An algorithm based on graph theory for the assembly of contigs in physical mapping of DNA

Peisen Zhang¹, Eric A. Schon¹,³, Stuart G. Fischer², Efthia Cayanis², Janie Weiss⁴, Susan Kistler⁴, and Philip E. Bourne²,⁵*

Abstract

An algorithm is described for mapping DNA contigs based on an interval graph (IG) representation. In general terms, the input to the algorithm is a set of binary overlapping relations among finite intervals spread along a real line, from which the algorithm generates sets of ordered overlapping fragments spanning that line. The implications of a more general case of the IG, called a probe interval graph (PIG), in which only a subset of cosmids are used as probes, are also discussed. In the specific case of cosmids graph (PIG), in which only a subset of cosmids are used as hybridization information using the cosmids as probes, and relations among finite intervals spread along a real line, on an interval graph (IG) representation. In general terms, an alignment of 67 cosmids spanning a YAC took 0.28 seconds of CPU time on a Convex 220 computer.

Introduction

A major aspect of genome mapping is the physical mapping of large fragments of DNA into overlapping segments, or contigs. Currently, most contigs are based on the assembly of DNA segments contained within two types of cloning vectors, yeast artificial chromosomes (YACs), which may contain 300–1500 kb of DNA, and cosmids, which typically contain 30–50 kb of foreign DNA. Both types of libraries can be used to isolate specific DNA sequences representing a region of a chromosome. By isolating suitable cosmid clones containing DNA associated with a specific YAC insert, it is possible to assemble a cosmid contig spanning that YAC. Similarly, overlapping YAC's may be ordered into more extensive cosmid contigs.

One often-used procedure (Olson et al., 1986; Coulson et al., 1987; Gemmill et al., 1987; Carrano et al., 1989a) to generate contigs relies on matching ‘DNA fingerprints’ among overlapping clones. Typically, a set of clones is digested with one or more restriction enzymes, and those clones sharing a subset of similarly-sized fragments are deemed to overlap. Using automated procedures, large numbers of cosmids have been fingerprinted to generate cosmid contigs of chromosome 19 (Carrano et al., 1989b; Branscomb et al., 1990). However, the assignment of overlaps is based on experimentally-determined restriction fragment lengths which are subject to both error and statistical variation due to problems in sizing fragments and in assigning identity to similarly-sized fragments (Branscomb et al., 1990). In practice, as much as 50% overlap of clones is required for the contig to be considered reliable.

Human chromosome 13 is being mapped using chromosome 13-specific YAC and cosmid libraries (Fischer et al., 1994). A key part of this strategy is the generation of cosmid contigs that span each YAC, and the eventual generation of overlapping YAC contigs to span the chromosome. Rather than using fingerprinting, the approach to contig generation is based on a complementary method—hybridization (Fischer et al., 1994)—which does not require traditional mapping data: individual members of the cosmid or YAC libraries are scored in a binary fashion as to whether or not they hybridize to each other.

A major problem in assembling such contigs is the complexity of the information. This problem is further complicated by redundancies in the YAC and cosmid libraries, which usually contain 3–5 times as many clones as are required to cover all the DNA just once. Thus, a YAC library with 3x coverage and an average insert size of 400 kb for a 100-Mb human chromosome will contain 750 clones, rather than the 250 clones necessary to span this distance. Similarly, a library of cosmids with 5x coverage and an average insert size of 40 kb spanning the average YAC will contain 50 rather than 10 members.

We describe here a binary approach based on graph theory which generates sets of cosmid contigs along a

Departments of ¹Genetics and Development, ²Biochemistry and Molecular Biophysics, and ³Neurology, the ⁴Cancer Center, and the ⁵Howard Hughes Medical Institute, Columbia University College of Physicians and Surgeons, 630 West 168th Street, New York, NY 10032

*To whom correspondence should be addressed.
YAC, including that set (if it exists) which spans the entire YAC. The same approach can be extended to the generation of YAC contigs. This approach is compared to other algorithms which also use hybridization data to generate contigs.

System and methods

Overview

In the strategy to isolate cosmid contigs spanning YACs, a chromosome 13-specific YAC that has been identified by in-situ hybridization is used as a hybridization probe of a chromosome 13-specific cosmid library to identify and isolate a set of 40–50 cosmids (Fischer et al., 1994). The cosmids in the set are spotted in an array on a filter for subsequent colony hybridization. The filter is probed with the insert of each cosmid, which has been labeled at its edges using T3 and T7 polymerase priming sites at the vector/insert boundaries. Thus, the filter is hybridized with each of the approximately 50 probes to identify sets of cosmids which overlap each other in the cosmid hybridization matrix. The problem now becomes how to order the overlapping cosmids in such a way as to maximize the span of each resulting cosmid contig.

The cosmid matrix hybridization method produces binary overlap information, by answering Yes or No to the following question: For each pair of fragments considered, do these two fragments overlap? Earlier approaches to ordering sequence fragments from binary overlap data attempted to compute the comprehensive arrangement of all fragments using interval graphs (IG’s) (Booth and Luecker, 1976; Rose et al., 1976; Korte and Mohring, 1989). A PQ-tree algorithm based on an IG will compute this order in linear time with respect to the number of overlapping fragments (Booth and Luecker, 1976). However, it was unclear whether the same linearity also applied in the case where not all the data were used in the analysis. This question is relevant to cosmid contig mapping, because the mapping task could be made more efficient if one could reorder the overlapping cosmids by using only a subset of the cosmids as probes in the cosmid-hybridization matrix.

We denote the graphical representation of a subset of cosmid probes, which is a more general case of an IG, as a probe interval graph (PIG), and demonstrate here that: (i) an ordering of cosmids along the YAC can still be accomplished, and (ii) such an ordering can be computed in a time that is linear with respect to the total number of cosmids. Thus, in our strategy for contig mapping, establishing the complete arrangement of all mutually overlapping fragments is not necessary. Importantly, to determine the approximate placement of markers along the chromosome, where those markers have already been localized to specific fragments, it is only necessary to construct a ‘path’ through the pattern of fragments, building it by successively attaching fragments which overlap each other. A connected series of fragments spanning a region of interest has been called a ‘spanning path,’ or SP (T. Slezak, personal communication). Note that with a cosmid library with 3–5x coverage, there still may not be a single SP covering the whole YAC. For marker placement purposes, the relative position of the many other, non-path fragments is of no consequence, as knowing the location of the non-path fragments would provide little additional mapping information. We can, however, relate non-path but overlapping cosmids to the SP cosmids, thereby obtaining information about the relative linear arrangement and location of all the cosmids.

Principle of the method

The method for identifying an SP, given pairwise overlap data as input, is embodied in an interval graph, which represents effectively the overlap relationships between fragments. It is easiest to present the method by illustrating the evolution of an overlap graph as a particular set of hypothetical data is analyzed. The Appendix contains the definitions upon which this analysis is based.

Consider the configuration of cosmid clones depicted schematically in Figure 1A. Each cosmid’s insert is represented by a numbered horizontal segment. The labelled segments at the ends of the cosmids, which are used as probes in the hybridization, are represented by the thick lines. Of course, this arrangement of overlapping cosmids is not known initially. From the matrix hybridization data, a list of cosmid-cosmid overlaps can be generated (Figure 1B).

Note that the example shown in Figure 1 is an IG, as all the cosmids have been used as probes. Note also that in the example presented here, all possible cosmid hybridization pairs have a reciprocal relationship to each other (e.g. cosmid 27 hybridizes to cosmid 32, and reciprocally, 32 hybridizes to 27). This reciprocity is always true in an IG in which the cosmids are labelled across their entire length. However, in the more restrictive case where all the cosmids are labeled only at their edges, reciprocity is usually, but not always, true. This issue, which has implications for the analysis of both false positives and false negatives, is discussed in greater detail below.

Next, one cosmid is picked at random (cosmid #28 in this example). A tree is built using this cosmid as the root vertex, as follows. The algorithm examines the overlap list (Figure 1B) to identify the other fragments that overlap...
Fig. 1. Deducing a cosmid SP using hypothetical IG data. (A) Individual cosmids (numbered) labeled by T3 and T7 polymerase (filled rectangles at cosmid positions below the SP based on the overlap data in B. The SP only indicates overlaps; no spacing between adjacent cosmids should be inferred. (B) Initial cosmid hybridization matrix, based on data in panel B; asterisks along the main diagonal denote cosmids hybridizing to themselves; dashes denote all other cosmid-cosmid hybridization pairs. (C) Potential SP's (i.e. ordered cosmids in ovals) derived from the second breadth-first search in C (bold lines). The SP only indicates overlaps; no spacing between adjacent cosmids should be inferred. (D) Potential SP's derived from the second breadth-first search in C (boxed in Figure 1C). Having rebuilt the tree in the second breadth-first search from a boundary vertex, the algorithm concludes by choosing any vertex at the lowest level of the rebuilt tree (cosmids 31 or 33 in this example), which is potentially located at the other boundary of the contig (Definition 5, Appendix). In other words, starting with a randomly-chosen cosmid, the first breadth-first search identified the cosmid(s) at one end of the contig, and those ends became the starting root(s) for the second breadth-first searches to identify the cosmid(s) located at the opposite end of the contig. As the second breadth-first search tree contains all the cosmids encountered between the two boundaries in hierarchial order, all spanning paths can be traced up the tree between the boundaries. In this example, there are two possible SPs (Definition 6, Appendix; Figure 1D). Given only binary data, it is not possible to determine which cosmid (31 or 33 at the left end; 23 or 24 at the right end) is at the true boundary. This process is repeated until all possible SPs along the length of the YAC have been examined.

**Reciprocity**

As noted above, labeling cosmids only at their edges can result in non-reciprocal cosmid hybridization pairs. For example, if cosmid \( x \) is located entirely inside of cosmid \( y \), \( x \) will certainly hybridize to \( y \), but \( y \) may not hybridize to \( x \), because cosmid \( y \) is labeled only at its ends and its probe cannot identify cosmid \( x \). Since, in the case of an IG, all the cosmids are used as probes, the reciprocal relationship
between cosmids $x$ and $y$ can still be inferred (i.e. if $x$ identifies $y$, then $y$ should identify $x$). However, in practice we ignore non-reciprocal hybridization 'hits' between cosmide probe pairs, for the simple reason that it is unknown whether such non-reciprocity is real (e.g. cosmide $x$ lies completely inside cosmide $y$) or artifactual (e.g. cosmide $x$ is a false positive, or conversely, cosmide $y$ is a false negative). As indicated below, non-reciprocal hybridization is not ignored when a member of the cosmide pair is not a probe.

Given the redundancy in the YAC and cosmide libraries, this requirement for reciprocity between cosmide probes is, in fact, a powerful tool to eliminate false overlaps caused by repetitive elements. For example, Figure 2A shows a hypothetical situation in which one probe on cosmide 30 contains a repetitive element (shaded) that hybridizes to five other cosmides (17, 27, 31, 32, 33), three of which (cosmides 17, 31, and 33) are not authentic overlaps. Because there is no reciprocity between cosmide 30–17, 30–31, or 30–33, those three pairs are eliminated from the data set (note that authentic overlaps between 30–27 and 30–32 exist, whether or not the repetitive element is present). Once these three non-reciprocal cosmide probe pairs have been eliminated, it is easy to show that a correct SP can be generated.

**Partial overlaps**

Figure 3 illustrates the extension of this approach to a situation in which only partial overlap data are available. In accordance with Definition 2 (Appendix), such a situation generates a probe interval graph (PIG), because even though most overlaps can be deduced by reciprocity, overlap 17–31 remains undetected (Figure 3B). Nevertheless, by following the same approach as described for the IG, a spanning path can be deduced.

**False overlaps**

One can still have a situation in which a repetitive element is shared among two or more probes, thus generating false reciprocal hybridization data (e.g. cosmides 31 and 32 in Figure 2B). However, this problem can be dealt with. In practice, we have one simple test for false overlaps: if at least one SP cannot be constructed from the IG or PIG, we claim an inconsistency in the data set. It is easy to prove that for a PIG (and certainly for an IG), there is at least one SP in the set of paths generated by the algorithm. With false positives caused by repetitive elements, the set of paths generated (only one such path in our example, 36–28–19–27–30–23–24; Figure 2E) may not contain at least one SP (see Definition 6 in the Appendix). Thus, the failure to find an SP enables us to recognize the presence of false positives.

When an SP is not found, we use a heuristic method to find an SP, by eliminating some vertices from the data set. The most likely vertex to eliminate is that which has the largest number of edges, because it is the presence of the
4 non-probe cosmids are included as gaps in the matrix for clarity. (F) cosmid 31 is the deepest vertex, it is not the deepest p-vertex, and is inferred. (Q Breadth-first search trees. (D) An SP. Note that although (right). Only the overlap between non-probe cosmids 17 and 31 cannot be generated relatively unambiguous contigs.

Thus, the combination of stringent algorithmic requirement plus a highly redundant cosmid library enables us to generate relatively unambiguous contigs. 

5. In step 3, we weighted those non-SP cosmids that had a direct relationship only to SP-cosmids. However, there may be situations where a non-SP cosmid overlaps any SP-member cosmid (e.g., as a result of false negatives, or in the case where only a subset of cosmids are available as probes). In these situations, each of these initially-unordered cosmids is assigned a weighted distance equal to that of the first non-SP cosmid in the order with which it overlaps, and is placed in the order immediately following that cosmid.

Reordering the cosmids

Once one or more SPs are obtained, each can be used as a framework for reordering the original cosmid-hybridization matrix (Figure 1E) according to distance along the YAC. This reordering of the matrix is done by a 'weighted distance' scheme, as follows (see the example of Figure 1).

1. Every cosmid in the SP is given a 'weighted distance' corresponding to its order in the path. For example, in the first SP shown in Figure 1D (i.e. 31-17-36-28-19-27-30-23-24), cosmid 31 has weight 1, cosmid 17 has weight 2, and so on, with cosmid 24 having weight 9.

2. For the remaining non-path cosmids (i.e. 21, 32, and 33), each cosmid is inspected individually. If it overlaps with one or more SP-member cosmids (see Figure 1B), we count its weight as the average of the weighted distances of the SP-member cosmids that it overlaps. For example, non-path cosmid 21 has a weighted distance of 4.5, as it overlaps cosmids 19 (weight 5) and 28 (weight 4). Similarly, cosmid 32 has a weighted distance of 6.5 (average of distances 6 [cosmid 27] and 7 [cosmid 30]); and cosmid 33 has a distance of 1.5 (average of distances 1 [cosmid 31] and 2 [cosmid 17]).

3. The weighted distances of the non-SP cosmids generated in step 2 are added at the appropriate positions to the list generated in step 1. Thus, the weighted distances are now: 1 (cosmid 31), 1.5 (33), 2 (17), 3 (36), 4 (28), 4.5 (21), 5 (19), 6 (27), 6.5 (32), 7 (30), 8 (23), and 9 (24). Sometimes 2 cosmids may have the same value; in that case they are arranged arbitrarily in alphanumeric order.

4. In our example, all 12 cosmids have now been ordered, as each of the 3 non-SP cosmids overlaps at least one SP-member cosmid. However, there may be cases where there are no data available to indicate where a non-SP cosmid overlaps any SP-member cosmid (e.g., as a result of false negatives, or in the case where only a subset of cosmids are available as probes). In these situations, each of these initially-unordered cosmids is assigned a weighted distances equal to that of the first non-SP cosmid in the order with which it overlaps, and is placed in the order immediately following that cosmid.


deductive process.

In dealing with multi-fold clone libraries (e.g. a cosmid library with 5x coverage), the ability to detect an SP is not affected by eliminating a few cosmids from the data set. Thus, the combination of stringent algorithmic requirement plus a highly redundant cosmid library enables us to generate relatively unambiguous contigs.

reweighting of non-SP cosmids in step 3. Second, it is
Reweighted based on its relationship to all the cosmids (i.e., SP and non-SP) following the reweighting in step 3. The new list is now renumbered in order.

6. For large data sets, step 5 may need to be reiterated once or twice before a stable order is reached.

The reordered matrix based on this scheme (Figure 1F) shows all the cosmids (that is, both those contributing to the SP [shown in bold in Figure 1F] and those not contributing to the SP) arranged contiguously along the length of the YAC. The advantage of this matrix is that it allows one to: (1) visualize the location of all the cosmids in the contig, (2) identify 'weak points' in the contig where only a relatively small number of cosmids reside, (3) visualize the representation of the cosmid library and the 'clonability' of certain regions, (4) identify cosmids for sequencing as a source of statistically-spaced STS's, and (5) identify outliers which may indicate erroneously-assigned hybridizing pairs. In practice, removing an outlier and repeating the complete calculation will generate a significantly improved contig map.

**Algorithmic complexity**

The CPU time required for one breadth-first search is linear with respect to the sum of the number of vertices (i.e., cosmids) and edges (i.e., cosmid overlaps) in the graph. However, this does not mean that the overall algorithm is linear with respect to the number of cosmids analyzed, because with multiple potential boundaries more than just two breadth-first searches are required to establish all possible SPs. The maximum number of such searches is of course less than the number of cosmids analyzed. Thus, the lower bound of the algorithmic complexity is linear; the upper bound is quadratic.

**Language and system**

The algorithm was coded in the C language on a Convex C-220 running the Convex Unix operating system. We believe the code is portable to most computers with a C compiler. The copyrighted program is available from the first author, who may be contacted via e-mail at zhangp@cuhhca.hhmi.columbia.edu (128.59.98.1).

**Results**

**Application to physical mapping of human chromosome 13**

The algorithm has been applied to matrix hybridization data generated for assembling cosmid and YAC contigs representing regions of chromosome 13 (Fischer et al., 1994). We have implemented this algorithm for two different but related sets of data: (i) overlapping cosmids spanning a YAC in order to generate a cosmid contig, and (ii) fingerprints of cosmid data spanning two different but overlapping YACs in order to generate a YAC contig. We show here an example of the application of the algorithm for deducing a cosmid contig spanning YAC2, a YAC localized to human chromosome 13 (Fischer et al., 1994).

The binary overlap data from the cosmid-cosmid overlap table of YAC2 and its associated hybridization matrix were the input to the algorithm. Hybridization of YAC2 to the cosmid library identified 67 cosmids. During generation of the cosmid hybridization matrix, 24 of the 67 cosmids identified no other cosmids, and were eliminated from further analysis by the algorithm. The remaining 43 cosmids were analyzed by the algorithm, and generated 2 contigs, one of 41 cosmids and one of 2 cosmids (i.e., 2 'islands' along YAC2). Only the 41-cosmid contig is displayed here, first as a hybridization matrix (Figure 4A), then as a spanning path (Figure 4B), and finally as a reordered matrix (Figure 4C).

Certain features of this example are noteworthy. First, the YAC contig was generated using only partial overlap data (i.e., using a PIG). Second, only 17 of the 41 cosmids were required to generate the SP (Figure 4B). Third, when the SP was generated, two non-path cosmids—2d2 and 2d3—were identified that were not ordered based on the initial weighted-distance scheme using steps 1–3 above (see Figure 4C): cosmid 2d2 hybridized only to non-SP cosmids 2f3 and 2a3; cosmid 2d3 hybridized only to non-path cosmid 2b9. These 2 cosmids were ordered after a reiteration of the weighting scheme (steps 4–6 above).

Alignment of the 41 cosmids spanning YAC2 derived from the initial 67 cosmids took 0.28 seconds of CPU time on a Convex 220 computer. An alignment of 202 cosmids spanning a 3-YAC contig (not shown) took 1.35 seconds of CPU time, and 433 cosmids took 3.15 sec (not shown).
Discussion

The described algorithm simplifies the logical process of deducing spanning paths, by transforming the overlap data into a graph abstraction. Importantly, the time required to conduct each breadth-first search is linear with respect to the number of vertices and edges. Thus, for most practical problems, the SPs can be deduced in a few seconds of CPU time. All of the cosmids (both SP- and non-SP members) can be ordered along the length of the contig by weighting each cosmid in relation to its proximity to a terminal cosmid.

The SPs derived from hypothetical and experimental data can be obtained via probe data not revealing all of the true cosmid-cosmid overlaps, that is, by using a PIG rather than an IG. This is especially true for overlap data obtained from robust cosmid and YAC libraries, which are deliberately created with 3x-5x fragment densities to minimize gaps in their sequence coverage.

The algorithm has the ability to deal with false negatives and false positives. False negatives (i.e. absence of a hybridization signal between two cosmids which do overlap) usually present no problem to the construction of an SP, due to the redundancy of clones in 3x- to 5x-deep YAC and cosmid libraries. On the other hand, false positives (i.e. presence of a hybridization signal between two cosmids which do not overlap), which can arise due to cross-hybridization between repetitive elements, can present a serious problem which must be dealt with, as they may produce erroneous edges in the IG or PIG. Any breadth-first search which uses those edges may result in an incorrect SP. We overcome this problem in a non-rigorous, yet effective, way. We require that all pairs of cosmid overlaps be reciprocal if both cosmids in the pair are used as probes, that is, if cosmid x hybridizes to cosmid y, then y must hybridize to x. If reciprocity is not achieved, that particular overlap is eliminated from the data set.

The data on the chromosome 13 cosmid contigs analyzed to date confirm the linearity of the algorithm, at least when the number of cosmids being analyzed is relatively small. However, there will certainly be circumstances in which there are deviations from linearity. For a segment of DNA as large as chromosome 13 (~100 Mb), the program should theoretically be able to calculate a complete 10,000-cosmid contig across the entire chromosome in ~2 min, assuming linearity. Of course, this is far too optimistic. However, even in the worst case (i.e. a quadratic relationship between CPUs and cosmid number), the CPU time would be only 1–2 h.

The algorithm described here and the associated software have proved successful in mapping cosmid contigs on human chromosome 13 using cosmid hybridization data. Using hybridization data, Mott et al. (1993) have recently shown similar success in the physical mapping of the small, 14-Mb, genome of *Schizosaccharomyces pombe*. Similarly, Mizukami et al. (1993) adopted a branch-and-bound algorithm based on hybridization data to obtain a physical map of the *S. pombe* genome. The approach of Mott et al. (1993) used a variety of probes—entire fragments of *S. pombe* DNA inserted into cosmid, YAC, and PI vectors, YAC and cosmid end probes, and genetically mapped markers—to detect overlapping members of cosmid, YAC, and PI libraries of *S. pombe* DNA. They used two approaches to establish contigs, with comparable results. The first was based on a maximum-likelihood estimate of the distance between probes. The second was described as ‘ordering probes using heuristics,’ and has some similarity with the algorithm described here. However, whereas both methods described by Mott et al. (1993) and the method described by Mizukami et al. (1993) order only probes as a first step, the PIG approach described here requires that only a subset of cosmid end-probes need be hybridized against the YAC library, and orders both probes and unlabelled cosmids in a single step. The approaches taken by all groups subsequently order all clones from the hybridization experiment relative to the spanning path(s) determined, and all approaches filter out probes considered as possible false positives. Although comparable timings are not available, our method illustrates that contigs can be generated in real-time from existing clone libraries using the available hybridization data.

Acknowledgements

We thank Mr. T. Mark Reboul and Dr. James Russo for critical comments. This work was supported by grants from the National Institutes of Health (NS28828, AG08702, and HG00424).

Appendix

A formal proof for the validity of using an IG or PIG to create a cosmid contig is not given. Rather, the IG and PIG are defined using set notation, and further details of the method are provided.

Definitions

Inter-clone hybridization data can be represented in graphs such that *vertices* correspond to the intervals spanned by the clones, and *edges* correspond to overlaps between pairs of clones. If every clone were used as a probe, and each clone were labelled across its entire length, so that all possible overlapping clones were identified in the matrix hybridization, we would obtain what has been called an *interval graph*, or IG (Golumbic, 1980). However, by labelling only a fraction of all clones,
the matrix of overlapping clones is only a subset of all theoretically possible overlaps. Nevertheless, the subset of hybridization data can still be used to generate a contig map, even though the interval graph information is incomplete. We call this type of graph, based upon incomplete overlap information, a probe interval graph, or PIG.

Definition 1: Let V be a finite set of intervals of a real line. Then G(V) denotes the graph whose set of vertices is V, where \( v_1, v_2 \in V \) are joined by an edge, if and only if, \( v_1 \cap v_2 \neq \emptyset \). The graph G(V) is called the interval graph associated with V.

Definition 2: Let \( P \) be a subset of V. Then G(V,P) denotes the graph whose set of vertices is V, where \( v_1, v_2 \in V \) are joined by an edge if, and only if, the following 2 conditions are satisfied:

(i) \( v_1 \cap v_2 \neq \emptyset \)

(ii) \( v_1 \in P \) or \( v_2 \in P \)

The elements of P are called p-vertices. The graph G(V,P) is called the probe interval graph associated with the pair (V,P). Note that G(V,V) = G(V), hence an interval graph may be regarded as a special kind of probe interval graph.

Definition 3: A path \( \sigma \) along either an interval or probe interval graph is a finite sequence of vertices, \( v_1, v_2, \ldots, v_n \), such that \( v_i \) is connected to \( v_{i+1} \) by an edge for all i such that \( 1 \leq i \leq n-1 \). The path \( \sigma \) is said to be a simple path if \( v_i \neq v_j \) for \( i \neq j \). That is, in a simple path a vertex cannot be repeated.

Definition 4: Given the set of intervals V and \( V^* \), then \( V^* \) is a cover set of V if the union of the intervals in set V is a subset of the union of the intervals in set \( V^* \). For example, vertices \( \{36,19,28\} \) is a cover set of vertex \( \{21\} \) (Fig. 1).

Definition 5: Let \( \sigma = v_1, v_2, \ldots, v_n \) be a simple path on a probe interval graph, G(V,P). Let \( W = \{w_1, w_2, \ldots, w_m\} \subset V \) be the set of all vertices which have a non-empty intersection with either \( v_i \) or \( v_n \) (the boundaries). The expanded union of \( \sigma \) is defined to be the union of all intervals represented by the vertices contained in \( W \cup \sigma \), that is, the expanded union of \( \sigma = w_1 \cup w_2 \ldots \cup w_m \cup v_1 \cup v_2 \ldots \cup v_n \). Thus, in Fig. 1, \( \sigma \) could be the path \( \{19,27,30\} \), and therefore the expanded union of \( \sigma \) includes the overlaps with the vertices \( \{21,23,28,32\} \). Thus the expanded union is \( \{19,21,23,27,30,32\} \).

Definition 6: Given a probe interval graph G(V,P), a simple path from one p-vertex to another p-vertex is a spanning path (SP), that is, a path spanning the line as described by G, if the expanded union of this path is a cover set of V, that is, if it includes the boundary vertices.

References


