Linear-Time Recognition of Probe Interval Graphs

Ross M. McConnell[1] and Yahav Nussbaum[2]

[1] Computer Science Department, Colorado State University, Fort Collins, CO 80528, USA, rmm@cs.colostate.edu
[2] The Blavatnik School of Computer Science, Tel Aviv University, 69978 Tel Aviv, Israel, yahav.nussbaum@cs.tau.ac.il

Abstract. The interval graph for a set of intervals on a line consists of one vertex for each interval, and an edge for each intersecting pair of intervals. A probe interval graph is a variant that is motivated by an application to genomics, where the intervals are partitioned into two sets: probes and non-probes. The graph has an edge between two vertices if they intersect and at least one of them is a probe. We give a linear-time algorithm for determining whether a given graph and partition of vertices into probes and non-probes is a probe interval graph. If it is, we give a layout of intervals that proves that it is. In contrast to previous algorithms for the problem, our algorithm can determine whether the layout is uniquely constrained. This is important for the biological application, where one seeks the true layout of the intervals in a genome. As part of the algorithm we solve the consecutive-ones probe matrix problem.

1 Introduction

An *interval graph* is the intersection graph of a set of intervals on a line. That is, it has one vertex for each interval and two vertices are adjacent if the corresponding intervals intersect. The set of intervals constitutes an *interval model* of the graph. Interval graphs play an important role in many problems, for example scheduling problems [5, 7, 9]. The problem of recognizing whether a graph is an interval graph played a key role in the 1950's in proving the linear topology of DNA [1]; the intervals were fragments of genetic material, and it was shown empirically that the intersections of these fragments give rise to an interval graph.

This gave rise to interest in algorithms for determining whether a graph is an interval graph [6]. Booth and Lueker gave the first linear-time algorithm for recognizing interval graphs and constructing interval models for the graphs in the 1970's [2]. A *consecutive-ones ordering* of columns of a 0-1 matrix is one such that, for every row, the 1's in the row are consecutive. Booth and Lueker's approach was to reduce the problem to that of finding a consecutive-ones ordering of a 0-1 matrix, and to give a linear time bound for finding such an ordering.

Another related application for interval graphs is *physical mapping*, which can be used for DNA sequencing. In this process, biologists create *clones*, which are copies of fragments of DNA. Physical mapping is the problem of reconstructing the original sequence of the DNA from the clones. The order of the clones is lost, but for some clones, called *probes*, the intersection data between them and other clones can be collected. If all clones are probes, then we can construct an interval graph from the clones, and an interval model for this graph gives the original sequence. However, because of practical considerations, not all clones can be used as probes.

A *probe interval graphs* [15, 18] (sometimes called interval probe graph) is a graph in which the vertex set is partitioned into *probes* and *non-probes*. It is a generalization of intersection graph of an interval model, such that the graph has an edge between two vertices if their intervals intersect *and at least one of them is a probe*. In other words, missing from the graph is information about which pairs of non-probe intervals intersect.

There has been recent work on topological and combinatorial properties of these graphs; see [9, 14] for a survey. The problem of recognizing whether a graph is a probe interval graph, and finding a corresponding arrangement of intervals if it is, was first shown to be polynomial by Johnson and Spinrad [10], who gave an $O(V^2)$ algorithm. Using a different approach, McConnell and Spinrad gave an $O(V + E \log V)$ algorithm [13]. The latter algorithm was a critical step in the first linear-time algorithm for recognizing *circular-arc graphs*, which are the intersection graphs of arcs on a circle [12]. (The algorithm makes use of a probe interval subgraph of size $O(V + E/\log V)$). Motivated by the biological application, where the partition into probes and non-probes is known in advance, both algorithms get as an input a graph whose vertex set is partitioned into probes and non-probes. Chang et al. [4] consider the problem of recognizing this graph class when the partition into probes and non-probes is not given.

In this paper, we give the first linear-time algorithm for recognizing whether a graph is a probe interval graph when the partition into probes and non-probes is given, and for finding a corresponding set of intervals if it is. In view of the complexity of the previous work, it is surprising that we are able to reduce the problem to that of finding consecutive-ones orderings of two easily-constructed consecutive-ones matrices, which can then be solved by Booth and Lueker's algorithm.

In the physical mapping problem, part of the clones are used as probes. The number of clones which are probes is enough only if we were able to construct the original DNA sequence accurately. Previous algorithms for finding probe interval arrangements have the defect that they cannot determine whether the graph uniquely constrains the arrangement, and therefore cannot be said to solve the physical mapping problem. Uehara [17] has addressed the issue and gave a polynomial-time algorithm that determines whether a give probe interval graph has a unique model. Our algorithm solves this problem as a by-product of the recognition problem, and it is the first linear-time algorithm that solves it.

A 0-1 matrix is a *consecutive-ones matrix* if it has a consecutive-ones ordering. Booth and Lueker's algorithm recognizes consecutive-ones matrices, and finds a consecutive-ones ordering for them. The consecutive-ones sandwich problem is an extension of this problem where the matrix has 0, 1 or $*$. A $*$ is a "don't care"; it can stand for either a 0 or a 1. This problem is NP-Complete [8]. If we require that the $*$'s form a submatrix then we get the consecutive-ones probe matrix problem (see also [3]). We solve this problem in linear time, for any 0, 1, $*$ probe matrix.

2 Preliminaries

Except for some additional definitions, we use standard terminology and notation from [5].

We will assume the standard adjacency-list representation of a graph. This imposes a numbering from 1 to n on the vertices, which we will call their *vertex numbers*.

A graph $G = (V, E)$ is a *probe graph* if the vertex set is partitioned into P, the set of probes, and N the set of non-probes. In this case, every edge of E is adjacent to at least one probe. We denote this by $G = (P, N, E)$.

If X is a nonempty subset of V, let $G[X]$ denote the subgraph of G induced by X. If G is a probe graph, then by $G[X]$ we mean a graph that also contains the classification in G of members of X as probes or non-probes.

An *interval model* of an *interval graph* is a set of intervals, one for each vertex, such that two vertices are adjacent if and only if their intervals intersect. Similarly, an *interval model* of a probe interval graph is a set of intervals, one for each vertex, such that two vertices are adjacent if and only if their intervals intersect *and at least one of the vertices is a probe*. If R is an interval model of a (probe) interval graph G and X is a nonempty subset of V, let $R[X]$ denote the set of intervals representing members of X. Note that if R is an interval model of G, then $R[X]$ is an interval model of $G[X]$.

Let $N(v)$ denote the *open neighborhood* of v, that is, the set of neighbors of v in G, and let $N[v]$ denote its *closed neighborhood*, that is, $\{v\} \cup N(v)$.

We define the *cliques* of a graph to be its *maximal* complete subgraphs. That is, X is a clique if $G[X]$ is complete and there is no $Y \subseteq V$ such that $X \subset Y$ and $G[Y]$ is complete. The *clique matrix* of a graph is a 0-1 matrix that has one column for each clique, one row for each vertex, and a 1 in row i, column j if and only if vertex i is a member of clique j.

In an interval model R of an interval graph G, each clique of G corresponds to the set of vertices whose intervals cover a unique *clique segment* in R. A clique segment occurs where a right endpoint is immediately to the right of a left endpoint. Ordering the columns of the clique matrix of G in the left-to-right order of the clique segments of a model R gives a consecutive-ones ordering of the clique matrix, since this makes the clique segments covered by each interval (hence the cliques that contain each vertex) consecutive. Conversely, given a consecutive-ones ordering of the clique matrix of a graph, the intervals occupied by the 1's in the rows constitute an interval model of the graph, since two vertices are adjacent if and only if they are members of a common clique. Interval graphs are therefore exactly the set of graphs whose clique matrices have consecutive-ones orderings [6].

An interval model consists of alternating blocks of consecutive left endpoints and of consecutive right endpoints. The order of endpoints within a block does not change the realized graph. Therefore, we represent an interval model by giving an ordered list of blocks, listing for each block the vertices that have endpoints in the block. This gives a combinatorial definition of an interval model, independent of geometry. In fact, a consecutive-ones ordering of the clique matrix of an interval graph is such a model, where the set of left endpoints in a column and the set of right endpoints in the column are each interpreted to be a block, where the block of left endpoints in the column implicitly precedes the block of right endpoints.

A *chordal graph* is a is a graph with no induced cycle of size greater than three. Every interval graph is a chordal graph. A chordal graph has $O(V)$ cliques. It is possible to find a sparse representation of the clique matrix of a chordal graph in $O(V)$ time [16]. Booth and Lueker's algorithm [2] for recognizing interval graphs uses this to find the clique matrix or else determine that the graph is not chordal, hence not an interval graph. If it is chordal, it reduces the problem to that of either finding a consecutive-ones ordering of this clique matrix, giving an interval model, or determining that no such ordering exists, in which case the graph is not an interval graph.

The algorithm of [2] actually gives a compact representation of *all* consecutive-ones orderings of a matrix, called a *PQ-tree.* The leaves of the PQ-tree are the columns of the matrix. Any set of orderings of the children of the internal nodes of a tree gives a unique ordering of the leaves, and the PQ-tree gives all consecutive-ones orderings by constraining the orderings of these children as follows. Some of the internal nodes are labeled *P nodes.* For such a node there is no constraint on the order of its children. Others are labeled *Q nodes.* For such a node an ordering $(x_1, x_2, \ldots x_k)$ of its children is given; in this case, the only permissible orderings of its children are $(x_1, x_2, \ldots, x_k)$ and $(x_k, x_{k-1}, \ldots, x_1)$. If T is a PQ-tree, let $\Pi(T)$ denote the set of all possible orderings of its leaves, given these constraints. Booth and Lueker [2] show that if T is the unique PQ-tree representing $\Pi(T)$, to get this property they define some restrictions on the degrees of internal nodes in a PQ-tree. Their algorithm either finds the PQ-tree for consecutive-ones orderings of columns of a matrix, or determines that the matrix is not a consecutive-ones matrix. Given a sparse representation of a 0-1 matrix, this takes in $O(i + j + k)$ time, where i is the number of rows, j is the number of columns, and k is the number of 1's in the matrix.

Let $\Pi = \Pi(T)$. We may consider each $\pi \in \Pi$ to be a bijective function that maps elements of C, the set of columns, to elements of $\{1, 2, \ldots, |C|\}$, where for all $c \in C$, $\pi(c)$ tells the position of c in a consecutive-ones ordering represented by π. If X is a nonempty subset of C, let π_X be the bijective function that maps elements of X to $\{1, 2, \ldots, |X|\}$, giving the relative order of elements of X in π. Formally, for $c \in X$, $\pi_X(c) = 1 + |\{d | d \in X \text{ and } \pi(d) < \pi(c)\}|$. Let $\Pi[X]$ denote $\{\pi_X | \pi \in \Pi\}$, namely, the relative orderings of elements of X given by orderings in Π. It is not hard to show that $\Pi[X]$ is the set of orderings of a PQ-tree with leaf set X; let us call this tree the *restriction* $T[X]$ *of* T *to* X. In [11], an $O(j)$ algorithm is given for finding $T[X]$, given X and T, where $j = |C|$ is the number of leaves of T.

If T_1 and T_2 are two PQ-trees whose leaf sets are both C, it is not hard to show that $\Pi(T_1) \cap \Pi(T_2)$ is a set of permutations that can also be represented by a PQ-tree. Let us call this tree the *intersection* $T_1 \cap T_2$ *of* T_1 *and* T_2. In [11], an $O(j)$ algorithm is given for finding $T_1 \cap T_2$, given T_1 and T_2, where $j = |C|$ is the number of leaves of each tree.

A *probe matrix* is a generalization of 0-1 matrix, which has the values $0, 1, *$, such that the $*$'s form a submatrix. An $*$ is a "don't care" value which might be interpreted either as 0 or as 1. The *consecutive-ones probe matrix* problem is a generalization of the consecutive-ones problem. In this problem we look for an ordering of the columns of the matrix such that there is an interpretation of the values of the $*$'s such that the 1's in every row are consecutive.

We want to represent the probe matrix in space proportional to the size of the matrix and the number of 1's in it, in other words, we do not want to represent the $*$'s explicitly. In order to do that, we split a probe matrix M into two submatrices. Let M_R be the submatrix of M whose rows are the rows that do not have $*$'s, and whose columns are all columns of M. Let M_C be the submatrix of M whose columns are the columns that do not have $*$'s, and whose rows are all rows of M. We represent M using sparse representations of M_R and M_C.

In the rest of the paper we present a linear-time recognition algorithm for probe interval graphs. In Sect. 3 we construct a probe matrix for the input graph that generalizes the clique matrix used for interval graphs. In Sect. 4 we show how to construct an interval model from a consecutive-ones ordering of the probe matrix. In Sect. 5 we present a linear-time algorithm for the consecutive-ones probe matrix problem. Last, in Sect. 6 we show that using our algorithm we can determine if the interval model that was found is unique.

3 Extension of the Clique Matrix

In this section we show how to build a probe matrix M that has the consecutive-ones property if G is a probe interval graph. The basis of this matrix is M_P, the clique matrix of $G[P]$. In addition, for every non-probe we define either new columns or a new row. A new column has a value of 0 or 1 for every row of M_P. Similarly, a new row has a value of 0 or 1 for every column of M_P. The submatrix of M induced by the new rows and the new columns consists exclusively of $*$'s, and so M is a probe matrix. We view each row of M as a *constraint*, since it limits the possible consecutive-ones orderings of M.

The graph $G[P]$ has no non-probes. Therefore, if G is a probe interval graph, then $G[P]$ is an interval graph and M_P has the consecutive-ones property. Let $x \in N$ and denote the set $P \cup \{x\}$ by $P + x$. If G is a probe interval graph then $G[P + x]$ is an interval graph, because a pair of non-probes is required to give rise to an interval intersection that is not an edge. This last observation is the basis of our probe matrix construction.

Therefore, we begin by finding a consecutive-ones ordering of M_P. Using [2] we can either find such an ordering or determine that G is not a probe interval graph, in $O(V + E)$ time.

Let $\mathcal{C}$ denote the set of cliques of $G[P]$. For each probe p, let $\mathcal{Q}(p)$ denote the set of cliques of $\mathcal{C}$ that contain p. In an interval model of G, the interval for p must cover the clique segments of members of $\mathcal{Q}(p)$. Because p is not contained in any other clique, the clique segments of $\mathcal{Q}(p)$ must be consecutive in the left-to-right ordering of clique segments. In M we represent these constraints for all P by the rows of M_P. We call these constraints *probe - clique* constraints.

Similarly, for each non-probe x, let $\mathcal{Q}(x)$ denote the set of cliques of $\mathcal{C}$ that are subsets of $N(x)$, and Q_x denote $\bigcup \mathcal{Q}(x)$. We call a vertex v *simplicial* if $N(x)$ induces a complete subgraph. For the construction of M, we split the set of non-probes into three sets: N_1 is the set of non-probes x such that $|\mathcal{Q}(x)| \geq 1$; N_2 is the set of non-simplicial vertices with $\mathcal{Q}(x) = \emptyset$; and N_3 is the set of simplicial vertices. Note that, according to this definition, a simplicial non-probe x such that $|\mathcal{Q}(x)| = 1$ is contained both in N_1 and in N_3. It does not matter into which of the two sets we put x. Moreover, we show below that x does not impose any constraints on orderings of M that are not already imposed by other vertices.

The vertices of N_1 add two kinds of rows (constraints) to M, the vertices of N_2 add one kind of row to M, and the vertices of N_3 add a single column to M. We show the details below, but first we show how to split N into these three sets. In order to split N, we should find for every $x \in N$ the set $\mathcal{Q}(x)$ and determine whether x is simplicial or not.

Let the *left endpoint* of a row of a consecutive-ones ordering of M_P be the column of the leftmost 1 in the row, and the *right endpoint* be the rightmost. Let $x \in N$, and assume that $G[P]$ is an interval graph. In the consecutive-ones ordering of M_P we find for every $p \in N(x)$ the left endpoint and the right endpoint of the row of p. We keep the column numbers of these two endpoints, together with their side (left or right) in a list L_x. We sort L_x for all x in linear time using a single radix sort, with x as the primary sort key, column number as the secondary sort key, and left *versus* right endpoint as the tertiary key so that if a left endpoint and right endpoint have the same primary and secondary key, the left endpoint goes to the left of the right.

We sweep through L_x from left to right, keeping a running count of the number of neighbors of x in the current column. We start a counter at zero. Each time we encounter a left endpoint in L_x we increment the counter, and each time we encounter a right endpoint we decrement it. Each time we encounter a right endpoint e that follows a left endpoint f, we compare the counter with the size of the clique C represented by the column of the endpoint e, and include C in $\mathcal{Q}(x)$ if the counter is equal to the size of C.

For correctness, let $C \subseteq N(x)$. The column for each clique C is the left endpoint for some member of C and the right endpoint for some member of C. It is necessary for x to have a neighbor whose left endpoint is in C's column and a neighbor whose right endpoint is in C's column. A check of the counter against the size of C will therefore occur when C is reached. The procedure identifies all cliques in $\mathcal{Q}(x)$. It does not falsely identify any clique C' as a member of $\mathcal{Q}(x)$, since if it doesn't belong, the counter will be equal to $|C' \cap N(x)| < |C'|$ when the sweep passes C', and the test for C', if it is performed, will fail.

Every time that we change the value of the counter, we compare it to $|N(x)|$, if these values are equal at some column C, then $N(x) \subseteq C$ and therefore x is simplicial. If x is simplicial then there must be at least one clique of $G[P]$ that contains $N(x)$. The column of at least one of the cliques which contain $N(x)$ must be the left endpoint of a row of a member of $N(x)$, and therefore we find all simplicial vertices this way.

The procedure for x takes time proportional to $|N[x]|$ because it spends $O(|N[x]|)$ time retrieving and sorting L_x, and $O(1)$ time at each entry of L_x, for a total of $O(|N[x]|)$ time. Summing over all x, we have an $O(N+E)$ bound for splitting N into N_1, N_2 and N_3. This bound is possible because the procedure often avoids checks at columns corresponding to cliques that are not subsets of $N(x)$.

We conclude with the following lemma:

Lemma 1. *In linear time we can either split N into N_1, N_2 and N_3 and find $\mathcal{Q}(x)$ for every $x \in N$, or else determine that G is not a probe interval graph.*

3.1 Non-Probe - Clique Constraints

Assume that G is indeed a probe interval graph and consider the interval of $x \in N_1$ in a interval model R. The interval of x must cover the clique segments that correspond to members of $\mathcal{Q}(x)$, since each member of $\mathcal{Q}(x)$ is a subset of $N(x)$ and $C \in \mathcal{Q}(x)$ has a member such that the clique segment of C is the leftmost clique segment in its interval and a member such that the clique segment of C is the rightmost clique segment in its interval. For any clique C' whose clique segment is intersected by x's interval, $C' \subseteq N(x)$, hence $C' \in \mathcal{Q}(x)$. We conclude that the clique segments of $\mathcal{Q}(x)$ must be consecutive in the ordering of clique segments of $G[P]$ given by any interval model of G.

The number of cliques containing a vertex v in an interval graph is bounded by $|N[v]|$, since a neighbor of v ends at the clique segment for each clique that contains v. Since $G[P+x]$ is an interval graph, for any $x \in N_1$, the number of cliques in $\mathcal{Q}(x)$ is bounded by $|N[x]|$.

We therefore add to M a row for each $x \in N_1$ that has a 1 in the column for C if $C \in \mathcal{Q}(x)$ and a 0 otherwise. Using a sparse representation of the matrix, this adds $O(|N[x]|)$ to the size of the matrix. We call these new rows *non-probe - clique constraints.*

3.2 Non-Probe - Probe Binding Constraints

The non-probe - clique constraints defined for members of N_1 are not enough. These constraints allow the interval of $x \in N_1$ to cover the clique segments of

$\mathcal{Q}(x)$ and thus intersect the intervals of Q_x, but there might be some vertices in $N(x) \setminus Q_x$. For these we add more constraints to M.

Let $x \in N_1$ and let $p \in N(x) \setminus Q_x$. Since x and p are adjacent, we know that their intervals must intersect in any interval model of G, and therefore $\mathcal{Q}(x) \cup \mathcal{Q}(p)$ must be consecutive. In any interval model R of G, these intersections must be realized between pairs of consecutive clique segments of $R[P]$. Let us call this additional constraint a *non-probe - probe binding constraint* imposed by x and p, since one of them is a non-probe and the other is a probe. Adding such a constraint for every such x and p will make M too large. We show that a set of new rows with a linear number of 1's is enough to enforce the non-probe - probe binding constraints.

We know that $\mathcal{Q}(x) \cap \mathcal{Q}(p) = \emptyset$, because if $C \in \mathcal{Q}(x) \cap \mathcal{Q}(p)$ then $p \in C$ and thus $p \in Q_x$. Therefore, in any interval model of G, the interval of p covers exactly one endpoint of the interval of x. Moreover, in the order of the clique segments in any model, either the rightmost member of $\mathcal{Q}(p)$ must be consecutive with the leftmost member of $\mathcal{Q}(x)$ or vice versa.

The set of probes that x is bound to is $N(x) \setminus Q(x)$. Denote this set by Y. In a interval model of G, we can divide Y into the set Y_1 that covers the left endpoint of x and the set Y_2 that covers the right endpoint of x. Note that although we defined Y using a specific model of G, the same Y_1 and Y_2 arise in every model, up to interchange between the two, since each of the two subsets induces a complete subgraph and two vertices from different subsets are nonadjacent.

Recall that the vertices are numbered arbitrarily from 1 through n. For two vertices v and u, let $v \prec u$ denote that either $\mathcal{Q}(v) \subset \mathcal{Q}(u)$ or that $\mathcal{Q}(v) = \mathcal{Q}(u)$ and v has a smaller vertex number than u does.

Since the members of Y_1 all end at the clique segment to the left of x's left endpoint and they all occupy consecutive cliques, it follows that for any two $y, y' \in Y_1$, either $\mathcal{Q}(y) \subseteq \mathcal{Q}(y')$ or $\mathcal{Q}(y') \subseteq \mathcal{Q}(y)$. It follows that Y_1 induces a linear order in the $\prec$ relation, so it has a unique a minimal member y_1 in this relation. Similarly, Y_2 has a unique minimal member y_2 in the $\prec$ relation. For every $y \in Y$ either $\mathcal{Q}(y_1) \subseteq \mathcal{Q}(y)$ or $\mathcal{Q}(y_2) \subseteq \mathcal{Q}(y)$, but not both.

By similar reasoning, each for each probe p, the $\prec$ relation on non-probes that p is bound to has at most two nonadjacent minimal members x_1 and x_2. Let us say that x and p are a *representative bound pair* if p is a minimal bound neighbor of x and x is a minimal bound neighbor of p in the $\prec$ relation.

Consider the current status of the matrix M. The matrix includes the probe - clique constraints and the non-probe - clique constraints. In $O(V + E)$ time we can either find a consecutive-ones ordering of M or determine that G is not a probe interval graph, since we cannot satisfy all the constraints. Using this ordering of M, we can determine in $O(1)$ time for two vertices $v, u \in P \cup N_1$ whether $\mathcal{Q}(v) \subseteq \mathcal{Q}(u)$ by examining the position of leftmost and rightmost 1's in the rows of u and v. So, the relation $\prec$ for two vertices can be determined in $O(1)$ time as well. We get that we can find the minimal bound neighbors of every vertex, and thus all the representative bound pairs, in $O(V + E)$ time.

We add to M a row for any representative pair $\{x, p\}$ that has a 1 in the column for C if $C \in \mathcal{Q}(x) \cup \mathcal{Q}(p)$ and a 0 otherwise. Using a sparse representation of the matrix, this adds $O(|N[x]| + |N[p]|)$ to the size of the matrix. Since every vertex adds at most two new rows to M, the size of M remains linear in the size of G.

We claim that it is enough to add probe - non-probe binding constraints only for representative bound pairs. In other words, a consecutive-ones ordering of M satisfies the binding constraint not just for representative bound pairs, but for all such bound pairs of vertices. Recall that because of the probe - clique constraints and the non-probe - clique constraints, M already has a row with the characteristic vector of $\mathcal{Q}(v)$ for each probe or non-probe vertex v.

Suppose that x and p are bound. Let x' be the minimal vertex in the neighborhood containment relation that is bound to p such that $x' \prec x$. Next, let p' be the minimal vertex that is bound to x' such that $p' \prec p$. We continue this process until we get $x'' \prec x$ and $p'' \prec p$ such that x and p are a representative bound pair. Because of the $\prec$ relation, $\mathcal{Q}(x'') \subseteq \mathcal{Q}(x)$ and $\mathcal{Q}(p'') \subseteq \mathcal{Q}(p)$, from the previous constraints $\mathcal{Q}(x)$ is consecutive and $\mathcal{Q}(p)$ is consecutive, and from the non-probe - probe binding constraints $\mathcal{Q}(x'') \cup \mathcal{Q}(p'')$ is consecutive. It follows that $\mathcal{Q}(x) \cup \mathcal{Q}(y)$ is consecutive, as required.

3.3 Probe - Probe Binding Constraints

Consider $x \in N_2$. In this case, $\mathcal{Q}(x)$ is empty and $N(x)$ is not a complete subgraph. If G is a probe interval graph, then in an interval model R of G, the interval of x lies between two consecutive clique segments of $R[P]$. Let C_1 and C_2 be the cliques of $G[P]$ corresponding to these clique segments, where C_1's segment lies to the left of C_2's. Let $Y_1 = N(x) \setminus C_2$ and let $Y_2 = N(x) \setminus C_1$. The sets Y_1 and Y_2 satisfies $Y_1 \subseteq C_1$, $Y_2 \subseteq C_2$ and $Y_1 \cap Y_2 = \emptyset$. Also, since x is not simplicial, neither Y_1 nor Y_2 is empty. Note that although we used a specific model to define Y_1 and Y_2 for x, these sets are unique for every $x \in N_2$, up to interchange between the two.

Let $y \in Y_1$ and $y' \in Y_2$. Since x is adjacent to both y and y', and does not cover any clique segment, we know that $\mathcal{Q}(y) \cup \mathcal{Q}(y')$ must be consecutive in any interval model of G. We call this additional constraint a *probe - probe binding constraint* imposed by y and y'. Adding such a constraint for every such y and y' for every $x \in N_2$ will make M too large. But as with the non-probe - probe binding constraints, we can use the same relation $\prec$ and add to M such a constraint only for a pair p and p' such that both are minimal bound neighbors of each other.

We add to M a row for each representative pair p, p' for probe - probe binding constraints that has a 1 in the column for C if $C \in \mathcal{Q}(p) \cup \mathcal{Q}(p')$ and a 0 otherwise. Using a sparse representation of the matrix, this adds $O(|N[p]| + |N[p']|)$ to the size of the matrix. Since every vertex adds at most two new rows to M, the size of M remains linear in the size of G.

To find these in $O(V + E)$ time, we find for each $x \in N_2$, the sets Y_1 and Y_2 of probes that are bound because of x. All elements of Y_1 are bound to elements of

Y_2, but we cannot consider all of these pairs and observe the linear time bound. The sets Y_1 and Y_2 are linearly ordered by the relation $\prec$. Therefore it is enough to bind only the minimum members of the two sets to each other. This gives $O(|N_2|)$ candidate pairs for bindings. However, a probe may be a member of several candidate bindings. We proceed on these as in the case of probe - non-probe bindings to find representative pairs, where each is a minimal element that is bound to the other.

3.4 Additional Segments

Last, we consider N_3. For this set of probes we do not define further constraints, but refine the probe - clique constraints. This is done by adding columns to M. As mentioned earlier, the new columns have 0 or 1 in rows of M_P and $*$ in rows that we added for N_1 and N_2.

Let $x \in N_3$, and assume that G is a probe interval graph. Let R be an interval model of G. The set $N[x]$ is a clique in G. Therefore there is a clique segment in R that is covered by the intervals of $N[x]$. Note that intervals of additional non-probes might cover this clique segment as well.

Let $\mathcal{C}' = \{N[x] \mid x \in N_3\}$. The members of $\mathcal{C}'$ are the cliques of G that are not in $\mathcal{C}$. For each vertex v, let $\mathcal{Q}'(v)$ denote the set of members of $\mathcal{C}'$ that contain v (for $x \in N_1 \cup N_2$ we get $\mathcal{Q}'(x) = \emptyset$, and for $x \in N_3$, $\mathcal{Q}'(x) = \{N[x]\}$). In an interval model of G, the interval for a probe p must cover the clique segments that correspond to members of $\mathcal{Q}(p) \cup \mathcal{Q}'(p)$. Because p is not contained in any other clique of $\mathcal{C} \cup \mathcal{C}'$, the clique segments of $\mathcal{Q}(p) \cup \mathcal{Q}'(p)$ must be consecutive in the left-to-right ordering of clique segments. This gives us a refinement of the probe - clique constraints.

To represent the cliques of $\mathcal{C}'$, we add a new column for every $x \in N_3$, that has a 1 in the row of a probe p if $p \in N[x]$, and 0 otherwise. Using a sparse representation of the matrix, this adds $O(|N[x]|)$ to the size of the matrix.

This concludes the construction of M. If G is a probe interval graph, then there must be a consecutive-ones ordering of the columns of M that obeys all constraints. We summarize the section in the following lemma:

Lemma 2. *It takes $O(V + E)$ time to construct the probe matrix M or else decide that G is not a probe interval graph. Moreover, if G is a probe interval graph then M is a consecutive-ones probe matrix.*

We use the algorithm of Sect. 5 to find a consecutive-ones ordering of M. If such an ordering does not exist then G is not a probe interval graph. Note that in some cases M has a consecutive-ones ordering even if G is not a probe interval graph.

4 Constructing an Interval Model

In this section we use the consecutive-ones ordering of M that we found in the previous section to find an interval model of G, if one exists.

The construction is similar to the construction of [2] of an interval model from the clique matrix of an interval graph. Each interval must cover the clique segments it belongs to. In addition, realizing bindings requires some differential stretching of endpoints inside the zone between two consecutive clique segments.

Recall that we represent an interval model combinatorially by a list of alternating blocks of left and right endpoints. We begin by defining two sets of endpoints for every column C of M: C_ℓ and C_r. We show below how we populate these sets. We order the sets according to the consecutive-ones ordering of the columns of M, such that C_ℓ is to the left of C_r.

Let $v \in V \setminus N_2$. In this case, $\mathcal{Q}(v) \cup \mathcal{Q}'(v)$ is not empty. If $v \in P \cup N_1$, then $\mathcal{Q}(v) \cup \mathcal{Q}'(v)$ has a row in M. Let C be the leftmost column with 1 in this row and D be the rightmost column with 1 in this row. If $v \in N_3$, we let C and D both be the column of the clique $N[v]$. We put the left endpoint of v in C_ℓ and the right endpoint of v in D_r. Because M has a consecutive-ones ordering, this takes linear time. Let us denote the resulting interval model by R_1.

In R_1, for every column C, the segment on the line between C_ℓ and C_r is the clique segment of C. If the intervals of v and u intersect in R_1, and at least one of the two vertices is a probe, then v and u are adjacent, because $(\mathcal{Q}(u) \cup \mathcal{Q}'(u)) \cap (\mathcal{Q}(v) \cup \mathcal{Q}'(v)) \neq \emptyset$.

However, there still might be some edges in E that are not realized by R_1. These edges are between $x \in N_1 \cup N_2$ and $p \in N(x) \setminus Q_x$. In order to realize these adjacencies we place the endpoints of vertices of N_2 and stretch the intervals of N_1, N_2 and $N(x) \setminus Q_x$ for $x \in N_1 \cup N_2$ between the clique segments of $\mathcal{C} \cup \mathcal{C}'$.

Let $x \in N_2$ and let Y_1 and Y_2 be as defined in Sect. 3.3, that is the two sets for which x defines probe - probe constraints, such that the intervals of Y_1 are to the left of the intervals of Y_2. Let C be the rightmost column in which all rows of members of Y_1 have a 1. This is the same columns for all Y_1 because of the probe - probe constraints. Let D be the leftmost column in which all rows of members of Y_2 have a 1. The column D is the column immediately to right of C because of the probe - probe constraints. We place the left endpoint of x in D_ℓ and the right endpoint of x in C_r. Denote the construction so far by R_2. Note that R_2 is not an interval model, since we place the left endpoint of x to the right of its right endpoint. We will resolve this problem when we stretch the intervals.

The last step of the construction is to stretch intervals of vertices of N_1, N_2 and $N(x) \setminus Q_x$ for $x \in N_1 \cup N_2$. Consider two vertices v and u that are adjacent in G, but whose adjacency is not realized in R_2. Assume that v is to the left of u. (If one of them is in N_2 then it does not have a real interval, but it is still clear which one is to the left.) Because of the non-probe - probe constraints and the probe - probe constraint, we know that the set C_r, which contains the right endpoint of v, is immediately to the left of D_ℓ, which contains the left endpoint of u. We must stretch the endpoints of intervals that have unrealized intersections, between the clique segments.

For every C_r and the set to its right, D_ℓ, we split the two sets and order them as follows. We split C_r into subsets $A_0, A_1, \ldots, A_{|D_\ell|}$ and A' such that an endpoint $f \in C_r$ is in A_i if it is an endpoint of an interval of a probe p with $|N(p) \cap D_\ell| = i$, and in A' if it is an endpoint of an interval of a non-probe. Similarly we split D_ℓ into subsets $B_0, B_1, \ldots, B_{|C_r|}$ and B'. Note that some of the the subsets might be empty. We replace C_r with the A_i's, where A_0 is the leftmost. We replace D_ℓ with B_i's where B_0 is the rightmost. For every endpoint $f \in B'$, we place f in a set to the right of A_j where j is the largest index such that the vertex of f is non-adjacent to all vertices of $A_0, A_1, \ldots, A_j$ and adjacent to all vertices of $A_{j+1}, A_{j+2}, \ldots, A_{|D_\ell|}$. Similarly we place every endpoint $f \in A'$ in a set on the left of the appropriate B_j. Note that the set between $A_{|D_\ell|}$ and $B_{|C_r|}$ contains both right and left endpoints. To avoid a set that contains both left and right endpoint, we split this set F into a set F_ℓ of left endpoints and a set F_r of right endpoints. Let us denote the resulting construction by R. See Fig. 1.

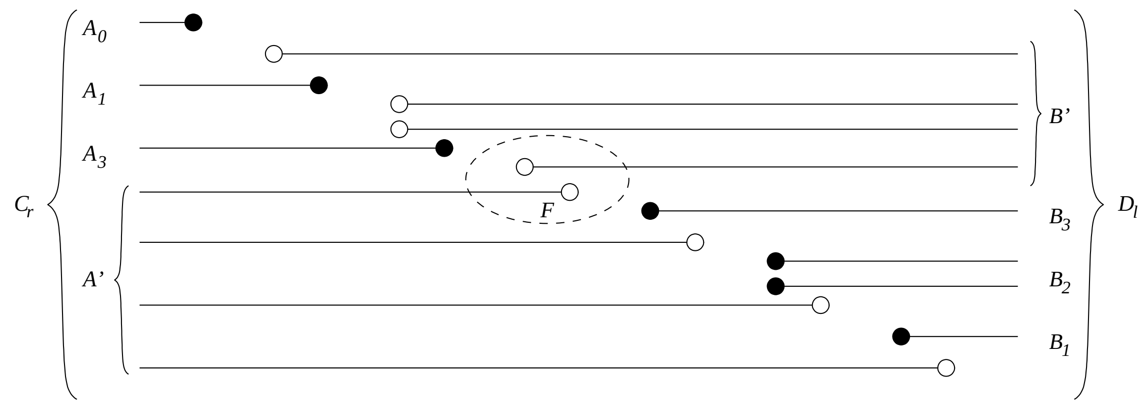

Fig. 1. Reordering the endpoints of C_r and D_ℓ. The endpoints of C_r are right endpoints and the endpoints of D_ℓ are left endpoints. Endpoints of non-probes are empty, endpoints of probes are full. The endpoints of C_r are split into A_0, A_1, A_3 for probes and A' for non-probes. The endpoints of D_ℓ are split into B_1, B_2, B_3 for probes and B' for non-probes. The set F contains the endpoints which are to the right of A_3 and to the left of B_3. We spit it into F_ℓ and F_r.

We show that if G is a probe interval graph, then we can place every members of A' and B', and therefore we can construct R. Assume that we cannot place $f \in B'$ on the right side of any A_i. Let x be the vertex of f. The fact that we cannot place f means that there are A_i and A_j with $i \leq j$, for which there is a probe p with an endpoint in A_i and a probe p' with an endpoint in A_j, such that x is adjacent to p, but not to p'. Both p and p' are members of the clique which C represents, so they are adjacent. Because p' is adjacent to at least as many non-probes of A' as p, we know that there is x' with an endpoint in A' that is adjacent to p' but not to p. In any interval model of G, the intervals of x and x' cannot intersect, since otherwise x, x', p', p would be a chordless cycle.

There is a vertex q with an endpoint in D_ℓ that is adjacent to x. If there is such a q that is adjacent also to x', then x, x', p, p' and q create a chordless cycle in any interval model of G. Otherwise, there is a vertex q' with endpoint D_ℓ which is adjacent to x' and not to x. In this case, q and q' must be adjacent since they are both members of the clique represented by column D. So the vertices x, x', p, p', q and q' create a chordless cycle in any interval model of G. Therefore, if we cannot place $f \in B'$ then G is not a probe interval graph. The proof for the case where we cannot place $f \in A'$ is symmetric.

Since every interval has two endpoints, and the splitting of C_r and D_ℓ takes time proportional to the number of edges between vertices which have endpoints in these sets, in $O(V + E)$ time we can either construct R or decide that R cannot be constructed. If R cannot be constructed, G is not a probe interval graph.

If we managed to construct R, then it is an interval model of G. First note that the endpoints of every vertex of N_2 are now ordered properly, that is, the left endpoint is to the left of the right endpoint. To show that R realizes G, we consider the following cases for a probe p and a vertex v, and show that their intervals in R intersect if and only if they are adjacent. If $v \in P$ then the claim is true because $R[P]$ is a model of $G[P]$. If $v \in N_1$ and $p \in Q_v$ or if $v \in N_3$ then the claim is true because the intervals intersect if and only if $(\mathcal{Q}(v) \cup \mathcal{Q}'(v)) \cap (\mathcal{Q}(p) \cup \mathcal{Q}'(p)) \neq \emptyset$. Otherwise, the claim follows by the way we stretch intervals into the region between the clique segments.

We conclude with the main theorem:

Theorem 3. *Let G be a probe graph. In $O(V + E)$ time we can construct a interval model for G, or decide that G is not a probe interval graph.*

5 Consecutive-Ones Probe Matrices

In this section we present a linear-time algorithm for the consecutive-ones probe matrix problem. Let M be a probe matrix with i rows, j columns and k 1's. We determine if M is a consecutive-ones probe matrix in $O(i + j + k)$ time. We do so by finding the PQ-tree of two 0-1 submatrices of M using [2], and combining the trees using tools of [11]. If M is a consecutive-ones probe matrix then we find a consecutive-ones ordering of it. With a modification of [11] we can find a PQ-tree that represents all consecutive-ones orderings of M. We do not present it here, because a single consecutive-ones ordering is enough, and we want to keep the description simple.

Let M_R be the submatrix of M whose rows are the rows that do not have $*$'s, and whose columns are all columns of M. Let M_C be the submatrix of M whose columns are the columns that do not have $*$'s, and whose rows are all rows of M. Let X be the set of columns of M_C.

If π is a consecutive-ones ordering of M, then π is also a consecutive-ones ordering of M_R and π_X is a consecutive-ones ordering of M_C. On the contrary, if π is a consecutive-ones ordering of M_R, and π_X is a consecutive-ones ordering of M_C, then π is a consecutive-ones ordering of M. Therefore, our goal is to find such a π.

Let T_R be the PQ-tree of M_R and let T_C be the PQ-tree of M_C. Let $T' = T_R[X] \cap T_C$. Each permutation $\sigma \in \Pi(T')$ is a permutation in both $\Pi(T_R[X])$ and in $\Pi(T_C)$. This means that $\sigma = \pi_X$ where $\pi \in \Pi(T_R)$. Assume that such a permutation $\sigma \in \Pi(T')$ exists.

We can order T_R so that the relative order of leaves that are members of X is σ, because $\sigma \in \Pi(T_R[X])$, so it is a restriction of some $\pi \in \Pi(T_R)$. It can be accomplished by the following at each internal node of T_R. At a P node, order the set of children that contain leaf descendants in X according to the order of those members in σ. At a Q node, if two children contain leaf descendants in X, from the two allowed linear orders of children, choose the one that is consistent with σ. The result is π, a consecutive-ones ordering of columns of M_R, such that $\sigma = \pi_X$ is a consecutive-ones ordering of M_C. Therefore, π is a consecutive-ones ordering of M.

On the other hand, if $\Pi(T') = \emptyset$, then there is no ordering of X that imposes a consecutive-ones ordering both for $T_R[X]$ and for T_C, and therefore M is not a probe interval matrix.

Using Booth and Lueker [2] we can find T_R and T_C in $O(i+j+k)$ time, and using [11] we can find $T_R[X]$ and from it T' in the same time bound. Choosing σ and π takes $O(j)$ time.

Theorem 4. *Let M be a probe matrix. In time linear in the size of the matrix and the number of 1's in it we can either find a consecutive-ones ordering of M, or else decide that M is not a consecutive-ones probe matrix.*

6 Determining Whether a Model is Uniquely Constrained

Recall that we represent an interval model combinatorially by a list of alternating blocks of left and right endpoints, as the order of endpoints within a block is inconsequential. Let R and R' be two interval models. We say that R and R' are *equivalent* if they are identical, or if we can get R' from R by reversing the order of its blocks and exchanging the blocks between the two endpoints of each interval. If every model of G is equivalent to R, then R is a *unique* model of G.

In this section we show that our algorithm for constructing a interval model for a probe interval graph can determine whether this model is unique for the graph. This is important for the physical mapping problem that we mentioned in the introduction, since the answer to this question tells the biologist if we found the correct sequence, or if we require more probes to uniquely constrain the model. Let T' be as in Sect. 5, for the matrix M, found in Sect. 3.

An interval model of an interval graph is unique only if the consecutive-ones ordering of its clique matrix is unique up to reversal. This is because a consecutive-ones ordering of the clique matrix defines the block of each endpoint, and also the order of the blocks. Similarly, a probe interval graph G has a unique model only if the probe matrix M has a unique consecutive-ones ordering up to reversal.

A consecutive-ones ordering of a matrix is unique up to reversal if and only if the PQ-tree has a single internal node that is a Q node. Let us call such a tree a *prime PQ-tree*. As mentioned before, using a modification of [11] we can find the PQ-tree of M, and in this way determine if M has a unique consecutive-ones ordering. Even if we use the algorithm of Sect. 5, we can still determine that as follows. If T' is not a prime PQ-tree, then there are nonequivalent ways to order the the columns of M, and G has more than one unique model. The same happens also when T' is a prime PQ-tree, but there is more one way to produce π from σ. This happens if and only if the order of children is not uniquely constrained at each node. Specifically, if there is a P node with a child that does not contain an element of X or if there is a Q node for which less than two children contain an element of X. This can be checked in time linear in the size of T', which is linear in the size of G. In other cases, there is a unique way to order the columns of M.

Even if M does have a unique consecutive-ones arrangement, there is still a possibility of different models, due to the fact that there may still be more than one way to add edges between the non-probes to produce an interval graph. The matrix M defines a unique order for the cliques of G, and therefore a unique order of the clique segments in any interval model of G. So, for every two vertices of V such that at least one is a probe, there is a unique order defined among their endpoints. However, if there are two non-probes x and x' such that the set that contains the left endpoint of x in the model R is next to the set that contains the right endpoint of x', then we can change the order between the two endpoints. So in this case as well, R is not a unique model of G. This last case can also be detected in time linear in the size of G.

References

1. S. Benzer. On the topology of the genetic fine structure. *Proc. Nat. Acad. Sci. U.S.A.*, 45:1607–1620, 1959.
2. S. Booth and S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *J. Comput. Syst. Sci.*, 13:335–379, 1976.
3. D. B. Chandler, J. Guo, T. Kloks, and R. Niedermeier. Probe matrix problems: Totally balanced matrices. In *Third International Conference on Algorithmic Aspects in Information and Management (AAIM)*, Lecture Notes in Computer Science 4508:368–377, 2007.
4. G. J. Chang, T. Kloks, J. Liu, and S.-L. Peng. The PIGSs full monty - a floor show of minimal separators. In *22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, Lecture Notes in Computer Science 3403:521–532, 2005.
5. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* McGraw Hill, Boston, 2001.
6. D. R. Fulkerson and O. Gross. Incidence matrices and interval graphs. *Pacific J. Math.*, 15:835–855, 1965.
7. M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, New York, 1980.

8. M. C. Golumbic. Matrix sandwich problems. *Linear Algebra and Applications*, 277:239–251, 1998.
9. M. C. Golumbic and A. N. Trenk. *Tolerance graphs.* Cambridge studies in advanced mathematics 89, New York, 2004.
10. J. L. Johnson and J. P. Spinrad. A polynomial time recognition algorithm for probe interval graphs. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 12:477–486, 2001.
11. R. M. McConnell and F. de Montgolfier. Algebraic operations on PQ trees and modular decomposition trees. In *31st International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, Lecture Notes in Computer Science 3787:421–432, 2005.
12. R. M. McConnell. Linear-time recognition of circular-arc graphs. *Algorithmica*, 37:93–147, 2003.
13. R. M. McConnell and J. P. Spinrad. Construction of probe interval models. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 866–875, 2002.
14. T. A. McKee and F. R. McMorris. *Topics in Intersection Graph Theory.* Society for Industrial and Applied Mathematics, Philadelphia, 1999.
15. F. R. McMorris, C. Wang, and P. Zhang. On probe interval graphs. *Discrete Applied Mathematics*, 88:315–324, 1998.
16. D. Rose, R. E. Tarjan, and G. S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM J. Comput.*, 5:266–283, 1976.
17. R. Uehara. Canonical data structure for interval probe graphs. In *15th International Symposium on Algorithms and Computation (ISAAC)*, Lecture Notes in Computer Science 3341:859–870, 2004.
18. P. Zhang. United states patent 5667970: Method of mapping DNA fragments. July 3, 2000.