Powers of Geometric Intersection Graphs and Dispersion Algorithms

Geir Agnarsson[*] *Peter Damaschke*[†] *Magnús M. Halldórsson*[‡]

Abstract

We study powers of certain geometric intersection graphs: interval graphs, m-trapezoid graphs and circular-arc graphs. We define the *pseudo product*, $(G, G') \to G*G'$, of two graphs G and G' on the same set of vertices, and show that $G*G'$ is contained in one of the three classes of graphs mentioned here above, if both G and G' are also in that class and fulfill certain conditions. This gives a new proof of the fact that these classes are closed under taking power; more importantly, we get efficient methods for computing the representation for G^k if $k \geq 1$ is an integer and G belongs to one of these classes, with a given representation sorted by endpoints. We then use these results to give efficient algorithms for the *k-independent set*, *dispersion* and *weighted dispersion* problem on these classes of graphs, provided that their geometric representations are given.

2000 MSC: 05C12, 05C62, 05C69, 05C85

Keywords: Powers of graphs, intersection graphs, interval graphs, circular-arc graphs, trapezoid graphs, dispersion.

1 Introduction

The subject of this paper is dispersion problems in certain geometric intersection graphs. The dispersion problem is to select a given number of vertices in a graph so as to maximize the minimum distance between them. The problem is dual to the *maximum k-independent set problem*, which is that of finding a maximum collection of vertices whose inter-vertex distance is greater than a given bound k. That problem in turn is equivalent to the well-studied maximum independent set problem on the power graph G^k of the original graph. Thus, in order to give efficient dispersion algorithms, we are led to study efficient methods for computing k-independent set and methods for constructing power graphs, as well as to study structural properties of these powers.

In this article we present efficient methods to compute the powers of some geometric intersection graphs, including interval graphs, circular-arc graphs, and m-trapezoid graphs. The containment of graph classes under study is as follows. Proper interval graphs, interval graphs, trapezoid graphs, m-trapezoid graphs, cocomparability graphs, all form one proper containment chain. More precisely, m-trapezoid graphs are interval graphs when $m = 0$, trapezoid graphs when $m \leq 1$,

[*]Computer Science Program, School of Computing, Armstrong Atlantic State University, Savannah, Georgia 31419-1997. `geir@armstrong.edu`

[†]Department of Computer Science, Chalmers University, 41296 Göteborg, Sweden. `ptr@cs.chalmers.se`

[‡]Department of Computer Science, University of Iceland, IS-107 Reykjavík, Iceland, and Iceland Genomics Corp. (UVS), Snorrabraut 60, IS-105 Reykjavik. `mmh@hi.is`

and cocomparability graphs for any m. Similarly, proper circular-arc graphs, circular-arc graphs, circular m-trapezoid graphs form another chain, and they also properly contain the respective non-circular class.

These and various other classes of graphs have been shown to be closed under taking power. This includes cocomparability graphs [6], strongly chordal graphs [17, 5], interval graphs [16], m-trapezoid and trapezoid graphs [8], circular-arc graphs [9]. For all of these classes, except circular-arc graphs, the following stronger fact is known that whenever G^k is in the class, then so is G^{k+1}.

Generally, these proofs of containment do not immediately yield efficient algorithms. This led us to derive an efficient method for computing the power graph G^k of an interval graph G in time $O(n \log k)$ [2]. We both improve and generalize this result in the present article.

To this end we define the *pseudo product* $(G, G') \to G * G'$ for two general graphs on the same set of vertices. This composition turns out to be commutative, but not associative in general. However, when we restrict to the class of various powers of a fixed graph, then the pseudo product is also associative, in fact if s and t are positive integers then $G^s * G^t = G^{s+t}$.

Our fast power computations lead to efficient algorithms for the *dispersion* problem in m-trapezoid, trapezoid, interval, and circular-arc graphs. The problem is defined as follows:

Dispersion
Given: Graph G and integer q.
Find: A set of at least q vertices, such that the minimum distance between the chosen vertices is maximized.

The generalized version is:

Weighted Dispersion
Given: Graph G with vertex weights $w : V(G) \to \mathbf{R}$, and real number q.
Find: A set of vertices with total weight at least q, with the minimum distance between the chosen vertices at maximum.

Dispersion is NP-hard for general graphs, since Maximum Independent Set is reducible to it, while it can be approximated in polynomial time within factor 2, see e.g. [18]. *Dispersion* restricted to $q = 2$ is nothing else than the problem of computing the diameter of G.

There is an obvious relationship between dispersion, k-independent sets and k-th powers of graphs: Let $\mathcal{G}$ be some graph class. If maximum independent sets can be found in polynomial time in the class of powers of graphs from $\mathcal{G}$ then *dispersion* in $\mathcal{G}$ can be solved in polynomial time, too. If $\mathcal{G}$ is closed under taking power, we merely have to compute maximum independent sets in the k-th power for several k. However the straightforward use of this observation yields $\Omega(n^2)$ time dispersion algorithms in such classes $\mathcal{G}$. For faster algorithms we have to avoid explicit insertion of edges when considering the k-th powers of G, such that fast power computations is exactly what we need here.

We want to make clear that we always assume that our geometric intersection graphs are given by their geometric representations, rather than by their edge lists. Thus they are described in $O(n)$ space, by the extreme points of geometric sets representing the vertices. Without such a representation, it is impossible to achieve equally fast algorithms for the problems we study. (For m-trapezoid graphs, already the recognition problem is NP-hard [8, 20].)

We further assume everywhere in this paper that the endpoints of the intervals/arcs/trapezoids are given in a *sorted* list. This is common in the literature on algorithms in these classes and allows us to derive $O(n \log n)$ algorithms for many of these problems. Solving *dispersion* on interval graphs is at least as hard as sorting, thus requiring $\Omega(n \log n)$ time. We note that there are natural cases where faster sorting is possible. For example, if all endpoints are integers within a range of s, then intervals can be bucket-sorted in $O(n+s)$ time, which is $O(n)$ if $s = O(n)$.

Only a few subquadratic dispersion algorithms have been provided before: Dispersion is solvable in $O(n)$ time for trees [3], while weighted dispersion in $O(n \log n)$ time for paths [18], and in $O(n \log^4 n)$ time for trees [4].

1.1 Outline of paper

In Section 2 we define the pseudo product and apply it to obtain a fast way to compute arbitrary powers of m-trapezoid graphs. The purpose here is to present the key elements of the pseudo product. We show that the pseudo product of two m-trapezoid graphs, that fulfill certain conditions, which are automatically satisfied for interval graphs, is also an m-trapezoid graph.

In Section 3 we use the pseudo product to obtain an $O(n \log k)$ algorithm to compute the k-th power of an m-trapezoid graph on n vertices. Here we view m as a fixed integer that merely defines our class of m-trapezoid graphs.

In Section 4 we present an $O(n)$ algorithm to compute the k-th power of interval (the case $m = 0$) and circular-arc graphs.

Finally, in Section 5 we use the results in previous sections to solve efficiently the k-Independent Set, or k-IS problem, and the *dispersion* and *weighted dispersion* problems for interval graphs, circular-arc graphs and m-trapezoid graphs.

We use the characterization of dispersion in terms of power graphs: The maximum dispersion is the largest k such that $\alpha(G^k, w) \geq q$, where α denotes the (weighted) independence number of G.

The following table summarizes the asymptotic complexity of the three problems that we consider on the three geometric classes of graphs, provided that a representation as stated above is given. In the table, n denotes the number of vertices in G, and lg refers to the usual base-2 logarithm. We assume m is a constant.

Class	**Power**	**k-IS**	**Dispersion**	**W-Dispersion**
Interval	n	n	n	$n \lg k$
Circular-arc	n	n	n	
Trapezoid	$n \lg k$	$n(\lg k + \lg \lg n)$	$n(\lg k + \lg \lg n)$	$n \lg k \lg \lg n$
m-Trapezoid	$n \lg k$	$n(\lg k + (\lg \lg n)^m)$	$n(\lg k + (\lg \lg n)^m)$	$n \lg k (\lg \lg n)^m$

In summary, we obtain linear time algorithms for computing arbitrary k-power of interval and circular-arc graphs, and $O(n \log k)$ algorithm for m-trapezoid graphs (m fixed). The computation of k-IS is equivalent to the power computation, plus the cost of an independent set computation. Dispersion is obtained by a constant number of k-IS computation in all these graph classes, which weighted dispersion requires $\lg k$ computations of weighted independent sets.

1.2 Notation

We will denote the positive integers $\{1, 2, 3, \ldots\}$ by $\mathbf{N}$, the nonnegative integers $\{0, 1, 2, \ldots\}$ by $\mathbf{N}_0$, the set of real numbers by $\mathbf{R}$, the Cartesian product $\mathbf{R} \times \mathbf{R}$ by $\mathbf{R}^2$, and the set of the closed interval $\{x : a \leq x \leq b\}$ by $[a; b]$. All graphs we consider are simple unless otherwise stated. For a graph G the set of its vertices will be denoted by $V(G)$, and the set of its edges by $E(G)$. The open neighborhood of a vertex v in G, that is, the set of neighbors of v not including v, will be denoted by $N_G(v)$. The closed neighborhood of a vertex v in G, that is, the set of neighbors of v, including v itself, will be denoted by $N_G[v]$. For two vertices u and v in G, the distance between them will be denoted by $d_G(u, v)$ or simply by $d(u, v)$ when unambiguous. We use notation compatible with [19].

Recall for a graph G and an integer k, the *k-th power* of G is the graph G^k on the same set of vertices as G, and where every pair of vertices of distance k or less in G are connected by an edge. Also, a graph G is called the *intersection graph* of a collection of sets $\{S_1, \ldots, S_n\}$ if $V(G) = \{v_1, \ldots, v_n\}$ and $\{v_i, v_j\} \in E(G) \Leftrightarrow S_i \cap S_j \neq \emptyset$, for all distinct $i, j \in \{1, \ldots, n\}$. Note that when G is represented by $\{S_1, \ldots, S_n\}$, then $d(S_i, S_j)$ is just the distance $d_G(v_i, v_j)$ between v_i and v_j in the intersection graph G.

2 Powers of m-Trapezoid Graphs

In this section we discuss a way to calculate the k-th power of an m-trapezoid graphs efficiently, and, as a special case, an interval or a circular-arc graph by means of the pseudo product which we define here below. Recall that any power of an interval graph (resp. circular-arc graph) is again an interval graph (resp. circular-arc graph) [16, 17, 2].

We start with the definition of the pseudo product for general graphs, a composition that captivates the essence of powers of graphs, as we will see in Observation 2.2.

Definition 2.1 *Let G and G' be simple graphs on the same set of vertices $V(G) = V(G') = V$, where $|V| = n \geq 1$. Define the* pseudo product *of G and G' to be the simple graph $G * G'$ on the vertex set V with edge set $E(G * G') = E(G) \cup E(G') \cup E^*$ where*

$$\begin{aligned} E^* \;=\; & \{\{u, v\} : \exists w \in V : \{u, w\} \in E(G), \{w, v\} \in E(G'), \\ & \textit{and } \exists w' \in V : \{u, w'\} \in E(G'), \{w', v\} \in E(G)\}. \end{aligned}$$

The word *"pseudo"* in our definition of the pseudo product of G and G', is fitting, since if we view $*$ as a group-like operation among all the simple graphs on V, then it is *not* an associative operation. In other words, the formula $(G * G') * G'' = G * (G' * G'')$ does not hold in general. We have however the following, which is a direct consequence of Definition 2.1.

Observation 2.2 *For a simple graph G and nonnegative integers s and t, we have $G^s * G^t = G^{s+t}$. In particular, the pseudo product is an associative operation on the set $\{G^k : k \in \{0, 1, 2, \ldots\}\}$ for any fixed simple graph G.*

Assume that for each $l \in \{0, 1, \ldots, m\}$ we have two real numbers, a_l and b_l where $a_l < b_l$. As defined in [8], an *m-trapezoid* T is simply the closed interior of the polygon formed by the points $S = \{(a_l, l), (b_l, l) : l \in \{0, 1, \ldots, m\}\} \subseteq \mathbf{R}^2$. This means, the left side of the polygon is the chain of straight-line segments connecting (a_l, l) and $(a_{l+1}, l + 1)$, where l ranges from 0 to $m - 1$, and similarly for the right side and numbers b_l. The lower and upper boundary of T is the horizontal line with ordinate 0 and m, respectively. This we denote by $T = \text{inter}(S)$. The horizontal lines with ordinates $l \in \{0, 1, \ldots, m\}$ will be called *lanes*.

An *m-trapezoid graph* is a graph G on n vertices $\{v_1, \ldots, v_n\}$ which is an intersection graph of a set $\{T_1, \ldots, T_n\}$ of m-trapezoids, that is, $\{v_i, v_j\} \in E(G) \Leftrightarrow T_i \cap T_j \neq \emptyset$. Let G be an m-trapezoid graph represented by $\{T_1, \ldots, T_n\}$ where each

$$T_i = \text{inter}(\{(a_{li}, l), (b_{li}, l) : l \in \{0, 1, \ldots, m\}\}). \tag{1}$$

We will write $\tilde{a}_{li}$ (resp. $\tilde{b}_{li}$) for the point (a_{li}, l) (resp. (b_{li}, l)) in $\mathbf{R}^2$. We say that the left sides of T_i and T_j *cross* (or synonymously, *intersect*) if there are distinct indices $p, q \in \{0, 1, \ldots, m\}$ such that $a_{pi} < a_{pj}$ and $a_{qi} > a_{qj}$.

If G and G' are two m-trapezoid graphs, both on n vertices, represented by sets of m-trapezoids $\mathcal{T} = \{T_1, \ldots, T_n\}$ and $\mathcal{T}' = \{T_1', \ldots, T_n'\}$ respectively, where the left side of T_i and the left side of T_i' coincide, that is $T_i = \mathrm{inter}(\{\tilde{a}_{li}, \tilde{b}_{li} : l \in \{0, 1, \ldots, m\}\})$ and $T_i' = \mathrm{inter}(\{\tilde{a}_{li}, \tilde{b}_{li}' : l \in \{0, 1, \ldots, m\}\})$, for all $i \in \{1, \ldots, n\}$, then we will say that $\mathcal{T}$ and $\mathcal{T}'$ are *left-coincidal*.

Recall that $d(T_i, T_\beta)$ and $d(T_i', T_\alpha')$ denote the distances between corresponding vertices in G and G' respectively. We now put $b_{li}^* = \max_{d(T_i', T_\alpha') \leq 1}\{b_{l\alpha}\}$ and ${b_{li}^*}' = \max_{d(T_i, T_\beta) \leq 1}\{b_{l\beta}'\}$ for each $i \in \{1, \ldots, n\}$ and $l \in \{0, 1, \ldots, m\}$. With this setup we have the following theorem.

Theorem 2.3 *For an integer $m \geq 0$ let G and G' be two m-trapezoid graphs on the same number of vertices, with left-coincidal representations $\{T_1, \ldots, T_n\}$ and $\{T_1', \ldots, T_n'\}$ respectively. Assume further that for each i we have either $b_{li}^* \leq {b_{li}^*}'$ for all $l \in \{0, 1, \ldots, m\}$, or $b_{li}^* \geq {b_{li}^*}'$ for all $l \in \{0, 1, \ldots, m\}$. In this case, the pseudo product $G * G'$ is also an m-trapezoid graph with an m-trapezoid representation $\mathcal{T}^* = \{T_1^*, \ldots, T_n^*\}$, which is left-coincidal with both $\mathcal{T}$ and $\mathcal{T}'$, and where the right sides of each T_i^* are determined by b_{li}^{**} where $b_{li}^{**} = \max\{b_{li}, b_{li}', \min\{b_{li}^*, {b_{li}^*}'\}\}$, for all $i \in \{1, \ldots, n\}$ and $l \in \{0, 1, \ldots, m\}$.*

REMARKS: (i) In the case $m = 0$ then Theorem 2.3 reduces precisely to the statement for left-coincidal interval graphs, for which the additional conditions of the above theorem are automatically satisfied. That again, can be applied to circular arc graphs in a natural fashion, where we extend the arcs in a clockwise direction. (ii) In the case where we are considering the pseudo product $G^s * G^t$ of two powers of the same graph m-trapezoid graph G, then the condition in the above theorem is satisfied. In fact, we will see in Observation 2.4 here below, that $b_{li}^* = {b_{li}^*}'$ holds then for each l and i.

Proof. To prove Theorem 2.3 we need to show

$$\{v_i, v_j\} \in E(G * G') \Leftrightarrow T_i^* \cap T_j^* \neq \emptyset. \tag{2}$$

In the case where the left sides of T_i and T_j cross, there is nothing to prove, since all the sets $T_i \cap T_j$, $T_i' \cap T_j'$ and $T_i^* \cap T_j^*$ are then nonempty. Hence we can assume throughout the proof that the left sides of T_i and T_j do not cross, say $a_{li} < a_{lj}$ for all $l \in \{0, 1, \ldots, m\}$. Furthermore, if $\{v_i, v_j\}$ is either in $E(G)$ or in $E(G')$ then $T_i^* \cap T_j^* \neq \emptyset$ by definition. Hence, we can further assume

$$a_{li} < b_{li} < a_{lj} \text{ and } a_{li} < b_{li}' < a_{lj} \tag{3}$$

to hold for all $l \in \{0, 1, \ldots, m\}$ throughout the proof.

To prove the "$\Rightarrow$"-direction of (2) assume that $\{v_i, v_j\} \in E(G * G')$. By definition of $E(G * G')$, there are v_α and v_β such that $\{v_\alpha, v_j\}, \{v_i, v_\beta\} \in E(G)$ and $\{v_i, v_\alpha\}, \{v_\beta, v_j\} \in E(G')$. This, together with (3), means that there are indices $p, q \in \{0, 1, \ldots, m\}$ such that $T_i' \cap T_\alpha' \neq \emptyset$, $a_{pj} < b_{p\alpha}$, $T_i \cap T_\beta \neq \emptyset$ and $a_{qj} < b_{q\beta}'$.

If $b_{pi}^* \leq {b_{pi}^*}'$ then we have $b_{pi}^{**} = b_{pi}^* \geq b_{p\alpha} > a_{pj}$, and hence $T_i^* \cap T_j^* \neq \emptyset$.

If however $b_{pi}^* \geq {b_{pi}^*}'$ then by assumption in the theorem we have that $b_{qi}^* \geq {b_{qi}^*}'$ also holds and hence we have $b_{qi}^{**} = {b_{qi}^*}' \geq b_{q\beta}' > a_{qj}$, which implies that $T_i^* \cap T_j^* \neq \emptyset$, thereby completing the "$\Rightarrow$"-part of (2).

Now for the other part, assume $T_i^* \cap T_j^* \neq \emptyset$. By (3) this means that there is an $l \in \{0, 1, \ldots, m\}$ such that $b_{li}^{**} > a_{lj}$. By definition of b_{li}^* and $b_{li}^{*\prime}$ we can find α and β such that $T_i' \cap T_\alpha' \neq \emptyset$, $b_{l\alpha} = b_{li}^*$, $T_i \cap T_\beta \neq \emptyset$ and $b'_{l\beta} = b_{li}^{*\prime}$.

Since now both $b_{l\alpha}$ and $b'_{l\beta}$ are greater or equal to b_{li}^{**} we have $T_i' \cap T_\alpha' \neq \emptyset$, $b_{l\alpha} > a_{lj}$, $T_i \cap T_\beta \neq \emptyset$ and $b'_{l\beta} > a_{lj}$. By our assumption in (3) we have $T_i' \cap T_\alpha' \neq \emptyset$, $T_\alpha \cap T_j \neq \emptyset$, $T_i \cap T_\beta \neq \emptyset$ and $T_\beta' \cap T_j' \neq \emptyset$, which implies that $\{v_i, v_j\} \in E(G * G')$. This proves the "$\Leftarrow$"-part of (2), thereby completing our proof. $\square$

Let us now consider the more special cases of a pseudo product of two powers of a fixed m-trapezoid graph G. By Observation 2.2 we have that $G^s * G^t = G^{s+t}$, and hence Theorem 2.3 gives us a way to obtain the representation of G^{s+t} *directly* from the representations of G^s and G^t. In [1] it is shown that if G is an m-trapezoid graph represented by a set $\{T_1, \ldots, T_n\}$ of m-trapezoids (as in (1)) and $k \geq 1$ is an integer, then G^k is represented by m-trapezoids $\{T_1(k), \ldots, T_n(k)\}$ which are given by

$$T_i(k) = \text{inter}(\{\tilde{a}_{li}, \tilde{b}_{li}(k) : l \in \{0, \ldots, m\}\}), \tag{4}$$

where $\tilde{b}_{li}(k) = \max_{d(T_\alpha, T_i) \leq k-1}\{b_{l\alpha}\}$. Although (4) provides a formula for the representation of G^k from the representation of G, this is not computationally feasible, since the definition of $\tilde{b}_{li}(k)$ is complex from a computational point of view. We are, however, able to compute precisely this representation much more efficiently, by applying the pseudo product.

Let $s, t \geq 1$ be integers, and G a fixed m-trapezoid graph. If G^s and G^t have $\{T_1(s), \ldots, T_n(s)\}$ and $\{T_1(t), \ldots, T_n(t)\}$, respectively, as their representations, then we can get the representation of the pseudo product $G^{s+t} = G^s * G^t$, given in Theorem 2.3, by calculating b_{li}^* explicitly and get

$$b_{li}^* = \max_{d(T_i(t), T_\alpha(t)) \leq 1}\{b_{l\alpha}(s)\} = \max_{d(T_i, T_\alpha) \leq t}\left\{\max_{d(T_\alpha, T_\beta) \leq s-1}\{b_{l\beta}\}\right\} = \max_{d(T_i, T_\beta) \leq s+t-1}\{b_{l\beta}\} = b_{li}(s+t).$$

In the same way we get that $b_{li}^{*\prime} = b_{li}(s+t)$, and hence we have in the case for pseudo product of G^s and G^t that $b_{li}^{**} = \max\{b_{li}, b'_{li}, \min\{b_{li}^*, b_{li}^{*\prime}\}\} = \max\{b_{li}(s), b_{li}(t), b_{li}(s+t)\} = b_{li}(s+t)$. Hence, the representation of G^{s+t}, which we obtain by repeated use of the pseudo product is $\{T_1(s+t), \ldots, T_n(s+t)\}$, which is identical with the presentation given in (4).

We see from the above that Theorem 2.3 applies when considering various powers of a fixed graph G, as the following observation shows.

Observation 2.4 *If both G and G' are powers of the same m-trapezoid graph on n vertices, then $b_{li}^* = b_{li}^{*\prime}$ holds for all $l \in \{0, 1, \ldots, m\}$ and $i \in \{1, \ldots, n\}$.*

Recall that although the pseudo product is not generally an associative operation, it is associative on the set of powers of a fixed graph G. This means that $(G^r * G^s) * G^t = G^r * (G^s * G^t)$, and therefore the notion $G^{r_1} * \cdots * G^{r_k}$ (k times) is perfectly sensible. Hence, we have the following corollary.

Corollary 2.5 *Let $k = \sum_{i=1}^{s} 2^{t_i}$ be the binary representation of k. For an m-trapezoid graph G represented by $\{T_1, \ldots, T_n\}$, the representation for $G^k = G^{2^{t_1}} * \cdots * G^{2^{t_s}}$ from Theorem 2.3, is $\{T_1(k), \ldots, T_n(k)\}$, the representation of G^k given in (4).*

As we will see in Section 3, this yields an $O(n \log k)$ algorithm to calculate the presentation of G^k, if $k \geq 1$ is an integer and G is a fixed m-trapezoid graph. To compute the representation of each G^{2^t} we need to calculate the representation repeatedly no more than t times, using $G^{2^i} * G^{2^i} = G^{2^{i+1}}$ for i from 0 to $t-1$. This will be discussed more precisely in the Section 3.

3 Computing Powers of m-Trapezoid Graphs

In this section we implement the theory of Section 2, to obtain a fast method of computing the representation of G^k, where $k \in \mathbf{N}$ and G is an m-trapezoid graph with a given representation. Let $\mathcal{T} = \{T_1, \ldots, T_n\}$ and $\mathcal{T}' = \{T_1', \ldots, T_n'\}$ be two such left-coincidal representations for G and G' respectively, as given by (1). Here we shall assume that for each lane $l \in \{0, 1, \ldots, m\}$ the endpoints a_{li} and b_{li}, where $i = 1, 2, \ldots, n$, have been translated to the set $\{1, 2, \ldots, 2n\}$.

We want to compute the pseudo product $G * G'$ of two powers of the same m-trapezoid graph, whose right endpoints are denoted by b_{li}^{**} as in Theorem 2.3. We shall compute a series of $m+1$ by $2n$ matrices A_p, where for $p, l = 0, 1, \ldots, m$ and $q = 1, \ldots, 2n$, the entry $A_p[l, q]$ equals the rightmost coordinate along lane p among trapezoids T_α' in G' with $a_{l\alpha} \leq q$. Each trapezoid T_α that intersects T_i must satisfy $a_{l\alpha} < b_{li}$, for some l. Thus, given the values of A_p, we compute b_{pi}^{**} from Theorem 2.3 by setting $b_{pi}^{**} = \max\{{b_{pi}^*}', b_{pi}\}$ where ${b_{pi}^*}' \leftarrow \max_{l \in \{0, \ldots, m\}} A_p[l, b_{li}]$, which takes $m+1$ operations. To compute A_p, we first initialize with zero and insert values for each trapezoid:

```
for each $\alpha \in \{1, \ldots, n\}$ and $l \in \{0, \ldots, m\}$ do
  $A_p[l, a_{l\alpha}] \leftarrow b_{l\alpha}'$
```

This, together with the zero initialization, uses a total of $2(m+1)n$ operations. We can then complete it in one pass from left to right, using the trivial observations that coordinates to the left of $q-1$ are also to the left of q. That is, we form a prefix maxima of A_p by $A_p[l, q] \leftarrow \max(A_p[l, q], A_p[l, q-1])$. This second loop also uses $2(m+1)n$ operations as l goes through $\{0, \ldots, m\}$ and q through $\{1, \ldots, 2n\}$, so we perform $4(m+1)n$ operations to compute each matrix A_p. Thus, the computation of A_p where $p \in \{0, \ldots, m\}$ takes a total of $4(m+1)^2 n$ operations. Hence, by Observation 2.2 and Theorem 2.3 we have the following.

Theorem 3.1 *Given powers G^s and G^t of an m-trapezoid graph G, the power graph G^{s+t} can be computed in $O(m^2 n)$ time.*

The given algorithm is easily parallelizable, as the second step is a standard parallel prefix operation. This gives an optimal $O(\log n)$ time $O(n)$ work algorithm on the EREW model [12].

This generalizes the algorithm given in [2] for interval graphs. The same construction holds also for circular-arc and circular-trapezoid graphs, where the max operator is viewed in modular arithmetic.

If $k \in \mathbf{N}$ and $k = \sum_{i=1}^{s} 2^{t_i}$ is its binary representation, then $G^k = G^{2^{t_1}} * G^{2^{t_2}} * \cdots * G^{2^{t_s}}$. Using fast multiplication G^k can be computed in at most $t_s + s - 1 \leq 2 \log k - 1$ pseudo products. By Theorem 3.1 and Corollary 2.5 we have the following.

Corollary 3.2 *The representation of G^k where G is an m-trapezoid graph, can be computed in $O(m^2 n \log k)$ time.*

4 Computing Powers of Interval Graphs and Circular-Arc Graphs

Let G be an interval graph on n vertices, represented by a set $\mathcal{I}_G$ of n intervals. We may assume that all the intervals have their $2n$ endpoints distinct among the numbers $\{1, 2, \ldots, 2n\}$. For each interval $I \in \mathcal{I}_G$ there is a unique interval $I' \in \mathcal{I}_G$ with the rightmost endpoint of any interval which intersects I. This yields a mapping $f : \mathcal{I}_G \to \mathcal{I}_G$, defined by $f(I) = I'$. This mapping is acyclic and thus induces a directed forest $\vec{F}_G$ on $\mathcal{I}_G$ (which is a directed tree if G is connected), with a directed edge from each $I \in \mathcal{I}_G$ to $f(I)$. Note that the root of any tree of $\vec{F}_G$ will point to itself.

The representation of G^k can now be obtained quickly: For each interval $I = [a_I; b_I] \in \mathcal{I}_G$ we obtain an interval $I(k) = [a_{I(k)}; b_{I(k)}]$ where $a_{I(k)} = a_I$ and $b_{I(k)} = b_{I_k}$, where I_k is the k-th ancestor of I in the tree of the above forest $\vec{F}_G$ (where the parent of the root is the root itself).

This is computed in a single traversal of the tree. As we traverse the tree, we keep the nodes on the path from the root to the current node on an indexable stack. This is a data structure supporting all the stack operation, as well as constant-time indexing of elements in the stack. Namely, we use an array X, and as we traverse a node I at depth d_I, we store it in $X[d]$. Then, the root is stored in $X[0]$, and the k-th ancestor of I is stored at $X[d_I - k]$, for $k \leq d$. We obtain I_k simply as $X(\max\{d_I - k, 0\})$, and for each node I, we output new interval $I(k)$ obtained by $I(k) = [a_I; b_{X(\max\{d_I - k, 0\})}]$.

When G is a circular-arc graph, mapping f is a *pseudo forest* (or a *pseudo tree* if G is connected), i.e. each component contains exactly one cycle, as the number of edges equals the number of vertices. We must now treat nodes at depth less than k differently. Select any node R on the sole cycle to be a "root", and set its depth to be 0. Extend the array X to negative indices, and let $X[-1] = f(R)$ and generally $X[-i] = f^{(i)}(R)$. We now traverse the tree rooted at R, as before, and set I_k to be $X[d_I - k]$ for each node I of depth d_I from R. Otherwise, the process is identical. We have therefore the following.

Theorem 4.1 *Let G be a circular-arc graph with a given representation. For any k, we can compute the representation of the power graph G^k in $O(n)$ time.*

5 k-Independent Set and Dispersion Algorithms

By computing the k-th power of a graph, we reduce the problem of finding a k-Independent Set or a k-IS for short, (resp. k-Weighted Independent Set or k-WIS for short), to the Maximum Independent Set or the MIS problem (resp. the Maximum Weighted Independent Set or the MWIS problem for short) on the corresponding class of graphs, within an additive factor of $O(n \log k)$. The following is known about those problems.

Fact 5.1 *MWIS can be computed in $O(n)$ time for interval graphs, and in $O(n \log\log n)$ time for trapezoid graphs. MIS can be computed in $O(n)$ time for circular-arc graphs.*

For the MWIS result on interval graphs see [13]. The MIS result on circular-arc graphs has been rediscovered several times [11, 14, 15, 21]. Felsner et al. [7] showed that MWIS of trapezoid graphs can be computed in $O(n \log n)$ time, when the representation is given. Their algorithm uses a data structure supporting Insert, Delete, Predecessor, and Successor operations of endpoints, and the complexity is equal to the complexity of n of each of these operations. Under our assumption that the list of endpoints is given sorted (which could be obtained by $O(n \log n)$ preprocessing), we may assume that all endpoints are integers from 1 to $2n$. Then, the data structure of van Emde Boas supports these operations in $\log\log n$ steps. Hence, we can compute the MWIS of trapezoid graphs in $O(n \log\log n)$ time. Thus we obtain:

Theorem 5.2 *k-WIS can be found in $O(n)$ time for interval graphs, and in $O(n(\log\log n + \log k))$ time for trapezoid graphs. k-IS can be found in $O(n)$ time for circular-arc graphs.*

Proof. By Theorem 4.1 and Corollary 3.2 we can compute the k-th power of an interval graph and of a trapezoid graph in $O(n)$ time and $O(n \log k)$ time, respectively, and MWIS on the k-th power is equivalent to k-MWIS. This and Fact 5.1 gives the results for these classes. The bound on k-IS for circular-arc graphs follows similarly. □

5.1 Dispersion via binary search for k

The following algorithm simply looks for the largest power G^k of G that still admits an independent set of weight at least q, by repeated doubling followed by binary search. This k is, of course, the solution to the dispersion problem.

```
WDisp($G$,$q$)
  $d \leftarrow 1$, $G^1 \leftarrow G$
  while (MWIS($G^d$) $\geq q$)
    $G^{2d} \leftarrow G^d * G^d$
    $d \leftarrow 2d$
  $k \leftarrow 0$
  $H \leftarrow G^{d/2}$
  for $(i = d/2; i \geq 1; i = i/2)$ do
    { Invariant: $H = G^k$, MWIS($H$) $\geq q >$ MWIS($H * G^{2i}$) }
    $H' \leftarrow H * G^i$
    if (MWIS($H'$) $\geq q$)
      $H \leftarrow H'$
      $k \leftarrow k + i$
  output $H$, $k$
```

The time complexity is dominated by the number of computations of maximum (weighted) independent sets. Here, it is at most $2 \log k$. Hence, we have the following.

Theorem 5.3 *Weighted Dispersion can be solved in $O(n \log k)$ time on interval graphs and $O(n \log k \log \log n)$ time on trapezoid graphs.*

A lower bound of Fredman [10] for maximum increasing subsequences yields a $\Omega(n \log n)$ lower bound for finding unweighted IS in permutation graphs, a subclass of trapezoid graphs. Thus, in the current setup, the dependence on the RAM model is necessary.

5.2 Unweighted dispersion of geometric graphs

For convenience let k-IS(G) denote the size of a minimum k-independent set in graph G. Recall the notion of a lane from Section 2.

Lemma 5.4 *Let G be an m-trapezoid graph, and d be the distance between trapezoids that are furthest in each direction along some lane. Then $\lfloor d/(k+1) \rfloor + 1 \leq k$-IS$(G) \leq \lfloor d/(k-1) \rfloor + 1$.*

Proof. Let u (resp. u') be the trapezoid furthest to the left (resp. right) along a given lane, and let $P = \langle u = u_0, u_1, u_2, \ldots, u_d = u' \rangle$ be a shortest path between u and u'. The set $\{u_{i(k+1)} | i = 0, 1, 2, \ldots, \lfloor d/(k+1) \rfloor\}$ then forms a k-IS, thus showing the first part of the claim.

On the other hand, suppose $\{v_1, v_2, \ldots, v_t\}$ is a k-IS. For each v_i, the trapezoid representing v_i intersects some trapezoid representing a node u_{x_i} in the abovementioned path P. Since v_i and v_{i+1} are of distance at least $k+1$, we have that $x_{i+1} \geq x_i + k - 1$. It follows by induction that $d \geq x_t \geq x_1 + (t-1)(k-1) \geq (t-1)(k-1)$. Thus, $t \leq \lfloor d/(k-1) \rfloor + 1$, yielding the second part of the claim. □

Theorem 5.5 *Let G be an m-trapezoid graph, d be the distance between vertices respectively with the leftmost and rightmost endpoint along some lane, and K be $\lfloor d/(q-1) \rfloor$. Then, the optimum dispersion of G is one of the three values $\{K-1, K, K+1\}$.*

Proof. Let OPT be the optimum dispersion of G, i.e. the largest value t such that t-IS$(G) \geq q$. By the definition of K, $K(q-1) \leq d$, and thus by Lemma 5.4, $q \leq \lfloor d/K \rfloor + 1 \leq (K-1)$-IS$(G)$. That is, $OPT \geq K-1$. By the definition of K, $d/(K+1) < q$, so K is the largest number such that $\lfloor \frac{d}{K} \rfloor \geq q-1$. By Lemma 5.4, $q \leq OPT$-IS$(G) \leq \lfloor d/(OPT-1) \rfloor + 1$. Thus, $OPT \leq K+1$. □

This can be extended to circular-arc graphs. A greedy covering of the circle is defined as follows: Start with an arbitrary arc I, add $f(I)$ and let $I := f(I)$, until the whole circle is covered. (Do not put the initial I in the set.) Such a covering exists unless the graph is actually an interval graph. Note that a greedy covering is a chordless cycle in the graph and can be computed in $O(n)$ time. The following result holds by an argument similar to Lemma 5.4.

Lemma 5.6 *Let c be the size of a greedy covering of a circular-arc graph G. Then $\lfloor c/(k+1) \rfloor \leq$ k-IS$(G) \leq \lfloor c/(k-1) \rfloor$.*

This can be further extended to circular m-trapezoid graphs. For this, we need a covering of minimum diameter. Start with an arbitrary trapezoid I, we compute in linear time by dynamic programming an array $d[i, j]$ containing the trapezoid that extend furthest clockwise on the cycle along lane j among those of distance i from I. We stop when we have found one that intersects a trapezoid I' in $d[1, j]$, in which case we trace the path backwards to I' omitting I.

Theorem 5.7 *Let c be the size of a greedy covering of a circular m-trapezoid graph G, and let K be $\lfloor c/q \rfloor$. Then, the optimum dispersion of G is one of the three values $\{K-1, K, K+1\}$.*

The proof follows the lines of Theorem 5.5. On circular-arc graphs, we can compute each k-IS in linear time, as mentioned earlier. Thus we finally get the following

Corollary 5.8 *Dispersion has equivalent complexity as k-IS on interval, circular-arc, m-trapezoid, and circular m-trapezoid graphs. In particular, it can be computed in $O(n(\log k + (\log\log n)^m))$ time on m-trapezoid graphs, and in $O(n)$ time on interval and circular-arc graphs.*

5.3 Unweighted dispersion of interval graphs revisited

We show here how our representation of interval and circular-arc graphs yields efficient computation of k-IS for several k.

Lemma 5.9 *Let G be an interval graph with nodes sorted by nondecreasing right endpoints. Then, for any k, k-IS(G) can be found in time $O(n(\log k)/k)$, using $O(n)$ precomputation.*

Proof. Recall the directed forest $\vec{F}_G$ that G induced, as explained in Section 4. Mark the depth of each node in its tree. For a node v of depth $depth(v)$, let $b(v)$ be the smallest bit in $depth(v)$ set to 1. Now, let $anc(v)$ be the ancestor of v of depth $depth(v) - 2^{b(v)}$. It can be computed in a top down traversal of the tree, when placing nodes on the tree on the indexable stack X, since $anc(v) = X(depth(v) - 1)$ when $b(v) = 0$, and $anc(v) = anc(X(depth(v) - 2^{b(v)-1}))$ otherwise. Also mark each node with $par(v)$, the parent of v in the tree.

We additionally compute for each node v, the node u that extends the least to right among those that have a left endpoint to the right of v. Let $next(v)$ be this interval if it exists, and nil otherwise. This completes the precomputation needed.

Given anc, we can compute the k-th ancestor $anc(v; k)$ of a node v, by at most $\log k$ iterations, where in each iteration we follow at most $2 \log k$ links of anc and one par link.

The algorithm is simply:

```
$v \leftarrow$node with leftmost right endpoint
$I \leftarrow \{v\}$
$v \leftarrow next(anc(v; k))$
while $v \neq nil$ do
  $I \leftarrow I \cup v$
  $v \leftarrow next(anc(v; k))$
od
```

By the foregoing argument, each iteration of the loop runs in $O(\log^2 k)$ time. There are at most $diam(G)/k$ iterations. □

Using binary search, we can use the above to obtain an alternative $O(n)$ algorithm for *dispersion* on interval graphs.

Theorem 5.10 *Dispersion on interval graphs can be solved in $O(n)$ time.*

Proof. We can search for the optimal value k^* of k using modified binary search. First, find by linear search the smallest j such that the maximum 2^j-independent set of the graph is too small (i.e. less than q), while the maximum 2^{j-1}-independent set is sufficiently large (i.e. at least q). Then, use binary search on k within the interval $[2^{j-1}, 2^j)$. Namely, this is the same algorithm as $WDisp(G, q)$, except we don't compute the product graphs G^k, but compute the k-IS instead directly. The complexity for the first step is at most

$$\sum_{i=1}^{\infty} O(n(\log 2^i)/2^i) = O(n) \sum_i i/2^i = O(n),$$

and for the second step at most

$$O(jn(\log 2^j)/2^j) = O(nj^2/2^j) = O(n).$$

□

Acknowledgments

Parts of this work were done while Geir was a Visiting Scholar at Los Alamos National Laboratory in Los Alamos, New Mexico, Summer of 2000. He is grateful to Madhav Marathe for his hospitality. We thank Rasmus Pagh for advice.

References

[1] G. Agnarsson. On Powers of some Intersection Graphs, *Congressus Numerantium*, **151**:97–107, (2001).

[2] G. Agnarsson, R. Greenlaw, M. M. Halldórsson. On Powers of Chordal Graphs and Their Colorings, *Congressus Numerantium*, **144**:41–65, (2000).

[3] B. K. Bhattacharya, M. E. Houle. Generalized Maximum Independent Sets for Trees. *Computing - The 3rd Australian Theory Symposium CATS'97.*

[4] B. K. Bhattacharya, M. E. Houle. Generalized Maximum Independent Sets for Trees in Subquadratic Time. *10th Int. Symp. on Algorithms and Computation ISAAC'99*, *LNCS* 1741 (Springer), 435–445.

[5] E. Dahlhaus and P. Duchet. On Strongly Chordal Graphs. *Ars Combinatoria*, **24 B**:23–30, (1987).

[6] P. Damaschke. Distances in Cocomparability Graphs and Their Powers. *Discrete Applied Mathematics*, **35**:67–72, (1992).

[7] S. Felsner, R. Müller and L. Wernisch. Trapezoid Graphs and Generalizations, Geometry and Algorithms. *Discrete Applied Mathematics*, **74**:13–32, (1997).

[8] C. Flotow. On Powers of m-Trapezoid Graphs. *Discrete Applied Mathematics*, **63**:187–192, (1995).

[9] C. Flotow. On Powers of Circular Arc graphs and Proper Circular Arc Graphs. *Discrete Applied Mathematics*, **74**:199–207, (1996).

[10] M. L. Fredman. On Computing the Length of Longest Increasing Subsequences. *Discrete Applied Mathematics*, **21**:35–46, (1988).

[11] W. L. Hsu, K. H. Tsai. Linear-time Algorithms on Circular Arc Graphs. *Information Proc. Letters*, **40**:123–129, (1991).

[12] J. JáJá. An Introduction to Parallel Algorithms. *Addison-Wesley*, 1992.

[13] J. Y. Hsiao, C. Y. Tang, R. S. Chang. An Efficient Algorithm for Finding a Maximum Weight 2-Independent Setw on Interval Graphs. *Information Proc. Letters*, **43**:229–235, (1992).

[14] D. T. Lee, M. Sarrafzadeh, Y. F. Wu. Minimum Cuts for Circular-Arc Graphs. *SIAM J. Computing*, **19**:1041–1050, (1990).

[15] S. Masuda and K. Nakajima. An Optimal Algorithm for Finding a Maximum Independent Set of a Circular-Arc Graph. *SIAM J. Computing*, **17**:219–230, (1988).

[16] A. Raychaudhuri. On Powers of Interval and Unit Interval Graphs. *Congressus Numerantium*, **59**:235–242, (1987).

[17] A. Raychaudhuri. On Powers of Strongly Chordal and Circular Graphs. *Ars Combinatoria*, **34**:147–160, (1992).

[18] D. J. Rosenkrantz, G. K. Tayi, S. S. Ravi. Facility Dispersion Problems Under Capacity and Cost Constraints. *J. of Combinatorial Optimization*, **4**:7–33 (2000).

[19] D. B. West. Introduction to Graph Theory. *Prentice-Hall Inc.*, Upper Saddle River, New Jersey, (1996).

[20] M. Yannakakis. The complexity of the partial order dimension problem. *SIAM J. Alg. Disc. Meth.* **3**:351–358, 1982.

[21] S. Q. Zheng. Maximum Independent Sets of Circular Arc Graphs: Simplified Algorithms and Proofs. *Networks*, **28**:15–19, (1996).

May 23, 2002