The Generalized Definitions of the Two-Dimensional Largest Common Substructure Problems *

Huang-Ting Chan[a], Chang-Biau Yang[a†] and Yung-Hsing Peng[b]

[a]Department of Computer Science and Engineering
National Sun Yat-sen University, Kaohsiung, Taiwan

[b]Innovative DigiTech-Enabled Applications & Services Institute
Institute for Information Industry, Kaohsiung, Taiwan

Abstract

The traditional longest common subsequence *(LCS) problem is to find the maximum number of ordered matches in two sequences. The similarity of two one-dimensional sequences can be measured by the LCS algorithms, which have been extensively studied. However, for the two-dimensional data, computing the similarity with an LCS-like approach remains worthy of investigation. In this paper, we utilize a systematic way to give the generalized definition of the* two-dimensional largest common substructure *(TLCS) problem by referring to the traditional LCS concept. With different matching rules, we thus define four versions of TLCS problems. We also show that two of the TLCS problems are $\mathcal{NP}$-hard by reducing the k-clique decision problem to them.*

1 Introduction

The similarity or distance of one-dimensional data can usually be measured by the algorithms for the *longest common subsequence* (LCS) problem [2, 12–16, 23] or the *edit distance* problem [22]. These problems have been extensively studied for several decades since 1970.

However, with the increasing motivation for computing the similarity for higher dimensional data, the LCS algorithms are no longer feasible. For example, to find the functionality of the secondary and tertiary structure of the proteins and RNA [5], and to compute the similarity of the pictures for identifying pictures in clinical diagnosis and criminal investigation are two application examples with data dimension higher than one. The increasing demand for computing the similarity of two-dimensional data is essential. In 1977, Knuth *et al.* [17] presented a method to find the pattern matching in one-dimensional data. And then the researches for computing two-dimensional data focused on pattern matching [4, 18].

In 1987, Chang *et al.* [7] presented a way of representing a two-dimensional picture by 2D strings. With this method, one can transform a picture into a one-dimensional string with iconic indices [26]. They also presented three types of similarity relation between two iconic objects. Chang *et al.* [6] proposed a similarity retrieval algorithm, called 2D-string-LCS, to retrieve the most similar picture whose type similarity is the largest in the image database. In 1992, Lee and Hsu [20] proposed another similarity retrieval definition with 2D C-strings. The most previous researches of 2D strings focused on the similarity retrieval of images, pattern matching, image database design and its variants [6, 8, 9, 11, 19–21, 25]. With theoretical interest, in 2000, Guan *et al.* [11] proved that the problems of finding maximum similar subpictures of relations type-0 and type-1 of 2D strings are NP-hard.

Without transforming the two-dimensional data into 2D strings and iconic indices, in 1998, Baeza-Yates [3] presented a method to compute the edit distance between two matrices. In 2008, Amir *et al.* [1] gave another similarity definition of

*This research work was partially supported by the Ministry of Science and Technology of Taiwan under contract MOST 104-2221-E-110-018-MY3. This research was also partially supported by the "Online and Offline Integrated Smart Commerce Platform (3/4)" of the Institute for Information Industry, which is subsidized by the Ministry of Economy Affairs of Taiwan.

†Corresponding author. E-mail: cbyang@cse.nsysu.edu.tw (Chang-Biau Yang).

two matrices, which is the *two-dimensional largest common substructure* (TLCS, which was shortened as 2D LCS by Amir, but it is called TLCS in this paper.) problem. Amir's definition of the TLCS problem is exactly the same as the type-1 relation in 2D strings [7, 11]. However, there is no connection between 2D strings and Amir's work. Amir *et al.* proved the $\mathcal{NP}$-hardness of the TLCS problem with another way.

In this paper, we give more general definitions of the TLCS problem. We first present eight possible matching rules for the TLCS problem. Then we show that four of them are valid, and the others are invalid. Among these valid generalized TLCS problems, $P(ENE)$ (introduced later) is exactly the same as type-0 relation in 2D strings [7, 11]. Although the proof technique of $\mathcal{NP}$-hardness of type-0 relation in 2D strings [11] can be applied to the proof that $P(ENL)$ (introduced later) is $\mathcal{NP}$-hard, we still give a different way to prove that $P(ENL)$ is $\mathcal{NP}$-hard. The $\mathcal{NP}$-hardness proof of $P(ENL)$ can also be applied to $P(ENE)$.

The organization of this paper is given as follows. In Section 2, we will give some notations, and review the picture retrieval problem and the TLCS problem. In Section 3, we will present the formal definitions of *two-dimensional largest common substructure* (TLCS) problems with our matching rules. We will also show that the two versions of the TLCS problem are $\mathcal{NP}$-hard with a different way in Section 4. Finally, in Section 5, we will give our conclusions.

2 Preliminaries

In this section, we first present the notations used in this paper in the following. Let S be a sequence of elements, where $S = s_1 s_2 s_3 \dots s_{|S|}$.

- S is called a *sequence* or a *string*.
- s_i is the ith element of S.
- $|S|$ represents the length of S, i.e., the number of elements in S.
- $S_{i..j} = s_i s_{i+1} s_{i+2} \dots s_j$ denotes a substring starting from the ith element and ending at the jth element of S.

Let A and B be two matrices, where $A = a_{1,1}, a_{1,2}, \dots, a_{1,c_A}, a_{2,1}, a_{2,2}, \dots, a_{2,c_A}, \dots, a_{r_A,1}, a_{r_A,2}, \dots, a_{r_A,c_A}$ and $B = b_{1,1}, b_{1,2}, \dots, b_{1,c_B}, b_{2,1}, b_{2,2}, \dots, b_{2,c_B}, \dots, b_{r_B,1}, b_{r_B,2}, \dots, b_{r_B,c_B}$. In addition, each $a_{i,j} \in \Sigma, 1 \le i \le r_A, 1 \le j \le c_A$, and each $b_{p,q} \in \Sigma, 1 \le p \le r_B, 1 \le q \le c_B$.

de		
	b	c
a		a

Figure 1: An example of the 2D string defined by Chang *et al.* [7], where the 2D string is $(d : e < b = c < a = a,\ a = d : e < b < a = c)$.

- $a_{i,j}$ is the entry at the ith row and the jth column of matrix A.
- $a_{i,*}$ denotes the ith row and $a_{*,j}$ denotes the jth column of matrix A.
- r_A represents the number of the rows and c_A represents the number of columns of matrix A.
- $\pi_A = r_A \times c_A$ represents the size of matrix A, i.e., the number of elements in A.

2.1 The Picture Retrieval Problem

In the pictorial information retrieval problem, the goal is to to retrieve pictures which satisfy a certain picture query. For example, one may want to find all pictures with a car to the right of a tree. In 1987, Chang *et al.* [7] proposed a way for representing the two-dimensional data with 2D strings. The problem of pictorial information retrieval can be regarded as the 2D subsequence matching problem.

A 2D string can be represented with (S_r, S_c), where S_r and S_c denote the row relation and column relation, respectively. To represent the spatial relation, three special symbols $T = \{ = , < , : \}$ are used. For two objects, $=$ represents the same spatial relation, $<$ means the below-above spatial relation in S_r, or the left-right spatial relation in S_c. $:$ represents that two objects are at the same position, that is, the same coordinate. Suppose there are n objects. S_r and S_c can be represented as $s_1 s_2 ... s_{2n-1}$, where $s_i \in \Sigma$ when $i \bmod 2 = 1$, and $s_i \in T$ when $i \bmod 2 = 0$. Figure 1 shows an example for the 2D string.

There are many ways for describing 2D strings, including absolute 2D strings, normal 2D strings, 2D C-strings and so on [6–8, 19–21].

Furthermore, Chang *et al.* defined three different types of matches. Let a_{i_1,j_1} and a_{i_2,j_2} be two objects (elements) in matrix A, and b_{p_1,q_1} and b_{p_2,q_2} be two objects in matrix B. Suppose that a_{i_1,j_1} matches to b_{p_1,q_1} and a_{i_2,j_2} matches to b_{p_2,q_2} with type-i relation. The definitions are given as follows [7].

- type-0: $(i_1 - i_2) \times (p_1 - p_2) \geq 0$ and $(j_1 - j_2) \times (q_1 - q_2) \geq 0$
- type-1: $[(i_1 - i_2) \times (p_1 - p_2) > 0$ or $i_1 - i_2 = p_1 - p_2 = 0]$ and $[(j_1 - j_2) \times (q_1 - q_2) > 0$ or $j_1 - j_2 = q_1 - q_2 = 0]$
- type-2: $i_1 - i_2 = p_1 - p_2$ and $j_1 - j_2 = q_1 - q_2$

In 2000, Guan *et al.* [11] proved the problems of finding maximum similar subpictures of relations type-0 and type-1 are $\mathcal{NP}$-hard by reducing from the Boolean satisfiability problem (SAT) [10, 24]. In the SAT problem with each clause having exactly three literals, the set of Boolean variables is $\{x_1, x_2, \cdots, x_k\}$, and the Boolean formula is $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_n$, where each clause $c_i = v_{i,1} \vee v_{i,2} \vee v_{i,3}$, for $1 \leq i \leq n$. The transformation of matrix A and matrix B is given as follows.

- In matrix A: $[v_{i,j}, g_1(i,j), h_1(i,j)]$, which means that an object $v_{i,j}$ locates at entry $(g_1(i,j), h_1(i,j))$ of A.
- In matrix B: $[v_{i,j}, g_2(i,j), h_2(i,j)]$, which means that an object $v_{i,j}$ locates at entry $(g_2(i,j), h_2(i,j))$ of B.

$$g_1(i,j) = \begin{cases} 3i-2 & \text{if } j = 1, \\ 3i-1 & \text{if } j = 2, \\ 3i & \text{if } j = 3, \end{cases} \quad (1)$$

$$g_2(i,j) = \begin{cases} 3i & \text{if } j = 1, \\ 3i-1 & \text{if } j = 2, \\ 3i-2 & \text{if } j = 3, \end{cases} \quad (2)$$

$$h_1(i,j) = \begin{cases} 2p & \text{if } v_{i,j} \text{ is the Boolean variable } \bar{x_p}, \\ 2p-1 & \text{if } v_{i,j} \text{ is the Boolean variable } x_p, \end{cases} \quad (3)$$

$$h_2(i,j) = \begin{cases} 2p & \text{if } v_{i,j} \text{ is the Boolean variable } \bar{x_p}, \\ 2p-1 & \text{if } v_{i,j} \text{ is the Boolean variable } x_p, \end{cases} \quad (4)$$

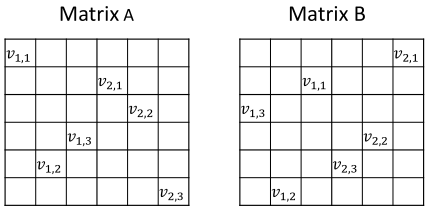


Figure 2: Matrices A and B constructed from the Boolean formula $\phi = (x_1 \vee x_3 \vee \bar{x_2}) \wedge (\bar{x_1} \vee x_2 \vee \bar{x_3})$.

In Figure 2, we show an example of the transformation. As one can see, there exist some maximum similar subpictures of relations type-0 and type-1 between matrices A and B with size n if and only if ϕ is satisfiable. The reason is that in each clause, there would be only one satisfiable relation match.

2.2 The Two-Dimensional Largest Common Substructure Problem

In 2008, Amir *et al.* [1] defined the *two-dimensional largest common substructure* (TLCS, which originally was named as two-dimensional longest common substructure and shortened as 2D-LCS by Amir *et al.*) problem. With the concept of the one-dimensional LCS, they defined the TLCS as the largest identical substructure which is obtained from two input matrices by deleting zero or more of their entries. A substructure is obtained from a matrix by deleting some entries but preserving the orientation of remaining entries in the plane. In a common substructure of two input matrices, the orientations of any two common elements (matches) are the same as those in the input matrices. In the one-dimensional LCS, the LCS is ordered on a line. The concept is also applicable in the two-dimensional problem as well. Hence, the TLCS problem is to find the maximal identical symbols which preserve their order in the two matrices. The formal TLCS definition and matching rules are given as follows.

Definition 1. [1] *Two-dimensional largest common substructure (TLCS) problem*
Input: *Matrix A of size $\pi_A = r_A \times c_A$ and matrix B of size $\pi_B = r_B \times c_B$.*
Output: *The maximum domain size of a one-to-one function $f : \{1, \ldots, r_A\} \times \{1, \ldots, c_A\} \rightarrow \{1, \ldots, r_B\} \times \{1, \ldots, c_B\}$ such that $a_{i,j} = b_{f(i,j)} = b_{p,q}$, $(i,j) \in domain(f)$. For every pair*

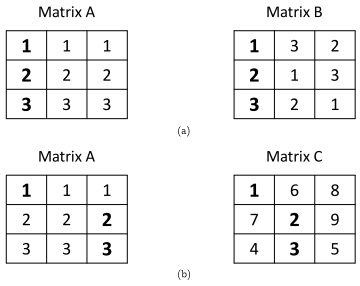


Figure 3: The TLCS of matrix A and matrix B or matrix C with the definition of Amir *et al.*, where the answers are shown by the boldface symbols. (a) The TLCS of matrices A and B. (b) The TLCS of matrices A and C.

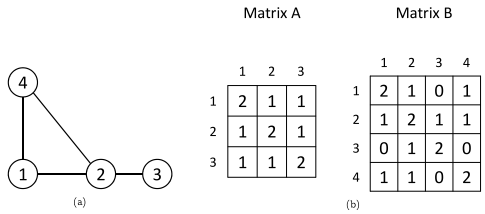


Figure 4: Matrices A and B constructed for the graph G and constant $k = 3$. (a) A graph G. (b) Matrix A constructed from a 3-clique and matrix B constructed from the graph G.

$(i_1, j_1), (i_2, j_2) \in domain(f)$ *that* $(p_1, q_1) = f(i_1, j_1)$ *and* $(p_2, q_2) = f(i_2, j_2)$*, the following holds:*

- $i_1 < i_2$ *if and only if* $p_1 < p_2$.
- $j_1 < j_2$ *if and only if* $q_1 < q_2$.
- $i_1 = i_2$ *if and only if* $p_1 = p_2$.
- $j_1 = j_2$ *if and only if* $q_1 = q_2$.

In fact, the TLCS problem defined by Amir *et al.* (Definition 1) is exactly the same as the similarity relation of type-1 on 2D strings, which has been proved to be $\mathcal{NP}$-hard [11]. However, Amir *et al.* did not refer to the papers studying the latter problem [7, 11]. As an example, Figure 3 shows three matrices A, B and C. It is clear that matrix B can be obtained by rotating some columns of matrix A. Thus, A and B are more similar than A and C. However, the TLCS of A and B is equal to the TLCS of A and C, where their sizes are both 3. Consequently, in this paper, we propose some different matching rules to disclose more similarity as how we look.

In 2008, Amir *et al.* proved the TLCS is NP-hard by reducing from the k-clique problem [1]. Suppose a graph $G = (V, E)$ and a constant k are given. Matrix A is transformed from a k-clique, of size $k \times k$, and matrix B from graph G, of size $|V| \times |V|$, as follows:

$$a_{i,j} = \begin{cases} 2 & \text{if } i = j \\ 1 & \text{otherwise} \end{cases} \quad (5)$$

$$b_{i,j} = \begin{cases} 2 & \text{if } i = j \\ 1 & \text{if } (V_i, V_j) \in E \\ 0 & \text{if } (V_i, V_j) \notin E \end{cases} \quad (6)$$

In Figure 4, we show an example of the transformation. As one can see, there exists a TLCS of size k^2 between matrices A and B if and only if there exists a clique of size k in G.

3 Problem Definitions

In this section, we present our formal definitions of the two-dimensional largest common substructure (TLCS) problem and give some properties.

3.1 Operator Definitions

To find the similarity of the two-dimensional data, we extend the concept for defining the one-dimensional LCS problem to the TLCS problem. The inputs of the TLCS problem are two matrices of characters, and we aim to find the *largest common substructure*. We regard matrix A of size $r_A \times c_A$ as the index set $T_A = \{(i, j) | 1 \le i \le r_A$ and $1 \le j \le c_A\}$. A *substructure* of A is defined as an index subset of T_A. A *common substructure* U of two matrices A and B is defined as an set of index pairs, one from A and the other from B, which obey the matching rules. Since each index of A or B has two tuples, U consists of four-tuples. That is, the set of *matching index pairs* is defined as $U = \{(i, j, p, q) \mid a_{i,j} = b_{p,q}$, and the matching rules are obeyed.$\}$. The matching rules will determine whether every two matches (two four-tuples) are allowed or not. We will introduce the matching rules in detail later. In the following, we first present the definition of the TLCS problem, which is the generalization of Definition 1 given by Amir *et al.* [1].

Definition 2. *Two-dimensional largest common substructure problem (TLCS)*

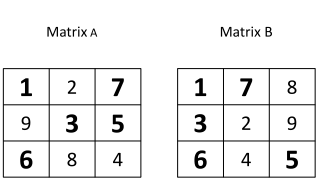


Figure 5: An example of the TLCS with $P(ENL)$ of two matrices.

Input: *Matrix A with size $\pi_A = r_A \times c_A$ and matrix B with size $\pi_B = r_B \times c_B$.*
Output: *The set CU of four-tuple indices of A and B with maximum cardinality, where every two elements in CU obey the matching rules (given later).*

The goal of the TLCS problem is to find the maximum number of matches such that every two matches satisfy the matching rules for determining whether the two matches are valid or not. In other words, the TLCS problem aims to find the *largest common substructure CU* between two character matrices. In Figure 5, we show an example for the TLCS definition with the matching rule $P(ENL)$ which is one of the matching rules we define (explained later).

In the traditional one-dimensional LCS problem, two matching rules have to be obeyed for one common sequence: (1) no duplicate match; and (2) no crossing matches. Similarly, in the TLCS problem, these two rules are still applicable. However, the concept of no crossing matches becomes more complicated in TLCS. In other words, the orientation of TLCS would be more various, and the matching rules would become more diverse. In the TLCS problem, there are eight possible directions in the orientation. They are left, right, upper, lower, upper left, upper right, lower left and lower right. Therefore, we divide the two-dimensional matching rules into two parts, *logical operator* and *orientation*. The logical operator combines the relationship between rows and columns. For logical operators, there are two ways, one is *And*(N) for considering both the orientation of rows and columns together, and the other is *Or*(R) for treating rows and columns as independent. The formal definition of logical operators is given as follows.

Definition 3. *Logical operator*
And (N): *Both row and column relationships are satisfied.*
Or (O): *The row relationship or the column relationships is satisfied.*

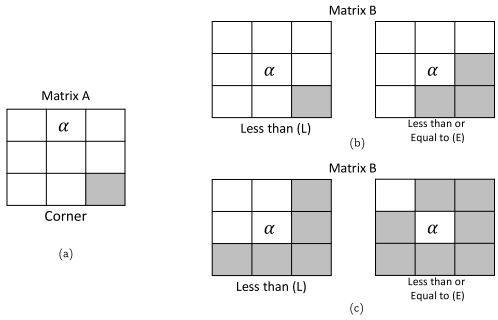


Figure 6: Examples for illustrating corners. (a) Matrix A. (b) The definition of corners with logical operator N. (c) The definition of corners with logical operator O.

For two elements a_{i_1,j_1} and a_{i_2,j_2}, there are two different conditions of the orientation. *Corner* means that $i_1 \neq i_2$ and $j_1 \neq j_2$, and *side* means that $i_1 = i_2$ or $j_1 = j_2$.

There are two possible corner rules. Suppose two elements a_{i_1,j_1} and a_{i_2,j_2} are in matrix A, and b_{p_1,q_1} and b_{p_2,q_2} are in matrix B, where (i_1, j_1, p_1, q_1), $(i_2, j_2, p_2, q_2) \in U$. For the row relationship, if $i_1 < i_2$ in matrix A, we have either $p_1 < p_2$ or $p_1 \leq p_2$, but not $p_1 > p_2$ in matrix B, because the orientation should be preserved. Similarly, for the column relationship, there are still two feasible cases, $q_1 < q_2$ or $q_1 \leq q_2$, when $j_1 < j_2$. On the other hand, if $i_1 > i_2$ in matrix A, we have either $p_1 > p_2$ or $p_1 \geq p_2$ in matrix B. We denote $p_1 < p_2$ or $p_1 > p_2$ as *less than* (L), and $p_1 \leq p_2$ or $p_1 \geq p_2$ as *less than or equal to* (E) when $i_1 < i_2$ or $i_1 > i_2$. The definition of the corner is given as follows.

Definition 4. *Corner: Let two elements a_{i_1,j_1} and a_{i_2,j_2} be in A where $i_1 \neq i_2$ and $j_1 \neq j_2$. And let two elements b_{p_1,q_1} and b_{p_2,q_2} be in matrix B. (i_1, j_1, p_1, q_1), $(i_2, j_2, p_2, q_2) \in U$.*
Less than (L): *$p_1 < p_2$ when $i_1 < i_2$, and $q_1 < q_2$ when $j_1 < j_2$.*
Less than or equal to (E): *$p_1 \leq p_2$ when $i_1 < i_2$, and $q_1 \leq q_2$ when $j_1 < j_2$.*

In Figure 6, we show examples for the definition of corners. Suppose the element a_{i_1,j_1} is of character α, a_{i_2,j_2} is marked by dark in matrix A. In matrix B, the allowed positions of element b_{p_2,q_2} are illustrated by dark cells if (i_1, j_1, p_1, q_1), $(i_2, j_2, p_2, q_2) \in U$.

For the definition of side, if $i_1 = i_2$ and $j_1 < j_2$, then we need keep only the column orientation because we could say that the element (i_1, j_1) is on

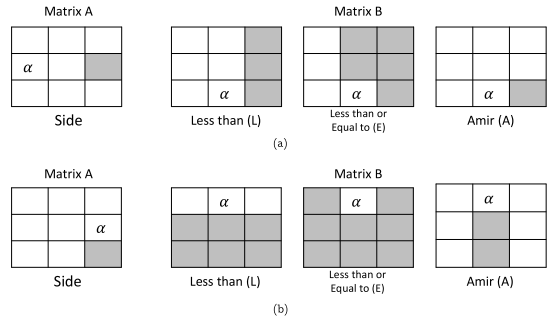


Figure 7: Two examples for illustrating the definition of sides.

the left side of (i_2, j_2). There are also two possible side rules, *less than* (L) and *less than or equal to* (E). In addition, we also define rule A for Amir *et al.*

Definition 5. *Side: Let two elements a_{i_1,j_1} and a_{i_2,j_2} be in matrix A where $i_1 = i_2$ or $j_1 = j_2$. And let two elements b_{p_1,q_1} and b_{p_2,q_2} be in matrix B. (i_1, j_1, p_1, q_1), $(i_2, j_2, p_2, q_2) \in U$.*
Less than (L): $p_1 < p_2$ when $i_1 < i_2$ and $j_1 = j_2$. $q_1 < q_2$ when $i_1 = i_2$ and $j_1 < j_2$.
Less than or Equal to (E): $p_1 \leq p_2$ when $i_1 < i_2$ and $j_1 = j_2$. $q_1 \leq q_2$ when $i_1 = i_2$ and $j_1 < j_2$.
Amir's rule (A): $p_1 < p_2$ and $q_1 = q_2$ when $i_1 < i_2$ and $j_1 = j_2$. $p_1 = p_2$ and $q_1 < q_2$ when $i_1 = i_2$ and $j_1 < j_2$.

Figure 7 shows two examples for the definition of sides. Suppose a_{i_1,j_1} is of character α and a_{i_2,j_2} is marked by dark in matrix A. In matrix B, the allowed positions of element b_{p_2,q_2} are presented by dark cells if (i_1, j_1, p_1, q_1), $(i_2, j_2, p_2, q_2) \in U$.

3.2 Problem Definitions

When we define a TLCS problem, we partition the definitions into three parts and denote the TLCS problem as $P(corner, operator, side)$, where *operator* denotes the logical operator for dealing with the relationship between the row index and column index, *corner* and *side* represent the row and column relationship. Accordingly, there are $2 \times 2 \times 2$ different versions of problems, two possible values (L, E) for *corner*, two possible values (And, Or) for *operator*, and two possible values (L, E) for *side*. Besides, the TLCS problem defined by Amir *et al.* is expressed as $P(LNA)$. Let a_{i_1,j_1} and a_{i_2,j_2} be in matrix A, and b_{p_1,q_1} and b_{p_2,q_2} be in matrix B, where (i_1, j_1, p_1, q_1), $(i_2, j_2, p_2, q_2) \in U$. Then we present the matching rules of the eight versions of the TLCS problem in the following.

*P(LNL): P(**L**ess than, a**N**d, **L**ess than)*

(i) $i_1 < i_2, j_1 < j_2 \rightarrow p_1 < p_2$ and $q_1 < q_2$
(ii) $i_1 < i_2, j_1 > j_2 \rightarrow p_1 < p_2$ and $q_1 > q_2$
(iii) $i_1 < i_2, j_1 = j_2 \rightarrow p_1 < p_2$
(iv) $i_1 = i_2, j_1 < j_2 \rightarrow q_1 < q_2$

*P(LNE): P(**L**ess than, a**N**d, less than or **E**qual to)*

(i) $i_1 < i_2, j_1 < j_2 \rightarrow p_1 < p_2$ and $q_1 < q_2$
(ii) $i_1 < i_2, j_1 > j_2 \rightarrow p_1 < p_2$ and $q_1 > q_2$
(iii) $i_1 < i_2, j_1 = j_2 \rightarrow p_1 \leq p_2$
(iv) $i_1 = i_2, j_1 < j_2 \rightarrow q_1 \leq q_2$

*P(ENL): P(less than or **E**qual to, a**N**d, **L**ess than)*

(i) $i_1 < i_2, j_1 < j_2 \rightarrow p_1 \leq p_2$ and $q_1 \leq q_2$
(ii) $i_1 < i_2, j_1 > j_2 \rightarrow p_1 \leq p_2$ and $q_1 \geq q_2$
(iii) $i_1 < i_2, j_1 = j_2 \rightarrow p_1 < p_2$
(iv) $i_1 = i_2, j_1 < j_2 \rightarrow q_1 < q_2$

*P(ENE): P(less than or **E**qual to, a**N**d, less than or **E**qual to)*

(i) $i_1 < i_2, j_1 < j_2 \rightarrow p_1 \leq p_2$ and $q_1 \leq q_2$
(ii) $i_1 < i_2, j_1 > j_2 \rightarrow p_1 \leq p_2$ and $q_1 \geq q_2$
(iii) $i_1 < i_2, j_1 = j_2 \rightarrow p_1 \leq p_2$
(iv) $i_1 = i_2, j_1 < j_2 \rightarrow q_1 \leq q_2$

*P(LOL): P(**L**ess than, **O**r, **L**ess than)*

(i) $i_1 < i_2, j_1 < j_2 \rightarrow p_1 < p_2$ or $q_1 < q_2$
(ii) $i_1 < i_2, j_1 > j_2 \rightarrow p_1 < p_2$ or $q_1 > q_2$
(iii) $i_1 < i_2, j_1 = j_2 \rightarrow p_1 < p_2$
(iv) $i_1 = i_2, j_1 < j_2 \rightarrow q_1 < q_2$

*P(LOE): P(**L**ess than, **O**r, less than or **E**qual to)*

(i) $i_1 < i_2, j_1 < j_2 \rightarrow p_1 < p_2$ or $q_1 < q_2$
(ii) $i_1 < i_2, j_1 > j_2 \rightarrow p_1 < p_2$ or $q_1 > q_2$
(iii) $i_1 < i_2, j_1 = j_2 \rightarrow p_1 \leq p_2$
(iv) $i_1 = i_2, j_1 < j_2 \rightarrow q_1 \leq q_2$

*P(EOL): P(less than or **E**qual to, **O**r, **L**ess than)*

(i) $i_1 < i_2, j_1 < j_2 \rightarrow p_1 \leq p_2$ or $q_1 \leq q_2$
(ii) $i_1 < i_2, j_1 > j_2 \rightarrow p_1 \leq p_2$ or $q_1 \geq q_2$
(iii) $i_1 < i_2, j_1 = j_2 \rightarrow p_1 < p_2$
(iv) $i_1 = i_2, j_1 < j_2 \rightarrow q_1 < q_2$

*P(EOE): P(less than or **E**qual to, **O**r, less than or **E**qual to)*

(i) $i_1 < i_2, j_1 < j_2 \rightarrow p_1 \leq p_2$ or $q_1 \leq q_2$
(ii) $i_1 < i_2, j_1 > j_2 \rightarrow p_1 \leq p_2$ or $q_1 \geq q_2$
(iii) $i_1 < i_2, j_1 = j_2 \rightarrow p_1 \leq p_2$

(iv) $i_1 = i_2, j_1 < j_2 \rightarrow q_1 \leq q_2$

In the eight matching rules, we can find that $P(ENE)$ is exactly the same as the type-0 relation of 2D strings [7, 11] and $P(LNA)$, defined by Amir *et al.*, is exactly the same as the type-1 relation [7, 11]

In the above definitions, not all matching rules can form a valid TLCS problem. A valid matching rule should be *symmetric*. That is, the optimal solution of two matrices A and B should also be the optimal solution of matrices B and A.

Let R_x denote the binary relation for satisfying $P(x)$ matching rule, where $x \in \{ LNL, LNE, ENL, ENE, LOL, LOE, EOL, EOE \}$. Let a_{i_1,j_1} and a_{i_2,j_2} be two distinct elements in matrix A, and b_{p_1,q_1} and b_{p_2,q_2} be two distinct elements in matrix B. We say that (i_1, j_1, i_2, j_2) R_x (p_1, q_1, p_2, q_2) if they satisfy the matching rule x. Furthermore, the TLCS problem with matching rule x is a valid problem if and only if the binary relation R_x is symmetric.

Lemma 1. *The binary relation R_{ENL} is symmetric.*

Proof. Suppose (i_1, j_1, i_2, j_2) R_{ENL} (p_1, q_1, p_2, q_2). Here, we want to prove that (p_1, q_1, p_2, q_2) R_{ENL} (i_1, j_1, i_2, j_2) is true.

To prove the symmetry of R_{ENL}, four cases are considered as follows.

Case 1: a_{i_1,j_1} and a_{i_2,j_2} are *corner*, b_{p_1,q_1} and b_{p_2,q_2} are *corner*.
Case 2: a_{i_1,j_1} and a_{i_2,j_2} are *side*, b_{p_1,q_1} and b_{p_2,q_2} are *side*.
Case 3: a_{i_1,j_1} and a_{i_2,j_2} are *corner*, b_{p_1,q_1} and b_{p_2,q_2} are *side*.
Case 4: a_{i_1,j_1} and a_{i_2,j_2} are *side*, b_{p_1,q_1} and b_{p_2,q_2} are *corner*.

In cases 1 and 2, the condition of the orientation of the a_{i_1,j_1}, a_{i_2,j_2} and b_{p_1,q_1}, b_{p_2,q_2} are the same. Hence, if (i_1, j_1, i_2, j_2) R_{ENL} (p_1, q_1, p_2, q_2), then (p_1, q_1, p_2, q_2) R_{ENL} (i_1, j_1, i_2, j_2). In rule (i) of $P(ENL)$, $p_1 \leq p_2$ and $q_1 \leq q_2$ when $i_1 < i_2$ and $j_1 < j_2$. Hence, in case 3, it should be: (1) $p_1 < p_2$, $q_1 = q_2$; or (2) $p_1 = p_2$, $q_1 < q_2$. In these two conditions, according to $P(ENL)$ rules (iii) and (iv) respectively, (p_1, q_1, p_2, q_2) R_{ENL} (i_1, j_1, i_2, j_2) is true. Consequently, R_{ENL} is symmetric in case 3. Similarly, case 4 is symmetric. Thus R_{ENL} is symmetric. □

With similar proofs (omitted here), we have the following lemma.

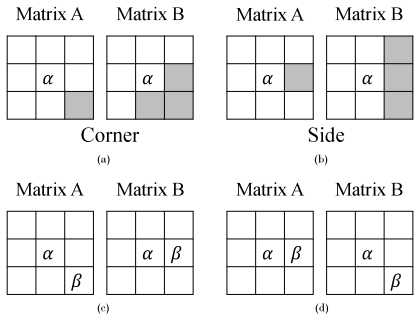


Figure 8: Examples of valid matching rules for $P(ENL)$. (a) The definition of corner for $P(ENL)$. (b) The definition of side for $P(ENL)$. (c) An example for $P(ENL)$. (d) Another example for $P(ENL)$.

Lemma 2. *The binary relations R_{ENE}, R_{LOL} and R_{LOE} are symmetric.*

By the above two lemmas, the following theorem can be obtained.

Theorem 1. *Each of the TLCS problem with $P(ENL)$, $P(ENE)$, $P(LOL)$ and $P(LOE)$ is valid.*

In Figure 8 we present an example for showing that R_{ENE} is symmetric. In Figure 8 (a) and (b), suppose $a_{i_1,j_1} = \alpha$ and a_{i_2,j_2} is marked by dark in matrix A. In matrix B, the allowed positions of element b_{p_2,q_2} are illustrated by dark cells. In Figure 8 (c) and (d), we can find that $(2, 2, 2, 3)$ R_{ENE} $(2, 2, 3, 3)$ and $(2, 2, 3, 3)$ R_{ENE} $(2, 2, 2, 3)$. This example can also be applied to R_{ENL}, R_{LOL} and R_{LOE}.

Lemma 3. *The binary relations R_{LNL}, R_{LNE}, R_{EOL} and R_{EOE} are not symmetric.*

Proof. For each binary relation, we give an example to show that it is not symmetric. It is clear that $(2, 2, 2, 3)$ R_{LNL} $(2, 2, 3, 3)$. However, $(2, 2, 3, 3)$ R_{LNL} $(2, 2, 2, 3)$ is not true. Hence, R_{LNL} is not symmetric.

Similarly, $(2, 2, 2, 3)$ R_{LNE} $(2, 2, 3, 3)$. However, $(2, 2, 3, 3)$ R_{LNE} $(2, 2, 2, 3)$ is not true. $(2, 2, 3, 3)$ R_{EOL} $(2, 2, 2, 1)$. Rule (iv) of $P(EOL)$ can be rewritten as $[i_1 = i_2, j_1 > j_2 \rightarrow q_1 > q_2]$. Thus, $(2, 2, 2, 1)$ R_{EOL} $(2, 2, 3, 3)$ is not true. $(2, 2, 3, 3)$ R_{EOE} $(2, 2, 2, 1)$. Rule (iv) of $P(EOE)$ can be rewritten as $[i_1 = i_2, j_1 > j_2 \rightarrow q_1 \geq q_2]$. However, $(2, 2, 2, 1)$ R_{EOE} $(2, 2, 3, 3)$ is not true.

Thus, this lemma holds. □

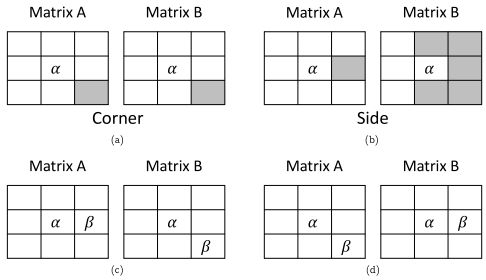


Figure 9: An example of the invalid matching rule for $P(LNE)$. (a) The definition of corner for $P(LNE)$. (b) The definition of side for $P(LNE)$. (c) An example for the valid $P(LNE)$ matching rule. (d) An example of an invalid matching rule for $P(LNE)$, which shows that R_{LNE} is not symmetric with respect to (c).

Figure 9 illustrates an example that R_{LNE} is not symmetric. In Figure 9 (a) and (b), suppose $a_{i_1,j_1} = \alpha$ and a_{i_2,j_2} is marked by dark in matrix A. In matrix B, the allowed positions of element b_{p_2,q_2} are illustrated by dark cells. It is clear that Figure 9 (c) $(2,2,2,3)$ R_{LNE} $(2,2,3,3)$, but in Figure 9 (d) $(2,2,3,3)$ R_{LNE} $(2,2,2,3)$ is not true.

With Lemma 3, we get the following result.

Theorem 2. *The problems $P(LNL)$, $P(LNE)$, $P(EOL)$ and $P(EOE)$ are not valid.*

We summarize the possible TLCS problems with various matching rules in Table 1. In the table, (i_1, j_1) and (i_2, j_2) are in matrix A, and the relation inside each cell indicates the relation between (p_1, q_1) and (p_2, q_2) in matrix B. Taking $P(ENL)$ as an example, two matching elements in the corner positons of matrix A with $i_1 < i_2$ and $j_1 < j_2$ means that $p_1 \leq p_2$ and $q_1 \leq q_2$ in matrix B, respectively. And two matching elements in the side positons of matrix A with $i_1 < i_2$ and $j_1 = j_2$ means that $p_1 < p_2$in matrix B. RA and CA represent that row alignment and column alignment, respectively. Row alignment means that the two-dimensional matrix is mapped into one-dimensional by the row-major scheme. Hence, the row alignment only considers the order of the rows. Similarly, the column alignment corresponds to the column-major scheme.

Definition 6. *Let CU_x denote the largest common substructure (optimal solution) of $P(x)$, $x \in \{ENL, ENE, LOL, LOE, RA, CA, LNA\}$, where RA and CA denote the problem with the row alignment and column alignment, respectively. And, let $|CU_x|$ denote its size.*

Table 1: The TLCS problems with various matching rules. Here, (i_1, j_1) and (i_2, j_2) are in matrix A, and each cell in column (i_1, i_2) indicates the relation between (p_1, p_2) of matrix B and in column (j_1, j_2) indicates the relation between (q_1, q_2).

Problem	Operator	Corner				Side				Valid
		$i_1 < i_2$	$j_1 < j_2$	$i_1 < i_2$	$j_1 > j_2$	$i_1 < i_2$	$j_1 = j_2$	$i_1 = i_2$	$j_1 < j_2$	
RA	and	$<$		$<$		$<$		$=$	$<$	Y
CA			$<$		$>$	$<$	$=$		$<$	Y
LNA		$<$	$<$	$<$	$>$	$<$	$=$	$=$	$<$	Y
LNL		$<$	$<$	$<$	$>$	$<$			$<$	N
LNE		$<$	$<$	$<$	$>$	$\leq$			$\leq$	N
ENL		$\leq$	$\leq$	$\leq$	$\geq$	$<$			$<$	Y
ENE		$\leq$	$\leq$	$\leq$	$\geq$	$\leq$			$\leq$	Y
LOL	or	$<$	$<$	$<$	$>$	$<$			$<$	Y
LOE		$<$	$<$	$<$	$>$	$\leq$			$\leq$	Y
EOL		$\leq$	$\leq$	$\leq$	$\geq$	$<$			$<$	N
EOE		$\leq$	$\leq$	$\leq$	$\geq$	$\leq$			$\leq$	N

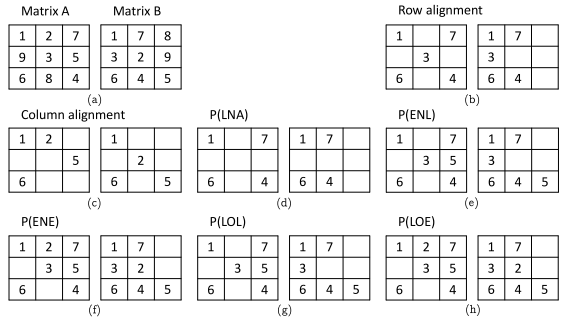


Figure 10: Comparison of solution sizes of various TLCS problems. (a) Matrices A and B. (b)-(h) The solutions of various TLCS problems.

In the above definition, for example, CU_{ENL} denotes the largest common substructure of $P(ENL)$, and $|CU_{ENL}|$ denotes its size. CU_{ENL} is represented by the maximal set of the four-tuples of the matching index pairs $\{(i,j,p,q)|$ $a_{i,j} = b_{p,q}$, and the $P(ENL)$ matching rule is obeyed.$\}$. The solution sizes of these TLCS problems can be partially ordered. Figure 10 illustrates the solutions with two input matrices A and B shown in Figure 10 (a). We can see that with different matching rules, it is possible to obtain a different solution with the same input. We have the following Theorem.

Theorem 3. *$|CU_{LNA}| \leq |CU_{ENL}| \leq (|CU_{ENE}|, |CU_{LOL}|) \leq |CU_{LOE}|$, and $|CU_{LNA}| \leq (|CU_{RA}|, |CU_{CA}|) \leq |CU_{LOL}| \leq |CU_{LOE}|$.*

Here, we do not give the formal proof. With the definitions of these TLCS problems, one can easily show the correctness of the above theorem. We illustrate the conceptual solutions in Figure 11, from which the above theorem can also be verified.

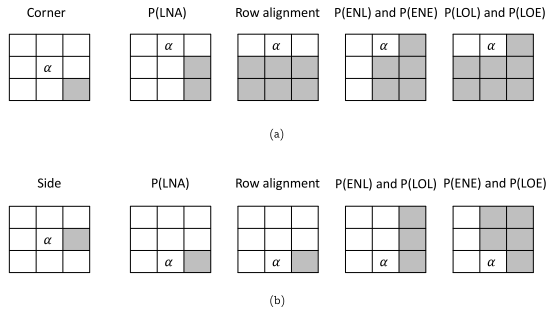


Figure 11: The illustration of conceptual solutions of various TLCS problems. (a) The solution of corners for various TLCS problems. (b) The solution of sides for various TLCS problems.

4 $\mathcal{NP}$-hardness Proof

In 2000, Guan *et al.* [11] used the same proof technique to show that the problems of finding maximum similar subpictures of relations type-0 and type-1 are $\mathcal{NP}$-hard by reducing from the Boolean satisfiability problem (SAT). In 2008, Amir *et al.* [1] proved that $P(LNA)$ is $\mathcal{NP}$-hard by reducing from the clique problem. Note that relation type-1 is exactly the same as $P(LNA)$, and type-0 is exactly the same as $P(ENE)$. The proof technique of type-0 can also applied to $P(ENL)$. But, the proof technique of $P(LNA)$ cannot be directly applied to $P(ENL)$ nor $P(ENE)$.

Inspired by Amir *et al.*, we provide another way to show that $P(ENL)$ is $\mathcal{NP}$-hard by reducing the k-clique decision problem to it. The transformation Γ_{ENL} for $P(ENL)$ is described in the following. The input instance of the k-clique problem is an undirected graph $G = (V, E)$ and a constant integer k, where $V = \{v_1, v_2, \cdots, v_n\}$, $|V| = n$. We can construct two matrices from graph G and constant k. Matrix A (size $3k \times 3k$) is transformed from a k-clique, and matrix B (size $3n \times 3n$) from graph G. Matrices A and B are defined as follows.

$$a_{i,j} = \begin{cases} \begin{cases} 2 & \text{if } i = j \\ 1 & \text{otherwise} \end{cases} \\ \text{if } i \bmod 3 = 2 \text{ and } j \bmod 3 = 2 \\ \begin{cases} \alpha & \text{if } i - j = -2, 0, 2 \\ \beta & \text{otherwise} \end{cases} \\ \text{if } i \bmod 3 = 0, 1 \text{ and } j \bmod 3 = 0, 1 \\ \gamma \quad \text{otherwise.} \end{cases} \tag{7}$$

$$b_{i,j} = \begin{cases} \begin{cases} 2 & \text{if } i = j \\ 1 & \text{if } (v_{\frac{i}{3}+1}, v_{\frac{j}{3}+1}) \in E \\ 0 & \text{if } (v_{\frac{i}{3}+1}, v_{\frac{j}{3}+1}) \notin E \end{cases} \\ \text{if } i \bmod 3 = 2 \text{ and } j \bmod 3 = 2 \\ \begin{cases} \alpha & \text{if } i - j = -2, 0, 2 \\ \beta & \text{otherwise} \end{cases} \\ \text{if } i \bmod 3 = 0, 1 \text{ and } j \bmod 3 = 0, 1 \\ \delta \quad \text{otherwise.} \end{cases} \tag{8}$$

Figure 12 illustrates an example of Γ_{ENL}. Every submatrix of size 3×3 in A and B corresponds to one edge of the graph. The center of each submatrix in A or B is set to 1 if and only if the corresponding edge exists in a k-clique or the graph G. Note that in A, no element is set to 0, since a k-clique is a complete graph with k vertices. It is obvious that in Γ_{ENL}, the required time for producing matrices A and B is $O(k^2)$ and $O(n^2)$, respectively.

It is clear that $|CU_{ENL}| \leq Min(|A|, |B|) = |A|$. Furthermore, $a_{x,y} \neq b_{x\prime,y\prime}$ when $(x, y), (x\prime, y\prime) \in \{(3i-2, 3j-1), (3i, 3j-1), (3i-1, 3j-2), (3i-1, 3j) | 1 \leq i, j \leq n\}$. Only other elements may be matched between A and B. Thus, $|CU_{ENL}|$ is bounded by $5k^2$.

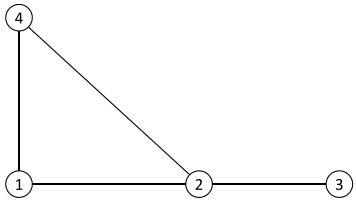
In the following, we will prove that there $|CU_{ENL}| = 5k^2$ if and only if there exists a k-clique in graph G. More precisely, if $|CU_{ENL}| = 5k^2$ and element $b_{p,p} = 2$ of B is one common element in CU_{ENL}, then vertex v_p should be picked as one vertex of the k-clique. The following lemmas prove the above statement.

Lemma 4. *Suppose* $(a_{i,i}, b_{p,p}) \in CU_{ENL}$ *with* $a_{i,i} = b_{p,p} = 2$. *If* $|CU_{ENL}| = 5k^2$, *then* $(i-1, i+1, p-1, p+1), (i+1, i-1, p+1, p-1) \in CU_{ENL}$.

Proof. It is evident that $a_{i-1,i+1} = a_{i+1,i-1} = b_{p-1,p+1} = b_{p+1,p-1} = \alpha$ when $a_{i,i} = b_{p,p} = 2$. If $|CU_{ENL}| = 5k^2$, each element of α in A has to match with one element in B. In $P(ENL)$, rule (ii) $[i_1 < i_2, j_1 > j_2 \rightarrow p_1 \leq p_2$ and $q_1 \geq q_2]$ forces $a_{i-1,i+1}$ to match with $b_{p-1,p+1}$ when $(i, i, p, p) \in CU_{ENL}$. Thus, $(i-1, i+1, p-1, p+1) \in CU_{ENL}$. Similarly, $(i+1, i-1, p+1, p-1) \in CU_{ENL}$. □

Lemma 5. *Suppose* $(a_{i_1,i_1}, b_{p_1,p_1})$, $(a_{i_2,i_2}, b_{p_2,p_2}) \in CU_{ENL}$ *with* $a_{i_1,i_1} = a_{i_2,i_2} = b_{p_1,p_1} = b_{p_2,p_2} = 2$. *If* $|CU_{ENL}| = 5k^2$, *then* (i_1, i_2, p_1, p_2), $(i_2, i_1, p_2, p_1) \in CU_{ENL}$.

Proof. We prove it by contradiction. Suppose that $(i_1, i_2, p_1, p_2) \notin CU_{ENL}$. Without loss of generality, assume that $i_1 \leq i_2 - 3$ and $p_1 \leq p_2 - 3$ in



(a)

Matrix A

	1	2	3	4	5	6	7	8	9
1	α	γ	α	β	γ	β	β	γ	β
2	γ	2	γ	γ	1	γ	γ	1	γ
3	α	γ	α	β	γ	β	β	γ	β
4	β	γ	β	α	γ	α	β	γ	β
5	γ	1	γ	γ	2	γ	γ	1	γ
6	β	γ	β	α	γ	α	β	γ	β
7	β	γ	β	β	γ	β	α	γ	α
8	γ	1	γ	γ	1	γ	γ	2	γ
9	β	γ	β	β	γ	β	α	γ	α

Matrix B

	1	2	3	4	5	6	7	8	9	10	11	12
1	α	δ	α	β	δ	β	β	δ	β	β	δ	β
2	δ	2	δ	δ	1	δ	δ	0	δ	δ	1	δ
3	α	δ	α	β	δ	β	β	δ	β	β	δ	β
4	β	δ	β	α	δ	α	β	δ	β	β	δ	β
5	δ	1	δ	δ	2	δ	δ	1	δ	δ	1	δ
6	β	δ	β	α	δ	α	β	δ	β	β	δ	β
7	β	δ	β	β	δ	β	α	δ	α	β	δ	β
8	δ	0	δ	δ	1	δ	δ	2	δ	δ	0	δ
9	β	δ	β	β	δ	β	α	δ	α	β	δ	β
10	β	δ	β	β	δ	β	β	δ	β	α	δ	α
11	δ	1	δ	δ	1	δ	δ	0	δ	δ	2	δ
12	β	δ	β	β	δ	β	β	δ	β	α	δ	α

(b)

Figure 12: An example of Γ_{ENL} for proving the $\mathcal{NP}$-hardness of $P(ENL)$. (a)A graph G. (b) Matrix A constructed from a 3-clique and matrix B constructed from the graph G.

Γ_{ENL}. To ensure $|CU_{ENL}| = 5k^2$, $a_{i_1,i_2} = 1$ is forced to match with one element $b_{p\prime,q\prime}$ in B, where $p\prime \neq p_1$ or $q\prime \neq p_2$.

If $p\prime \neq p_1$, there are two cases: (1) $p\prime \leq p_1 - 3$ and (2) $p\prime \geq p_1 + 3$ (In Γ_{ENL}, $|p\prime - p_1| \geq 3$ because $b_{p\prime,q\prime} = 1$ and $b_{p_1,p_2} = 1$.). In case (1), $(i_1-1, i_1+1, p_1-1, p_1+1) \in CU_{ENL}$ (by Lemma 4). If $(i_1, i_2, p\prime, q\prime) \in CU_{ENL}$, then $p_1 - 1 \leq p\prime$ is got since $i_1 - 1 < i_1$ and $i_1 + 1 < i_2$ (by rule (i) of $P(ENL)$). It is a contradiction. And in case (2), $(i_1 + 1, i_1 - 1, p_1 + 1, p_1 - 1) \in CU_{ENL}$ (by Lemma 4). $If (i_1, i_2, p\prime, q\prime) \in CU_{ENL}$, then $p_1 + 1 \geq p\prime$ since $i_1 + 1 > i_1$ and $i_1 - 1 < i_2$ (by rule (ii) rewritten as $[i_1 > i_2,\ j_1 < j_2 \rightarrow p_1 \geq p_2$ and $q_1 \leq q_2]$). It is a contradiction. By these two cases, we conclude that $p\prime = p_1$.

Similarly, we can obtain that $q\prime = p_2$. Thus, $(i_1, i_2, p_1, p_2) \in CU_{ENL}$ is true. With a similar proof, we can also get that $(i_2, i_1, p_2, p_1) \in CU_{ENL}$. □

Lemma 6. *$|CU_{ENL}| = 5k^2$ if and only if there exists a k-clique in graph G.*

Proof. If there is a k-clique in G, it is obvious that in Γ_{ENL}, $5k^2$ elements in B can be matched with elements in matrix A.

If $|CU_{ENL}| = 5k^2$, it means that we have to pick up k elements of value 2 in matrix B because there are also k elements of value 2 in matrix A. We denote these k elements in matrix B as $b_{p_1,p_1}, b_{p_2,p_2}, \ldots, b_{p_k,p_k}$. With Lemma 5, we know that each $b_{p_i,p_j} = 1$, $1 \leq i, j \leq k$, has to match with one element of A. Therefore, by Γ_{ENL}, the matches we pick up in matrix B correspond to a k-clique in G. □

With Lemma 6, we have the following result.

Theorem 4. *The TLCS problem with $P(ENL)$ is $\mathcal{NP}$-hard.*

Similarly, the reduction and Lemma 6 can also be applied to $P(ENE)$, thus we have the following result.

Theorem 5. *The TLCS problem with $P(ENE)$ is $\mathcal{NP}$-hard.*

5 Conclusion

In this paper, we present the more general definitions of the *two-dimensional largest common substructure* (TLCS) problems with various matching rules. To meet different demands, we present different types of corners, operators and sides. With various combinations of corners, operators and sides, we define several TLCS problems, including $P(ENL)$, $P(ENE)$, $P(LOL)$ and $P(LOE)$. We show that two TLCS problems $P(ENL)$ and $P(ENE)$ are $\mathcal{NP}$-hard by reducing the k-clique decision problem to them.

We have tried to prove that $P(LOL)$ and $P(LOE)$ are $\mathcal{NP}$-hard. However, we did not succeed. Our conjecture is that $P(LOL)$ and $P(LOE)$ are $\mathcal{NP}$-hard. It is worthy to discover whether these two problems are $\mathcal{NP}$-hard or not.

References

[1] A. Amir, T. Hartmana, O. Kapaha, B. R. Shaloma, and D. Tsur, "Generalized LCS," *Theoretical Computer Science*, Vol. 409, pp. 438–449, 2008.

[2] H.-Y. Ann, C.-B. Yang, C.-T. Tseng, and C.-Y. Hor, "A fast and simple algorithm for computing the longest common subsequence of run-length encoded strings," *Information Processing Letters*, Vol. 108, pp. 360–364, 2008.

[3] R. A. Baeza-Yates, "Similarity in two-dimensional strings," *Proc. of the 4th Annual International Conference on Computing and Combinatorics(COCOON '98)*, pp. 319–328, London, UK, 1998.

[4] R. Bird, "Two dimensional pattern matching," *Theoretical Computer Science*, Vol. 6, pp. 168–170, 1977.

[5] C.-I. Brndn and J. Tooze, *Introduction to Protein Structure*. Garland, 1999.

[6] S.-K. Chang, E. Jungert, and Y. Li, "Representation and retrieval of symbolic pictures using generalized 2D strings," *SPIE Proc. Visual Communications and Image Processing*, pp. 1360–1372, Philadelphia, 1989.

[7] S.-K. Chang, Q.-Y. Shi, and C.-W. Yan, "Iconic indexing by 2-D strings," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, No. 3, pp. 413–428, 1987.

[8] S. Chang and Y. Li, "Representation of multi-resolution symbolic and binary pictures using 2D h-strings," *Languages for Automation: Symbiotic and Intelligent Robots, 1988., IEEE Workshop on*, pp. 190–195, 1988.

[9] S. Chang, C. Yan, D. Dimitroff, and T. Arndt, "An intelligent image database system," *Software Engineering, IEEE Transactions on*, Vol. 14, No. 5, pp. 681–688, May 1988.

[10] S. A. Cook, "The complexity of theorem-proving procedures," *Proc. of the third annual ACM symposium on Theory of computing*, New York, USA, pp. 151–158, 1971.

[11] D. Guan, C.-Y. Chou, and C.-W. Chen, "Computational complexity of similarity retrieval in a pictorial database," *Information Processing Letters*, Vol. 75, No. 3, pp. 113 – 117, 2000.

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

[13] K.-S. Huang, C.-B. Yang, K.-T. Tseng, H.-Y. Ann, and Y.-H. Peng, "Efficient algorithms for finding interleaving relationship between sequences," *Information Processing Letters*, Vol. 105, pp. 188–193, 2008.

[14] K.-S. Huang, C.-B. Yang, K.-T. Tseng, Y.-H. Peng, and H.-Y. Ann, "Dynamic programming algorithms for the mosaic longest common subsequence problem," *Information Processing Letters*, Vol. 102, pp. 99–103, 2007.

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

[16] C. S. Iliopoulos and M. S. Rahman, "Algorithms for computing variants of the longest common subsequence problem," *Theoretical Computer Science*, Vol. 395, pp. 255–267, 2008.

[17] D. Knuth, J. Morris, and V. Pratt, "Fast pattern matching in strings," *SIAM Journal on Computing*, Vol. 6, pp. 323–350, 1977.

[18] K. Krithivasan and R. Sitalakshmi, "Efficient two-dimensional pattern matching in the presence of errors," *Information Sciences*, Vol. 43, pp. 169–184, 1987.

[19] S.-Y. Lee and F.-J. Hsu, "2D c-string: A new spatial knowledge representation for image database systems," *Pattern Recognition*, Vol. 23, No. 10, pp. 1077 – 1087, 1990.

[20] S.-Y. Lee and F.-J. Hsu, "Spatial reasoning and similarity retrieval of images using 2D c-string knowledge representation," *Pattern Recognition*, Vol. 25, No. 3, pp. 305 – 318, 1992.

[21] S.-Y. Lee, M.-K. Shan, and W.-P. Yang, "Similarity retrieval of iconic image database," *Pattern Recognition*, Vol. 22, No. 6, pp. 675 – 682, 1989.

[22] M. Pawlik and N. Augsten, "RTED: a robust algorithm for the tree edit distance," *Proc. of the VLDB Endowment*, Vol. 5, No. 4, pp. 334–345, 2011.

[23] Y.-H. Peng, C.-B. Yang, K.-S. Huang, C.-T. Tseng, and C.-Y. Hor, "Efficient sparse dynamic programming for the merged LCS problem with block constraints," *International Journal of Innovative Computing, Information and Control*, Vol. 6, pp. 1935–1947, 2010.

[24] T. J. Schaefer, "The complexity of satisfiability problems," *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing, STOC '78*, pp. 216–226, ACM, 1978.

[25] H. Tamura and N. Yokoya, "Image database systems: A survey," *Pattern Recognition*, Vol. 17, No. 1, pp. 29 – 43, 1984.

[26] S. Tanimoto, "An iconic/symbolic data structuring scheme," *Pattern recognition and artificial intelligence*, C.H. Chen , Ed. Newyork: Academic, 1976.