Computing the Independence Number of Intersection Graphs

Jacob Fox[*] János Pach[†]

Abstract

Computing the maximum number of disjoint elements in a collection C of geometric objects is a classical problem in computational geometry with applications ranging from frequency assignment in cellular networks to map labeling in computational cartography. The problem is equivalent to finding the independence number, $\alpha(G_C)$, of the intersection graph G_C of C, obtained by connecting two elements of C with an edge if and only if their intersection is nonempty. This is known to be an NP-hard task even for systems of segments in the plane with at most two different slopes. The best known polynomial time approximation algorithm for systems of arbitrary segments is due to Agarwal and Mustafa, and returns in the worst case an $n^{1/2+o(1)}$-approximation for α. Using extensions of the Lipton-Tarjan separator theorem, we improve this result and present, for every $\epsilon > 0$, a polynomial time algorithm for computing $\alpha(G_C)$ with approximation ratio at most n^{ϵ}. In contrast, for general graphs, for any $\epsilon > 0$ it is NP-hard to approximate the independence number within a factor of $n^{1-\epsilon}$. We also give a subexponential time exact algorithm for computing the independence number of intersection graphs of arcwise connected sets in the plane.

1 Introduction

An *independent set* of a graph $G = (V(G), E(G))$ is a subset of the vertex set of G that contains no pair of adjacent vertices. The *independence number* $\alpha(G)$ of G is the size of the largest independent set in G. Determining or estimating $\alpha(G)$ is a fundamental problem in theoretical computer science, which is known to be NP-hard [19]. The fastest known algorithm for determining $\alpha(G)$ runs in exponential time in $n = |V(G)|$. The approximation ratio of the best known polynomial time approximation algorithm is $n/\text{polylog}\ n$. The reason for this disappointingly weak performance is that approximating $\alpha(G)$ is extremely hard. Building on earlier works [4, 5, 10, 16], Zuckerman [32] proved that, for any $\epsilon > 0$, it is impossible to approximate in polynomial time the independence number of G within a factor of $n^{1-\epsilon}$, provided that P $\neq$ NP.

Computing the maximum number of disjoint elements in a collection C of geometric objects belongs to the oldest problems in computational geometry, with many applications, including frequency assignment in cellular networks [8, 9, 24], map labeling in computational cartography [1, 11, 31], interval scheduling in manufacturing [30], and chip manufacturing [17]. The question is equivalent to computing the independence number of the *intersection graph* G_C of C, defined as a graph on the vertex set $V(G_C) = C$, in which and two vertices are adjacent if and only if the corresponding elements of C have nonempty intersection. For collections of segments in the plane, this problem was already addressed in the first monograph devoted to computational geometry [27]. Determining $\alpha(G_C)$ is NP-hard even if C is a collection of segments in the plane lying in at most two directions [21] (see also [18, 25]). On the other hand, Agarwal and Mustafa [2] designed a $O(n^3)$-time algorithm which, given any collection C of n segments in the plane with $\alpha(G_C) = \alpha$, finds an independent set of size $(\alpha/\log(2n/\alpha))^{1/2}$. Expressed as a function of n, the approximation ratio of this algorithm is $n^{1/2+o(1)}$ in the worst case.

The aim of this note is to give better exact and approximation algorithms for this problem. Our results can be extended to intersection graphs of curves. A *curve* is a subset of the plane $\mathbf{R}^2$ which is homeomorphic to the unit interval $[0, 1]$. The intersection graph of a collection of curves is called a *string graph.* In fact, it is easy to show that the intersection graph of any collection of *arcwise connected* sets in the plane is a string graph. String graphs have been intensely studied both for practical applications and for theoretical interest. Benzer [7] was the first to introduce these graphs in 1959, while exploring the topology of genetic structures. In 1966, interested in electrical networks realizable by printed circuits, Sinden [29] investigated the same graphs at Bell Labs. He showed that not every graph is a string graph but all planar graphs are. He also raised the question whether there exists any algorithm for recognizing string graphs. Today we know

[*]Department of Mathematics, Princeton, Princeton, NJ. Email: `jacobfox@math.princeton.edu`. Research supported by an NSF Graduate Research Fellowship and a Princeton Centennial Fellowship.

[†]City College, CUNY and Courant Institute, NYU, New York, NY. Email: `pach@cims.nyu.edu`. Supported by NSF Grant CCF-08-30272, and by grants from NSA, SNF, PSC-CUNY, OTKA, and BSF.

that the answer is yes, but the problem is NP-hard [28].

A collection of curves is *k-intersecting* if every pair of its members have at most k points in common. A collection of segments in general position is 1-intersecting. A string graph is *k-intersecting* if it is the intersection graph of a k-intersecting collection of curves.

We give, for any fixed $\epsilon > 0$, an n^ϵ-approximation algorithm for estimating the independence number of k-intersecting string graphs in polynomial time.

THEOREM 1.1. *Fix $\epsilon > 0$. Let C be a k-intersecting collection of n curves in the plane, and $G = G_C$ denote its intersection graph. In time $n^{O_k((4/\epsilon)^{-2/\epsilon})}$ we can compute an independent set whose size is at least $n^{-\epsilon}\alpha(G)$.*

We also design an algorithm to compute exactly the independence number of a string graph in subexponential time.

THEOREM 1.2. *Given a string graph G with n vertices, we can compute a maximum independent set in G in time $2^{n^{4/5} \text{polylog } n}$.*

The running time of this algorithm can be improved for k-intersecting string graphs.

THEOREM 1.3. *Given a k-intersecting string graph G with n vertices, we can compute a maximum independent set in G in time $2^{O_k(n^{2/3})}$.*

The *chromatic number* $\chi(G)$ of G is the minimum number of colors needed to color the vertices of G so that no two adjacent vertices receive the same color. It is easy to see that any approximation algorithm for $\alpha(G)$ can be converted into an approximation algorithm for $\chi(G)$, with only a polynomial loss in time and an extra $\log n$ factor in the approximation ratio. Indeed, we can repeatedly pull out large independent sets and color their elements with a new color. This will result in a proper coloring of G. Thus, Theorem 1.1 implies a polynomial time n^ϵ-approximation algorithm for the chromatic number of a k-intersecting string graph on n vertices.

THEOREM 1.4. *For any fixed $\epsilon > 0$, we can properly color a k-intersecting string graph G with n vertices with $n^\epsilon\chi(G)$ colors in time $n^{O_k((4/\epsilon)^{-2/\epsilon})}$.*

When discussing algorithms on curves, it is important to clarify the precise way how the curves are given. In the above theorems on computing and approximating the independence number and chromatic number of k-intersecting string graphs (Theorems 1.1, 1.3, 1.4), we assume that we have access to the underlying plane graph, whose vertices are the intersection points of the curves and two vertices are adjacent if and only if they are consecutive intersection points along a curve, as well as the paths in the plane graph corresponding to the curves. Note that these paths partition the edge set of the underlying plane graph. For collections of segments, it is easy to compute in polynomial time (as a function of the endpoint coordinates) this underlying plane graph. For Theorem 1.2, we do not really need any special representation of G: the algorithm uses the string graph itself. We could also establish Theorem 1.3 without using the underlying plane graph, with a loss of an additional $\log n$ factor in the exponent.

2 Tools

There are two main lemmas which we use for Theorems 1.1 and 1.3. The first lemma shows that if a k-intersecting collection of curves has many pairs of intersecting curves, then we can find two large subcollections such that every curve in the first subcollection intersects every curve in the second subcollection.

LEMMA 2.1. *[14] For every positive integer k, there is a constant $c_k > 0$ with the following property. Every k-intersecting collection C of n curves with m intersecting pairs contains disjoint subcollections $A, B \subset C$ with $|A| = |B| \geq c_k m/n$ such that every curve in A intersects every curve in B. Moreover, two such subcollections can be found in polynomial time.*

Lemma 2.1 appears in [14]; its proof was based on a lemma from [15]. While it was not explicitly stated in these papers that the sets A and B can be found in polynomial time, the rather involved proof using a density increment argument indeed constructs two subcollections $A, B \subseteq C$ with the desired properties in polynomial time.

A *separator* in a graph $G = (V, E)$ is a vertex subset V_0 such that there is a partition $V = V_0 \cup V_1 \cup V_2$ with $|V_1|, |V_2| \leq 2|V|/3$ and there are no edges with one vertex in V_1 and the other in V_2. A classical result of Lipton and Tarjan [22] states that any planar graph on n vertices has a separator of order $O(\sqrt{n})$. Further, they showed that such a separator could be found in time $O(n)$. This result has been extended to intersection graphs of curves.

LEMMA 2.2. *[12] The intersection graph of a collection C of curves in the plane with a total of x intersection points among them has a separator of size at most $c'\sqrt{x}$. Such a separator can be found in polynomial time.*

The proof of Lemma 2.2 in [12] uses the Lipton-Tarjan separator result for planar graphs. While not explicitly stated in [12], the proof indeed finds a separator in polynomial time.

The proof of Theorem 1.2, which gives a subexponential time algorithm for computing a maximum independent set in a string graph, uses another separator lemma from [14].

Lemma 2.3. *[14] Every string graph with m edges and maximum degree Δ contains a separator of order at most $c''\Delta m^{1/2}\log m$.*

3 Approximation Algorithm

The aim of this section is to prove Theorem 1.1. We describe an algorithm which, given a k-intersecting collection C of n curves with intersection graph $G = G_C$, outputs a subcollection $I \subset C$ of disjoint curves, with $|I| \geq n^{-\epsilon}\alpha(G)$. The running time of this algorithm is at most $f(n) := n^{d_k(4/\epsilon)^{-2/\epsilon}}$, where the constant d_k is chosen sufficiently large, but only depending on k. Our algorithm is recursive. In the base cases $|C| = 0$ or 1, the algorithm simply outputs $I = C$.

For $|C| \geq 2$, the algorithm breaks into two cases. Note that the number of vertices of the intersection graph G is $|C| = n$. Let m denote the number of edges of G, and α denote the independence number of G. Let $c = k^{-1}c'^{-2}\left(1-\left(\frac{2}{3}\right)^{\epsilon}\right)^{2/\epsilon}$, so $\left(c'(kc)^{1/2}\right)^{\epsilon} + \left(\frac{2}{3}\right)^{\epsilon} = 1$, where c' is the absolute constant in Lemma 2.2. Note that

$$c^{-1} \leq c'^{-2}k\,(4/\epsilon)^{2/\epsilon}, \tag{3.1}$$

which follows from the inequality $1-(2/3)^{\epsilon} \geq \epsilon/4$ for $0 \leq \epsilon \leq 1$.

Case 1: $m \geq cn^2$. By Lemma 2.1, there are disjoint subcollections $A, B \subset C$ with $|A| = |B| \geq c_k m/n \geq c_k cn$ such that every curve in A intersects every curve in B. Moreover, these subcollections can be found in polynomial time. Since every curve in A intersects every curve in B, any independent set in C cannot contain both a curve in A and a curve in B. Therefore, every independent set is contained in $C \setminus A$ or $C \setminus B$. We run the algorithm on both $C \setminus A$ and $C \setminus B$. Let $n_0 = |C \setminus A| \leq n - c_k cn = (1-c_k c)n$. The running time of the algorithm is at most

$$2f(n_0) + n^{O(1)} = 2n_0^{d_k(4/\epsilon)^{-2/\epsilon}} + n^{O(1)}$$
$$\leq 2(1-c_k c)^{d_k(4/\epsilon)^{-2/\epsilon}} n^{d_k(4/\epsilon)^{-2/\epsilon}} + n^{O(1)}$$
$$\leq n^{d_k(4/\epsilon)^{-2/\epsilon}} = f(n).$$

We get the last inequality from $(1-c_k c)^{d_k\epsilon^{-2/\epsilon}} < 1/4$, which follows from (3.1) and the fact that d_k is chosen sufficiently large as a function of k. We get independent sets in $C \setminus A$ and $C \setminus B$, and we output the set I which is the larger of these two independent sets. Since a maximum independent set in G is contained in $C \setminus A$ or $C \setminus B$, we have $|I| \geq n_0^{-\epsilon}\alpha(G) \geq n^{-\epsilon}\alpha(G)$.

Case 2: $m < cn^2$. Since the collection C is k-intersecting, the number x of intersections is at most $km < kcn^2$. By Lemma 2.2, the intersection graph G has a separator $C_0 \subset C$ with $|C_0| \leq c'\sqrt{x} < c'(kc)^{1/2}n$, and C_0 can be found in polynomial time. It is easy to partition a graph into its connected components in polynomial time, and in particular, for the intersection graph of $C \setminus C_0$. We can thus find in polynomial time a partition $C = C_0 \cup C_1 \cup C_2$ with $|C_1|, |C_2| \leq 2n/3$, and no curve in C_1 intersects a curve in C_2. For $i \in \{0,1,2\}$, let $n_i = |C_i|$ and α_i denote the independence number of the intersection graph of C_i. Since no curve in C_1 intersects a curve in C_2, the union of an independent set in C_1 and an independent set in C_2 is an independent set in C. Hence, $\alpha \geq \alpha_1 + \alpha_2$. Also, trivially, $\alpha_0 \leq \alpha \leq \alpha_0 + \alpha_1 + \alpha_2$.

We run the algorithm on the subcollections C_0, C_1, C_2. The running time up to this point is

$$f(n_0) + f(n_1) + f(n_2) + n^{O(1)} \leq 4f(2n/3) < f(n)/2.$$

We obtain three independent sets $I_i \in C_i$ for $i \in \{0,1,2\}$ with $|I_i| \geq n_i^{-\epsilon}\alpha_i$. If $|I_1|+|I_2| \geq |I_0|$, then we output the independent set $I = I_1 \cup I_2$. Otherwise, we output the independent set $I = I_0$. As $|I| = \max(|I_0|, |I_1| + |I_2|)$, we have $|I| \geq |I_0|$ and $|I| \geq |I_1| + |I_2|$. This yields

$$\begin{aligned}
\alpha &\leq \alpha_0 + \alpha_1 + \alpha_2 \\
&\leq n_0^{\epsilon}|I_0| + n_1^{\epsilon}|I_1| + n_2^{\epsilon}|I_2| \\
&\leq n_0^{\epsilon}|I_0| + \left(\frac{2}{3}\right)^{\epsilon} n^{\epsilon}\left(|I_1| + |I_2|\right) \\
&\leq n^{\epsilon}\left(\left(c'(kc)^{1/2}\right)^{\epsilon}|I_0| + \left(\frac{2}{3}\right)^{\epsilon}\left(|I_1| + |I_2|\right)\right) \\
&\leq n^{\epsilon}\left(\left(c'(kc)^{1/2}\right)^{\epsilon} + \left(\frac{2}{3}\right)^{\epsilon}\right)|I| \\
&= n^{\epsilon}|I|.
\end{aligned}$$

Here we used $\left(c'(kc)^{1/2}\right)^{\epsilon} + \left(\frac{2}{3}\right)^{\epsilon} = 1$. Hence, $|I| \geq n^{-\epsilon}\alpha$, and the running time is at most $f(n)$, which completes the proof.

4 Exact Algorithms

In this section, we prove Theorems 1.2 and 1.3.

Proof of Theorem 1.2: We give a recursive algorithm which, given a string graph $G = (V, E)$ on n vertices, outputs a maximum independent set in G in time at most $g(n) := 2^{c_0 n^{4/5}\log^{7/5} n}$, where c_0 is a sufficiently large absolute constant. Let $d = \lceil n^{1/5}\log^{-2/5} n\rceil$. It is not difficult to check that the function $g(n) - g(n-1) - g(n-d)$ grows faster that any polynomial in n.

If G has at most one vertex, the algorithm simply outputs the vertex set V. Suppose now that $|V| \geq 2$. Let Δ denote the maximum degree of the vertices in G.

Case 1: $\Delta \geq d$. Pick a vertex $v \in V$ of degree Δ, and let $N(v)$ denote the set of neighbors of v. A maximum independent set containing v lies either in $V \setminus N(v)$ or in $V \setminus v$. We find a maximum independent set in $V \setminus N(v)$ in time at most $g(n-\Delta)$, a maximum independent set in $V \setminus v$ in time at most $g(n-1)$, and output the larger of the two. The running time of this algorithm is at most

$$g(n-1) + g(n-\Delta) + n^{O(1)} \leq g(n).$$

Case 2: $\Delta < d$. The number m of edges of G is less than $n\Delta/2$. By Lemma 2.3, G has a separator V_0 of size at most $s = \lfloor c''\Delta m^{1/2} \log m \rfloor < 2c''\Delta^{3/2} n^{1/2} \log n$. We can find such a separator V_0 by trying all subsets of size at most s in time at most $n^s = 2^{O(n^{4/5} \log^{7/5} n)}$. We obtain a partition $V = V_0 \cup V_1 \cup V_2$ such that $|V_0| \leq s$, $|V_1|, |V_2| \leq 2n/3$, and no vertex in V_1 is adjacent to any vertex in V_2. For any independent set $I \subset V_0$ and any $j \in \{1,2\}$, let $V_{I,j} \subset V_j$ denote the set of all vertices in V_j that have no neighbor in I. Let $X_{I,j}$ be a maximum independent set in $V_{I,j}$. For a given I, the time to compute the $V_{I,j}$ is $O(n^2)$. The set $I' = I \cup X_{I,1} \cup X_{I,2}$ is an independent set, as there are no edges between V_1 and V_2. For at least one independent set $I \subset V_0$, the set I' is a *maximum* independent set, and we output this set. Once we have picked I, the time required to find $X_{I,1}$ and $X_{I,2}$ is bounded from above by $2g(2n/3)$. The number of possible choices for I is $2^{|V_0|} \leq 2^s$. Hence, the total running time of our algorithm is at most

$$O(n^2 2^s g(2n/3)) \leq g(n).$$

This completes the proof of Theorem 1.2.

Using Lemma 2.2 instead of Lemma 2.3 and picking d appropriately, results in an algorithm for computing the independence number of a k-intersecting string graph G on n vertices. This algorithm runs in time $h(n) = 2^{O_k(n^{2/3} \text{polylog } n)}$. To obtain Theorem 1.3, we have to drop the polylog n factor in the exponent. To achieve this, we distinguish two cases based on the number of edges of G rather than the maximum degree of its vertices.

In the dense case, we use Lemma 2.1 to obtain two large subfamilies of curves which cross each other. Denoting the size of these subfamilies by t, and using the fact that no independent set contains at least one curve from each of these subfamilies, we obtain an upper bound of roughly $2h(n-t)$ on the running time. Otherwise, we use Lemma 2.2 to pick out a small separator, and as in the proof of Theorem 1.2, we find a maximum independent set by extending all possible independent subsets of the separator.

Remark: The only property of string graphs we used in the proof of Theorem 1.2 (and in the proof of the slightly weaker version of Theorem 1.3, where we lost a polylog n factor in the exponent) is that they satisfy a separator theorem. The proof thus extends to give a subexponential time algorithm for any class of graphs for which separator theorems are known. For example, Miller, Teng, Thurston, and Vavasis [26] established a separator theorem for intersection graphs of balls in Euclidean space. It states that any intersection graph of n balls in $\mathbf{R}^d$ with no point belonging to k balls has a separator of order $O_d(k^{1/d} n^{1-1/d})$. The above algorithm together with this result gives an algorithm for computing the independence number of an intersection graph of n balls in $\mathbf{R}^d$ which runs in time $2^{O_d\left(n^{1-\frac{1}{d+1}} \text{polylog } n\right)}$.

5 Concluding Remarks

The *clique number* and the *clique cover number* of a graph G are the independence number and the chromatic number of its complement $\overline{G}$. Of course, the complement of a string graph is not necessarily a string graph. Nevertheless, the idea of the proof of Theorem 1.1, based on Lemmas 2.1 and 2.2, can be modified to produce a polynomial time algorithm which approximates these parameters within a factor $n^{1-\epsilon_k}$ for some fixed $\epsilon_k > 0$.

THEOREM 5.1. *For each positive integer k there is $\epsilon_k > 0$ such that we can approximate the clique number of a k-intersecting string graph on n vertices within a factor $n^{1-\epsilon_k}$ in time $n^{O(1)}$.*

It would be interesting to improve this to a polynomial time n^ϵ-approximation algorithm.

Planar graphs are an interesting special case of string graphs. Indeed, the Koebe circle packing theorem [20] states that a graph is planar if and only if it is the intersection graph of a collection of nonoverlapping disks in the plane. It follows from the Four-Color Theorem [3] that every planar graph on n vertices has an independent set of size at least $n/4$, so we immediately know the independence number up to a factor of 4. Using their planar separator theorem, Lipton and Tarjan [23] gave an algorithm which gives a $(1+o(1))$-approximation for the independence number of a planar graph on n vertices in time $O(n \log n)$. They also used their separator theorem to give an exact algorithm that runs in time $2^{O(\sqrt{n})}$. Baker [6] found an alternative algorithm which gives a better trade-off between the approximation ratio and the running time. It would be interesting to improve these bounds further.

References

[1] P.K. Agarwal, M. van Kreveld, and S. Suri, Label placement by maximum independent set in rectangles, *Comput. Geom.* **11** (1998), 209–218.

[2] P. K. Agarwal and N. H. Mustafa, Independent set of intersection graphs of convex objects in 2D, *Comput. Geom.* **34** (2006), 83–95.

[3] K. Appel, W. Haken, and J. Koch, Every planar map is four colorable, *Illinois J. Math.* **21** (1977), 439–567.

[4] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, Proof verification and the hardness of approximation problems, *J. ACM* **45** (1998), 501–555.

[5] S. Arora and S. Safra, Probabilistic checking of proofs: a new characterization of NP, *J. ACM* **45** (1998). 70–122.

[6] B. S. Baker, Approximation algorithms for NP-complete problems on planar graphs, *J. ACM* **41** (1994), 153–180.

[7] S. Benzer, On the topology of the genetic fine structure, *Proc. Nat. Acad. Sci.* **45** (1959), 1607–1620.

[8] B. N. Clark, C. J. Colbourn, and D. S. Johnson, Unit disk graphs, *Discrete Math.* **86** (1990), 165–177.

[9] G. Even, Z. Lotker, D. Ron, and S. Smorodinsky, Conflict-free colorings of simple geometric regions with applications to frequency assignment in cellular networks, *SIAM J. Computing* **33** (2003), 94–136.

[10] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy, Interactive proofs and the hardness of approximating cliques, *J. ACM* **43** (1996), 268–292.

[11] L. de Floriani, P. Magillo, and E. Puppo, Applications of computational geometry to geographic information systems, Chapter 7 in: *Handbook of Computational Geometry* (J. Sack, J. Urrutia, eds.), North-Holland, Amsterdam, 2000, 333–383.

[12] J. Fox and J. Pach, Separator theorems and Turán-type results for planar intersection graphs, *Advances in Mathematics* **219** (2008), 1070–1080.

[13] J. Fox and J. Pach, String graphs and incomparability graphs, submitted.

[14] J. Fox and J. Pach, A separator theorem for string graphs and its application, *Combin. Probab. Comput.* **19** (2010), 371–390.

[15] J. Fox, J. Pach, and C. D. Tóth, Intersection patterns of curves, *J. London Math. Soc.*, to appear.

[16] J. Håstad, Clique is hard to approximate within $n^{1-\epsilon}$, *Acta Math.* **182** (1999), 105–142.

[17] D.S. Hochbaum and W. Maass, Approximation schemes for covering and packing problems in image processing and VLSI, *J. ACM* **32** (1985), 130–136.

[18] J. Kara and J. Kratochvíl, Fixed parameter tractability of independent set in segment intersection graphs, in: *2nd International Workshop on Parameterized and Exact Computation (IWPEC 2006), Lecture Notes in Comput. Sci.* **4169**, Springer, Berlin, 2006, 166–174.

[19] R. Karp, Reducibility among combinatorial problems, in *Complexity of Computer Computations* (R.E. Miller and J. W. Thatcher, eds.), Plenum, New York, 1972, 85-103.

[20] P. Koebe, Kontaktprobleme der Konformen Abbildung, *Ber. Sächs. Akad. Wiss. Leipzig, Math.-Phys. Kl.* **88** (1936), 141–164.

[21] J. Kratochvíl and J. Nešetril, INDEPENDENT SET and CLIQUE problems in intersection-defined classes of graphs, *Comment. Math. Univ. Carolin.* **31** (1990), 85–93.

[22] R. J. Lipton and R. E. Tarjan, A separator theorem for planar graphs, *SIAM J. Appl. Math.* **36** (1979), 177–189.

[23] R. J. Lipton and R. E. Tarjan, Applications of a planar separator theorem, *SIAM J. Comput.* **9** (1980), 615–627.

[24] E. Malesinska, *Graph-Theoretical Models for Frequency Assignment Problems,* Ph.D. Thesis, Technische Universität Berlin, 1997.

[25] D. Marx, Parameterized complexity of independence and domination on geometric graphs, in: *2nd International Workshop on Parameterized and Exact Computation (IWPEC 2006), Lecture Notes in Comput. Sci.* **4169**, Springer, Berlin, 2006, 154–165.

[26] G. L. Miller, S.-H. Teng, W. Thurston, and S. A. Vavasis, Separators for sphere-packings and nearest neighbor graphs, *J. ACM* **44** (1997), 1–29.

[27] F. Preparata and M. Shamos, *Computational Geometry: An Introduction,* Texts and Monographs in Computer Science, Springer-Verlag, New York, 1985.

[28] M. Schaefer, E. Sedgwick, and D. Štefankovič, Recognizing string graphs in NP, in: Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing, ACM Press, New York, 2002, 1–6.

[29] F. W. Sinden, Topology of thin film RC-circuits, *Bell System Technological Journal* (1966), 1639–1662.

[30] F. Spieksma, Approximating an interval scheduling problem, in: *Proceedings of the First International Workshop on Approximation Algorithms for Combinatorial Optimization* (APPROX.98), Lecture Notes in Computer Science, vol. 1444, Springer, Berlin, 1998, 169–180.

[31] B. Verweij and K. Aardal, An optimisation algorithm for maximum independent set with applications in map labelling, in: *Proceedings of the Seventh Annual European Symposium on Algorithms* (ESA.99), Lecture Notes in Computer Science, vol. 1643, Springer, Berlin, 1999, 426–437.

[32] D. Zuckerman, Linear degree extractors and the inapproximability of max clique and chromatic number, *Theory Comput.* **3** (2007), 103–128.