

The Consecutive Ones Submatrix Problem for Sparse Matrices

Jinsong Tan¹ and Louxin Zhang²

Abstract. A 0-1 matrix has the Consecutive Ones Property (C1P) if there is a permutation of its columns that leaves the 1's consecutive in each row. The Consecutive Ones Submatrix (C1S) problem is, given a 0-1 matrix A , to find the largest number of columns of A that form a submatrix with the C1P property. Such a problem finds application in physical mapping with hybridization data in genome sequencing. Let (a, b) -matrices be the 0-1 matrices in which there are at most a 1's in each column and at most b 1's in each row. This paper proves that the C1S problem remains NP-hard for (i) $(2, 3)$ -matrices and (ii) $(3, 2)$ -matrices. This solves an open problem posed in a recent paper of Hajiaghayi and Ganjali [1]. We further prove that the C1S problem is polynomial-time 0.8-approximatable for $(2, 3)$ -matrices in which no two columns are identical and 0.5-approximatable for $(2, \infty)$ -matrices in general. We also show that the C1S problem is polynomial-time 0.5-approximatable for $(3, 2)$ -matrices. However, there exists an $\varepsilon > 0$ such that approximating the C1S problem for $(\infty, 2)$ -matrices within a factor of n^ε (where n is the number of columns of the input matrix) is NP-hard.

Key Words. NP-hardness, Approximation algorithm, Consecutive ones property, Consecutive ones submatrix, Caterpillar spanning tree.

1. Introduction. A 0-1 matrix is said to have the Consecutive Ones Property (C1P) if there is a permutation of its columns that leaves the 1's consecutive in each row. This property was first mentioned by an archaeologist named Petrie in 1899 [2]. In the last three decades, the study of the C1P property for a given 0-1 matrix has found different applications in graph theory [2]–[4], computer science [5], [6] and genome sequencing [7].

For sequencing large genomes, one first constructs a clone library consisting of thousands of clones. Each clone represents a DNA fragment of small size. Then one maps each clone and assembles those resulting maps to determine the map of the entire genome using overlapping between the clones (overlapping can be achieved by using several restriction enzymes or other physical means). One approach to determine whether two clones overlap or not is based on hybridization with short probes. In this approach, a clone is exposed to a number of probes and which of these probes hybridize to the clone is determined. Therefore, for the map assembly problem with m clones and n probes, the experimental data is an $n \times m$ matrix $A = (a_{ij})$, where $a_{ij} = 1$ if clone c_i hybridizes with probe p_j , and $a_{ij} = 0$ otherwise. If the probes are STSs, each probe is unlikely to occur twice in the genome and hence this leads to physical mapping with unique probes [8]. If

¹ Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104-6389, USA. jinsong@seas.upenn.edu. The work was done when he was at the National University of Singapore.

² Department of Mathematics, National University of Singapore, Singapore 117543. matzlx@nus.edu.sg. The work was partially supported by the Singapore BMRC (BMRC01/1/21/19/140).

an experimental data matrix A is error-free, it has the C1P property and there are efficient algorithms for constructing a correct ordering of clones and probes [3], [9]–[12]. However, in reality, physical mapping is prone to various errors and the physical mapping problem with error and uncertainties becomes extremely difficult [12]–[15]. This raises the following problem

CONSECUTIVE ONES SUBMATRIX (C1S).

Instance: A 0-1 matrix A .

Question: Find the largest number of columns of A that form a submatrix with the C1P property.

The decision version of this problem is one of the earliest NP-complete problems appearing in Garey and Johnson's book [16]. However, the NP-completeness proof was misreferred to in the book. Recently, Hajiaghayi and Ganjali gave a proof [1]. Let (a, b) -matrices be the 0-1 matrices in which there are at most a 1's in each column and at most b 1's in each row. Actually, they proved that the C1S problem is NP-hard even for $(2, 4)$ -matrices. On the other hand, the C1S problem can be solved in polynomial time for $(2, 2)$ -matrices. Therefore, their work raises the problem of whether the C1S problem remains NP-complete or not for (i) $(3, 2)$ -matrices and (ii) $(2, 3)$ -matrices.

Studying the C1S problem for matrices with a small number of 1's in column and/or rows is not only theoretically interesting, but also practically important. Actually, this sparsity restriction was proposed by Lander and Istrail with hopes of making the mapping problem tractable (personal communication). However, as indicated in the following study, the C1S problem is still hard to solve for sparse matrices.

This paper is divided into five sections. We answer the open problem by proving its NP-completeness for the two special cases just mentioned (in Section 2). Furthermore, in Sections 3 and 4, we prove that the C1S problem is 0.5-approximatable for $(2, \infty)$ -matrices and for $(3, 2)$ -matrices, we also show that there exists a 0.8-approximation algorithm for the $(2, 3)$ -matrices in which no two columns are identical. On the contrary, we show that there exists an $\varepsilon > 0$ such that approximating the C1S problem for $(\infty, 2)$ -matrices within a factor of n^ε (n is the number of columns of the input matrix) is NP-hard. Finally, we conclude this work with several remarks in Section 5. Two closely related works are that the different versions of physical mapping with errors are shown to be NP-hard for sparse matrices in [17] and [18].

For basic concepts and knowledge in NP-hardness, polynomial time approximation, we refer readers to the books by Garey and Johnson [16] and Ausiello *et al.* [19].

2. NP-Hardness of the Problems. In this section we study the decision version of the C1S problem: given a 0-1 matrix A and a positive integer k , are there k columns of A that form a submatrix with the C1P property? A 0-1 matrix is an (a, b) -matrix if it has at most a 1's in each column and at most b 1's in each row, where a and b are integers or ∞ .

2.1. C1S for $(2, 3)$ -Matrices Is NP-Complete. To prove the C1S problem is NP-complete for $(2, 3)$ -matrices, we first define a special spanning tree problem and prove it is NP-complete. Formally, it is defined as follows.

DEFINITION 2.1. A tree is caterpillar if each node of degree $c \geq 3$ is adjacent to at least $c - 2$ leaf nodes.

SPANNING CATERPILLAR TREE IN A DEGREE-3 GRAPH.

Instance: A graph $G = (V, E)$ in which each node has degree at most 3.

Question: Does G contain a spanning caterpillar subtree T ?

LEMMA 2.1. *The Spanning Caterpillar Tree in the Degree-3 Graph problem is NP-complete.*

PROOF. The proof is by a reduction from the Hamiltonian Path problem for cubic graphs in which every node has degree 3 (see the comment to problem GT39 in [16]). Given a cubic graph $G = (V, E)$, we construct a new graph $G' = (V', E')$ by inserting a new node in each edge of G . Formally, we have

$$\begin{aligned} V' &= V \cup \{x_{uv} \mid (u, v) \in E\}, \\ E' &= \{(u, x_{uv}), (x_{uv}, v) \mid (u, v) \in E\}, \end{aligned}$$

where x_{uv} is called an *edge node*. The reduction follows from the fact that there exists a Hamiltonian path in G if and only if there exists a spanning caterpillar subtree of G' . We conclude the proof by proving this fact as follows.

The “only if” direction is easily seen. If there is a Hamiltonian path P in the cubic G , we obtain a desired spanning caterpillar subtree of G' by replacing each edge (u, v) by a two-edge path $ux_{uv}v$ if (u, v) is in the path P and attaching the inserted node x_{uv} to u otherwise.

For the “if” direction, we assume that such a spanning caterpillar subtree T exists in G' . By construction of G' , any degree-3 node in T must be a degree-3 node in G' and hence corresponds to a node in G . This also implies that each leaf adjacent to a degree-3 node in T must be an edge node. Therefore, by removing the leaves adjacent to the degree-3 nodes from T , we obtain a path P' in G' . Obviously, by construction, P' corresponds to a Hamiltonian path in G . \square

Next, we proceed to show the NP-completeness of the C1S problem for $(2, 3)$ -matrices by a reduction from the Spanning Caterpillar Tree in the Degree-3 Graph problem.

THEOREM 2.1. *The decision version of the C1S problem is NP-complete for $(2, 3)$ -matrices.*

PROOF. The proof is a refinement of the reduction used to prove the NP-completeness of the C1S problem in [1]. Given a $G = (V, E)$ with n nodes and e edges in which each node has degree at most 3, we consider its incidence matrix $B(G) = (b_{ij})$. Recall that $B(G)$ has a row for each node $v_i \in G$ and a column for each edge $e_j \in G$; and b_{ij} is 1 if v_i is incident to e_j in G and 0 otherwise. Since each node in G has degree at most 3, the incidence matrix $B(G)$ has exact two 1's in each column and at most three 1's in each row.

It can be easily seen that a subset of columns corresponds to a subgraph induced by the corresponding edges in G . Moreover, we claim that G has a spanning caterpillar subtree if and only if $B(G)$ has a submatrix of size $n \times (n - 1)$ with the C1P property. We conclude the proof by proving this claim.

We start with the “only if” direction. Suppose G has a spanning caterpillar subtree T in which each degree-3 node is adjacent to at least one leaf. Since G has n nodes, T contains $n - 1$ edges. Moreover, the submatrix induced by the columns corresponding to these $n - 1$ edges satisfies the C1P property. To prove this fact, we consider two cases.

Case 1. If T is just a Hamiltonian path $u_1, u_2, u_3, \dots, u_n$ of G , then the columns corresponding to edges $(u_1, u_2), (u_2, u_3), \dots, (u_{n-1}, u_n)$ (in this order) form an $n \times (n - 1)$ submatrix in which all the 1's are consecutive in each row.

Case 2. If T contains degree-3 nodes, then we consider the a longest path in T . Assume such a longest path P is $u_1, u_2, u_3, \dots, u_m$, where $m < n$. Since G has n nodes and T is a spanning caterpillar subtree, P contains exactly $n - m$ nodes $u_{i_1}, u_{i_2}, \dots, u_{i_{n-m}}$ of degree 3 in T such that each u_{i_j} is adjacent to a unique leaf v_{i_j} that is not in the path P . By arranging columns corresponding to edges $(u_1, u_2), (u_2, u_3), \dots, (u_{m_1}, u_{m_1})$ in the same order and inserting the column corresponding to (u_{i_j}, v_{i_j}) between those corresponding to (u_{i_j-1}, u_{i_j}) and (u_{i_j}, u_{i_j+1}) for each j , we obtain an $n \times (n - 1)$ submatrix in which all the 1's are consecutive in each row. This is because the row corresponding to a leaf contains only one 1.

Now we show the “if” direction. Suppose there exists a submatrix of size $n \times (n - 1)$ that has the C1P property in $B(G)$. The subgraph induced by the corresponding $n - 1$ edges must not admit a cycle (otherwise, at least one of the n rows cannot satisfy the C1P property). Hence, the $(n - 1)$ edges induce a spanning subtree of G since these edges are incident to n nodes. Suppose T does not satisfy our special requirement that each degree-3 node in it is adjacent to at least a leaf node. Then there exists a node $v \in T$ such that v is adjacent to three nodes $v_1, v_2, v_3 \in T$ with $\text{degree}(v_i) \geq 2$, $i = 1, 2, 3$. It is easy to see that there is no way to arrange the columns corresponding to (v_1, v) , (v_2, v) and (v_3, v) in $B(G)$ such that all four rows corresponding to v, v_1, v_2 and v_3 satisfy the C1P property. Therefore, the spanning subtree T is a caterpillar, i.e., each degree-3 node is adjacent to at least a leaf in T . \square

2.2. C1S for $(3, 2)$ -Matrices Is NP-Complete. Now we prove the NP-completeness of the C1S problem for $(3, 2)$ -matrices by a reduction from the following NP-complete problem for cubic graphs.

INDUCED DISJOINT-PATH-UNION SUBGRAPH (IDPUS).

Instance: Graph $G = (V, E)$ in which each node has degree 3, and a positive integer $k \leq |V|$.

Question: Does there exist a k -node subset $V' \subseteq V$ that induces a subgraph $G(V')$ (not necessarily connected) whose connected components are paths?

LEMMA 2.2. *IDPUS is NP-complete for cubic graphs.*

PROOF. According to the theorems by Yannakakis [20] and Lewis [21] (see also problem GT21 and the related comment on page 195 in the book [16] by Garey and Johnson), the problem of finding an induced subgraph with property Π of a particular given size in a cubic graph is NP-complete if Π is:

- (1) *Non-trivial.* Π is true for a single node and not satisfied by all the graphs in the given domain.
- (2) *Interesting.* There are arbitrarily large graphs satisfying Π .
- (3) *Easy.* For a given graph, the property can be verified in polynomial time.
- (4) *Hereditary.* If a graph satisfies property Π , then any of its induced subgraphs must also satisfy property Π .

We now prove our problem is NP-complete by showing that our property π , being a disjoint union of paths, satisfies the above conditions.

Let \mathcal{G} denote the set of all cubic graphs. The set \mathcal{G}' of graphs that are isomorphic to an induced subgraph of G is the domain of our property π .

It is easy to see that property π holds for a single node and an arbitrarily large graph in the domain, but not all graphs in the domain; also, π is hereditary since whenever G is a disjoint union of paths, so is its any subgraph. In addition, property π can be easily verified in polynomial time. Hence, our problem is NP-complete. \square

THEOREM 2.2. *The decision version of the C1S problem is NP-complete for (3, 2)-matrices.*

PROOF. We prove this NP-completeness result by giving a simple reduction from the IDPUS problem for cubic graphs. Given a cubic graph G with n nodes and m edges, we consider the transpose $B'(G) = (b_{ij})$ of the incidence matrix of G . In other words, $B'(G)$ is an $m \times n$ matrix in which each row corresponds to an edge of G and each column to a node of G . The entry b_{ij} is 1 if edge e_i has v_j as an endnode; it is 0 otherwise. Since G is a cubic graph, $B'(G)$ has exactly three 1's in each column and two 1's in each row.

We claim that a k -node subset V' induces a subgraph $G(V')$ of G that is a disjoint union of paths if and only if the k columns corresponding to nodes in V' form an $m \times k$ submatrix with the C1P property.

The “only if” direction is similar to Case 1 in the “only if” condition proof in Theorem 2.1, so we focus on the “if” direction. Suppose $B'(G)$ contains an $m \times k$ submatrix C with the C1P property. Let V' be the subset of nodes corresponding to the k columns in C . We show that V' induces a subgraph with the desired property. Let v be a node in V' . Since G is cubic, let the neighbors of v be v_1, v_2 and v_3 in G . If all its neighbors v_1, v_2 and v_3 are also in V' , then there is no way to arrange these four columns corresponding to v, v_1, v_2 and v_3 so that the two 1's are consecutive in rows corresponding to (v_1, v) and (v_2, v) and (v_3, v) . (Note that each column in the submatrix C contains exactly three 1's.) Hence, at most two of the v_i 's are in V' . This implies that V' induces a subgraph in which each node has degree at most 2. Furthermore, it is easy to see that the induced subgraph $G(V')$ does not contain a cycle (otherwise, at least one of the rows corresponding to the edges in the cycle cannot have its 1's consecutive under any column permutation). This concludes the proof. \square

3. Approximation Algorithms for $(2, \infty)$ -Matrices. We present a 0.8-approximation algorithm for the C1S problem for $(2, 3)$ -matrices: Given a $(2, 3)$ -matrix A , find a largest submatrix B of A consisting of a subset of A 's columns with the C1P property. We also show that a direct generalization of the algorithm turns out to have an approximation ratio 0.5 for $(2, \infty)$ -matrices. Without loss of generality, we may assume a $(2, \infty)$ -matrix $A = (a_{ij})$ satisfies the following properties in this section:

1. Row-distinguishability. No two rows are identical.
2. Column-distinguishability. No two columns are identical.

Recall that a $(2, \infty)$ -matrix A has at most two 1's in each column. If A contains any column with only one 1, we expand A into a 'full' $(2, \infty)$ -matrix A' whose columns contains exactly two 1's as follows. For each column j containing only one 1, we add an extra row i' that has 1 at the column j and 0 elsewhere. Assume A has k columns with a single 1. Then its expansion A' has the same number of columns as A and k more rows than A . Finally, we assume A has n columns. For any subset $C \subseteq \{1, 2, \dots, n\}$, we use $A(C)$ to denote the submatrix of A consisting of columns with indices in C . Then we give the following simple observation without proof.

PROPOSITION 3.1. *For any subset $C \subseteq \{1, 2, \dots, n\}$, $A(C)$ has the C1P property if and only if $A'(C)$ has the property.*

By Proposition 3.1, the C1S problem has the same solution to A and A' , and a good approximation solution to A' is also a good approximation to A . Therefore, assume that A has exactly two 1's in each column. Obviously, such a $(2, \infty)$ -matrix A defines uniquely a graph $G(A) = (V, E)$ that has A as the incidence matrix: Each row and column of A corresponds to a node and edge in $G(A)$, respectively. We assume $E = \{1, 2, \dots, n\}$. For a subset $C \subseteq E$, the subgraph of $G(A)$ induced by C has node set V and edge set C and is denoted by $G_C(A)$.

PROPOSITION 3.2. *Let $C \subseteq \{1, 2, \dots, n\}$. Then $A(C)$ has the C1P property if and only if $G_C(A) = (V, C)$ is a union of caterpillar subtrees. Here a single node is also considered as a caterpillar tree.*

PROOF. For convenience, we use the term edges and nodes in $G(A)$ and columns and rows in A interchangeably. Suppose $G_C(A)$ is a union of caterpillar subtrees C_1, C_2, \dots, C_h . For each $1 \leq t \leq h$, using the same discussion as in the proof of Theorem 2.1, the edges in $C_t = (V_t, E_t)$ form a submatrix $A_{V_t \times E_t}(C)$ with the C1P property. By arranging the columns in each caterpillar subtree in a block and arranging the columns within each block according to their connection in the corresponding subtree, we obtain a submatrix in which all the 1's in each row are arranged consecutively. This is because $A(i, j) = 0$ if node i and edge j do not belong to the same caterpillar tree. Hence, $A(C)$ has the C1P property.

Conversely, suppose $A(C)$ has the C1P property. Using the same discussion as in the proof of Theorem 2.1, each component of $G_C(A)$ must be a caterpillar subtree. Therefore, $G_C(A)$ is a union of caterpillar trees. \square

3.1. A 0.8-approximation for (2, 3)-Matrices

THEOREM 3.1. *There is a polynomial time 0.8-approximation algorithm for the CIS problem for any (2, 3)-matrix A .*

PROOF. Let the (2, 3)-matrix A have m rows and n columns. Then $G(A) = (V, E)$ has m nodes and n edges. Since each row contains at most three 1's in A , each node has degree at most 3.

For any subset $C \subseteq \{1, 2, \dots, n\}$, by Proposition 3.2, if $A(C)$ has the CIP property, $G_C(A)$ is a union of caterpillar subtrees. Therefore, $G_C(A)$ has at most $m - 1$ edges, and hence C contains at most $(m - 1)$ columns.

To find a 0.8-approximating solution to the CIS problem for A , we first find a spanning tree T in $G(A)$. Then we apply ALGORITHM A given below to T to find a union of caterpillar subtrees that have at least $0.8(m - 1)$ edges. The set of edges in the union gives a desired solution by Proposition 3.2. To describe ALGORITHM A, we recall some basic concepts in graph theory. In a rooted tree, the *depth* of a node is equal to the distance between the root and itself. A non-leaf node is said to be *internal*. For an internal node x in a rooted tree T' , we use $T'(x)$ to denote the subtrees rooted at x and use $p(x)$ to denote the *parent* of x that is the first node on the unique path from x to the root. Finally, an internal node is said to be *complete* if it is adjacent to 3 internal nodes. Obviously, a complete internal node is of degree 3.

ALGORITHM A

Input: A tree T with m nodes, in which each node has degree at most 3.

Output: A union of caterpillar subtrees with at least $0.8(m - 1)$ edges.

- 1 Pick a leaf r of T and root T at r . Let T_r be this rooted tree;
- 2 Initially, set $RT = T_r$, $RE = \varphi$;
- 3 Do a Breadth First Search on T_r , for each node v encountered;
- 4 Record its depth in T_r ;
- 5 If v is *complete* in T_r , *push* it to the end of list L_c
- 6 Repeat the following action until RT contains at most six nodes:
- 7 *Pop* a node x from the end of L_c ;
- 8 if x is *complete* in RT
- 9 Remove the edge between x and its parent $p(x)$;
- 10 $RT = RT - RT(x) - (x, p(x))$, $RE = RE \cup \{(x, p(x))\}$;
- 11 Output $T - RE$;

It is easy to see that the algorithm takes $O(m)$ time. Notice that a tree with at most six nodes is a caterpillar if each node has degree at most 3. Now we prove its connectedness. Consider a complete internal node x with a largest depth in the tree RT . (Note each node *poped* from list L_c at line 7 is with a largest depth in RT). First, the subtree rooted at x , $RT(x)$, is a caterpillar subtree since by assumption every internal node of it is adjacent

to at most two internal nodes. Second, since x is a complete internal node, there are at least two internal nodes in $RT(x)$ and hence $RT(x)$ contains at least four edges. This implies that each repeat of lines 6–10 removes at least five edges (including $(x, p(x))$). Therefore, lines 6–10 of the algorithm can be repeated at most $(m - 1)/5$ times and at most $0.2(m - 1)$ edges are removed from the input tree. \square

3.2. *0.5-Approximation for $(2, \infty)$ -Matrices.* Now we show that a direct generalization of ALGORITHM A has an approximation ratio 0.5 for $(2, \infty)$ -matrices.

Let A be a $(2, \infty)$ -matrix with m rows and n columns. Then the corresponding graph $G(A)$ has m nodes and n edges, and each node has degree at most Δ . We propose the following generalization to ALGORITHM A.

First, we find a spanning tree T of $G(A)$ and root it at a leaf r . Let the resulting rooted tree be T_r . Recall that a non-leaf node is called an internal node. For an internal node x , we use l_x to denote the number of leaves adjacent to it and d_x to denote its degree; x is said to be *complete* if it is adjacent to at least three internal nodes, i.e., $d_x - l_x \geq 3$. We find a union of caterpillar subtrees of T (hence $G(A)$) by repeating the following action until no complete internal nodes exist in T_r :

Pick a complete internal node x in T_r with the largest depth and then remove the edge $(x, p(x))$ and any other $d_x - l_x - 3$ edges between x and its internal neighbors.

In each repeat, we take away at least $l_x + 1 + 2(d_x - l_x - 1) = 2d_x - l_x - 1$ edges from T_r by removing $d_x - l_x - 2$ edges. Thus, at least half of the edges in T remain in the output union of caterpillar subtrees. Hence we proved that

THEOREM 3.2. *There is a polynomial time 0.5-approximation algorithm for the C1S problem when the input matrix has at most two 1's in each column.*

3.3. *0.5-Approximation for Arbitrary $(2, \infty)$ -Matrices.* In this subsection we present a 0.5-approximation algorithm for the C1S problem for general $(2, \infty)$ -matrices in which column-distinguishability is no longer assumed, that is, columns can be identical.

ALGORITHM B

Input: A tree $T = (V, E)$ with $\sum_{e \in E} \text{weight}(e) = W$.

Output: A union of caterpillar subtrees with total weight at least $0.5W$.

- 1 Pick a leaf r of T and root T at r , denote it by T_r ;
- 2 Initialize $E_1 = E_2 = \emptyset$
- 3 Do a Breath First Search on T_r , for each edge (i, j) encountered;
- 4 If $\text{MIN}(\text{depth}(i), \text{depth}(j))$ is odd, $E_1 = E_1 \cup e$;
- 5 Else, $E_2 = E_2 \cup e$;
- 6 If $\sum_{e \in E_1} \text{weight}(e) \geq \sum_{e \in E_2} \text{weight}(e)$, output E_1 ; else output E_2

THEOREM 3.3. *There is a polynomial time 0.5-approximation algorithm for the C1S problem when the input matrix has at most two 1's in each column.*

PROOF. Suppose the $(2, 3)$ -matrix A has m rows and n columns. Define graph $G(A) = (V, E)$, where each node corresponds to a row in A and each edge corresponds to a unique column in A ; further define $weight(e)$ ($e \in E$) to be the number of occurrences of the corresponding unique column in A .

Similar to the argument in Proposition 3.2, it is easy to see that for any subset $C \subseteq \{1, 2, \dots, n\}$, if $A(C)$ has the C1P property, then $G_C(A)$ is a union of caterpillar subtrees with total edge weight $|C|$. To find a 0.5-approximation solution to the C1S problem for A , we first find a maximum weighted spanning tree T in $G(A)$. Then we apply ALGORITHM B to T to find a union of caterpillar subtrees with total edge weight at least $0.5n$. The set of columns corresponding to the edges in this union form a desired solution to the C1S problem. If we define the *depth* of an edge (i, j) in a rooted tree to be the smaller one of $depth(i)$ and $depth(j)$, ALGORITHM B partitions the edges into two sets: one with odd depths and the other with even depths. It is easy to see that the edge-induced subgraph of either set is a union of caterpillar trees and the larger one must have a total edge weight of at least $0.5n$. Therefore, ALGORITHM B gives the desired solution. \square

4. A 0.5-Approximation Algorithm for $(3,2)$ -Matrices. In this section we use a partition theorem of Lovász in graph theory to obtain a 0.5-approximation algorithm for the C1S problem for $(3, 2)$ -matrices. The idea can also be generalized to $(\infty, 2)$ -matrices.

4.1. *The Algorithm.* Recall that a $(3, 2)$ -matrix contains at most three 1's in each column and at most two 1's in each row. Noting that any column permutation preserves the consecutiveness of 1's in a row with at most one 1, we only focus on the $(3, 2)$ -matrices which have exactly two 1's in each row in the rest of this section. Without loss of generality, we again assume the row-distinguishability property of the input matrix.

Let A be such a $(3, 2)$ -matrix. Then it defines uniquely a graph $G(A)$ with maximum degree 3 in which nodes correspond one-to-one to the columns in A and edges to the rows in A . Without loss of generality, we assume A has m rows and n columns and the corresponding graph $G(A)$ has node set $V = \{1, 2, \dots, n\}$. Furthermore, we have the following fact.

LEMMA 4.1. *Let $C = \{i_1, i_2, \dots, i_k\}$ be a subset of columns of A . Then the submatrix $A(C)$ of A consisting of the columns in C has the C1P property if and only if the subgraph $G(A)|_C$ induced by node subset $C \subseteq V$ is a union of paths. Here an isolated node is considered as a trivial path.*

PROOF. Assume $A(C)$ has the C1P property. Consider a node $i' \in C$. If it has three adjacent nodes i_j, i_k, i_l , then any permutation of C cannot keep the 1's consecutive on the rows corresponding to (i', i_j) , (i', i_k) and (i', i_l) since there are only two 1's in each row. Thus, each node in $G(A)|_C$ has degree at most 2. Similarly, $G(A)|_C$ cannot contain any cycles. Therefore, the node induced subgraph is a union of paths and isolated nodes.

Conversely, if $G(A)|_C$ is a union of paths, then if we arrange all the columns in each path together in the same order as they appear in the path, the resulting matrix has 1's consecutive on each row. This finishes the proof. \square

THEOREM 4.1. *There is a linear-time algorithm that always outputs a C1P submatrix consisting of at least $n/2$ columns given a $(3, 2)$ -matrix A with n columns.*

PROOF. Let A be a $(3, 2)$ -matrix. Recall that we assume that each row contains exactly two 1's in A . By Lemma 4.1, we only need to find a subset C containing at least $n/2$ nodes in $G(A)$ such that the induced subgraph $G(A)|_C$ is a union of paths and isolated nodes. The following ALGORITHM C is such an algorithm.

ALGORITHM C

Input: A $(3, 2)$ -matrix A .

Output: A subset C of columns of A such that $A(C)$ has the C1P property.

- 1 Construct the graph $G(A) = (V, E)$ as described above;
($G(A)$ has A as its incidence matrix; each node has degree at most 3.)
- 2 Initially, set $V' = \varnothing$ and $V'' = V$;
- 3 Repeat the following action until both $G(A)|_{V'}$ and $G(A)|_{V''}$ contain no nodes of degree more than 1;
- 4 Pick a node of degree at least 2 in $G(A)|_{V'}$ and move it to V'' , or
- 5 Pick a node of degree at least 2 in $G(A)|_{V''}$ and move it to V' ;
- 6 Output $C = V'$ if $|V'| \geq n/2$ or V'' otherwise.

Each execution of lines 4–5 increases the number of cut edges between V' and V'' by at least 1; and hence it will repeat at most $|E|$ times. Therefore, the algorithm takes linear time. Since $G(A)|_{V'}$ and $G(A)|_{V''}$ contains no nodes with degree more than 2, by Lemma 4.1, the output C is a desired subset of columns, i.e. $A(C)$ has the C1P property. This finishes the proof. \square

4.2. Generalization to $(\infty, 2)$ -Matrices. Given a graph G , we use $\Delta(G)$ to denote the largest degree of a node in G . ALGORITHM C indicates that the node set V of any graph G with $\Delta(G) = 3$ can be partitioned into V' and V'' such that $\Delta(G|_{V'}) \leq 1$ and $\Delta(G|_{V''}) \leq 1$. In fact, this is a special case of the following important partition theorem by setting $t_1 = t_2 = 1$.

THEOREM 4.2 [22]. *Let $G = (V, E)$ be a graph. Let t_1, t_2, \dots, t_k be non-negative integers such that $\sum_{i=1}^k (t_i + 1) - 1 = \Delta(G)$. Then V can be partitioned into k subsets that induce subgraphs G_1, G_2, \dots, G_k with $\Delta(G_i) \leq t_i$, for $i = 1, 2, \dots, k$.*

In addition, a desired partition can also be found in polynomial time [23]. By this theorem, the node set of a graph G can be decomposed into $\lceil (\Delta(G) + 1)/2 \rceil$ subsets that induce subgraphs with maximum degree of at most 1. This implies the following result.

PROPOSITION 4.1. *There is a polynomial-time algorithm that always outputs a subset C of at least $n/\lceil(\Delta + 1)/2\rceil$ columns such that $A(C)$ has the C1P property given an input $(\Delta, 2)$ -matrix A of n columns.*

4.3. Inapproximability of $(\infty, 2)$ -Matrices. In this section we show the inapproximability of the C1S problem for $(\infty, 2)$ -matrices.

LEMMA 4.2. *There exists an $\varepsilon > 0$ such that approximating the maximum IDPUS problem within a factor of n^ε is NP-hard, where n is the number of nodes in the input graph.*

PROOF. If an induced subgraph is a path of p nodes, we can find an independent set of size $\lceil p/2 \rceil$ by partitioning the nodes into two sets, each consisting of alternating nodes on the path and taking the larger one. Therefore, any solution to the maximum IDPUS of size k implies an independent set of size at least $\lceil k/2 \rceil$. Let the optimal solution to the independent set be OPT_{IS} and let the optimal solution to maximum IDPUS be OPT_{IDPUS} , we have $|OPT_{IDPUS}| \geq |OPT_{IS}|$ since any solution to the independent set is also a solution to maximum IDPUS. Therefore, it means that any $(1/\alpha)$ -approximation algorithm for IDPUS implies a $(1/2\alpha)$ -approximation algorithm for the Independent Set problem. Noticing that approximating the Independent Set problem within a factor of n^ε is NP-hard for some $\varepsilon > 0$ [24] (where n is the number of nodes in the input graph), we obtain the lemma. \square

THEOREM 4.3. *There exists an $\varepsilon > 0$ such that approximating the C1S problem for $(\infty, 2)$ -matrices within a factor of n^ε is NP-hard, where n is the number of columns in the input matrix.*

PROOF. Consider the reduction from the IDPUS problem for cubic graphs to the C1S problem for $(3, 2)$ -matrices in Theorem 2.2. If we remove the constraint that the input graph is cubic from the former, we obtain a reduction from the IDPUS problem for general graphs to the C1S problem for $(\infty, 2)$ -matrices. Similarly, we can also show that there is a size k (k nodes) solution to IDPUS if and only if there is a size k (k columns) solution to the C1S problem for $(\infty, 2)$ -matrices in the general case. This implies that the optimal solutions to both problems must be of the same size. Therefore, the theorem follows from Lemma 4.2. \square

5. Conclusion. In this paper we have answered an open problem posed in [1] by proving that the decision version of the C1S problem remains NP-complete for $(2, 3)$ - and $(3, 2)$ -matrices. To prove these results, we formulate two simple, but interesting NP-complete problems for cubic graphs. These two problems—Spanning Caterpillar Tree and Induced Disjoint-Path-Union Subgraph—may find applications in studying the complexity issues of other algorithmic problems.

We further study the approximation issue of the C1S problem. It is proved that the C1S problem is polynomial-time 0.8-approximatable for column-distinguishable $(2, 3)$ -

matrices, 0.5-approximatable for both $(2, \infty)$ -matrices and $(3, 2)$ -matrices. However, we also show that there exists an $\varepsilon > 0$ such that approximating the C1S problem for $(\infty, 2)$ -matrices within a factor of n^ε is NP-hard.

The C1S problem finds applications in physical mapping in genome sequencing. Our results in the complexity and approximation issues of the C1S problem demonstrate the difficulty of the physical mapping problem, which, we believe, are of value for computational biologists and bioinformaticians.

References

- [1] M. T. Hajiaghayi and Y. Ganjali. A note on the consecutive ones submatrix problem. *Inform. Process. Lett.*, **83** (2002), 163–166.
- [2] D. G. Kendall. Incidence matrices, interval graphs and seriation in archaeology. *Pacific J. Math.*, **28** (1969), 565–570.
- [3] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific J. Math.*, **15** (1965), 835–855.
- [4] J. S. Deogun and K. Gopalakrishnan. Consecutive retrieval property revisited. *Inform. Process. Lett.*, **69** (1999), 15–20.
- [5] S. P. Ghosh. File organization: the consecutive retrieval property. *Commun. ACM*, **15** (1972), 802–808.
- [6] M. Flammini, G. Gambosi and S. Salomone. *Boolean Routing*, pp. 219–233. Lecture Notes in Computer Science, vol. 725. Springer-Verlag, Berlin, 1993.
- [7] P. A. Pevzner. *Computational Molecular Biology: An Algorithmic Approach*. The MIT Press, Cambridge, MA, 2000.
- [8] S. Foote, D. Vollrath, A. Hilton and D. C. Page. The human Y chromosome: overlapping DNA clones spanning the euchromatic region. *Science*, **258** (1992), 60–66.
- [9] K. S. Booth and G. S. Lueker. Test for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *J. Comput. Systems Sci.*, **13** (1976), 335–379.
- [10] J. Meidanis, O. Porto and G. P. Telles. On the consecutive ones property. *Discrete Appl. Math.*, **88** (1998), 325–354.
- [11] M. Habib, R. McConnell, C. Paul and L. Viennot. Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing. *Theoret. Comput. Sci.*, **234** (2000), 59–84.
- [12] F. Alizadeh, R. M. Karp, D. K. Weissner and G. Zweig. Physical mapping of chromosomes using unique probes. *J. Comput. Biol.*, **2** (1995), 159–184.
- [13] W.-F. Lu and W.-L. Hsu. A test for the consecutive ones property on noisy data—application to physical mapping and sequence assembly. *J. Comput. Biol.*, **10**(5) (2003), 709–735.
- [14] D. S. Greenberg and S. Istrail. Physical mapping by STS hybridization: algorithmic strategies and the challenge of software evaluation. *J. Comput. Biol.*, **2** (1995), 219–273.
- [15] R. Mott, A. Grigoriev, and H. Lehrach. An algorithm to detect chimeric clones and random noise in genomic mapping. *Genetics*, **22** (1994), 482–486.
- [16] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA, 1979.
- [17] J. Atkins and M. Middendorf. On physical mapping and the consecutive ones property for sparse matrices. *Discrete Appl. Math.*, **71** (1996), 23–40.
- [18] S. Weis and R. Reischuk. The complexity of physical mapping with strict chimerism. in *Proc. 6th International Symposium on Computing and Combinatorics (COCOON 2000)*, pp. 383–395. Lecture Notes in Computer Science, vol. 1858. Springer-Verlag, Berlin, 2000.
- [19] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer-Verlag, New York, 2000.
- [20] M. Yannakakis. Node- and edge-deletion NP-complete problems. In *Proc. 10th Annual ACM Symposium on Theory of Computing* (1978), pp. 253–264.

- [21] J. M. Lewis. On the complexity of the maximum subgraph problem. In *Proc. 10th Annual ACM Symposium on Theory of Computing* (1978), pp. 265–274.
- [22] L. Lovász. On decomposition of graphs. *Stud. Sci. Math. Hung.*, **1** (1966), 237–238.
- [23] M. M. Halldórsson and H. C. Lau. Low-degree graph partitioning via local search with applications to constraint satisfaction, max cut, and 3-coloring. *J. Graph Algorithm Appl.*, **1**(3) (1997), 1–13.
- [24] D. S. Hochbaum (Editor) *Approximation Algorithms for NP-Hard Problems*. PWS Publishing, Boston, MA, 1995.