Discrete Applied Mathematics 160 (2012) 2760–2768

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Hardness results on the gapped consecutive-ones property problem

Ján Maňuch [a,b,1], Murray Patterson [b,*,1], Cedric Chauve [a]

[a] Department of Mathematics, Simon Fraser University, Burnaby, British Columbia, Canada

[b] Department of Computer Science, UBC, Vancouver, British Columbia, Canada

ARTICLE INFO

Article history:
Received 30 April 2010
Received in revised form 2 February 2012
Accepted 15 March 2012
Available online 10 April 2012

Keywords:
Consecutive-ones property
Ancestral genome reconstruction
Computational complexity
Graph bandwidth
Algorithm

ABSTRACT

Motivated by problems in comparative genomics and paleogenomics, we study the computational complexity of the Gapped Consecutive-Ones Property ((k, δ)-C1P) Problem: given a binary matrix M and two integers k and δ, decide if the columns of M can be permuted such that each row contains at most k blocks of ones and no two neighboring blocks of ones are separated by a gap of more than δ zeros. The classical C1P decision problem, which is known to be polynomial-time solvable is equivalent to the $(1, 0)$-C1P problem. We extend our earlier results on this problem [C. Chauve, J. Mauch, M. Patterson, On the gapped consecutive-ones property, in: Proceedings of the European Conference on Combinatorics, Graphs Theory and Applications (EuroComb), in: Electronic Notes in Discrete Mathematics, vol. 34, 2009, pp. 121–125] to show that for every $k \geq 2, \delta \geq 1$, $(k, \delta) \neq (2, 1)$, the (k, δ)-C1P Problem is NP-complete, and that for every $\delta \geq 1$, the (∞, δ)-C1P Problem is NP-complete. On the positive side, we also show that if k, δ and the maximum *degree* of M are constant, the problem is related to the classical Graph Bandwidth Problem and can be solved in polynomial time using a variant of an algorithm of Saxe [J.B. Saxe, Dynamic-programming algorithms for recognizing small-bandwidth graphs in polynomial time, SIAM Journal on Algebraic and Discrete Methods 1 (4) (1980) 363–369].

1. Introduction

Let M be a binary matrix with m rows and n columns. A *block* in a row of M is a maximal sequence of consecutive entries containing 1. A *gap* is a sequence of consecutive zeros that separates two blocks; the size of the gap is the length of this sequence of zeros. Matrix M is said to have the *Consecutive-Ones Property* (C1P) if its columns can be permuted such that each row contains at most one block (there are no gaps in this case). We call a permutation of the columns of M that witnesses this property a *consecutive-ones ordering* of M, and the resulting matrix of such a permutation is *consecutive*, and we say that M has the C1P or that M is a C1P matrix. The Consecutive-Ones Property was introduced, at least under this name, by Fulkerson and Gross [10] in 1965, motivated by applications in genetics. It has since then been the subject of intense research. From a combinatorial point of view, the structure of C1P matrices has been described in terms of forbidden submatrices and asteroidal triples by Tucker in [20]. The set of all consecutive-ones orderings of a C1P binary matrix can also be encoded in linear space using a data structure called a PQ-tree. From an algorithmic point of view, efficient algorithms have been developed to decide if a binary matrix M is a C1P matrix [2,12,17,16,15].

The C1P has also been widely used in molecular biology, in relation with physical mapping [1] and the reconstruction of ancestral genomes [6]. These applications aim at reconstructing genomes that cannot be sequenced. They rely on the

* Corresponding author. Tel.: +1 778 549 4587; fax: +1 604 822 5485.
E-mail addresses: jmanuch@cs.ubc.ca (J. Maňuch), murrayp@cs.ubc.ca, murray.patterson@gmail.com (M. Patterson), cedric.chauve@sfu.ca (C. Chauve).

[1] These authors have contributed equally.

0166-218X/$ – see front matter
doi:10.1016/j.dam.2012.03.019

following encoding of genomic information in a binary matrix: each column of the matrix represents a genomic marker (in general a DNA sequence, but other kinds of data can be used) that is believed to have been present and unique in the considered unsequenced genome, and each row of the matrix represents a set of markers that are believed to have been contiguous along a chromosome of the unsequenced genome. The goal is then to find one (or several, if possible) total order on the markers that respects all rows (i.e., that keeps all entries 1 together in each row). We refer the reader to [11] for a discussion of physical mapping and to [6] for a discussion of ancestral genome reconstruction. Hence, given a binary matrix M encoding such data, if each row of M encodes true information regarding the unsequenced genome (i.e., a set of markers that were really contiguous in this genome), since M is a C1P matrix, the PQ-tree of its consecutive-ones orderings represent all possible orders of the markers that respect all rows of M. However, a common problem in such applications is that matrices obtained from experiments do not have the C1P [11,6], due to some rows that do not encode sets of markers that were contiguous in the unknown genome.

A first general approach to handle a matrix M that does not have the C1P consists of transforming M into a matrix that has the C1P, while minimizing the modifications to M; such modifications can involve either removing rows, or columns, or both, or flipping some entries from 0 to 1 or 1 to 0. In all cases, the corresponding optimization problems have been proven NP-hard [8,9,14]. A second approach, that we follow here, consists of relaxing the condition of consecutivity of the ones of each row, by allowing gaps, with some restriction on the nature of these gaps. The question is then to decide if there is an ordering of the columns of M that satisfies these relaxed C1P conditions. As far as we know, the only restriction that has been considered is the number of gaps, either per row or in M. In [11], the authors introduced the notion of the *k-Consecutive-Ones Property* (k-C1P). A binary matrix M has the k-C1P when its set of columns can be permuted such that each row contains at most k blocks. They call a permutation of the columns of M that witnesses this property a *k-consecutive-ones ordering* of M, and the resulting matrix of such a permutation is *k-consecutive*. In [11], the authors show that deciding if a binary matrix M has the k-C1P is NP-complete, even if $k = 2$. Also, finding an ordering of the columns that minimizes the number of gaps in M is NP-complete even if each row of M has at most two ones [13].

In the present work, we follow the second approach, motivated by the problem of reconstructing ancestral genomes using max-gap clusters [6]: the restrictions to the allowed gaps are that both the number of gaps per row and the size of each gap are bounded. Formally, let k and δ be two integers. A binary matrix M is said to have the *(k, δ)-Consecutive-Ones Property*, denoted by (k, δ)-C1P, if its columns can be permuted such that each row contains at most k blocks and no gap is larger than δ. If any of the two parameters is unbounded, we replace k or δ with ∞. For instance, the k-C1P is equivalent to the (k, ∞)-C1P. Here, we call a permutation of the columns of M that witnesses this property a *(k, δ)-consecutive-ones ordering* of M, and the resulting matrix of such a permutation is *(k, δ)-consecutive*. The (k, δ)-C1P Problem is to decide if a given matrix M has the (k, δ)-C1P. This problem is naturally related to the classical Graph Bandwidth Problem, that asks, given a graph G and an integer δ, if the vertices of G can be linearly ordered in such a way that no edge spans more than δ vertices. Indeed, if M has two entries 1 in each row, it can be viewed as the incidence matrix of a graph G and G has bandwidth δ if and only if it satisfies the (∞, δ)-C1P. From an application point of view (i.e., paleogenomics and the reconstruction of ancestral genomes), answering the (k, δ)-C1P Problem for small values of both k and δ is very relevant. Indeed, in most cases, it is errors in computing the initial matrix M that results in M not having the C1P; these errors correspond to small gaps in some rows of this matrix that are due to small overlapping genome rearrangements or mistakes in identifying proper ancestral genomic markers.

In [5], we introduced the (k, δ)-C1P Problem and gave preliminary complexity and algorithmic results. In particular we showed that for every $\delta \geq 2$, the $(2, \delta)$-C1P Problem is NP-complete and that the $(3, 1)$-C1P problem is NP-complete. In the present work, we settle the complexity for all possible values of k and δ but one: we show that for every $k \geq 2, \delta \geq 1, (k, \delta) \neq (2, 1)$, that testing for the (k, δ)-C1P is NP-complete (Section 4). We also prove that the (∞, δ)-C1P Problem is NP-complete (Section 5). In both cases our proofs rely on reduction from 3SAT, although the techniques are relatively different. This leaves only one case open: the $(2, 1)$-C1P Problem. On the positive side, in Section 6 we show that if every row of an $m \times n$ matrix M has at most d entries 1, then the (k, δ)-C1P Problem can be solved in polynomial time $O(mn^{d+(k-1)\delta+1})$ by a variant of an algorithm described in [19] for recognizing graphs with a constant bandwidth. This algorithm was also described in [5].

2. Notation and conventions

First, we introduce some notation and conventions that we use throughout this paper. Given integers a, b, where $a \leq b$, $\langle a, b\rangle$ denotes the set $\{a, a+1, \ldots, b\}$. Let M be a binary $m \times n$ matrix with columns labeled by $\langle 1, n\rangle$. In the constructions used to show NP-completeness, we will divide columns of this matrix into ordered sequences of blocks $B_1, \ldots, B_p$ by designing rows enforcing the columns of each block to appear together and the blocks to appear in the order $B_1, \ldots, B_p$ (resp., in the reversed order), i.e., for any $i < j$, column $c \in B_i$ and $d \in B_j$, c appears before (resp., after) d in any (k, δ)-consecutive ordering of M. The columns of a block B_i will be denoted $B_i^1, \ldots, B_i^{|B_i|}$ and $B_i^{\langle a,b\rangle} = \{B_i^a, B_i^{a+1}, \ldots, B_i^b\}$, where $a \leq b$.

To specify a row in the matrix M, we use the convention of only listing in the square brackets, the columns that contain 1 in this row. For example, [1, 8, 5] represents a row with ones in columns 1, 5, and 8, and zeros everywhere else. We will also use blocks of M to specify columns in the block, for example, if $B_1 = \{1, 2, 3, 4, 5\}$, then $[B_1, 7]$ would mean [1, 2, 3, 4, 5, 7], $[B_1 \setminus \{B_1^2\}, 6, 7]$ would mean [1, 3, 4, 5, 6, 7], and $[B_1^{\langle 2,4\rangle}, 6]$ would mean [2, 3, 4, 6].

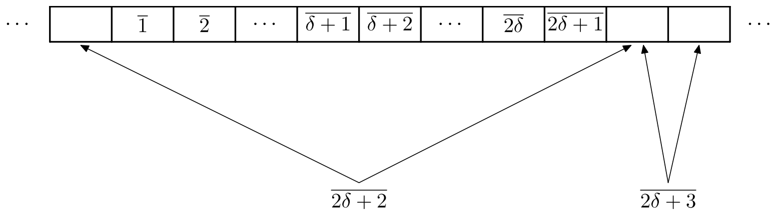

Fig. 1. Possible positions of columns $\overline{2\delta+2}$ and $\overline{2\delta+3}$.

3. Fixing the order of selected columns in a matrix

For every $k \geq 2$, $\delta \geq 1$, we have the following important property of matrices that have the (k, δ)-C1P. Note that the following construction does not depend on k as it uses only two ones per row.

Theorem 1. *For every $k \geq 2$ or $k = \infty$, $\delta \geq 1$ and $s \geq 2\delta + 3$, given binary matrix M on $n \geq s$ columns, $s + \delta + 1$ rows can be added to M to force s selected columns to appear together and in fixed order (or the reverse order) in any (k, δ)-consecutive ordering of M.*

Proof. Let $k \geq 2$ (or $k = \infty$), $\delta \geq 1$, $s \geq 2\delta + 3$ and $n \geq s$. Without loss of generality, let $S = \{\overline{1}, \ldots, \overline{s}\}$ be the subset of s columns that we want to force to appear together and in this order (or the reverse order) in any (k, δ)-consecutive ordering of M. We will show by induction on s that there are $s+\delta+1$ rows of the type $[\overline{c}, \overline{d}]$, where $1 \leq c < d \leq s$ and $|c-d| \leq \delta+1$, which force this order.

For the base case, let us assume that $s = 2\delta + 3$. We will show the base case by induction on δ. If $\delta = 1$, then $s = 2 \cdot 1 + 3 = 5$, and we add to M the following seven rows: $[\overline{1}, \overline{2}]$, $[\overline{2}, \overline{3}]$, $[\overline{3}, \overline{4}]$, $[\overline{4}, \overline{5}]$, $[\overline{1}, \overline{3}]$, $[\overline{2}, \overline{4}]$, and $[\overline{3}, \overline{5}]$. It is easy to check that the claim holds and that the number of rows used is exactly $s + \delta + 1$. Now assume that the claim holds for $\delta = \delta_0$ and $s = s_0 = 2\delta_0+3$, where $\delta_0 \geq 1$. We will show that it holds also for $\delta = \delta_0+1$ and $s = 2\delta+3 = 2\delta_0+5$. Using the induction hypothesis, there are $s_0 + \delta_0 + 1 = s - 2 + \delta - 1 + 1 = s + \delta - 2$ rows, which will force the correct order for columns $\overline{1}, \ldots, \overline{2\delta+1}$. Note that all of these rows $[\overline{c}, \overline{d}]$ satisfy the condition $|c-d| \leq \delta + 1$, and hence, they can be added to M for parameters $\delta = \delta_0 + 1$ and $s = 2\delta_0 + 5$. In addition, we add to M three new rows: $[\overline{\delta+1}, \overline{2\delta+2}]$, $[\overline{\delta+2}, \overline{2\delta+3}]$ and $[\overline{2\delta+2}, \overline{2\delta+3}]$. The total number of rows added to M is now $s + \delta + 1$. Fig. 1 shows the possible positions of columns $\overline{2\delta+2}$ and $\overline{2\delta+3}$ forced by rows $[\overline{\delta+1}, \overline{2\delta+2}]$ and $[\overline{\delta+2}, \overline{2\delta+3}]$ if we assume that rows $\overline{1}, \ldots, \overline{2\delta+1}$ appear in the correct order. It is easy to see that the row $[\overline{2\delta+2}, \overline{2\delta+3}]$ is (k, δ)-consecutive only if columns $\overline{2\delta+2}$ and $\overline{2\delta+3}$ appear in the correct positions as well. This completes the induction on δ and we have that the claim holds for any $\delta \geq 1$ and $s = 2\delta + 3$, i.e., the base case for the induction on s.

Now, assuming that the claim holds for $s - 1$, where $s - 1 \geq 2\delta + 3$, we show that it holds also for s columns. By the induction hypothesis, there are $s + \delta$ rows which will force columns $\overline{1}, \ldots, \overline{s-1}$ to appear in the correct order. We add one new row: $[\overline{s-\delta-1}, \overline{s}]$. Since $s - \delta - 1 \geq \delta + 3$, there is only one position where column $\overline{s}$ can appear: next to $\overline{s-1}$, i.e., all columns in S appear in the correct order. The number of rows used is exactly $s + \delta + 1$. This completes the induction on s, and the claim follows. □

4. Complexity of the (k, δ)-C1P Problem

In this section we will show that for every $k \geq 2$, $\delta \geq 1$, $(k, \delta) \neq (2, 1)$, the (k, δ)-C1P Problem is NP-complete.

4.1. Complexity of the (k, δ)-C1P Problem for every $k, \delta \geq 2$

For every $k, \delta \geq 2$, we use Theorem 1 in a reduction from 3SAT to the problem of testing for the (k, δ)-C1P to show that this problem is NP-complete.

Theorem 2. *For every $k, \delta \geq 2$, testing for the (k, δ)-C1P is NP-complete.*

Proof. Consider $k, \delta \geq 2$. Let ϕ be a 3CNF formula over the n variables $\{v_1, \ldots, v_n\}$, with m clauses $\{c_1, \ldots, c_m\}$. We construct a matrix M_ϕ with $2n + d + 6m$ columns and $n + 7m + d + \delta + 1$ rows, where $d = \max\{2k - 1, 2\delta + 3\}$, such that M_ϕ has the (k, δ)-C1P if and only if ϕ is satisfiable.

In [11], the authors show that for every $k \geq 2$, given a 3CNF formula ϕ, they can construct a matrix M_ϕ that has the k-C1P if and only if ϕ is satisfiable. Our construction is based on theirs. In our construction, we associate the first $2n$ columns $\langle 1, 2n\rangle$ of M_ϕ with the variables $\{v_1, \ldots, v_n\}$. In particular, we associate variable v_i with the pair of columns $b_i = \{2i-1, 2i\}$, for $i \in \langle 1, n\rangle$. Variable v_i equal to *true* represents the statement about the order of the columns: "$2i - 1$ is before $2i$" (v_i equal to *false* represents statement: "$2i - 1$ is after $2i$"). Since a truth assignment to the formula ϕ represents a statement about a permutation of the columns of M_ϕ, we want to relate M_ϕ to the clauses $\{c_1, \ldots, c_m\}$ of ϕ in such a way that only the permutations of M_ϕ that are (k, δ)-consecutive correspond to truth assignments that satisfy ϕ and vice versa. This

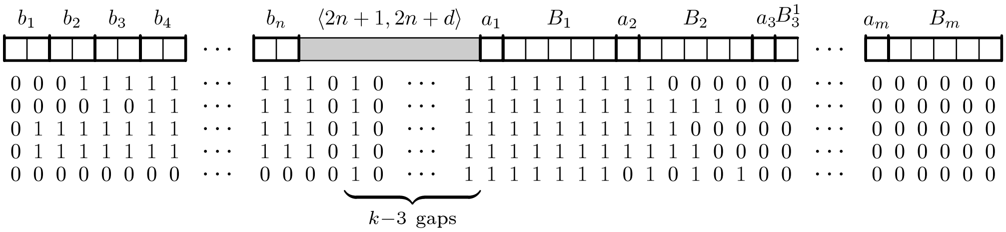

Fig. 2. The structure of M_ϕ and the five rows encoding clause $c_2 = \{v_2 \vee \neg v_3 \vee v_1\}$.

construction involves associating the last $6m$ columns $\langle 2n+d+1, 2n+d+6m\rangle$ with the clauses $\{c_1, \ldots, c_m\}$. In particular, we associate clause c_j with the block of five columns $B_j = \langle 2n+d+6j-4, 2n+d+6j\rangle$, while each block B_j is preceded by a column $a_j = \{2n+d+6j-5\}$. Finally, the set $\langle 2n+1, 2n+d\rangle$ of columns in the middle will be used to ensure that the construction works for parameters k and δ. The details are as follows.

The base of our construction is a subset of the columns of M_ϕ that we force to be together and in fixed order in any (k,δ)-consecutive ordering of M_ϕ, and then we will build off this base a construction similar to that of [11]. In particular, we impose this fixed order on this subset $\langle 2n+1, 2n+d\rangle$ of the columns in the middle of M_ϕ by adding $d+\delta+1$ rows to M_ϕ according to Theorem 1. While these d columns must be together and in fixed order (or the reverse) in any (k,δ)-consecutive ordering, we assume the former without loss of generality. We now build the remaining construction off this block of d columns.

To force the blocks $b_1, \ldots, b_n$ to appear together and in this order, and before the set $\langle 2n+1, 2n+d\rangle$ of d columns in M_ϕ, we add the n rows $[b_i, b_{i+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-3, 2n+2k-1]$ to M_ϕ, for $i \in \langle 1, n\rangle$. Observe that, if block b_n is not immediately to the left of the d columns, then there are more than $k-1$ gaps in the row $[b_n, 2n+1, 2n+3, \ldots, 2n+2k-3, 2n+2k-1]$, while, for each $i \in \langle 1, n-1\rangle$, if block b_i is not immediately to the left of b_{i+1}, then there are more than $k-1$ gaps in the row $[b_i, b_{i+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-3, 2n+2k-1]$.

Next, to force the blocks $a_1, B_1, \ldots, a_m, B_m$ to appear together and in this order, and *after* the set $\langle 2n+1, 2n+d\rangle$ of d columns in M_ϕ, we add the $2m$ rows $[2n+d-(2k-2), 2n+d-(2k-4), \ldots, 2n+d-4, 2n+d-2, 2n+d, a_1, B_1, \ldots, a_{j-1}, B_{j-1}, a_j]$ and $[2n+d-(2k-2), 2n+d-(2k-4), \ldots, 2n+d-4, 2n+d-2, 2n+d, a_1, B_1, \ldots, a_j, B_j]$ to M_ϕ, for $j \in \langle 1, m\rangle$.

Now the blocks of columns in any (k,δ)-consecutive ordering of the matrix M_ϕ are ordered as follows: the blocks $b_1, \ldots, b_n$ associated with the variables of ϕ, followed by the d columns $2n+1, \ldots, 2n+d$, followed by the blocks $a_1, B_1, \ldots, a_m, B_m$, where the blocks $B_1, \ldots, B_m$ are associated with the clauses of ϕ. Since the restrictions placed on variable blocks $\{b_1, \ldots, b_n\}$ and the clause blocks $\{B_1, \ldots, B_m\}$ are the same as in [11], we simply have to add rows, similar to those in [11], to M_ϕ to associate each clause to its three variables to properly simulate 3SAT. The difference from our construction to that of [11], is what values the row takes within this segment $\langle 2n+1, 2n+d\rangle$ of d columns and the m columns $a_1, \ldots, a_m$. We now present the details.

Suppose that clause c_j contains the literal v_α. We add the following (corresponding) row to M_ϕ: $[b_\alpha^2, b_{\alpha+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-7, 2n+2k-5, \langle 2n+2k-3, 2n+d\rangle, a_1, B_1, \ldots, a_j, B_j^1]$. If v_α is *false*, this forces B_j^1 to be the first column of block B_j in any (k,δ)-consecutive ordering of M_ϕ. Any other ordering of the columns of B_j would introduce a k-th gap in this row. If v_α appears negated in c_j, then we add the row $[b_\alpha^1, b_{\alpha+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-7, 2n+2k-5, \langle 2n+2k-3, 2n+d\rangle, a_1, B_1, \ldots, a_j, B_j^1]$ instead. Suppose another literal in c_j is v_β. We add the row $[b_\beta^2, b_{\beta+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-7, 2n+2k-5, \langle 2n+2k-3, 2n+d\rangle, a_1, B_1, \ldots, a_j, B_j^{\langle 1,4\rangle}]$. If v_β is *false*, this forces B_j^5 to be the last column of block B_j. Suppose the third literal of c_j is v_γ. We add the rows $[b_\gamma^2, b_{\gamma+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-7, 2n+2k-5, \langle 2n+2k-3, 2n+d\rangle, a_1, B_1, \ldots, a_j, B_j^{\langle 1,2\rangle}]$ and $[b_\gamma^2, b_{\gamma+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-7, 2n+2k-5, \langle 2n+2k-3, 2n+d\rangle, a_1, B_1, \ldots, a_j, B_j^{\langle 1,3\rangle}]$ to M_ϕ. If v_γ is *false*, this forces B_j^3 to be the middle column of block B_j. Finally, we add the row $[2n+3, 2n+5, \ldots, 2n+2k-7, 2n+2k-5, \langle 2n+2k-3, 2n+d\rangle, a_1, B_1, \ldots, a_{j-1}, B_{j-1}, B_j^1, B_j^3, B_j^5]$ to M_ϕ. This last row is not (k,δ)-consecutive exactly when B_j^1, B_j^3, and B_j^5 are the first, middle and last columns of block B_j, as it contains k gaps then. This fifth row enforces the constraint that not all three literals of c_j can be *false*. Fig. 2 illustrates the structure of matrix M_ϕ, along with these five rows that would be added to M_ϕ for clause $c_2 = \{v_2 \vee \neg v_3 \vee v_1\}$.

It remains to show that if any literal in c_j is *true*, then there is some ordering of the columns of block B_j such that these five rows are (k,δ)-consecutive. If v_α (resp., v_β) is *true*, we can order the columns $B_j^2, B_j^1, B_j^3, B_j^4, B_j^5$ (resp., $B_j^1, B_j^2, B_j^3, B_j^5, B_j^4$). If v_γ is *true*, the columns can be in any order that places B_j^1 (resp., B_j^5) in the first (resp., last) position, while placing B_j^2, B_j^3, B_j^4 in any of the four orderings that avoids placing B_j^3 in the middle (as this fifth row would have k gaps in this case). Note that these orderings work even when the corresponding variable is the only one that is *true*, and that in all of these orderings, no row has a gap of size larger than two. Finally, we remark that if v_γ is the only variable that satisfies clause c_j, for example, then in all of the four (possible) orderings of the columns where these five rows are (k,δ)-consecutive, there is a gap of size two in the fifth row. Hence this construction does not work for $\delta = 1$.

Since, for every $k, \delta \geq 2$, the (k, δ)-C1P Problem is clearly in NP, by the above reduction from 3SAT, it follows that for every $k, \delta \geq 2$, the (k, δ)-C1P Problem is NP-complete. □

4.2. Complexity of the $(k, 1)$-C1P Problem for every $k \geq 3$

We slightly modify the reduction from 3SAT in the proof of Theorem 2 to show that, for every $k \geq 3$, testing for the $(k, 1)$-C1P is NP-complete.

Theorem 3. *For every $k \geq 3$, testing for the $(k, 1)$-C1P is NP-complete.*

Proof. Consider $k \geq 3$. Let ϕ be a 3CNF formula over the n variables $\{v_1, \ldots, v_n\}$, with m clauses $\{c_1, \ldots, c_m\}$. We construct a matrix M_ϕ with $2n+d+4m$ columns and $n+4m+d+2$ rows, where $d = 2k-1$, such that M_ϕ has the $(k, 1)$-C1P if and only if ϕ is satisfiable. We do this as follows.

We again use Theorem 1 to force the columns $\langle 2n+1, 2n+d \rangle$ to appear together and in fixed order in any $(k, 1)$-consecutive ordering of M_ϕ, and build a construction off this block. We again associate columns $\langle 1, 2n \rangle$ with the variables of ϕ, and associate each clause c_j with block B_j. However, B_j now has four columns rather than five, that is $B_j = \langle 2n+d+4j-3, 2n+d+4j \rangle$. Note also that we do not have the blocks a_j in this construction. We again add the appropriate rows to M_ϕ so that the columns of any $(k, 1)$-consecutive ordering of the matrix M_ϕ are ordered $b_1, \ldots, b_n$, followed by $2n+1, \ldots, 2n+d$, followed by $B_1, \ldots, B_m$. The only major difference from Theorem 2 of this reduction is the manner in which we associate the clauses to their variables to properly simulate 3SAT. The details are as follows.

We need to introduce only three more rows to associate the clauses to their variables to properly simulate 3SAT. Suppose that clause c_j contains literals v_α, v_β and v_γ. We add the row $[b^2_\alpha, b_{\alpha+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-9, 2n+2k-7, \langle 2n+2k-5, 2n+d \rangle, B_1, \ldots, B_{j-1}, B_j^{\langle 1,2 \rangle}]$ to M_ϕ. If v_α is *false*, this forces B^1_j and B^2_j to be among the first three columns of block B_j in any $(k, 1)$-consecutive ordering of M_ϕ. Note that any other ordering of the columns of B_j would introduce either a gap of size 2, or a k-th gap in this row. Similarly, we add the rows $[b^2_\beta, b_{\beta+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-9, 2n+2k-7, \langle 2n+2k-5, 2n+d \rangle, B_1, \ldots, B_{j-1}, B^1_j, B^3_j]$ and $[b^2_\gamma, b_{\gamma+1}, \ldots, b_n, 2n+1, 2n+3, \ldots, 2n+2k-9, 2n+2k-7, \langle 2n+2k-5, 2n+d \rangle, B_1, \ldots, B_{j-1}, B^1_j, B^4_j]$ to M_ϕ. If v_β is *false*, this forces B^1_j and B^3_j to be among the first three columns of block B_j, and if v_γ is *false*, this forces B^1_j and B^4_j to be among the first three columns of block B_j. Finally, since $B^1_j, B^2_j, B^3_j, B^4_j$ cannot simultaneously be among the first three columns of block B_j, we have that not all three literals of c_j can be *false* in any $(k, 1)$-consecutive ordering of M_ϕ.

It remains to show that if any literal in c_j is *true*, then there is some ordering of the columns of block B_j such that these four rows are $(k, 1)$-consecutive. If v_α (resp., v_β, and v_γ) is *true*, we can order the columns $B^3_j, B^1_j, B^4_j, B^2_j$ (resp., $B^2_j, B^1_j, B^4_j, B^3_j$, and $B^2_j, B^1_j, B^3_j, B^4_j$). Note that these orderings work even when the corresponding variable is the only one that is *true*.

Since, for every $k \geq 3$, the $(k, 1)$-C1P Problem is clearly in NP, by the above reduction from 3SAT, it follows that for every $k \geq 3$, the $(k, 1)$-C1P Problem is NP-complete. □

In summary, by Theorems 2 and 3, it follows that for every $k \geq 2, \delta \geq 1, (k, \delta) \neq (2, 1)$, the (k, δ)-C1P Problem is NP-complete. Note that this leaves open only the case of the complexity of the (2,1)-C1P Problem, which we conjecture to be polynomial-time solvable.

5. Complexity of the (∞, δ)-C1P Problem

Here we show that for every $\delta \geq 1$, the (∞, δ)-C1P Problem is NP-complete. The first step is to reduce 3SAT(3), the version of the 3SAT Problem where no variable appears more than twice positively and more than once negatively to an auxiliary version of the 3SAT Problem. We then reduce this auxiliary version to the (∞, δ)-C1P Problem for the result.

5.1. The 3SAT(L:2,R:2) Problem

First we reduce from 3SAT(3), the version of the 3SAT Problem with 2-clauses and 3-clauses, and where no variable appears more than twice positively and more than once negatively [18, p. 183, Prop. 9.3], to an auxiliary version of the 3SAT Problem, namely 3SAT(L : 2, R : 2): the version of the 3SAT Problem with 2-clauses and 3-clauses, where each clause is assigned the label L or R (for left or right) such that for each label, no variable appears more than once positively and more than once negatively in the corresponding set of clauses.[3]

Lemma 4. *The 3SAT(L:2,R:2) Problem is NP-complete.*

[3] We remark that the exact formulation of 3SAT(3) in [18] allows also variables with one positive and two negated occurrences, however these can easily be converted to the other type of variables by replacing them with their negations in all clauses. Clearly, this does not affect the complexity of the problem.

Proof. We are given an instance to the 3SAT(3) Problem: a set V of variables and C of 2 and 3-clauses, such that for each $v \in V$, v appears no more than twice in C and $\neg v$ appears no more than once in C. For each $v \in V$ with two positive occurrences, we replace one of the occurrences of v with the new variable v'. We then label all the clauses of this new instance with L. Note that in this set of clauses labeled with L, no variable appears more than once positively and once negatively. Now, for each appearance of v', we add the two new clauses $c_v^1 = v' \vee \neg v$ and $c_v^2 = v \vee \neg v'$, and label them both with R. These two clauses enforce the constraint that $v = v'$ in any satisfying assignment to this new instance of the 3SAT Problem, thus this new instance is satisfiable if and only if the original 3SAT Problem instance is satisfiable. This new instance of the 3SAT Problem has 2- and 3-clauses, and for each of the labels L and R, no variable appears more than once positively and once negatively. Thus we have transformed in polynomial time the instance of the 3SAT(3) Problem to an instance of the 3SAT(L : 2, R : 2) Problem that is satisfiable if and only if the original 3SAT(3) instance is satisfiable. Since the 3SAT(L : 2, R : 2) Problem is clearly in NP, it follows that the 3SAT(L : 2, R : 2) Problem is NP-complete. □

5.2. Complexity of the $(\infty, 1)$-C1P Problem

We now show that that the $(\infty, 1)$-C1P Problem is NP-Complete by giving a reduction from 3SAT(L : 2, R : 2). We will later generalize this reduction to show that for every $\delta \geq 1$, the (∞, δ)-C1P Problem is NP-Complete.

Theorem 5. *Testing for the $(\infty, 1)$-C1P is NP-complete.*

Proof. We are given an instance ϕ of the 3SAT(L : 2, R : 2) Problem: a set V of variables and the sets C^L and C^R of 2- and 3-clauses, such that for each $v \in V$, v and $\neg v$ each appear no more than once in C^S, for $S \in \{L, R\}$. We use ϕ to build a matrix M_ϕ such that ϕ is satisfiable if and only if M_ϕ has the $(\infty, 1)$-C1P.

The idea of the construction is that for each variable $v_i \in V = \{v_1, \ldots, v_n\}$, the matrix M_ϕ will have the block of columns b_i, called the *variable block*, to represent the value of this variable. Matrix M_ϕ will also contain the blocks of columns $b_{0,1}, \ldots, b_{n,n+1}$ of *dummy blocks* that will interleave the variable blocks. We will add some rows to M_ϕ to force the individual columns of each of the variable and dummy blocks to appear together and in fixed order, or the reverse order. The direction of block b_i will represent the value of the variable v_i. We will then add some rows to M_ϕ to force only the order $b_{0,1}, b_1, b_{1,2}, \ldots, b_{n-1,n}, b_n, b_{n,n+1}$ (or the reverse order) of these blocks, while the individual variable blocks may switch direction relative to this order. If variable block b_i is in the same order relative to this ordering of all of the blocks then its corresponding variable v_i has value *true*, otherwise it has value *false*. The matrix M_ϕ will also have an additional $2n$ free columns. With each clause $c \in C = \{C^L \cup C^R\}$ we associate a unique empty free column f_c. This is possible since for every $S \in \{L, R\}$, each variable appears no more than once positively and once negatively in C^S, and each $c \in C^S$ contains at least two variables, and hence $|C^S| \leq 2n/2 = n$. Thus $|C^L| + |C^R| \leq 2n$. We then add some rows to M_ϕ to force these $2n$ free columns to fall (in any order) between the $2n$ pairs of adjacent $b_{i-1,i}, b_i$ and $b_i, b_{i,i+1}$ blocks, for $i \in \langle 1, n\rangle$, such that there is one free column for each hole.

For a clause $c \in C^L$ (resp., C^R) where c contains variables v_α, v_β (and v_γ for a 3-clause), we assign this clause to column f_c of the $2n$ free columns, and we add a row to M_ϕ that forces the column f_c to be to the left (resp., right) of either block b_α, b_β (or b_γ for a 3-clause). However, column f_c can only go to the left (resp., right) of the block of a variable when its corresponding literal is set to the value that satisfies clause c. Note by the construction that each variable can satisfy at most one left and one right clause, which is sufficient because each literal appears at most once in a right (resp., left) clause. These properties will imply that only when, for every $c \in C^L$ (resp., C^R), column f_c can be placed to the left (right) of a b_i, for $i \in \langle 1, n\rangle$, for a v_i that is set to a value that satisfies c, i.e., ϕ is satisfied, is there a $(\infty, 1)$-consecutive ordering of M_ϕ, and vice versa. We now give the full details of the construction in what follows.

For each variable $v_i \in V = \{v_1, \ldots, v_n\}$, we add the set of columns $b_i = \{b_i^1, \ldots, b_i^5\}$ to M_ϕ. In addition, for every $i \in \langle 1, n\rangle$, we add the set of columns $b_{i-1,i} = \{b_{i-1,i}^1, \ldots, b_{i-1,i}^5\}$ to M_ϕ. For each set of columns b_i for $i \in \langle 1, n\rangle$, and $b_{i-1,i}$ for $i \in \langle 1, n+1\rangle$, we add to M_ϕ the rows according to Theorem 1 to force the columns of each set to appear together and in fixed order (or the reverse) in any $(\infty, 1)$-consecutive ordering of M_ϕ, i.e., in any $(\infty, 1)$-consecutive ordering of M_ϕ, set b_i will appear either as the sequence $b_i^1, \ldots, b_i^5$ or $b_i^5, \ldots, b_i^1$ of consecutive columns, and similarly for the columns in sets $b_{i-1,i}$. We will refer to the b_i as *variable blocks* and the $b_{i-1,i}$ as *dummy blocks*. Note that Theorem 1 requires that a set of columns must have size $2\delta + 3$ before such an ordering can be enforced on it, this is why each block is of size five. In addition, we add $2n$ free columns to M_ϕ.

Now, for each pair of blocks $b_{i-1,i}, b_i$ and $b_i, b_{i,i+1}$ for $i \in \langle 1, n\rangle$, we add rows $[b_{i-1,i} \setminus \{b_{i-1,i}^1\} \cup b_i]$ and $[b_i \cup b_{i,i+1} \setminus \{b_{i,i+1}^5\}]$ to force these pairs to be together with at most one free column in between them. This enforces that the blocks appear in the order $b_{0,1}, b_1, b_{1,2}, \ldots, b_{n-1,n}, b_n, b_{n,n+1}$ (or the reverse) in any $(\infty, 1)$-consecutive ordering of M_ϕ. The first (resp., last) column of the dummy blocks is omitted to fix their direction (relative to the order of the blocks) under the assumption that there is a free column between each pair of neighboring blocks, which we will now enforce with the following row. We add to M_ϕ the row $[B \cup F]$, where $B = \{b_{0,1}^{\langle 2,4\rangle} \cup b_1^{\langle 2,4\rangle} \cup \cdots \cup b_n^{\langle 2,4\rangle} \cup b_{n,n+1}^{\langle 2,4\rangle}\}$, and F is the set of $2n$ free columns. It now follows that between each $b_{i-1,i}, b_i$ and $b_i, b_{i,i+1}$ pair for $i \in \langle 1, n\rangle$, there must lie at least one column from F, in any $(\infty, 1)$-consecutive ordering of M_ϕ. Since we have exactly $2n$ pairs, between each pair there must be exactly one. Fig. 3 depicts all $(\infty, 1)$-consecutive orderings of the current matrix M_ϕ. Note that the columns in each variable block can be oriented either in the same direction as the order of all of the blocks, or in the reverse direction. If variable block b_i is oriented in the same

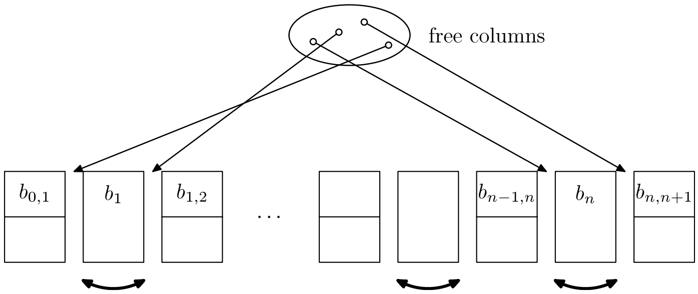

Fig. 3. The structure of matrix M_ϕ.

direction as the order of all of the blocks, this corresponds to the setting of the variable v_i to *true*, while the reverse direction corresponds to v_i being *false*. Now it remains to add rows to M_ϕ to force the free column associated with each clause to fall next to only the blocks of variables that are set to a value that satisfies the clause.

Let $c \in C^L$ (resp., C^R) contain the variables x_α, x_β (and x_γ for a 3-clause), and let $f_c \in F$ be the free column associated with clause c. We add the row $[B \cup F \setminus \{f_c\} \cup S_c]$ to M_ϕ, where S_c is defined as follows. If $c \in C^L$, then for each $j \in \{\alpha, \beta\}$ ($j \in \{\alpha, \beta, \gamma\}$ for a 3-clause), if v_j appears positively (resp., negatively) in c, set S_c contains the columns $\{b^5_{j-1,j}, b^1_j\}$ (resp., $\{b^5_{j-1,j}, b^5_j\}$). Otherwise, if $c \in C^R$, then for each j, if v_j appears positively (resp., negatively) in c, set S_c contains the columns $\{b^5_j, b^1_{j,j+1}\}$ (resp., $\{b^1_j, b^1_{j,j+1}\}$). Adding these extra ones around the variable blocks b_j for each j forces f_c to fall only to the immediate left (resp., right) of these b_j in any $(\infty, 1)$-consecutive ordering of M_ϕ. Furthermore, f_c can only fall to the immediate left (resp., right) of a b_j if it is oriented in a direction such that corresponding variable v_j is set to a value that sets its literal to *true*, i.e., if v_j satisfies c. Hence, the satisfying assignments of any individual clause c correspond to the $(\infty, 1)$-consecutive orderings of the submatrix of M_ϕ consisting of the row added for clause c, and all of the rows previously added to M_ϕ for the blocks b_i for $i \in \langle 1, n\rangle$, and $b_{i-1,i}$ for $i \in \langle 1, n+1\rangle$. Clearly, the $(\infty, 1)$-consecutive orderings of two rows added for clauses c and c' are independent of each other (unless c and c' have variables in common). We now show the correspondence between satisfying assignments of ϕ and $(\infty, 1)$-consecutive orderings of M_ϕ.

After adding the row for all clauses $c \in C^L \cup C^R$, the set of remaining $(\infty, 1)$-consecutive orderings of M_ϕ (if there exist any) correspond to the cases where for every clause $c \in C^L$ (resp., C^R), its corresponding column f_c is placed to the immediate left (resp., right) of a block of a variable that is set to a value (*true* or *false*) that satisfies c, that is, to satisfying assignments of ϕ. Conversely, if ϕ has a satisfying assignment, then we can assign each $c \in C^L$ (resp., C^R) to a unique $v \in V$ that satisfies c, in the sense that either v or $\neg v$ satisfies c, i.e., each $v \in V$ will satisfy at most one clause from C^L and at most one clause from C^R. We can make this claim because v and $\neg v$ each appear no more than once in C^L (resp., C^R), and at most one of v and $\neg v$ satisfies a given clause c. Thus we can assign each column f_c of M_ϕ to a unique slot to the immediate left (resp., right) of block b_i for $i \in \langle 1, n\rangle$, for the corresponding v_i that satisfies the clause c. Thus M_ϕ has an $(\infty, 1)$-consecutive ordering. Hence, ϕ is satisfiable if and only if M_ϕ has the $(\infty, 1)$-C1P.

In summary, given a 3SAT(L : 2, R : 2) formula ϕ with n variables and $m \leq 2n$ clauses, we have constructed a matrix M_ϕ with $12n + 5$ columns and $16n + m + 8$ rows such that M_ϕ has the $(\infty, 1)$-C1P if and only if ϕ is satisfiable. Given that the $(\infty, 1)$-C1P Problem is clearly in NP, and Lemma 4, it follows that the $(\infty, 1)$-C1P Problem is NP-complete. □

5.3. *Complexity of the (∞, δ)-C1P Problem*

We now generalize the construction given in Section 5.2 to show that for every $\delta \geq 1$, the (∞, δ)-C1P Problem is NP-complete by reduction from 3SAT(L : 2, R : 2).

Theorem 6. *For every $\delta \geq 1$, testing for the (∞, δ)-C1P is NP-complete.*

Proof. Consider $\delta \geq 1$. Here, given an instance ϕ of 3SAT(L : 2, R : 2), we build a matrix M_ϕ such that ϕ is satisfiable if and only if M_ϕ has the (∞, δ)-C1P. The idea of the construction is the same as that of the proof of Theorem 5: it will again have the blocks b_i for $i \in \langle 1, n\rangle$, and $b_{i-1,i}$ for $i \in \langle 1, n+1\rangle$ as well as $2n$ free columns for the clauses, only the blocks will need more columns, and we will need to add more rows to M_ϕ in order for it to behave in the same way for arbitrary δ.

For each block b_i for $i \in \langle 1, n\rangle$, and $b_{i-1,i}$ for $i \in \langle 1, n+1\rangle$ we again add to M_ϕ the rows according to Theorem 1 to force each individual block to be in fixed order (or the reverse) in any (∞, δ)-consecutive ordering of M_ϕ. Thus, each block will contain $2\delta + 3$ columns. In order to force each pair of blocks $b_{i-1,i}$, b_i and b_i, $b_{i,i+1}$ for $i \in \langle 1, n\rangle$, to be together, with at most one free column in between them, thus enforcing a total order on the blocks, we add the rows $[b^{\langle \delta+1, \delta+4\rangle}_{i-1,i} \cup b_i]$ and $[b_i \cup b^{\langle \delta, \delta+3\rangle}_{i,i+1}]$. Note here, that the first (resp., last) δ columns of the dummy blocks are omitted to fix their direction (relative to the order of the blocks) under the assumption that there is a free column between each pair of neighboring blocks, which we enforce by adding to M_ϕ the row $[B \cup F]$, where $B = b^{\langle \delta+1, \delta+3\rangle}_{0,1} \cup b^{\langle \delta+1, \delta+3\rangle}_1 \cup \cdots \cup b^{\langle \delta+1, \delta+3\rangle}_n \cup b^{\langle \delta+1, \delta+3\rangle}_{n,n+1}$, and F is a set of $2n$ free columns. Now M_ϕ again has the desired structure, as depicted in Fig. 3. Now it remains to add rows to M_ϕ for the clauses.

Let $c \in C^L$ (resp., C^R) contain the variables x_α, x_β (and x_γ for a 3-clause), and let $f_c \in F$ be the free column associated with clause c. We add the row $[B \cup F \setminus \{f_c\} \cup S_c]$ to M_ϕ, where S_c is defined as follows. If $c \in C^L$, then for each $j \in \{\alpha, \beta\}$ ($j \in \{\alpha, \beta, \gamma\}$ for a 3-clause), if v_j appears positively (resp., negatively) in c, set S_c contains the columns $\{b^{2\delta+3}_{j-1,j}, b^1_j\}$ (resp., $\{b^{2\delta+3}_{j-1,j}, b^{2\delta+3}_j\}$). Otherwise, if $c \in C^R$, then for each j, if v_j appears positively (resp., negatively) in c, set S_c contains the columns $\{b^{2\delta+3}_j, b^1_{j,j+1}\}$ (resp., $\{b^1_j, b^1_{j,j+1}\}$). Now this matrix M_ϕ will have the same behavior as in the proof of Theorem 5, hence ϕ is satisfiable if and only if M_ϕ has the (∞, δ)-C1P.

In summary, for every $\delta \geq 1$, given a 3SAT(L : 2, R : 2) formula ϕ with n variables and $m \leq 2n$ clauses, we have constructed a matrix M_ϕ with $(4\delta + 8)n + 2\delta + 3$ columns and $(6\delta + 10)n + m + 3\delta + 4$ rows such that M_ϕ has the (∞, δ)-C1P if and only if ϕ is satisfiable. Given that for every $\delta \geq 1$, the (∞, δ)-C1P Problem is clearly in NP, and Lemma 4, it follows that for every $\delta \geq 1$, the (∞, δ)-C1P Problem is NP-complete. □

6. An algorithm for matrices with low maximum degree

A binary matrix M has maximum degree d if every row contains at most d entries 1. We show now that, when d and δ are constant (which implies that k is also constant, since $k \leq d$), then the (k, δ)-C1P Problem is tractable. We rely on a connection to graph bandwidth.

A graph $G = (V, E)$ is said to have bandwidth at most b if there exists a total order on its vertices $V = \{v_1, \ldots, v_n\}$ such that every edge $\{v_i, v_j\}$ satisfies $|i - j| \leq b$. Let M be an $m \times n$ binary matrix and $G_M = (V_M, E_M)$ be the undirected graph defined as follows: $V_M = \{1, \ldots, n\}$ (each vertex of G_M represents a column of M), and there is an edge $\{i, j\}$ in E_M if and only if there is a row of M with entries 1 in columns i and j.

The following property then follows immediately from this definition: If M has maximum degree d and M has the (k, δ)-C1P, then G_M has bandwidth at most $d + (k - 1)\delta - 1$.

In [19], Saxe describes an algorithm that decides if a graph has bandwidth at most b with complexity $O(n^{b+1})$, in time and space. We now describe how it can be modified to test for the (k, δ)-C1P. This algorithm uses the property that, given a prefix of a total order on the vertices of a graph, if one wants to test that the bandwidth remains at most b after the addition of some new vertex (from the suffix), only the last b elements of the prefix are useful; the *active region* of this prefix is then composed of its last b vertices, and it defines unambiguously the content of its prefix. The principle of the algorithm is to consider, in a breadth-first search, only the active regions, each of them defining an equivalence class of prefixes, and given a current active region, to extend it by a vertex if it does not violate the bandwidth condition. In our problem, this algorithm needs only to be augmented by testing, each time an active region is extended, if this extension does not violate the gap conditions in any row, which adds an $O(mn)$ time and space cost factor to the algorithm.

Theorem 7. *Let M be an $m \times n$ binary matrix such that every row has at most d entries* 1. *Deciding if M has the (k, δ)-C1P can be done in time and space $O(mn^{d+(k-1)\delta+1})$.*

7. Conclusion

In this work, we have shown that for every $k \geq 2$, $\delta \geq 1$, $(k, \delta) \neq (2, 1)$, testing for the (k, δ)-C1P is NP-complete, and also that for every $\delta \geq 1$, testing for the (∞, δ)-C1P is NP-complete. Testing for the k-C1P, or equivalently the (k, ∞)-C1P, for $k \geq 2$ has been proved NP-complete in [11]. Hence, we have closed in the present work most questions that were open in [5], and the only remaining open problem related to the case of considering both the number of gaps and their size is the problem of testing for the $(2, 1)$-C1P. Since the two NP-completeness constructions presented here force either a gap of size two, or at least two gaps of size one in any legal configuration of M, if testing for the $(2, 1)$-C1P is NP-complete, it would certainly require a different type of construction. We conjecture that this problem is polynomial-time solvable.

A natural extension of this work would be to consider an additional parameter d, namely the maximum number of entries 1 that can be present in any row of M, or the maximum *degree* of M, defining the (d, k, δ)-C1P problem. This problem is motivated by the fact that in the framework described in [6], it is possible to constrain matrices used to reconstruct ancestral genomes to have small maximum degree. Note that with matrices of maximum degree 2, the number of gaps can be at most 1, and the $(2, \delta)$-C1P problem is then equivalent to the problem of deciding if the graph whose incidence matrix is M has bandwidth at most $(\delta + 1)$. For $\delta = 1$, the graph bandwidth problem can be solved in linear time [3], while in [19] a dynamic programming algorithm with time and space complexity exponential in δ was described. We showed in Section 6 that whenever d and δ are constant, the (d, k, δ)-C1P problem is tractable. We have implemented the algorithm described in Section 6 and used it to analyze yeast genome data. Preliminary results suggest that it can handle data with small values of d and δ, although its space complexity is an issue and raises non-trivial algorithm engineering challenges [7]. The design of efficient algorithms, both in time and space, for deciding the gapped consecutive-ones property for matrices of small maximum degree is a natural research avenue, with immediate applications in genomics. From a theoretical point of view, the complexity of testing for this property when δ is unbounded, namely the (d, k, ∞)-C1P is still open (work in progress).

From a purely combinatorial point of view, there has been a renewed interest in the characterization of non-C1P matrices in terms of forbidden submatrices introduced by Tucker [20]. It has recently been shown that this characterization could be used in the design of algorithms related to the C1P [8,4]. The question then is the following: is there a nice characterization of non-(k, δ)-C1P matrices in terms of forbidden matrices?

Finally it is also natural to ask if there exists a structure that can represent all orderings that satisfy some gaps conditions related to the consecutive-ones property. Such a structure exists for the ungapped C1P: for a matrix that has the C1P, its PQ-tree represents all its valid consecutive-ones orderings, and it can be computed in linear time [16]. This has even been extended to matrices that do not have the C1P through the notion of the PQR-tree [17,16]. Although the existence of such a structure with nice algorithmic properties is ruled out by the hardness of solving the Gapped C1P Problems, it remains open to find classes of matrices such that testing for the Gapped C1P is tractable, and in such a case, to represent all possible orderings in a compact structure. Here again, this question is motivated both by theoretical considerations (for example representing all possible layouts of a graph of bandwidth 2), but also by computational genomics problems [6].

Acknowledgment

We would like to thank an anonymous reviewer for some very useful comments and suggestions in improving this paper. The first author's research was supported by an NSERC Discovery Grant. The second author's research was supported by NSERC PGS-D3.

References

[1] F. Alizadeh, R. Karp, D. Weisser, G. Zweig, Physical mapping of chromosomes using unique probes, Journal of Computational Biology 2 (2) (1995) 159–184.
[2] K.S. Booth, G.S. Leuker, Testing for the consecutive ones property, interval graphs, and graph planarity, Journal of Computer and Systems Sciences 13 (2) (1976) 335–379.
[3] A. Caprara, F. Malucelli, D. Petrolani, On bandwidth-2 graphs, Discrete Applied Mathematics 34 (2002) 477–495.
[4] C. Chauve, U.-W. Haus, T. Stephen, V.P. You, Minimal conflicting sets for the consecutive-ones property in ancestral genome reconstruction, in: Proceedings of International Workshop RECOMB Comparative Genomics, in: Lecture Notes in Bioinformatics, vol. 5817, 2009, pp. 48–58.
[5] C. Chauve, J. Maňuch, M. Patterson, On the gapped consecutive-ones property, in: Proceedings of the European Conference on Combinatorics, Graphs Theory and Applications (EuroComb), in: Electronic Notes in Discrete Mathematics, vol. 34, 2009, pp. 121–125.
[6] C. Chauve, E. Tannier, A methodological framework for the reconstruction of contiguous regions of ancestral genomes and its application to mammalian genomes, PLoS Computational Biology 4 (2008) e1000234.
[7] R. Coghlan, A k-consecutive-ones property check algorithm, USRA Report, Simon Fraser University, 2010.
[8] M. Dom, Recognition, generation, and application of binary matrices with the consecutive-ones property, Ph.D. Thesis, Institut für Informatik, Friedrich–Schiller-Universität, Jena, 2008.
[9] M. Dom, J. Guo, R. Niedermeier, Approximability and parameterized complexity of the consecutive ones submatrix problems, in: Proceedings of the 4th International Conference on the Theory and Applications of Models of Computation, TAMC, in: LNCS, vol. 4484, Springer-Verlag, 2007, pp. 680–691.
[10] D. Fulkerson, O. Gross, Incidence matrices and interval graphs, Pacific Journal of Mathematics 15 (1965) 835–855.
[11] P.W. Goldberg, M.C. Golumbic, H. Kaplan, R. Shamir, Four strikes against physical mapping of DNA, Journal of Computational Biology 2 (1) (1995) 139–152.
[12] M. Habib, R.M. McConnell, C. Paul, L. Viennot, Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing, Theoretical Computer Science 234 (1–2) (2000) 59–84.
[13] S. Haddadi, A note on the NP-hardness of the consecutive block minimization problem, International Transactions in Operational Research 9 (6) (2002) 775–777.
[14] M.T. Hajiaghayi, Y. Ganjali, A note on the consecutive ones submatrix problem, Information Processing Letters 83 (3) (2002) 163–166.
[15] W.-L. Hsu, A simple test for the consecutive ones property, Journal of Algorithms 43 (1) (2002) 1–16.
[16] R.M. McConnell, A certifying algorithm for the consecutive-ones property, in: Proceedings of the Symposium on Discrete Algorithms, SODA, ACM, 2004, pp. 761–770.
[17] J. Meidanis, O. Porto, G.P. Telles, On the consecutive ones property, Discrete Applied Mathematics 88 (1–3) (1998) 325–354.
[18] C.H. Papadimitriou, Computational Complexity, Addison Wesley, 1994.
[19] J.B. Saxe, Dynamic-programming algorithms for recognizing small-bandwidth graphs in polynomial time, SIAM Journal on Algebraic and Discrete Methods 1 (4) (1980) 363–369.
[20] A.C. Tucker, A structure theorem for the consecutive 1's property, Journal of Combinatorial Theory, Series B 12 (1972) 153–162.