Physical Maps and Interval Sandwich Problems: Bounded Degrees Help

Haim Kaplan*
Department of Computer Science
Princeton University
Princeton, NJ 08544 USA
hkl@cs.princeton.edu.

Ron Shamir†
Department of Computer Science
Sackler Faculty of Exact Sciences
Tel Aviv University
Tel-Aviv 69978 ISRAEL
shamir@math.tau.ac.il.

Abstract

The problems of Interval Sandwich (IS) and Intervalizing Colored Graphs (ICG) have received a lot of attention recently, due to their applicability to DNA physical mapping problems with ambiguous data. Most of the results obtained so far on the problems were hardness results. Here we study the problems under assumptions of sparseness, which hold in the biological context. We prove that both problems are polynomial when either (1) the input graph degree and the solution graph clique size are bounded, or (2) the solution graph degree is bounded. In particular, this implies that ICG is polynomial on bounded degree graphs for every fixed number of colors, in contrast with the recent result of Bodlaender and de Fluiter.

1. Introduction

A graph is an *interval graph* if one can assign an interval on the real line to each vertex so that two vertices are adjacent if and only if their intervals have a nonempty intersection. Consider the following problem:

> INTERVAL SANDWICH (IS):
> **INSTANCE:** A triple $S = (V, E, F)$, where V is a set of vertices, and E and F are disjoint sets of edges on V.
> **QUESTION:** Decide whether there exists an interval graph $G' = (V, E')$ so that $E \subseteq E'$ and $E' \cap F = \emptyset$.

E and F are called the sets of *mandatory* and *forbidden* edges, respectively. The graph G', if it exists, is called a *sandwich graph* for the instance S. Without loss of generality, $G = (V, E)$ (which we call the *mandatory graph*, or simply the *input graph*) is assumed to be connected.

The interval sandwich problem was introduced by Golumbic and Shamir [12] in the context of temporal reasoning, and was shown to be NP-complete. See also [11] for a simpler proof. The following special case of the problem, which is motivated by molecular biology, was studied in [8] and [11]: Recall that a *coloring* of a graph $G = (V, E)$ is a function $c : V \to \{1, \dots, k\}$ such that for every edge $(x, y) \in E$, $c(x) \neq c(y)$.

> INTERVALIZING COLORED GRAPHS (ICG):
> **INSTANCE:** A graph $G = (V, E)$ and a coloring c of it.
> **QUESTION:** Is G a subgraph of an interval graph G' that is properly colored by c?

Clearly ICG is a restriction of IS where $F = \{(i, j) | c(i) = c(j)\}$. ICG was shown to be NP-complete independently in [8] and [11]. Parameterized hardness results for the problem, where the parameter is the number of colors, were given in [8] and strengthened in [4]. Recently, Bodlaender and de Fluiter [3] strengthened these results further by showing that ICG is NP-complete even when the number of colors k is fixed to any constant $k \geq 4$. On the other hand they showed that ICG is polynomial for $k \leq 3$, by giving a rather involved algorithm and proof for three colors.

In this note we show that two parameterized versions of IS (and hence also of ICG) are tractable. Both problems deal with bounded degree instances. The first problem asks for the existence of an interval sandwich graph with maximum clique size no greater than k in a sandwich instance in which the degree of any vertex in the input graph G is

*Research at Princeton University partially supported by the NSF, Grant No. CCR-8920505, and the Office of Naval Research, Contract No. N00014-91-J-1463.
†Research supported in part by the Ministry of Science and the Arts, Israel, Grant No. 4911294.

no greater than d. For fixed k and d we give an $O(n^{k-1})$ algorithm for the problem. Hence, if the degree is bounded, IS is polynomial whenever the clique size is bounded. In particular this implies that ICG is polynomial *for every fixed number of colors*, on bounded-degree input graphs. This is in contrast to the recent negative result of [3].

The second problem asks for the existence of an interval sandwich graph with maximum degree no greater than d. Note that the degree in the sandwich graph of the first problem is not bounded, so none of the two problems is a restriction of the other. For fixed d we give a $O(n^{d-1})$ algorithm for that problem. Again, this implies that ICG is polynomial when the interval graph must have bounded degree.

One of the motivations to studying IS and ICG is constructing physical DNA maps. In physical mapping in molecular biology, several copies of a long chain of DNA are cut, and fragments of the chains are extracted. Each fragment is a contiguous chain of DNA, called a *clone*. The order of the clones is lost, and the goal is to reconstruct that order, based on information on pairwise overlaps of clones. Biological experiments allow the determination whether two clones overlap (have a nonempty intersection). The experimental data of clone-clone overlap can be presented as a graph G whose vertices corresponding to clones, overlaps corresponding to edges in E, and non-overlaps corresponding to edges in F. As is often the case in practice, for some pairs of clones, the information whether they overlap or not is unknown, due to unperformed or inconclusive experiments. In this situation, deciding if the data is consistent with some clone order is equivalent to the IS problem. If clones can be partitioned into sets where in each set all clones originate from a single copy of the chromosome, then each such set forms an independent set in the graph, which can be assigned a single color. In that case the decision problem is equivalent to ICG. See [5, 20, 2, 11, 9] for more on the biology and computational aspects of this problem.

Our motivation to studying bounded degree problems is from physical mapping, although similar sparse problems arise naturally in many other applications. We have observed in [15] that most biological maps are very sparse: The size of the largest set of mutually overlapping clones in such maps is typically between 5 and 15, even when the maps contain thousands of clones [17, 18]. The question is whether by restricting the problems to satisfy this property, their complexity will be polynomial. In [15], sparseness was modeled by bounding the clique size of the sandwich graph, for the case of equal length clones. It was shown that the unit interval sandwich problem is polynomial whenever the clique size in the resulting realization is bounded by a constant [15, 16]. Here we deal with clones of arbitrary lengths. The problem discussed in Section 3.1 models sparseness by bounding the maximum clique size of the sandwich graph, and additionally, by bounding the maximum degree in the input graph. The first requirement means that the largest set of mutually overlapping clones in the solution map should be bounded. The second means that at the input, the maximum number of clones known to intersect a particular one is bounded. The problem discussed in Section 3.2 models sparseness by bounding the maximum degree of the required sandwich graph. In biological terms it means bounding the number of clones intersecting any single clone in the map we seek. The same modeling of sparseness was exploited in [13] for studying other mapping problems. The algorithms used here build on a dynamic programming scheme, motivated by ideas from [19] and [14].

For basic facts and terminology on graph theory and interval graphs, see [10]. For a description of parameterized complexity theory, see [6, 1, 7]. Due to space constraints, some proofs are omitted or sketched in this abstract.

2. The basic scheme

The basic idea is very simple: Generate solutions for subproblems induced on increasing number of vertices. Each solution (an interval sandwich graph) is represented by a set of intervals (a realization). A method of extending a solution for a subproblem by adding a single vertex is given. The key is to observe that only a relatively small amount of information in a representation of a solution determines which vertices it contains and if it can be extended. Hence, solutions can be considered equivalent if that information is identical in them, and the scheme is applied to representatives of the equivalence classes.

For an instance $S = (V, E, F)$ and $X \subseteq V$ denote by S_X the *subinstance* of the interval sandwich problem induced by the vertex set X, i.e., $S_X = (X, E_X, F_X)$ where $E_X = \{(x, y) \in E \mid x, y \in X\}$ and $F_X = \{(x, y) \in F \mid x, y \in X\}$. (Throughout, we use sets as subscripts to denote restriction.) A vertex $v \in X$ is *active* in X if there exists some $y \in V - X$ such that $(v, y) \in E$. The *active set* of X, denoted $\Delta(X)$, is the set of active vertices in X.

A *realization* of S_X is a set of intervals on the real line $I = \{Int(v) \mid v \in X\}$ where $Int(v) = [L(v), R(v)]$ satisfying:

1. For every $v \in X$, $L(v) \leq R(v)$.
2. For every $u, v \in X$, if $(u, v) \in E$, then $Int(u) \cap Int(v) \neq \emptyset$.
3. For every $u, v \in X$, if $(u, v) \in F$, then $Int(u) \cap Int(v) = \emptyset$.

Condition 1 makes sure that the intervals are properly defined. Conditions 2 and 3 guarantee that the realization contains all the mandatory edges and no forbidden edges. Given a realization I of S_X define a relation $\prec_I$ on X so that $a \prec_I b$ if and only if $R(a) < L(b)$. The relation $\prec_I$ is called the *interval order* of I. We will abbreviate $\prec_I$ to simply $\prec$ when I is clear from the context. A vertex $v \in X$ is *maximal* in I if for no other vertex u, $v \prec u$. v is *maximum* in I if $L(v) \geq L(u)$ for all $u \in X$.

A *layout* of X, where $\emptyset \neq X \subseteq V$, is a realization I of X that satisfies also:

4. Every active vertex in X is maximal in I.

The set X is called the *domain* of the realization. A *complete realization* has domain V. Note that every realization of S is a layout. The active set of a realization I with domain X is denoted $\Delta(I)$ and is simply $\Delta(X)$.

Let I^1 and I^2 be realizations of V^1 and V^2 respectively. I^2 *extends* I^1 *by* x if $V^2 = V^1 \cup \{x\}$ for some vertex $x \in V - V^1$, the interval order of I^1 and the interval order of I^2 restricted to V^1 are identical, and x is maximum in I^2.

Lemma 2.1 *If I is a layout of X with $|X| \geq 2$, then I extends some layout.*

Proof. Let M and A be the sets of maximal and active vertices in I, respectively. By definition, $A \subseteq M$. Let z be a maximum vertex in I. Denote $X' = X - \{z\}$, and let I' be the set of intervals obtained by restricting I to X'. By definition, I' is a realization of X' and I extends I' by z. We shall show that I' is indeed a layout.

Let M' be the set of maximal vertices in I'. Note that $z \in M$ and $M - \{z\} \subseteq M'$. Consider an active vertex t in X': If t is also active in X then $t \in M$, so $t \in M'$. Otherwise, t is not active in X, which implies that it is active in X' only since $(t, z) \in E$. But then $Int(t) \cap Int(z) \neq \emptyset$, and the choice of z guarantees that t is in M and M'. Hence, I' satisfies condition 4 and is a layout. ■

The algorithm will start with the *singleton layouts*, whose domains are single vertices, and recursively extend them by new vertices. One possible way to do this is as follows: Start with a layout $[1, 1]$ for every single vertex. Suppose I is a layout of X and $z \in V - X$ and let $X' = X \cup \{z\}$. If z is connected by an edge in F to any vertex in $\Delta(X)$, it cannot extend I. Otherwise, a layout I' of X' can be built as follows: Suppose I is defined by the intervals $Int(x) = [L(x), R(x)]$, with M the set of maximal intervals in I. Let $p = 1/2 + \max\{R(x)|x \in M\}$. The layout I' is defined by the intervals $Int'(x) = [L'(x), R'(x)]$ where:

$$L'(v) = \begin{cases} L(v) & \text{if } v \in X \\ p & \text{if } v = z \end{cases}$$
$$R'(v) = \begin{cases} R(v) & \text{if } v \notin \Delta(X) \text{ and } v \neq z \\ p+1 & \text{if } v \in \Delta(X) \text{ or } v = z \end{cases} \tag{A}$$

Figure 1 gives an example of a sequence of layouts extended by this rule. Intuitively, a layout I on domain X is just an interval realization that solves the sandwich subproblem S_X, and may be potentially "extended to the right". When I is extended by z, the new interval of z is added to the right of all the old intervals. All the old endpoints are "frozen", except the right endpoints of intervals that are active in X, which are "pushed to the right", to intersect the new interval. Note that some maximal intervals in the new layout (perhaps including z itself) may be non-active in X', but all active intervals are maximal, as required.

Let I be a layout of X. For each vertex $v \in \Delta(X)$ denote by $dangling_I(v)$ the set of edges $\{(v, w) \in E | w \in V - X\}$. Two layouts I and J are *equivalent*, denoted $I \sim J$, iff they have the same active set and for each active vertex v, $dangling_I(v) = dangling_J(v)$. The relation thus defined is clearly an equivalence relation that partitions the set of layouts into equivalence classes. We shall refer to non-empty equivalence classes simply as *classes*.

Observation 2.2 *Equivalent layouts have identical domains.*

Proof. By the connectivity of G, layouts whose active set is empty have no dangling edges and domain V. Consider a layout I with a nonempty active set. The union $\tilde{E}$ of $dangling_I(v)$ for all $v \in \Delta(I)$ is a cutset in G. Since G is connected, $domain(I)$ is the union of all connected components of $G(V, E - \tilde{E})$ which contain active vertices. ■

Hence, there is only one class with domain V. Since any realization with domain V satisfies conditions 1-3 (and 4, trivially) it is a layout and we can conclude:

Corollary 2.3 *There exists an interval sandwich iff there is a class with domain V.*

Observation 2.4 *If I and J are equivalent layouts and K extends I then there exists a layout K' such that $K' \sim K$ and K' extends J.*

Proof. Suppose $I \sim J$ and K extends I by z. By Observation 2.2, I and J have the same domain, say X. By condition 4 in the definition of a layout, all active vertices in X are maximal in I and in J. Apply the method of extension (A) to extend J by x. Denote the resulting layout K'. It is easy to verify that $K' \sim K$. ■

Note that the converse to Observation 2.4 is not true as two non-equivalent layouts could be extended to the same one.

Observation 2.4 implies that if there is a layout in class A that extends a layout in class B, then for every layout in B there exists a layout in A that extends it. Hence, in this case we can say that *class A extends class B*, and talk about the active set and the domain of a class. Moreover, in the algorithm we can extend any layout in a class in order to extend the class, and the choice of the layout does not matter. By this discussion, Lemma 2.1 implies:

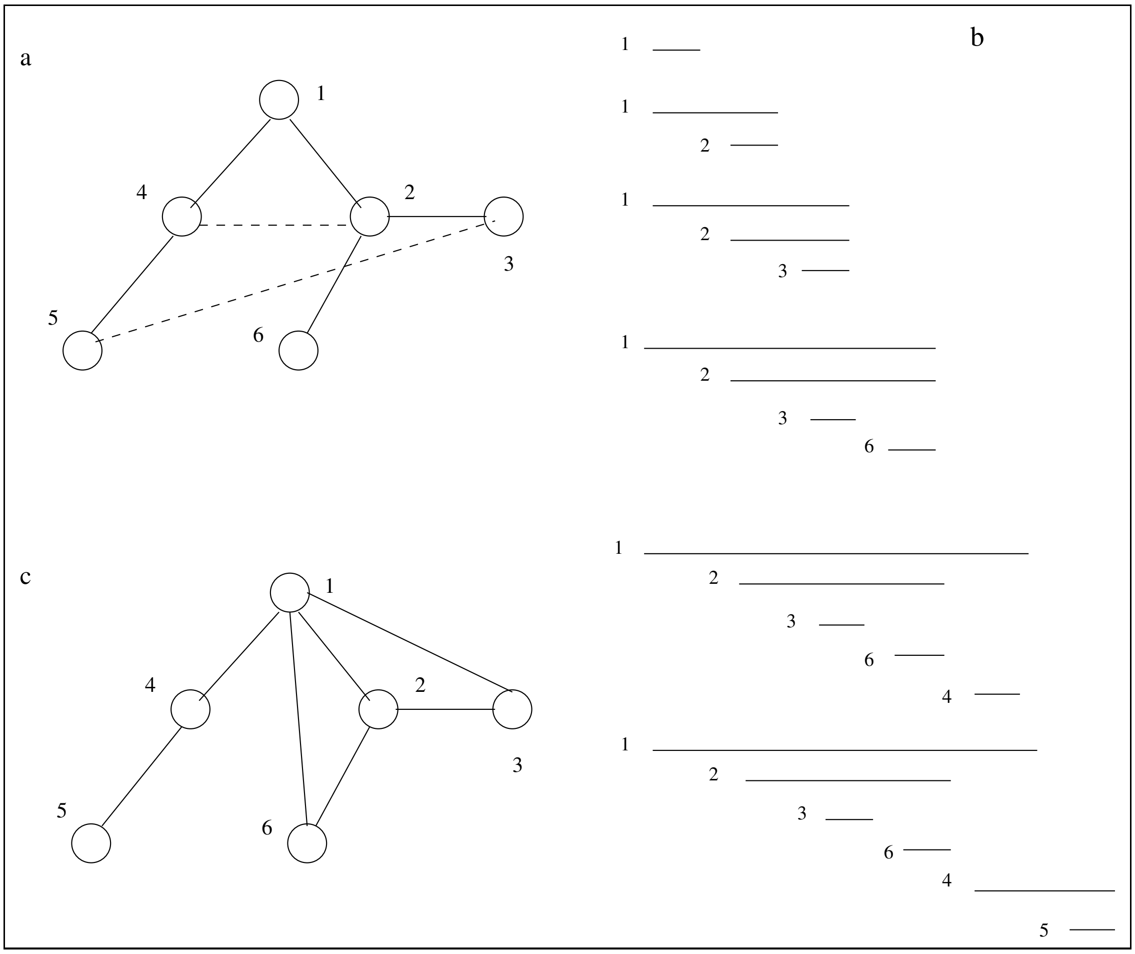

Figure 1. a) A sandwich instance. Solid edges are mandatory. Broken edges are forbidden. b) A sequence of layouts each extending the previous by a vertex according to the method of extension (A). Extension order is: 1,2,3,6,4,5. Note that the partial layout 1,2,3 cannot be extended by 4. c) The sandwich graph corresponding to the final layout.

Observation 2.5 *A class whose domain is not a singleton extends at least one other class.*

Observation 2.6 *Let B be a class, and let $x \in V - domain(B)$. There exists a class C that extends B by x iff*

1. For every $v \in \Delta(B)$, $dangling_C(v) = dangling_B(v) - \{(v, x)\}$.

2. $dangling_C(x) = \{(x, y) \in E \mid y \notin \Delta(B)\}$.

3. $\Delta(C) = \{w \in \Delta(B) \cup \{x\} \mid dangling_C(w) \neq \emptyset\ \}$

4. For every $t \in \Delta(B)$, $(x, t) \notin F$.

These observations imply an algorithm for finding an interval sandwich: Start with the classes whose domain is a singleton and use Observation 2.6 to recursively extend classes. Output "yes" iff there is a class with domain V. See Figure 2 for pseudo-code.

Theorem 2.7 *Algorithm SANDWICH generates every non-empty equivalence class.*

Corollary 2.8 *Algorithm SANDWICH correctly determines if there exists an interval sandwich graph.*

By storing one layout for each class, the algorithm can also produce a realization if there is one. We shall assume that for any pair of vertices the existence of a forbidden edge between them can be checked in constant time. Step 5 is polynomial per extension attempt. Assuming a randomly accessible table of all possible classes is kept, when a class

```
algorithm SANDWICH(S);

/* S = (V, E, F) is a sandwich instance */
/* initialization */
1. initialize an empty queue Q.
2. for each v ∈ V, create an equivalence class C with domain {v}
and dangling(C) = {(v, y) | (v, y) ∈ E}. Add C to the queue Q.
begin
        while Q ≠ ∅ do :
        /* iteration */
        begin
                3. Remove some class C with domain X from Q.
                4. For every z ∉ X do:
                begin
                        /* extension attempt */
                        5. Try to extend C by z (using Observation 2.6)
                        6. Suppose a class C' is generated.
                        if C' has domain V then output "success" and stop.
                        Otherwise, if C' is new, then add it to Q.
                end
        end
        7. output "failure" and stop.
end
```

Figure 2. A schematic algorithm for the IS problem

is extended, one can check in constant time if the resulting class is new. Under these assumptions, the algorithm is polynomial in the number of classes. In general this algorithm is superpolynomial since the number of classes is so. However, for instances with a polynomial number of classes the algorithm requires polynomial time. In the next section we show how to adapt the scheme to be polynomial for the problems we have defined in Section 2.

3. Polynomial problems

In this section we present two polynomial restrictions of the general Interval Sandwich problem. For the first, presented in Section 3.1, the relevant layouts are *bounded width layouts* (defined below). We show that the number of non-empty classes of the equivalence relation defined in Section 2 restricted to bounded width layouts is polynomial when the input graph has bounded degree. For the second, presented in Section 3.2, the relevant layouts are *degree-d layouts*. Here, in order to obtain the solution, we have to refine the equivalence relation, and then show that the refined relation has polynomially many classes of degree-d layouts.

3.1. Bounded degree input graph, bounded width

Recall that in this version of the problem we are given a sandwich instance where the degree of all vertices in the mandatory graph $G = (V, E)$ is bounded by d. A interval sandwich graph with maximum clique size at most k is required. Clearly, in this case one can restrict oneself to bounded width layouts: A layout I has *k-bounded width* if every $k + 1$ intervals in it have an empty intersection, and its active set contains at most $k - 1$ vertices. Note that $k > d$ and $k < d$ are both possible. Note also that if d is part of the input then the problem is NP-complete for every $k \geq 4$, by the results of [3], as the number of colors is an upper bound on the width of the graph.

The arguments leading to the algorithm in the previous section hold if one adds an additional restriction that the layouts have bounded width. (This can be added as an extra condition in Observation 2.6.) If one applies the dynamic programming scheme on classes of bounded width layout the following result is obtained:

Theorem 3.1 *For fixed k and d, algorithm SANDWICH restricted to k-bounded width layouts can be implemented to run in $O(n^{k-1})$ time on degree-d sandwich instances.*

Proof. The size of an active set cannot exceed $k - 1$. Thus, the number of possible different active sets in layouts is no greater than n^{k-1}. We partition the classes into two depending on the size of their active sets:

Let A be a class with an active set of size $k-1$. Let $w \notin domain(A)$. There is a class B that extends A by w iff the conditions of Observation 2.6 hold, and in addition the active set of B contains at most $k-1$ vertices. By the connectivity of G, this holds only if there is a dangling edge (v, w) with $v \in \Delta(A)$. If $w \in \Delta(B)$ then an extra condition is that there exists at least one vertex $v \in \Delta(A)$ with $dangling(v) = \{(v, w)\}$. In both cases there may be only $(k-1)d$ possible vertices w. Under the assumptions mentioned in the end of section 2, the effort for testing if A can be extended by w is $O(k)$, for a total of $O(dk^2)$ to reject all impossible extensions. The effort for the update of a possible extension is $O(kd)$, which we charge to the new class. There are $O(n^{k-1})$ active sets of this size. Using the above and the fact that the degrees are bounded by d one can show that the total work for classes of size $k-1$ is $O(n^{k-1})$.

For a class A with strictly less than $k-1$ vertices in its active set there might be $O(n)$ classes that extend it. Each vertex in the set $V - domain(A)$ is a potential candidate for extension. That set (and $domain(A)$) can be detected in $O(nd)$ time by performing a depth-first search starting from active vertices, on the graph G from which all dangling edges were omitted. One can check in $O(k)$ time per such vertex if A can be extended by w. If it can, it takes $O(kd)$ time to generate the new class, and as in the previous case this effort is charged to the new class. Thus the total work for such classes is again $O(n^{k-1})$. ■

Corollary 3.2 *For every fixed number of colors, Intervalizing Colored Graphs is polynomial if the degrees in the input graph are bounded by a constant.*

3.2. Bounded degree

Let I be a layout with domain $V' \subseteq V$. For a vertex $v \in V'$ denote by $degree(v)$ the number of vertices $w \in V' - \{v\}$ such that $Int(w) \cap Int(v) \neq \emptyset$. Note that since we require that every active vertex must be maximal, for $v \in \Delta(I)$, $degree(v) \geq |\Delta(I)|$. We say that layout I is of degree (at most) d if each interval intersects at most d other intervals and for each active vertex v, $degree(v) + |dangling(v)| \leq d$. Note that in particular $|\Delta(I)| \leq d$.

We address now the Interval Sandwich problem, where the sandwich graph G' must have maximum degree at most d. We need to modify here the definition of equivalence classes: Redefine an equivalence relation on the class of degree-d layouts as follows. Two layouts I and J are *equivalent*, denoted $I \sim J$, iff $\Delta(I) = \Delta(J)$, for each $v \in \Delta(I)$ $dangling_I(v) = dangling_J(v)$ and $degree_J(v) = degree_I(v)$. Observations 2.3 and 2.5 hold for classes of degree-d layouts as well. Observation 2.6 needs to be slightly modified:

Observation 3.3 *Let B be a class, and let $x \in V - domain(B)$. There exists a class C that extends B by x iff the conditions of Observation 2.6 are satisfied and in addition:*

$degree_C(x) = |\Delta(B)|$, and

For each $u \in \Delta(C) \cap \Delta(B)$, $degree_C(u) = 1 + degree_B(u)$.

Theorem 3.4 *For fixed d, algorithm SANDWICH restricted to degree-d layouts can be implemented to run in $O(n^{d-1})$ time.*

Proof. A layout L with an active set of size d can be extended iff $domain(L) = \Delta(L) = V - \{x\}$ for some $x \in V$ and $F = \emptyset$. The solution in that case is a clique of size $d+1$. One could check whether this case happens in constant time (independent of n). Hence, for the rest of the proof we assume that the size of an active set cannot exceed $d-1$.

Let A be a class with an active set of size $d-1$. For A to be extendible to a complete layout $\Delta(A)$ should be adjacent to no more than two vertices in $V - domain(A)$. If that is not the case one could avoid generating the class. Hence one can assume that an active set of size $d-1$ has no more than two different vertices in $V - domain(A)$ incident with its dangling edges. Only these two vertices are candidates for extension (Otherwise one would get a layout with active set of size d that we assumed not to exist). It takes $O(d)$ to generate each of the possible two extensions.

The number of possible different active sets of size $d-1$ is no greater than n^{d-1}. For each such active set there are $O(d^4)$ possible neighboring pairs. This pair determines the set of dangling edges incident with each active vertex. The degree of each active vertex is either $d-2$ or $d-1$. A brief calculation shows that the total work for all such classes is $O(n^{d-1})$.

For a class B with less than $d-1$ vertices in its active set, there might be $O(n)$ classes that extend it. The set $V - domain(B)$ can be calculated in $O(nd)$ time using depth first search. Each vertex in $V - domain(B)$ is a potential candidate for extension. One can check in $O(d)$ time per such vertex x if B can be extended by x. Hence, the total effort per class for identifying its extensions is $O(nd)$. It takes $O(d)$ time to generate an extension whose active set is of size $d-1$ and $O(d^2)$ to generate any other extension. We charge the effort for actually generating a class to the new class being generated. The number of possible active sets of size less than $d-1$ is $O(n^{d-2})$. For a vertex v in an active set the number of possible different combinations of $degree(v)$ and $dangling(v)$ is bounded by $d2^d$. One can use this to show that again the total work for such classes is $O(n^{d-1})$. ■

Corollary 3.5 *Intervalizing Colored Graphs is polynomial if the degrees in the interval graph G' are bounded by a constant.*

References

[1] K. Abrahamson, R. Downey, and M. Fellows. Fixed-parameter intractability II. In *Proceedings of the 10th Symposium on Theoretical Aspects of Computer Science (STACS'93)*, Lecture Notes in Computer Science vol. 665, pages 374–385. Springer-Verlag, Berlin, 1993.

[2] F. Alizadeh, R. M. Karp, L. W. Newberg, and D. K. Weisser. Physical mapping of chromosomes: A combinatorial problem in molecular biology. In *Proc. fourth annual ACM-SIAM Symp. on Discrete Algorithms (SODA 93)*, pages 371–381. ACM Press, 1993.

[3] H. L. Bodlaender and B. L. E. de Fluiter. Intervalizing k-colored graphs. In *Proc. ICALP 95*. LNCS, 1995.

[4] H. L. Bodlaender, M. R. Fellows, and M. T. Hallet. Beyond NP-Completeness for problems of bounded width: Hardness for the W hierarchy (extended abstract). In *Proceedings of the 26th Annual ACM Symposium on the Theory of Computing*, pages 449–458. ACM Press, New York, 1994.

[5] A. V. Carrano. Establishing the order of human chromosome-specific DNA fragments. In A. D. Woodhead and B. J. Barnhart, editors, *Biotechnology and the Human Genome*, pages 37–50. Plenum Press, New York, 1988.

[6] R. G. Downey and M. R. Fellows. Fixed-parameter intractability. In *Proceedings of the Seventh Annual Structure in Complexity Theory Conference (Structures'92)*, pages 36–49, Boston Massachusetts, 1992. IEEE Computer Society Press, Los Alamitos, California.

[7] R. G. Downey and M. R. Fellows. Fixed-parameter tractability and completeness III: Some structural aspects of the W hierarchy. In *Complexity Theory: Current Research (Proceedings of the 1992 Dagstuhl Workshop on Structural Complexity)*, pages 191–226. Cambridge University Press, Cambridge, 1993.

[8] M. R. Fellows, M. T. Hallet, and H. T. Wareham. DNA physical mapping: Three ways difficult. In *Proc. First European Symp. on Algorithms (ESA '93)*, pages 157–168. Springer, 1993. LNCS 726.

[9] P. W. Goldberg, M. C. Golumbic, H. Kaplan, and R. Shamir. Four strikes against physical mapping of DNA. *Journal of Computational Biology*, 2(1):139–152, 1995.

[10] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York, 1980.

[11] M. C. Golumbic, H. Kaplan, and R. Shamir. On the complexity of DNA physical mapping. *Advances in Applied Mathematics*, 15:251–261, 1994.

[12] M. C. Golumbic and R. Shamir. Complexity and algorithms for reasoning about time: A graph-theoretic approach. *J. ACM*, 40:1108–1133, 1993.

[13] D. Greenberg and S. Istrail. The chimeric mapping problem: algorithmic strategies and performance evaluation on synthetic genomic data. *Computers Chem.*, 18(3):207–220, 1994.

[14] E. Gurari and I. H. Sudborough. Improved dynamic programming algorithms for the bandwidth minimization and the mincut linear arrangement problem. *J. Algorithms*, 5:531–546, 1984.

[15] H. Kaplan and R. Shamir. Pathwidth, bandwidth and completion problems to proper interval graphs with small cliques. Technical report, CS Department, Tel Aviv University, November 1993. To appear in *SIAM J. Computing*.

[16] H. Kaplan, R. Shamir, and R. E. Tarjan. Tractability of parameterized completion problems on chordal and interval graphs: Minimum fill-in and physical mapping. In *Proceedings of the 35th Symposium on Foundations of Computer Science*, pages 780–791. IEEE Computer Science Press, Los Alamitos, California, 1994.

[17] Y. Kohara, K. Akiyama, and K. Isono. The physical map of the whole *E. coli* chromosome: application of a new strategy for rapid analysis and sorting of large genomic libraries. *Cell*, 50:495–508, 1987.

[18] M. V. Olson et al. Random-clone strategy for genomic restriction mapping in yeast. *Proc. Nat. Acad. Sci. USA*, 83:7826–7830, 1986.

[19] J. B. Saxe. Dynamic programming algorithms for recognizing small-bandwidth graphs in polynomial time. *SIAM J. Algebraic Discrete Methods*, 1:363–369, 1980.

[20] J. Watson, M. Gilman, J. Witkowski, and M. Zoller. *Recombinant DNA*. W.H. Freeman, New York, 2nd edition, 1992.