*International Journal of Innovative Computing, Information and Control*
Volume **6**, Number **4**, April **2010**

ICIC International ©2010 ISSN 1349-4198
pp. 1935–1947

# EFFICIENT SPARSE DYNAMIC PROGRAMMING FOR THE MERGED LCS PROBLEM WITH BLOCK CONSTRAINTS

Yung-Hsing Peng, Chang-Biau Yang*, Kuo-Si Huang, Chiou-Ting Tseng and Chiou-Yi Hor

Department of Computer Science and Engineering
National Sun Yat-sen University
No. 70, Lienhai Rd., Kaohsiung 80424, Taiwan
*Corresponding author: cbyang@cse.nsysu.edu.tw

Received September 2008; revised January 2009

Abstract. *Detecting the interleaving relationship between sequences has become important because of its wide applications to genomic and signal comparison. Given a target sequence $T$ and two merging sequences $A$ and $B$, recently Huang* et al. *propose algorithms for the merged LCS problem, without or with block constraint, whose aim is to find the longest common subsequence (LCS) with interleaving relationship. Without block constraint, Huang's algorithm requires $O(nmr)$-time and $O(mr)$-space, where $n = |T|$, $m$ and $r$ denote the longer and shorter length of $A$ and $B$, respectively. In this paper, for solving the problem without block constraint, we first propose an algorithm with $O(Lnr)$ time and $O(m + Lr)$ space. We also propose an algorithm to solve the problem with block constraint. Our algorithms are more efficient than previous results, especially for sequences over large alphabets.*
**Keywords:** Algorithm, Dynamic programming, Longest common subsequence, Bioinformatics, Merged sequence

## 1. Introduction.

In the field of sequence comparison, finding the *longest common subsequence* (LCS) between sequences is a classic approach for measuring the similarity of sequences. Given a sequence $S$, a subsequence $\bar{S}$ of $S$ can be obtained by deleting zero or more characters from $S$. Given two sequences $S_1$ and $S_2$, the LCS of $S_1$ and $S_2$, denoted by $LCS(S_1, S_2)$, is the longest sequence $\bar{S}'$ such that $\bar{S}'$ is a subsequence of both $S_1$ and $S_2$.

Algorithms for finding the LCS of two or more sequences have been extensively studied for several decades [1, 2, 3]. Most of these algorithms are based on either traditional dynamic programming [1] or sparse dynamic programming [4, 5, 6]. Traditional dynamic programming, which compares each character in $S_1$ with each character in $S_2$, takes $O(|S_1||S_2|)$ time, where $|S_1|$ and $|S_2|$ denote the lengths of $S_1$ and $S_2$, respectively. For alphabets that cannot be sorted, this approach is optimal [7]. For sortable alphabets, however, sparse dynamic programming is faster [5, 6], because the information of matches can be obtained more efficiently.

In addition to the traditional LCS problem, some extended versions, such as the constrained LCS problem [8, 9, 3, 10, 11, 12] and the mosaic LCS problem [13, 14], have been proposed to provide more flexible comparison on sequences. Recently, Huang *et al.* [15] proposed one interesting problems: *the merged LCS problem*, which is to detect the interleaving relationship between sequences. In the problem, the block constraint may be involved.

In biology, finding the interleaving relationship of sequences can be realized as detecting the *synteny* phenomenon, which means the order of specific genes in chromosome is