, and $s$ and $t$ the lengths of the two constraining sequences. Some concluding remarks are in section 4.

**Characterization of the Generalized LCS Problem**

A sequence is a string of characters over an alphabet $\sum$. A subsequence of a sequence $X$ is obtained by deleting zero or more characters from $X$ (not necessarily contiguous). A substring of a sequence $X$ is a subsequence of successive characters within $X$.

For a given sequence $X = x_1x_2\cdots x_n$ of length $n$, the $i$th character of $X$ is denoted as $x_i \in \sum$ for any $i = 1,\cdots,n$. A substring of $X$ from position $i$ to $j$ can be denoted as $X[i:j] = x_ix_{i+1}\cdots x_j$. If $i \neq 1$ or $j \neq n$, then the substring $X[i:j] = x_ix_{i+1}\cdots x_j$ is called a proper substring of $X$. A substring $X[i:j] = x_ix_{i+1}\cdots x_j$ is called a prefix or a suffix of $X$ if $i = 1$ or $j = n$, respectively.

An appearance of sequence $X = x_1x_2\cdots x_n$ in sequence $Y = y_1y_2\cdots y_m$, for any $X$ and $Y$, starting at position $j$ is a sequence of strictly increasing indexes $i_1,i_2,\cdots,i_n$ such that $i_1 = j$, and $X = y_{i_1}, y_{i_2},\cdots, y_{i_n}$. A compact appearance of $X$ in $Y$ starting at position $j$ is the appearance of the smallest last index $i_n$.

For the two input sequences $X = x_1x_2\cdots x_n$ and $Y = y_1y_2\cdots y_m$ of lengths $n$ and $m$, respectively, and two constrained sequences $P = p_1p_2\cdots p_s$ and $Q = q_1q_2\cdots q_t$ of lengths $s$ and $t$, the SEQ-IC-SEQ-IC-LCS problem is to find a constrained LCS of $X$ and $Y$ including $P$ and $Q$ as its subsequences.

**Definiton 1** *Let $S(i,j,k,r)$ denote the set of all LCSs of $X[1:i]$ and $Y[1:j]$ such that for each $z \in S(i,j,k,r)$, $z$ includes $P[1:k]$ and $Q[1:r]$ as subsequences, where $1 \le i \le n, 1 \le j \le m, 0 \le k \le s$, and $0 \le r \le t$. The length of an LCS in $S(i,j,k,r)$ is denoted as $f(i,j,k,r)$.*

The following theorem characterizes the structure of an optimal solution based on optimal solutions to subproblems, for computing the LCSs in $S(i,j,k,r)$, for any $1 \le i \le n, 1 \le j \le m, 0 \le k \le s$, and $0 \le r \le t$.

**Theorem 1** *If $Z[1:l] = z_1, z_2,\cdots, z_l \in S(i,j,k,r)$, then the following conditions hold:*

*1. If $k > 0, r > 0, x_i = y_j = p_k = q_r$, then $z_l = x_i$ and $Z[1:l-1] \in S(i-1, j-1, k-1, r-1)$.*

*2. If $k > 0, x_i = y_j = p_k$ and $(r = 0 \vee x_i \neq q_r)$, then $z_l = x_i$ and $Z[1:l-1] \in S(i-1, j-1, k-1, r)$.*

*3. If $r > 0, x_i = y_j = q_r$ and $(k = 0 \vee x_i \neq p_k)$, then $z_l = x_i$ and $Z[1:l-1] \in S(i-1, j-1, k, r-1)$.*

*4. If $x_i = y_j$ and $(k = 0 \vee x_i \neq p_k)$ and $(r = 0 \vee x_i \neq q_r)$, then $z_l = x_i$ and $Z[1:l-1] \in S(i-1, j-1, k, r)$.*

*5. If $x_i \neq y_j$, then $z_l \neq x_i$ implies $Z[1:l] \in S(i-1, j, k, r)$.*

*6. If $x_i \neq y_j$, then $z_l \neq y_j$ implies $Z[1:l] \in S(i, j-1, k, r)$.*

**Proof.**

1. In this case, we can add $x_i$ to $Z[1:l-1]$ such that $x_i = z_l$, and thus $Z[1:l-1]$ must be an LCS of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k-1]$ and $Q[1:r-1]$ as subsequences, i.e. $Z[1:l-1] \in S(i-1, j-1, k-1, r-1)$.

2. In this case we have no constraints on $Q$, and thus $x_i$ cab be added to $Z[1:l-1]$ and $Z[1:l-1]$ must be a common subsequence of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k-1]$ as a subsequence and including $Q[1:r]$ as a subsequence. We can show that $Z[1:l-1]$ is an LCS of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k-1]$ as a subsequence and including $Q[1:r]$ as a subsequence. Assume by contradiction that there exists a common subsequence $a$ of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k-1]$ as a subsequence and including $Q[1:r]$ as a subsequence, whose length is greater than $l-1$. Then the concatenation of $a$ and $x_i$ will result in a common subsequence

of $X[1:i]$ and $Y[1:j]$ including $P[1:k]$ and $Q[1:r]$ as subsequences, whose length is greater than $l$. This is a contradiction.

3. In this case we have no constraints on $P$, and thus $x_i$ cab be added to $Z[1:l-1]$ and $Z[1:l-1]$ must be a common subsequence of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k]$ as a subsequence and including $Q[1:r-1]$ as a subsequence. We can show that $Z[1:l-1]$ is an LCS of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k]$ as a subsequence and including $Q[1:r-1]$ as a subsequence. Assume by contradiction that there exists a common subsequence $a$ of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k]$ as a subsequence and including $Q[1:r-1]$ as a subsequence, whose length is greater than $l-1$. Then the concatenation of $a$ and $x_i$ will result in a common subsequence of $X[1:i]$ and $Y[1:j]$ including $P[1:k]$ and $Q[1:r]$ as subsequences, whose length is greater than $l$. This is a contradiction.

4. In this case we have no constraints on $P$ and $Q$, and thus $x_i$ cab be added to $Z[1:l-1]$ and $Z[1:l-1]$ must be a common subsequence of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k]$ and $Q[1:r]$ as subsequences. We can show that $Z[1:l-1]$ is an LCS of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k]$ and $Q[1:r]$ as subsequences. Assume by contradiction that there exists a common subsequence $a$ of $X[1:i-1]$ and $Y[1:j-1]$ including $P[1:k]$ and $Q[1:r]$ as subsequences, whose length is greater than $l-1$. Then the concatenation of $a$ and $x_i$ will result in a common subsequence of $X[1:i]$ and $Y[1:j]$ including $P[1:k]$ and $Q[1:r]$ as subsequences, whose length is greater than $l$. This is a contradiction.

5. Since $x_i \neq y_j$ and $z_l \neq x_i$, $Z[1:l]$ must be a common subsequence of $X[1:i-1]$ and $Y[1:j]$ including $P[1:k]$ and $Q[1:r]$ as subsequences. It is obvious that $Z[1:l]$ is also an LCS of $X[1:i-1]$ and $Y[1:j]$ including $P[1:k]$ and $Q[1:r]$ as subsequences.

6. Since $x_i \neq y_j$ and $z_l \neq y_j$, $Z[1:l]$ must be a common subsequence of $X[1:i]$ and $Y[1:j-1]$ including $P[1:k]$ and $Q[1:r]$ as subsequences. It is obvious that $Z[1:l]$ is also an LCS of $X[1:i]$ and $Y[1:j-1]$ including $P[1:k]$ and $Q[1:r]$ as subsequences.

The proof is completed. ■

## A Simple Dynamic Programming Algorithm

Let $f(i,j,k,r)$ denote the length of an LCS in $S(i,j,k,r)$. By the optimal substructure properties of the SEQ-IC-SEQ-IC-LCS problem shown in Theorem 1, we can build the following recursive formula for computing $f(i,j,k,r)$. For any $1 \le i \le n, 1 \le j \le m, 0 \le k \le s$, and $0 \le r \le t$, the values of $f(i,j,k,r)$ can be computed by the following recursive formula (1).

$$f(i,j,k,r) = \begin{cases} \max\{f(i-1,j,k,r), f(i,j-1,k,r)\} & \text{if } x_i \neq y_j \\ 1+f(i-1,j-1,k-1,r-1) & \text{if } k>0, r>0, x_i = y_j = p_k = q_r \\ 1+f(i-1,j-1,k-1,r) & \text{if } k>0, x_i = y_j = p_k \wedge (r=0 \vee x_i \neq q_r) \\ 1+f(i-1,j-1,k,r-1) & \text{if } r>0, x_i = y_j = q_r \wedge (k=0 \vee x_i \neq p_k) \\ 1+f(i-1,j-1,k,r) & \text{if } x_i = y_j \wedge (k=0 \vee x_i \neq p_k) \wedge (r=0 \vee x_i \neq q_r) \end{cases} \quad (1)$$

The boundary conditions of this recursive formula are $f(i,0,0,0) = f(0,j,0,0) = 0$ and $f(i,0,k,r) = f(0,j,k,r) = -\infty$ for any $0 \le i \le n, 0 \le j \le m, 0 \le k \le s$, and $0 \le r \le t$.

Based on this formula, our algorithm for computing $f(i,j,k,r)$ is a standard dynamic programming algorithm. By the recursive formula (1), the dynamic programming algorithm for computing $f(i,j,k,r)$ can be implemented as the following Algorithm 1.

**Algorithm 1** SEQ-IC-SEQ-IC-LCS

**Input:** Strings $X = x_1 \cdots x_n$, $Y = y_1 \cdots y_m$ of lengths $n$ and $m$, respectively, and two constrained sequences $P = p_1p_2 \cdots p_s$ and $Q = q_1q_2 \cdots q_t$ of lengths $s$ and $t$

**Output:** $f(i, j, k, r)$, the length of an LCS of $X[1 : i]$ and $Y[1 : j]$ including $P[1 : k]$ and $Q[1 : r]$ as subsequences, for all $1 \leq i \leq n, 1 \leq j \leq m, 0 \leq k \leq s$, and $0 \leq r \leq t$.

```
for all $i, j, k, r$ , $0 \leq i \leq n, 0 \leq j \leq m, 0 \leq k \leq s$ and $0 \leq r \leq t$ do
  $f(i, 0, k, r), f(0, j, k, r) \leftarrow -\infty, f(i, 0, 0, 0), f(0, j, 0, 0) \leftarrow 0$ {boundary condition}
end for
for all $i, j, k, r$ , $1 \leq i \leq n, 1 \leq j \leq m, 0 \leq k \leq s$ and $0 \leq r \leq t$ do
  if $x_i \neq y_j$ then
    $f(i, j, k, r) \leftarrow \max\{f(i-1, j, k, r), f(i, j-1, k, r)\}$
  else if $k > 0$ and $x_i = p_k$ then
    if $r > 0$ and $x_i = q_r$ then
      $f(i, j, k, r) \leftarrow 1 + f(i-1, j-1, k-1, r-1)$
    else
      $f(i, j, k, r) \leftarrow 1 + f(i-1, j-1, k-1, r)$
    end if
  else if $r > 0$ and $x_i = q_r$ then
    $f(i, j, k, r) \leftarrow 1 + f(i-1, j-1, k, r-1)$
  else
    $f(i, j, k, r) \leftarrow 1 + f(i-1, j-1, k, r)$
  end if
end for
```

It is obvious that the algorithm requires $O(nmst)$ time and space. For each value of $f(i, j, k, r)$ computed by algorithm *Suffix*, the corresponding LCS of $X[1:i]$ and $Y[1:j]$ including $P[1:k]$ and $Q[1:r]$ as subsequences, can be constructed by backtracking through the computation paths from $(i, j, k, r)$ to $(0,0,0,0)$. The following algorithm $back(i, j, k, r)$ is the backtracking algorithm to obtain the LCS, not only its length. The time complexity of the algorithm $back(i, j, k, r)$ is obviously $O(n+m)$.

**Algorithm 2** $back(i, j, k, r)$

**Input:** Integers $i, j, k, r$

**Output:** The LCS of $X[1 : i]$ and $Y[1 : j]$ including $P[1 : k]$ as a subsequence and $Q[1 : r]$ as a suffix

```
if $i < 1$ or $j < 1$ then
  return
end if
if $x_i \neq y_j$ then
  if $f(i-1, j, k, r) > f(i, j-1, k, r)$ then
    $back(i-1, j, k, r)$
  else
    $back(i, j-1, k, r)$
  end if
else if $k > 0$ and $x_i = p_k$ then
  if $r > 0$ and $x_i = q_r$ then
    $back(i-1, j-1, k-1, r-1)$
    print $x_i$
  else
    $back(i-1, j-1, k-1, r)$
    print $x_i$
  end if
else if $r > 0$ and $x_i = q_r$ then
  $back(i-1, j-1, k, r-1)$
  print $x_i$
else
  $back(i-1, j-1, k, r)$
  print $x_i$
end if
```

## Summary

We have suggested a new dynamic programming solution for the new generalized constrained longest common subsequence problem SEQ-IC-SEQ-IC-LCS. The new dynamic programming algorithm requires $O(nmst)$ in the worst case, where $n, m, s, t$ are the lengths of the four input sequences respectively. It is not difficult to show that this problem can also be solved in $O(\min(n, m)st)$ space based on Hirschberg's Algorithm [5].

Many other generalized constrained longest common subsequence (GC-LCS) problems have similar structures. It is not clear that whether the same technique of this paper can be applied to these problems to achieve efficient algorithms. We will investigate these problems further.

## Acknowledgement

This research was financially supported by the Natural Science Foundation of Fujian (Grant No.2013J01247), and Fujian Provincial Key Laboratory of Data-Intensive Computing and Fujian University Laboratory of Intelligent Computing and Information Processing.

## References

[1] Chen Y.C., Chao K.M. On the generalized constrained longest common subsequence problems, J. Comb. Optim. 21(3), 2011, pp. 383-392.

[2] Crochemore M.,Hancart C., and Lecroq T., Algorithms on strings, Cambridge University Press, Cambridge, UK, 2007.

[3] Deorowicz S., Quadratic-time algorithm for a string constrained LCS problem, Inform. Process. Lett. 112(11), 2012, pp. 423-426.

[4] Deorowicz S., Obstoj J., Constrained longest common subsequence computing algorithms in practice, Comput. Inform. 29(3), 2010, pp. 427-445.

[5] Hirschberg,D.S., Algorithms for the longest common subsequence problem, J. ACM 24 (1977) 664?C675.

[6] Gotthilf Z., Hermelin D., Lewenstein M., Constrained LCS: hardness and approximation. In: Proceedings of the 19th annual symposium on combinatorial pattern matching, CPM'08, Pisa, Italy, 2008, pp. 255-262.

[7] Gotthilf Z., Hermelin D., Landau G.M., Lewenstein M., Restricted LCS. In: Proceedings of the 17th international conference on string processing and information retrieval, SPIRE'10, Los Cabos, Mexico, 2010, pp. 250-257.

[8] Gusfield, D.,Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, Cambridge, UK, 1997.

[9] Peng Y.H., Yang C.B., Huang K.S., Tseng K.T., An algorithm and applications to sequence alignment with weighted constraints, Int. J. Found. Comput. Sci. 21(1),2010, pp. 51-59.

[10] Tang C.Y., Lu C.L., Constrained multiple sequence alignment tool development and its application to RNase family alignment, J. Bioinform. Comput. Biol. 1, 2003, pp. 267-287.

[11] Tseng C.T., Yang C.B., Ann H.Y., Efficient algorithms for the longest common subsequence problem with sequential subsequence constraints. J. Complexity 29, 2013, pp. 44-52.