COVER CONTACT GRAPHS[*]

Nieves Atienza,[†] *Natalia de Castro,*[†] *Carmen Cortés,*[†] *M. Ángeles Garrido,*[†]
Clara I. Grima,[†] *Gregorio Hernández,*[‡] *Alberto Márquez,*[†] *Auxiliadora Moreno-González,*[†]
Martin Nöllenburg,[§] *José Ramón Portillo,*[†] *Pedro Reyes,*[†] *Jesús Valenzuela,*[†]
Maria Trinidad Villar,[†] *Alexander Wolff*[¶]

Abstract. We study problems that arise in the context of covering certain geometric objects called *seeds* (e.g., points or disks) by a set of other geometric objects called *cover* (e.g., a set of disks or homothetic triangles). We insist that the interiors of the seeds and the cover elements are pairwise disjoint, respectively, but they can touch. We call the contact graph of a cover a *cover contact graph* (CCG).

We are interested in three types of tasks, both in the general case and in the special case of seeds on a line: (a) deciding whether a given seed set has a connected CCG, (b) deciding whether a given graph has a realization as a CCG on a given seed set, and (c) bounding the sizes of certain classes of CCG's.

Concerning (a) we give efficient algorithms for the case that seeds are points and show that the problem becomes hard if seeds and covers are disks. Concerning (b) we show that this problem is hard even for point seeds and disk covers (given a fixed correspondence between graph vertices and seeds). Concerning (c) we obtain upper and lower bounds on the number of CCG's for point seeds.

1 Introduction

Koebe's theorem [Koe36, PA95], a beautiful and classical result in graph theory, says that every planar graph can be represented as a *coin graph*, that is, a contact graph of disks in the plane. In other words, given any planar graph with n vertices, there is a set of n disjoint open disks in the plane that are in one-to-one correspondence to the vertices such that a pair of disks is tangent if and only if the corresponding vertices are adjacent. Conversely, every coin graph is obviously planar since if we connect the centers of touching disks by straight-line edges, no two edges intersect. Koebe's theorem has been rediscovered several times, see

[*]A preliminary version of this paper appeared in: Proc. 15th International Symposium on Graph Drawing (GD'07) [AdCC+08]. This research was partially supported by projects *PAI* FQM—0164, *ORI* MTM2008-05866-C03-01, and grant WO 758/4-3 of the German Research Foundation (DFG).

[†]*Universidad de Sevilla, Spain*, {natienza, natalia, ccortes, vizuete, grima, almar, auxiliadora, josera, preyes, jesusv, villar}@us.es

[‡]*Departamento de Matemática Aplicada, Facultad de Informática, Universidad Politécnica de Madrid, Spain*, gregorio@fi.upm.es

[§]*Institute of Theoretical Informatics, Karlsruhe Institute of Technology (KIT), Germany*, noellenburg@kit.edu

[¶]*Institut für Informatik, Universität Würzburg, Germany*, www1.informatik.uni-wuerzburg.de/en/staff the survey of Sachs [Sac94]. Collins and Stephenson [CS03] give an efficient algorithm for numerically approximating the radii and locations of the disks of such a representation of a planar graph. Their algorithm relies on an iterative process suggested by Thurston [Thu80].

Since Koebe there has been a lot of work in the graph-drawing community dedicated to the question which planar graphs can be represented as contact or intersection graphs of which geometric object. Intersection graphs, especially of disks, are used as graph models in multiple application areas, for example, wireless communication networks [Hal80, CCJ90]. Recently, Chalopin and Gonçalves [CG09] showed that, similarly to Koebe's characterization, every planar graph is the intersection graph of line segments in the plane.

On the other hand, there has been a lot of work in the geometric-optimization community dedicated to the question how to (optimally) cover geometric objects (usually points) by other geometric objects (for example, convex shapes, disks, or annuli). Typical objectives are minimizing the (maximum/total) radius of a set of k disks to cover n input points. Alternatively, the disk size might be fixed and the number of disks used to cover the input points is to be minimized. Applications of such covering problems are, for example, geometric facility location problems [RT90, Wel91].

In this paper we combine the two previous problems: we are given a set of pairwise disjoint geometric objects called *seeds* (for example, points or disks) that must be covered by other geometric objects called *covering objects* (for example, disks or triangles) whose interiors are pairwise disjoint. Unlike in geometric optimization, each of our covering objects must contain exactly one of the seeds. We are not interested in minimizing or maximizing the sizes of the covering objects; instead we want their contact graph to satisfy some graph-theoretic property (like connectivity) or to be isomorphic to a given graph. Compared to previous work on geometric representation of graphs, we are more restricted in the choice of our representatives by the requirement to cover the set of seeds.

Model. Given a set $S = \{p_1, p_2, \ldots, p_n\}$ of pairwise disjoint *seeds* in $\mathbb{R}^2$ of some type, a *cover* of S is a set $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ of closed objects of some type (also called *cover elements*) with the property that there is a bijection between seeds and cover elements such that each C_i contains its seed p_i and any two cover elements are allowed to intersect only on their boundaries. Figure 1b depicts a disk cover of the disk seeds in Figure 1a. The *cover contact graph* (CCG) induced by $\mathcal{C}$ is the contact graph of the elements of $\mathcal{C}$, that is, the graph $G = (\mathcal{C}, E)$ with $E = \{\{C_i, C_j\} \subseteq \mathcal{C} \mid C_i \neq C_j,\ C_i \cap C_j \neq \emptyset\}$. In other words, two vertices of a CCG are adjacent if the corresponding cover elements touch, that is, their boundaries intersect. Figure 1c depicts the CCG induced by the cover in Figure 1b. Note that the vertices of the CCG are in one-to-one correspondence to both seeds and covering objects. We consider seeds to be topologically open (except if they are single points). Then the closures of two open seeds are allowed to touch. Note that we require cover objects to be closed. This makes sure that a cover actually contains a point seed that lies on its boundary. We call a CCG whose cover consists of disks a *disk-CCG* and in the case of triangles a *triangle-CCG*.

In this paper, we investigate the following questions.

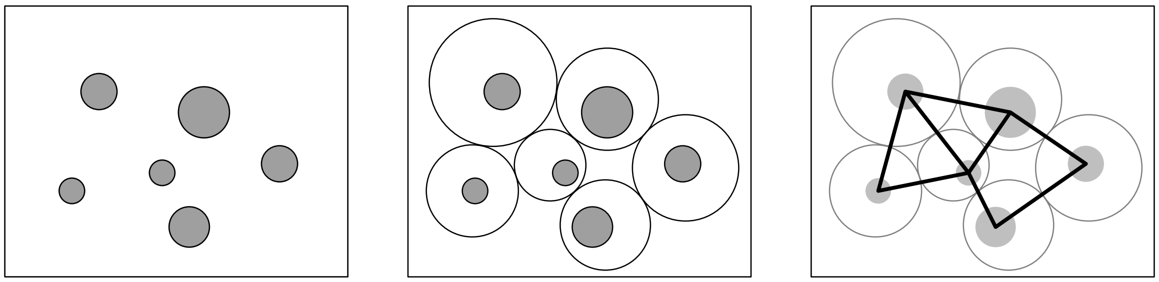

(a) Disk seeds (b) Disk cover of (a) (c) CCG induced by (b)

Fig. 1: Seeds, cover, and CCG.

Connectivity: Given a seed set, does it admit a (simply or bi-) connected CCG?

Realizability: Given a planar graph G and a set of seeds, can G be realized as a CCG on the given seeds?

Enumeration: For a given number of vertices, how many graphs of a certain graph class can be realized as a CCG?

Our focus in this paper is on the first two questions: connectivity and realizability.

We also consider an interesting restriction of the above problems where seeds and cover elements must lie in the half plane $\mathbb{R}^2_+$ above and including the x-axis. Seeds are additionally required to contain at least one point of the x-axis. In this restricted setting we call the contact graph of a cover a CCG$^+$. See Figures 11b and 13 for examples.

Our results. First, we consider arbitrary sets of point seeds in the plane; see Section 2. Concerning connectivity we show that we can always cover a set of point seeds using disks or using homothetic triangles such that the resulting CCG is 1- or even 2-connected. The two algorithms run in $O(n \log n)$ worst-case time (1-connected CCG) and $O(n^2 \log n)$ expected time (2-connected CCG). Concerning realizability, we give some necessary conditions and then show that it is NP-hard to decide whether a given graph can be realized as a disk-CCG if the correspondence between vertices and point seeds is given. We also remark that there is an upper bound on the number of planar graphs that are realizable as CCG's.

Next, we consider the restriction where the seed set is a set of points on the x-axis; see Section 3. We show that in this case 2-connectivity is easy: given any seed set on the x-axis, we can realize the n-cycle as CCG and any n-vertex tree as CCG$^+$. For the case that the correspondence between seeds and vertices is given, we show how to decide, in $O(n \log n)$ time, whether a given tree can be realized as a CCG$^+$. Concerning enumeration, we prove that the (n)-th Catalan number is a lower bound of the number of labeled trees realizable as a CCG$^+$ on an ordered seed set of cardinality $2n + 1$.

Finally, we consider disk and triangle seeds; see Section 4. We show that for homothetic triangle seeds on the x-axis, there is always a connected triangle-CCG$^+$ and that for disk seeds it is already NP-hard to decide whether they admit a connected disk-CCG. The hardness of the realization problem for point seeds and the enumeration results for points remain to be true for disk seeds.

Related work. Abellanas et al. [ABH$^+$06] proved that the following problem, which they call the *coin placement problem*, is NP-complete. Given n disks of given varying radii and n points in the plane, is there a way to place the disks such that each disk is centered at one of the given points and no two disks overlap?

Abellanas et al. [AdCH$^+$06] considered a related problem. They showed that given a set of points in the plane, it is NP-complete to decide whether there are disjoint disks centered at the points such that the contact graph of the disks is connected.

Given a pair of touching convex (or, more generally, star-shaped) cover elements, we can draw the corresponding edge in a drawing of the CCG by a two-segment polygonal line that connects the incident seeds and uses the contact point of the cover elements as bend. This is a link to the problem of point-set embeddability. We say that a planar graph $G = (V, E)$ with n vertices is *k-bend (point-set) embeddable* if for any set $P \subset \mathbb{R}^2$ of n points there is a one-to-one mapping from V to P such that the edges of G can be drawn as non-crossing polygonal lines with at most k bends. Kaufmann and Wiese [KW02] showed that (a) every 4-connected planar graph is 1-bend embeddable, (b) every planar graph is 2-bend embeddable, and (c) given a planar graph $G = (V, E)$ and a set P of n points on a line, it is NP-complete to decide whether G has a 1-bend embedding that maps V one-to-one on P.

For a given correspondence between vertices and points, Pach and Wenger [PW01] showed how to compute in $O(n^2)$ time a plane embedding of the given graph where each vertex is represented by the corresponding point and each edge is represented by a polygonal line with $O(n)$ bends. They also showed that $\Omega(n)$ bends are needed in the worst case.

2 Seeds are Points in the Plane

In this section we study point seeds with arbitrary positions in the plane. If not stated otherwise, our results hold for both disk covers and (homothetic) triangle covers. We start with the connectivity problem.

2.1 Connectivity

It is known to be NP-hard to decide whether a given set of points can be covered by a set of pairwise disjoint open disks, each centered on a point, such that the contact graph of the disks is connected [AdCH$^+$06]. In contrast to that result we give a simple sweep-line algorithm that covers point seeds by (non-centered) disks or triangles such that their contact graph is connected.

Theorem 2.1. *Every set S of n point seeds has a connected CCG. Such a CCG can be constructed in $O(n \log n)$ time and linear space.*

Proof. We first sort the given seed set S by non-increasing ordinate. Let $p_1, \ldots, p_n$ be the resulting sequence of points. Then we process the points in this order from top to bottom. We cover p_1 by a cover element C_1 such that p_1 is the bottommost point of C_1. In case of disk covers, C_1 can have arbitrary size; in case of homothetic triangle covers, the triangle C_1 must be large enough so that the “shadow” region defined by the top edge of C_1 and two rays through the top vertices that are parallel to the opposing triangle sides contains all other seeds, see Figure 2. Now assume that $k > 1$ and that the first $k-1$ points have been covered such that the contact graph of their cover is connected. Then we cover $p_k = (x_k, y_k)$ by an infinitesimally small cover element C_k and inflate C_k with p_k as the bottommost point until C_k eventually touches one of the previously placed cover elements. Figure 2 shows an example with homothetic triangles as cover elements.

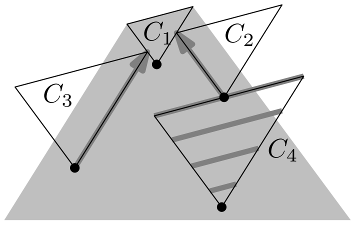


Fig. 2: Constructing a connected CCG by inflating triangles as cover elements. The size of the first triangle C_1 is selected so that its shadow contains all other seeds. Then three contact types are possible: the top edge of the new triangle touches the bottom vertex of a previous triangle (C_4), the top right vertex touches a left edge (C_3), or the top left vertex touches a right edge (C_2).

It is possible to do this construction by incrementally constructing an abstract Voronoi diagram with respect to accordingly defined bisectors. It is not clear, however, how to implement this in $O(n \log n)$ time. Instead we use the following simpler algorithms for triangles and disks, respectively.

For triangles, it is easy to determine the size of the k-th triangle C_k given that triangles $C_1, \ldots, C_{k-1}$ have been placed. The idea is that we maintain three data structures, one for each type of collision during the afore-mentioned inflation step, see Figure 2. We first precompute, in $O(n \log n)$ time and linear space, a data structure for so-called *segment-dragging queries* [Mit92]. Given a fixed direction d and a wedge W with apex $(0,0)$, a data structure for segment dragging yields, in $O(\log n)$ time, for a given query point $p \in \mathbb{R}^2$ the first point hit by a line segment of direction d whose endpoints run from p along the edges of the wedge $W + p$ with apex p. This data structure gives us seeds that are hit by (the relative interior of) the top edge of C_k (see C_4 in Figure 2). It remains to determine the first triangles that are hit by rays on which the top left and top right vertex of C_k travel when we inflate C_k. This can be done by two simple ray-shooting data structures. Both data structures can be implemented by dynamic balanced binary search trees since in each of them all query rays have the same direction, and so do the segments that are potentially hit. Thus we again have a query time of $O(\log n)$. Once a new triangle is added to the cover, we insert the new left and right triangle edges to the search trees in $O(\log n)$ time. Thus our algorithm for connected triangle-CCG’s runs in $O(n \log n)$ time and needs linear space.

For disks we do a top-to-bottom sweep with a horizontal sweep line $\ell\colon y = c$. As before, let $C_1, \ldots, C_{k-1}$ be the disks that have already been placed, that is, whose south poles lie on or above ℓ. We maintain the lower envelope of functions $f_1^c, \ldots, f_{k-1}^c$, where f_j^c is the locus of the centers of all disks that touch both disk C_j and the sweep line. It is easy to see that f_j^c is the parabola whose focus is the center of C_j and whose directrix is the horizontal line with y-coordinate $c - r_j$, see Figure 3. Note that, given a new site $p_k = (x_k, y_k)$ on the sweep line, the largest disk with south pole p_k that does not intersect the interior of any previously placed disk has center $(x_k, \tilde{y_k})$ and radius $\tilde{y_k} - c$, where $\tilde{y_k} = \min_{1 \le j < k} f_j^c(x_k)$. This disk can easily be computed given the lower envelope.

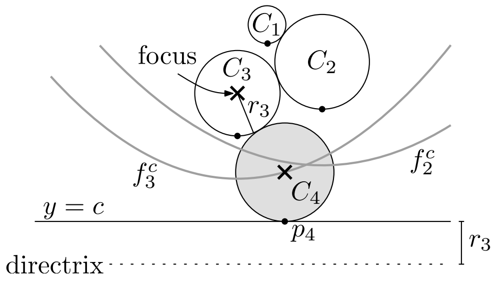


Fig. 3: Finding a covering disk C_4 for seed p_4 that touches a previous disk. The center of C_4 lies on the lower envelope of the parabolas f_1^c, f_2^c, and f_3^c. Focus and directrix of f_3^c are shown.

Our sweep is very similar to Fortune's sweep [For86] for computing the Voronoi diagram of weighted points (or disks). The only difference is that we do not know the weight of a point p beforehand; we compute the weight of p (by querying the lower envelope) as soon as we reach p. The handling of changes in the lower envelope and the insertion of new parabolas are essentially the same as in Fortune's sweep. Thus the running time of $O(n \log n)$ and the linear space consumption carry over. $\square$

In fact, even a biconnected CCG for any set of n point seeds exists as the following theorem assures.

Theorem 2.2. *Any set S of n point seeds has a biconnected CCG. Such a CCG can be constructed in $O(n^2 \log n)$ time using linear space.*

Proof. We first consider disks as cover elements. Let D_1, D_2, and D_3 be three congruent disks that touch each other. They delimit a pseudo-triangular shape R. Choose the three disks such that each disk D_i contains a unique point $p_i \in S$ and such that $S \setminus \{p_1, p_2, p_3\} \subset R$, see Figure 4a.

In order to cover the remaining points, we assume that disks $D_4, \ldots, D_{i-1}$ have been placed such that each covers a unique point of S and touches two previously placed disks, see Figure 4b. Thus the contact graph of $D_1, \ldots, D_{i-1}$ is biconnected. Let R_i be a connected component of $R \setminus \bigcup_{j=4}^{i-1} D_j$ that contains at least one uncovered point. We use Fortune's sweep [For86] to compute the combined Voronoi diagram of the disks incident to R_i and the points in $S \cap R_i$. This takes $O(n \log n)$ time and the resulting Voronoi diagram has complexity $O(n)$. The part of the Voronoi diagram in R_i is the locus of the centers of

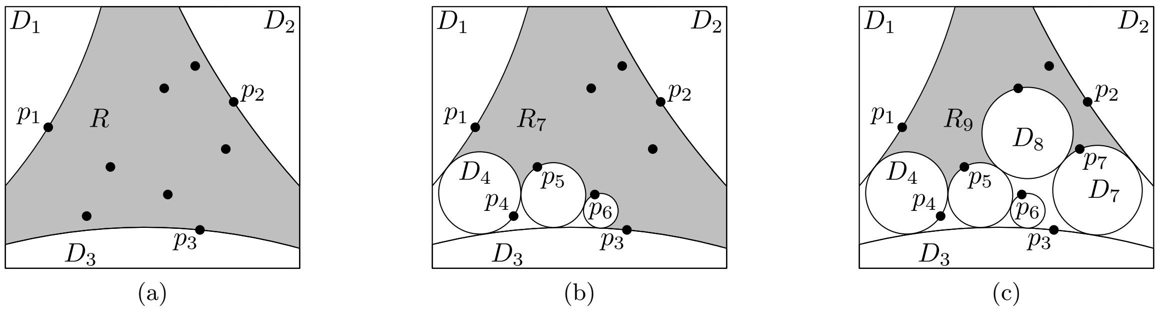


Fig. 4: Three steps in the construction of a biconnected disk-CCG.

all disks that lie in R_i and touch $\partial R_i \cup (S \cap R_i)$ in at least two points, where ∂R_i is the boundary of R_i.

Now we make a simple but crucial observation: if D is a disk that (a) lies in R_i, (b) contains a seed $s \in S \cap R_i$ on its boundary, and (c) touches two of the previous disks, then D is centered at a *vertex* of the Voronoi diagram. Thus a disk $D^\star$ fulfilling (a)–(c) can be found in linear time and, by construction, does not contain any point of S in its interior. (If, by any chance, all such disks touch more than one point of S, we re-start the whole computation with three slightly wiggled initial disks D_1, D_2, and D_3. Then the probability of this degeneracy becomes 0.) Now set $D_i = D^\star$, and repeat the process until all seeds are covered. This takes $O(n^2 \log n)$ time in total.

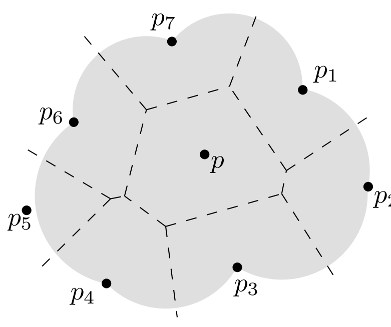
The case of triangles can be handled analogously. Choosing any reference point in the triangular shape, a structure similar to the medial axis can be computed in $O(n \log n)$ time and updated in $O(n)$ time in each of the $n - 3$ phases. □

2.2 Realizability

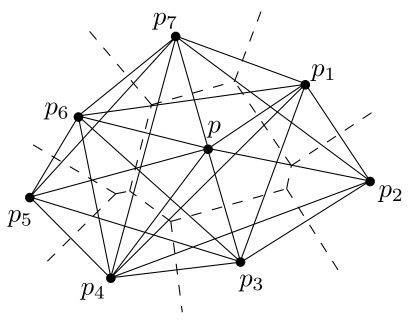
In this section we first give two necessary conditions that a planar graph must satisfy in order to be realizable as a disk-CCG on a given seed set. We show that there is a plane geometric graph on six vertices that cannot be represented as disk-CCG. Finally we investigate the complexity of deciding realizability.

To formulate our necessary conditions for realizability, we define the *hyperinfluence graph* on the given seed set S. This graph is inspired by the sphere-of-influence graph defined by Toussaint [Tou88] (see also [HJLM93, JLM95] for more results on sphere-of-influence graphs). Given a seed set S and a point $p \in S$, let the *influence area* of p be the closure of the union of all empty open disks D (that is, $D \cap S = \emptyset$) that are centered at vertices of the Voronoi region of p, see Figure 5a. We call the intersection graph of the influence areas of all seeds in S the *hyperinfluence graph* of S and denote it by $HI(S)$, see Figure 5b.

Proposition 2.1. *Let S be a set of point seeds and let G be a graph that is realizable as a disk-CCG on S. Then*



(a) Influence area of seed p.



(b) Hyperinfluence graph $HI(S)$.

Fig. 5: Influence area and hyperinfluence graph for seeds S.

(i) G is a subgraph of $HI(S)$, and

(ii) G has a plane drawing where each vertex is mapped to a unique seed in S and each edge is drawn as a polygonal line with at most one bend.

Proof. Both properties are straightforward to obtain. Property (i) is based on the observation that any possible covering disk of p is contained in the influence area of p. Thus, if the covering disks of two seeds are in contact, their influence areas intersect.

Property (ii) is obtained by representing each edge of the CCG by two line segments that connect the seeds with the point of tangency of the covering disks. □

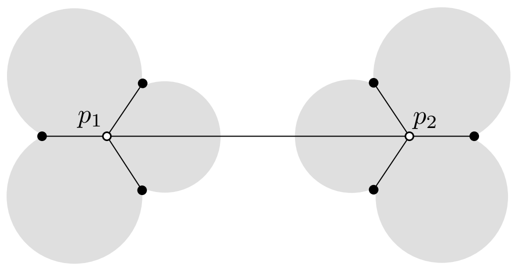


Fig. 6: A non-realizable graph. The influence areas of p_1 and p_2 do not intersect and thus no two covering disks of p_1 and p_2 can touch.

While it is NP-complete to verify property (ii) of Proposition 2.1 even if all seeds lie on a line [KW02], property (i) of Proposition 2.1 gives us a way to show non-realizability of certain geometric graphs, for example, the graph depicted in Figure 6. The edge p_1p_2 of the graph cannot be realized in a CCG with the given seeds because the shaded influence areas of p_1 and p_2 do not intersect. This graph is thus an example of a non-realizable graph with eight vertices. On the other hand it is easy to see that any three-vertex graph can be realized on any three-point seed set. Now it is interesting to ask for the least n for which there is a plane n-vertex geometric graph G that cannot be realized as CCG.

Proposition 2.2. *There is a set S of six point seeds in convex position such that their Delaunay triangulation is not representable as a CCG.*

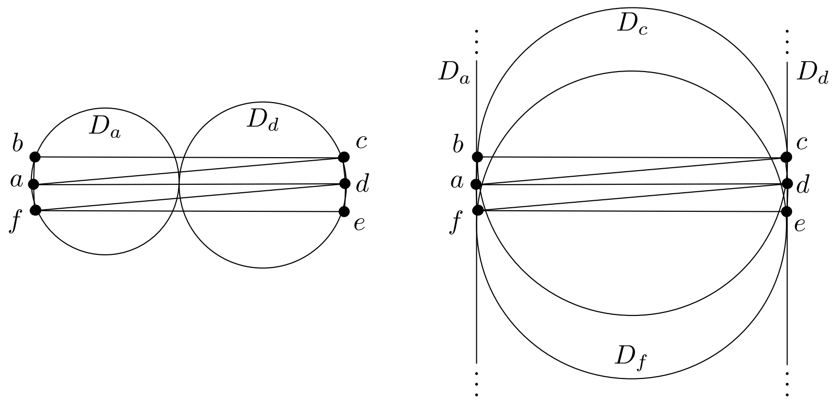


Fig. 7: Non-realizable Delaunay triangulation of six points in convex position.

Proof. Let $S = \{a, b, c, d, e, f\}$ be six points in convex position that are connected by the edges of their Delaunay triangulation as shown in Figure 7. Since the points a and d are connected, the covering disks D_a and D_d of the points a and d must touch each other in one of two ways. Either the tangent point of the disks lies inside the convex hull of S (left part of Figure 7), or D_a and D_d are very large and lie to the left of a and to the right of d, in which case they touch far above or below S as indicated in the right part of Figure 7. In the first case, we can either find a disk covering c that touches D_a and D_d or a disk covering f that touches D_a and D_d, but not both at the same time. In the second case, we can assume that the boundaries of D_a and D_d are two almost parallel lines in the vicinity of the six points. The disks D_c and D_f covering c and f must both touch D_a and D_d. If, however, c and f are close enough to a and d then D_c and D_f cannot be disjoint. □

So we have seen that there are pairs of (quite small) graphs and seed sets such that the graph cannot be realized on the seed set as disk-CCG. A natural question to ask is whether a given graph is realizable as CCG on a given seed set or not. Of course Koebe's theorem [Koe36] guarantees that, for any planar graph G, there is a seed set S such that it is possible to realize G on S. If the seeds and the vertex–seed correspondence are given, however, the problem becomes NP-hard as the next theorem shows.

Theorem 2.3. *Given a set S of points in the plane and a planar graph $G = (V, E)$ with a bijection between V and S, it is NP-hard to decide whether G is realizable as disk-CCG on S.*

Proof. We show the NP-hardness by reduction from Planar3Sat, which is NP-hard [Lic82]. Planar3Sat is defined as follows. We are given a planar bipartite graph H_φ, the *variable-clause graph*, corresponding to a *planar* Boolean formula φ in conjunctive normal form with three literals per clause. The vertices in one part of the bipartition represent the variables of φ, and the vertices in the other part correspond to the clauses of φ. Each clause is connected to the three variables that it contains. Such a graph H_φ can be drawn on a grid of polynomial size with all variable vertices placed on a horizontal line and the clause vertices connecting in a comb-shaped manner from above or below that line [KR92]. Similar to a hardness proof of Cabello et al. [CDR07], we use a slanted layout of H_φ, where all angles are multiples of 60 degrees.

Next, we construct gadgets for variables and clauses as seeds and edges on a triangular grid such that the resulting graph can be realized as CCG if and only if the Boolean formula φ is satisfiable. One basic ingredient are linear chains of adjacent points, each of which can be covered by a disk from one of two classes of disks depending on the truth value encoded by the respective chain structure. These chains will be used for the variables and for the literal connections to the clauses. The structure of the chains is exemplified in Figure 8 which shows two chain links and the underlying graph. The two desired covering disks D_p^{true} and D_p^{false} of the central point seed p are drawn in solid black and gray, respectively. They both touch the covering disks of the two *stopper elements* s and s' above and below p as required by the edges ps and ps'.

To see how a stopper element works, consider point s in Figure 8. The positions of the singleton seed q and the seed r, which is adjacent to s, force the centers of all valid covering disks to the left of p to lie within a small region I_p^{true}. This region can be made arbitrarily small by slightly shifting the seeds of the stopper element. The same holds for I_p^{false} and disks to the right of p.

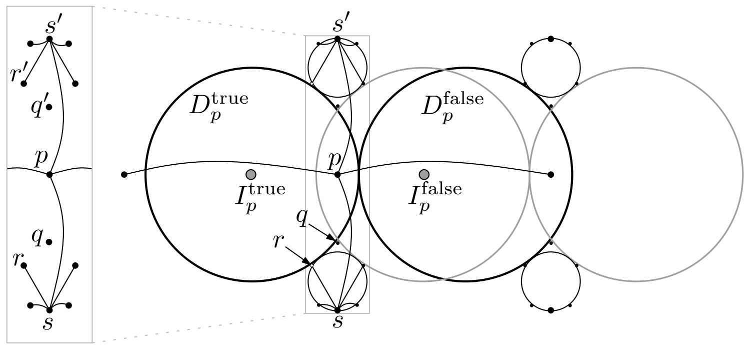
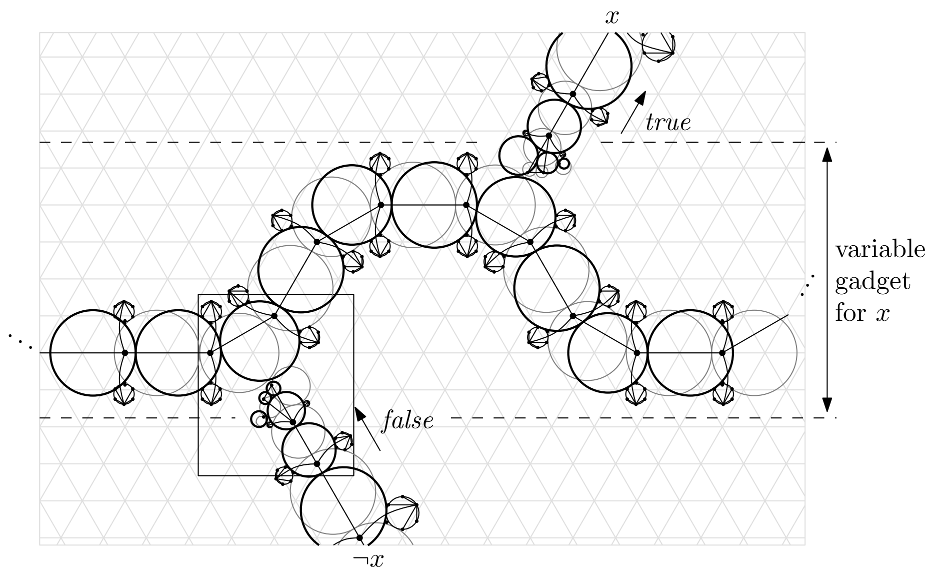
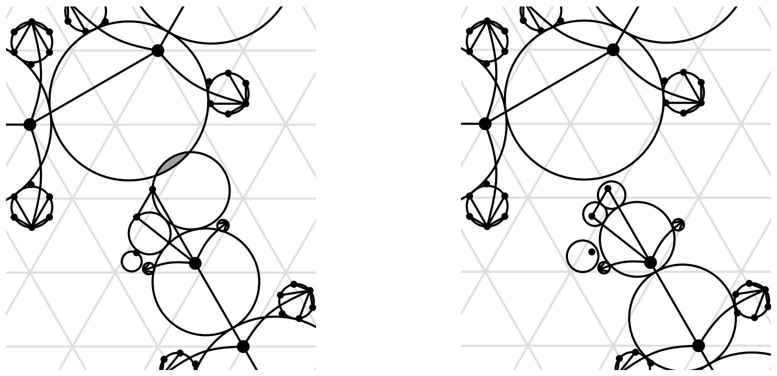


Fig. 8: Two chain links (black and gray) and the graph structure around p

Variable gadgets. By putting chain links together such that they touch their neighbors, we construct variable gadgets as shown in Figure 9. Each variable is represented by a chain of pairs of successive left and right turns of 60 degrees. Note that the turns of 60 degrees adhere to the grid. Once the covering disk of the first seed is fixed to one of the two possibilities, *all* successive disks in that chain are fixed because they have to touch their predecessor as well as their two stopper elements. We define the black variable configuration in Figure 9a as *true* and the gray one as *false*. At the bends of the variable gadget, literal chains can connect from above or below as depicted. If a literal has the value *false*, the covering disks of the literal chain are pulled towards the variable (see the black configuration of the lower literal chain). Otherwise both configurations of the literal chain are possible and we may choose the one where the covering disks are pushed away from the variable chain (see the black configuration of the upper literal chain). Figure 9b shows two close-up views of the truth value transfer for a negative literal connecting to the variable gadget in its *true* state. (Note that for positive literals, the three special seeds at the end of the literal chain are mirrored at the main axis of the literal chain.) The left-hand side of Figure 9b is an invalid configuration because one of the final disks of the literal chain intersects a covering disk of the variable gadget. Only the configuration on the right-hand side, where the covering disks are pulled towards the variable gadget and hence encode the value *false* correctly, is valid.



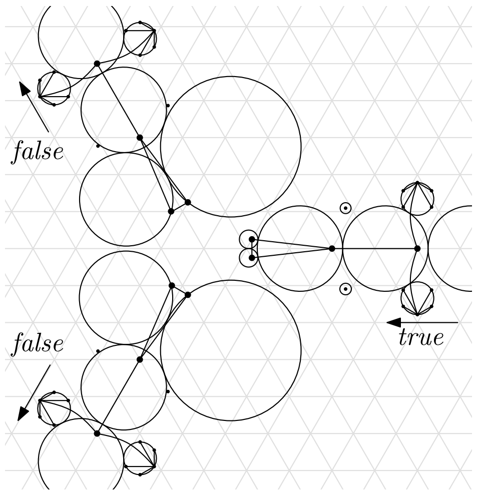
(a) Gadget for a variable x in *true* state (black cover) and in *false* state (gray cover).



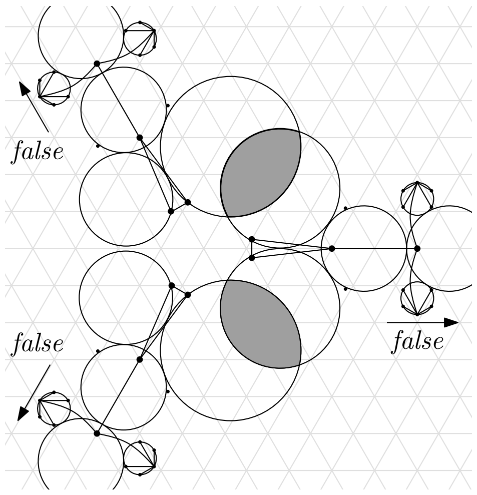
(b) Close-up view of the value transfer from variable chain to literal chain (see selection box in subfigure (a)). The left configuration has an invalid disk overlap; the right configuration is valid.

Fig. 9: Variable gadget

Clause gadgets. Figure 10 depicts the clause gadget. The three literal chains are meeting symmetrically at angles of 120 degrees. At the end of each chain, the graph G contains a triangle. Hence, the corresponding seeds must be covered by three disks such that each pair of disks touches. If a literal evaluates to *true*, that is, the covering disks of the literal chain are pushed towards the clause gadget, then the last disk of the chain touches the disks of the other two seeds in the vicinity of their position. In that case, the overall area used by the three disks of the triangle is relatively small (see the literal gadget on the right-hand side in Figure 10a). If, on the other hand, the literal evaluates to *false*, the disks of the literal chains are pulled towards the variable gadget. Hence, the last disk of the chain is pulled away from the other two seeds of the triangle; these two disks, consequently, need to grow strongly. Figure 10 shows that two false literals can still be accommodated (Figure 10a), whereas three false literals clearly cannot (Figure 10b).



(a) Clause evaluates to *true*.



(b) Clause evaluates to *false*.

Fig. 10: Clause gadget.

Reduction. From the construction of the gadgets we have the following:

(i) Each variable gadget realizes its respective subgraph by one of two combinatorially different configurations, which represent the truth values *true* and *false*.

(ii) Each literal gadget similarly realizes its subgraph in one of two configurations; if the literal evaluates to *false*, only one configuration is possible (otherwise two disks would intersect in their interior), if the literal evaluates to *true*, both options are possible.

(iii) Each clause gadget is a space-restricted area that can accommodate the covering disks of at most two false literal gadgets, otherwise there will be some disk overlap.

This concludes the proof of the reduction since we have shown that the constructed graph can be realized on the constructed seeds if and only if the corresponding planar Boolean 3SAT formula is satisfiable. We have embedded all seeds, except those belonging to stopper elements, on a hexagonal grid. By refining this grid and slightly perturbing the seeds of stopper element we can achieve that all seeds are on a grid. This grid has polynomial size since the variable-clause graph of φ is embeddable on a grid whose size is quadratic in the length of φ. Hence the reduction takes polynomial time. □

2.3 Enumeration

Every graph that can be realized as CCG is planar. Hence, the number of n-vertex graphs that can be realized as CCG is upperbounded by the number g_n of n-vertex planar graphs. The asymptotic growth of g_n has been determined by Gimenez and Noy [GN09]. They show that

$$g_n \sim g \cdot n^{-7/2} \gamma^n n!$$

where $g \approx 0.497 \cdot 10^{-5}$ and $\gamma \approx 27.2$ are constants given by explicit analytic expressions.

3 Seeds are Points on a Line

In this section, we restrict the seeds to be points on the x-axis and consider covers in the plane $\mathbb{R}^2$ as well as in the upper half plane $\mathbb{R}^2_+$. Since the connectivity question is answered by our realizability results, we focus on the latter problem and then give some enumeration results.

3.1 Realizability

We consider the following four questions. Note that seeds now correspond to real numbers, so we can use the natural order $<$ in $\mathbb{R}$ to compare them.

Q1. Given a graph class $\mathcal{C}$ (for example, the class of trees), does it hold that for any seed set S there is a graph in $\mathcal{C}$ that is realizable as CCG or CCG$^+$ on S?

We show: This is true for the combinations (cycles, CCG) and (trees, CCG$^+$), both for disk and triangle covers.

Q2. Given a graph class $\mathcal{C}$, does it hold that for any graph G in $\mathcal{C}$ there is a seed set S such that G can be realized as CCG or CCG$^+$ on S?

We show: This is true for the combination (trees, CCG$^+$), both for disk and triangle covers, where the triangles must have a unique bottom vertex.

Q3. Let $\mathcal{C}$ be a fixed graph class. Given a graph $G \in \mathcal{C}$ with a labeling $\lambda\colon V \to \{1, \dots, n\}$ of its vertices, is there a sequence $s_1 < \dots < s_n$ of seeds on the x-axis and a realization of G as CCG or CCG$^+$ that maps each vertex v to the corresponding seed $s_{\lambda(v)}$?

We show: There is an $O(n \log n)$-time decision algorithm for the combination (trees, disk-CCG$^+$).

Q4. Let $\mathcal{C}$ be a fixed graph class. Given a seed set S and a graph $G = (V, E) \in \mathcal{C}$ with a one-to-one correspondence between S and V, can G be realized on S as CCG or CCG$^+$?

We show: There is an $O(n \log n)$-time decision algorithm for the combination (trees, triangle-CCG$^+$). Even if the correspondence between seeds and vertices is not fixed, there are pairs of seed sets and trees that cannot be realized as triangle-CCG$^+$.

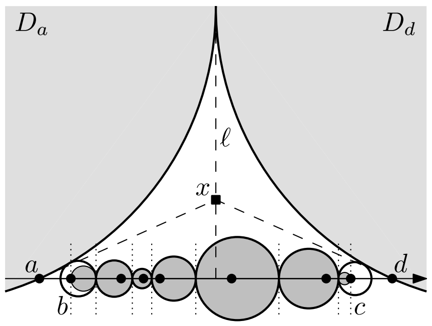
The above questions vary in the amount of information they require about the seed set, ranging from no information (Q2) via a fixed order (Q3) to complete information (Q1 and Q4).

Question Q1. We start with question Q1, which also answers the connectivity problem.

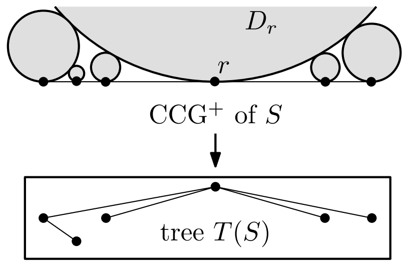
Proposition 3.1. *Let S be a set of n point seeds on a line, then*

(i) the n-vertex cycle can be realized as CCG on S, and
(ii) there is a tree $T(S)$ that can be realized as CCG$^+$ on S.

Proof. (i) Let S be ordered from left to right and let a, b, c, and d be the first, second, second last, and last point in S, see Figure 11a. Consider the one-dimensional Voronoi diagram of S. We shift the first Voronoi point between a and b to b and the last point between c and d to c. The resulting Voronoi points are marked by vertical dotted segments in Figure 11a. Each finite cell of the resulting diagram is a segment of the x-axis and contains a seed. We cover the seed by a disk whose diameter is the segment. The resulting disks are shaded in dark gray in Figure 11a. Clearly each seed in $S \setminus \{a, d\}$ is now covered by a disk that touches the disk of its predecessor and the disk of its successor (where those exist). Choose a point x on the perpendicular bisector ℓ of a and d such that all (closed) disks lie in the interior of quadrangle $(a, x, d, -x)$. Cover a and d by two congruent disks D_a and D_d that touch the quadrangle in a and d, respectively, and touch each other on ℓ. These two disks (light gray in Figure 11a) do not touch any of the other $n-2$ disks yet. In order to achieve this, we enlarge the disk D_b of b by moving the left endpoint of its diameter towards a until D_b touches D_a. The disk D_c of c is enlarged analogously towards d. This closes the cycle.



(a) Any cycle can be realized as CCG.



(b) tree $T(S)$ is realizable as CCG$^+$.

Fig. 11: Graphs that can be realized on a given one-dimensional n-point seed set S.

A similar construction that first connects the points from b to c by a path and then closes the cycle from a to d can be used for homothetic triangles as cover elements.

(ii) We pick any seed r as root and cover it by a disk D_r whose projection on the x-axis contains all seeds. Then we grow a disk from each seed until it touches one of the previously placed disks, see Figure 11b. A cycle can appear only if a new disk accidentally touches more than one previously placed disk. In this case, we increase the radius of D_r by a randomly chosen $\varepsilon > 0$ and repeat the process. Then the probability of constructing a tree is 1.

For triangular cover elements we start with a large triangle placed with its bottom-most vertex at the leftmost seed such that its projection on the x-axis contains all seeds (apply the same idea from right to left if the triangle is tilted to the left). Now we iteratively grow a triangle from the next seed until it touches one of the earlier triangles with its top left vertex. Note that the top right vertex can never touch a previous triangle and hence the CCG$^+$ obtained is a tree. In the special case of triangles with a horizontal bottom edge we always place the bottom left vertex at the seed and grow the triangle until the bottom right vertex touches the next seed in order to obtain a path (and thus a tree). □

Question Q2. In terms of this paper, a coin graph is obtained when seeds are points and cover elements are disks centered at seeds, and thus Koebe's theorem establishes that it is always possible to choose seeds in the plane such that any given plane graph is realizable as a coin graph on them. We have seen in Proposition 3.1 that any cycle is realizable as a CCG on any seed set on a line. One can ask whether a Koebe-type theorem also holds in this restricted setting. Kaufmann and Wiese [KW02] have shown, however, that there is a plane triangulated 12-vertex graph (see Figure 12) that cannot be drawn with at most one bend per edge if vertices are restricted to a line. Now Proposition 2.1 (ii) implies that this graph is not realizable as CCG if the seeds lie on a line. On the positive side, we can show that a Koebe-type theorem holds for the combination (trees, CCG$^+$). This is an answer to Q2 and in a way dual to Proposition 3.1 (ii). See Figure 13 for a sketch of our recursive construction.

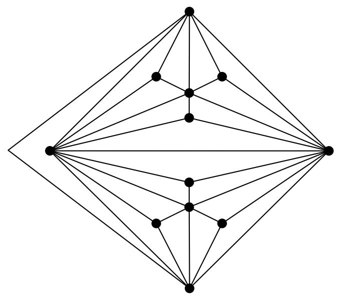


Fig. 12: Kaufmann–Wiese graph [KW02].

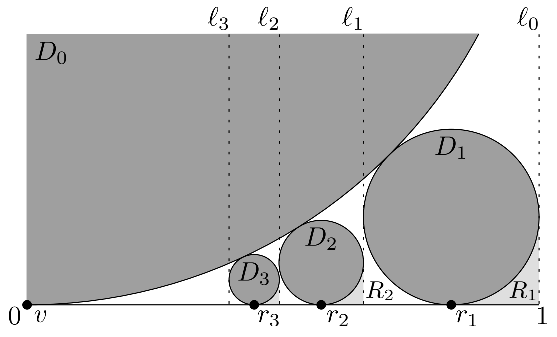


Fig. 13: Constructing a seed set $S(T)$.

Proposition 3.2. *For any tree T there is a seed set $S(T) \subset \mathbb{R}$ such that T is realizable as CCG$^+$ on $S(T)$.*

Proof. Our construction is recursive. We traverse the vertices of T in breadth-first order. We map the root v of T to 0 and cover it by a disk D_0 of radius 1, see Figure 13. If v has no children, we are done. Otherwise let $r_1, \dots, r_k$ be the children of v. Define ℓ_0 to be the line $x = 1$. Note that ℓ_0 touches D_0. Now for each $i = 1, \dots, k$ do the following. Place the largest disk D_i that fits into the region delimited by D_0, the x-axis, and the line ℓ_{i-1}. Map r_i to the lowest point of D_i, that is, the point where D_i touches the x-axis. Note that D_i also touches ℓ_{i-1} on its right-hand side. Define ℓ_i to be the left vertical line that touches D_i and continue with the next child of v. Figure 13 shows the placement of the seeds for the three children r_1, r_2, r_3 of v and their covering disks D_1, D_2, D_3.

It is not hard to see that we can place an arbitrary number of children of v in this way. By construction, the disks of any two children of v are disjoint, and they all lie in the region R_0 delimited by D_0, ℓ_0, and the x-axis. For $i = 1, \dots, k$ we define the region R_i delimited by D_i, ℓ_{i-1} and the x-axis. These regions are shaded in light gray in Figure 13. Each of them is similar to R_0. Thus we can recursively repeat the process of placing the children of v in R_0 for the children of $r_1, \dots, r_k$ in the respective regions $R_1, \dots, R_k$. These regions are pairwise disjoint, so the disks of two different grandchildren of v do not intersect.

The same idea can also be used to show the result for cover elements that are homothetic triangles with a unique bottom vertex. $\square$

Question Q3. In Proposition 3.2 above, we had complete freedom to choose the seeds. Now we turn to question Q3, where we are not just given a tree, but also an order of its vertices that must be respected by the corresponding seeds. Kaufmann and Wiese [KW02] have investigated a related problem. They showed that it is NP-complete to decide whether the vertices of a given (planar) graph can be put into one-to-one correspondence with a given set of points on a line such that there is a plane drawing of the graph with at most one bend per edge. We call such a drawing a 1d-1BD. If additionally all bends lie on one side of the line, we call the drawing a 1d-1BD$^+$. Note that a 1d-1BD of a graph G can be seen as a two-page book embedding [CLR87], where the edges drawn below the line that contains the vertices (called the *spine* in book embeddings) correspond to edges on one page while the edges above the spine correspond to edges on a second page. Similarly, a 1d-1BD$^+$ of G can be considered as a one-page book embedding.

Note that the hardness result of Kaufmann and Wiese does not yield the hardness of the one-dimensional CCG realizability problem since not every graph that can be one-bend embedded on a set of points on a line is realizable as CCG, let alone as CCG$^+$. Our next result explores the gap between Kaufmann and Wiese's one-dimensional embeddability problem and the situation in Proposition 3.2.

More formally, given an n-vertex tree T and a (bijective) labeling $\lambda \colon V \to \{1, \dots, n\}$ of its vertices, we say that T is λ-*realizable* (as CCG, CCG$^+$, 1d-1BD, 1d-1BD$^+$) if there is a sequence $s_1 < \dots < s_n$ of seeds in $\mathbb{R}^1$ and a realization of T (as CCG, CCG$^+$, 1d-1BD, 1d-1BD$^+$) that maps each vertex v to the corresponding seed $s_{\lambda(v)}$.

In order to obtain a characterization of trees that are λ-realizable as CCG$^+$, we need the following definition. Given a graph $G = (V, E)$ with vertex labeling λ, a *forbidden pair* is a pair of edges $\{\{a, b\}, \{c, d\}\}$ such that $\lambda(a) < \lambda(c) < \lambda(b) < \lambda(d)$. Note that it is impossible to embed the edges of a forbidden pair simultaneously above the x-axis.

Theorem 3.1. *For a λ-labeled tree T, the following statements are equivalent:*

(i) T is λ-realizable as a CCG$^+$.
(ii) T is λ-realizable as a 1d-1BD$^+$.
(iii) T does not contain any forbidden pair.

Proof. We first show that (i) and (ii) and then that (ii) and (iii) are equivalent.

(i) $\Rightarrow$ (ii): Given a λ-realization of T as CCG$^+$ on some seed set $S \subset \mathbb{R}^1$, we can use the idea of Proposition 2.1 (ii) to get a one-bend embedding in $\mathbb{R}^2_+$ on the same seed set by drawing each edge as the two-segment polyline from the first seed via the point of tangency of the corresponding covering disks to the second seed.

(ii) $\Rightarrow$ (i): Now we are given a λ-realization of T as 1d-1BD$^+$ on some seed set $S \subset \mathbb{R}^1$. We say that v is a *free* vertex of T if T has no edge $\{u, w\}$ with $\lambda(u) < \lambda(v) < \lambda(w)$. Note that T has at least two (but potentially more) free vertices, namely those with λ-values 1 and n. Pick any free vertex v as root of T. Represent v by a seed at the origin and cover that seed by the unique disk D_0 of radius 1 in $\mathbb{R}^2_+$ touching the origin. The root v partitions $T \setminus \{v\}$ into two (possibly empty) parts. The vertices in the right part have λ-values greater than $\lambda(v)$, and the vertices in the left part, have λ-values less than $\lambda(v)$. Let R_0 and R'_0 be the two regions delimited by the x-axis, D_0, and the vertical lines $x = 1$ and $x = -1$, respectively.

Now we place all seeds and the cover of the right part and the left part in R_0 and R'_0, respectively. Let the children of v in the right part be $r_1, \ldots, r_t$ ordered by decreasing λ-value. We place them from right to left as in the proof of Proposition 3.2 (see Figure 13), but with tiny gaps in between the left vertical tangency line of disk D_i and the right vertical tangency line of D_{i+1} for $i = 1, \ldots, t-1$. The children $l_1, \ldots, l_{t'}$ of v in the left part are defined and placed symmetrically (with respect to the line $x = 0$).

From the 1d-1BD$^+$ drawing of T we obtain that each subtree T_u rooted at a descendant u of a vertex v forms an interval in the sequence of the λ-values: Assume to the contrary that there is a descendant u of v, a non-descendant w of u, and two vertices u' and u'' in T_u with $\lambda(u') < \lambda(w) < \lambda(u'')$. Then w cannot be a free vertex in $T_u \cup \{w\}$ since T_u is connected. Hence, either T is disconnected or the 1d-1BD$^+$ of T is not plane, both of which is a contradiction.

Next we show that each child u of the root v is a free vertex in the subtree T_u rooted at u. Assume to the contrary that there are two descendants w and w' of u connected by an edge $\{w, w'\}$ such that $\lambda(w) < \lambda(u) < \lambda(w')$, that is, u is not free in T_u. But since we know that the subtree T_u forms an interval in the λ-values of T and that $\{v, u\}$ is an edge in T it follows that the edges $\{v, u\}$ and $\{w, w'\}$ must cross, which is a contradiction. So u is indeed a free vertex in T_u and we can recurse for each child u of v using the region defined by the disk of u, its vertical tangents, and the x-axis.

(ii) $\Rightarrow$ (iii): Trivial.

(iii) $\Rightarrow$ (ii): We map each vertex v of T to the seed $\lambda(v) \in \mathbb{R}^1$. Let all edges be directed from left to right with respect to the seed ordering on the x-axis and let v_i ($i = 1, \ldots, n$) be the vertex in T that is mapped to seed $\lambda(v_i) = i$. We insert the edges iteratively by increasing order of the source vertices. For each $i = 1, \ldots, n$ we draw any outgoing edge $\{v_i, v_j\}$ as the two-segment polyline whose bend is placed at position $(i+1/2, (j-i)/2^i)$. Figure 14 shows an example of this construction. Clearly, all edges with the same source do not intersect (except at their common vertex). It remains to show that none of the previously inserted edges is intersected by a new edge $\{v_i, v_j\}$. Since there are no forbidden pairs, it holds that for any previously inserted edge $\{v_k, v_l\}$ and the current edge $\{v_i, v_j\}$, either the four seeds are ordered as $k < l \leq i < j$, in which case the two edges obviously do not intersect, or they are ordered as $k < i < j \leq l$. The slope of the right leg of $\{v_k, v_l\}$ in the drawing is

$$-\frac{1}{2^k} \cdot \frac{l-k}{l-k-1/2} < -\frac{1}{2^k}$$

and the slope of the right leg of the new edge $v_i v_j$ is

$$-\frac{1}{2^i} \cdot \frac{j-i}{j-i-1/2} \geq -\frac{1}{2^{i-1}}.$$

Since $k \leq i-1$ this means that the right leg of any previously inserted edge is steeper than the right leg of $\{v_i, v_j\}$. Moreover, the bend of any previously inserted edge is to the left of i. Hence edge $\{v_i, v_j\}$ does not intersect any other edge when it is inserted. Once all edges are drawn, we have found a valid 1d-1BD$^+$ for T. $\square$

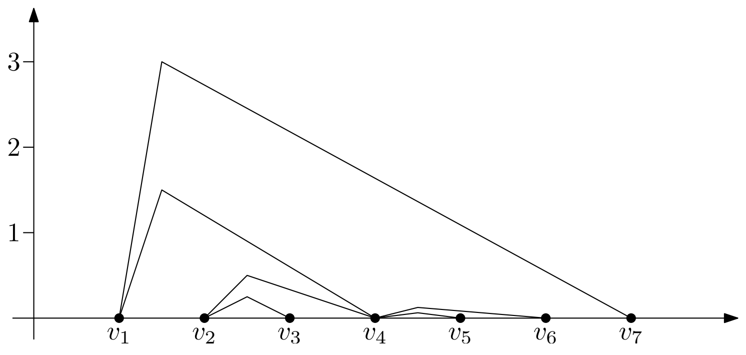


Fig. 14: Incremental 1d-1BD$^+$ drawing for a tree without forbidden pairs.

Given the tree, statement (iii) can be checked in $O(n \log n)$ time using an interval tree, therefore we immediately obtain the following corollary.

Corollary 3.1. *Given a λ-labeled tree T, we can decide in $O(n \log n)$ time whether T is λ-realizable as CCG$^+$.*

Question Q4. Now we are given a set S of seeds, a tree $T = (V, E)$, and a bijection between S and V that assigns each vertex to a seed. Our aim is to device a decision algorithm for the realizability of T as a triangle-CCG$^+$ on S.

We call a family of homothetic triangles *V-shaped* if each triangle is symmetric to a vertical line and its bottommost vertex is unique. In the following, we will consider only V-shaped triangles. First, note that there are pairs of trees and seed sets for which the answer to question Q4 is negative—even if the mapping between vertices and seeds is not fixed in advance.

Observation 3.1. *There is a complete binary tree T and a seed set $S \subset \mathbb{R}$ with $|S| = |V(T)|$ such that T is not realizable as a triangle-CCG$^+$ on S.*

Proof. Figure 15 shows a complete binary tree T on seven vertices and the seven-point seed set $S = \{a(0), b(2), c(5), d(11), e(13), f(16), g(33)\} \subset \mathbb{R}$. A case distinction on the seed that represents the root vertex 1 shows that it is not possible to find a representation of T as triangle-CCG$^+$ on S. The example in Figure 15 shows the case where seed g represents the root. In this case, any two covers of points in $S \setminus \{g\}$ that touch the covering triangle of g will overlap, even the covering triangles of the most distant seeds a and f. Hence, if g is the root, it is impossible to attach two children to the root. The other cases can be treated similarly. □

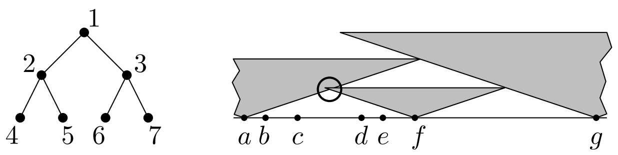
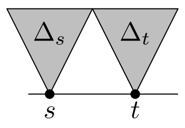
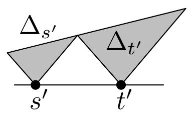


Fig. 15: Example of a binary tree T and a seed set $S = \{a(0), b(2), c(5), d(11), e(13), f(16), g(33)\}$ such that T is not realizable as triangle-CCG$^+$ on S.

On the other hand, there is always a tree that can be realized on a given set of seeds as Proposition 3.1 (ii) shows. Below we give an algorithm that decides this realizability question in $O(n \log n)$ time. We say that a set of points on a line is in *general position* if no point is equidistant to two other points, and we consider only such point sets. Furthermore, we define the *atomic* V-shaped covering triangles for a given V-shaped family of triangles and a pair of seeds $\{s, t\}$ as the unique congruent triangles Δ_s and Δ_t placed at s and t that touch each other in a triangle vertex, see Figure 16a. The concept of atomic triangles can also be generalized to families of non V-shaped covering triangles. Here the atomic triangles for two seeds $\{s', t'\}$ are the unique homothetic (but not necessarily congruent) triangles $\Delta_{s'}$ and $\Delta_{t'}$ placed at s' and t' that touch in a triangle vertex, see Figure 16b.



(a) V-shaped triangles.



(b) general case.

Fig. 16: Atomic triangles.

Algorithm 1 generates a cover that realizes a triangle-CCG$^+$ on the given seeds. Note that this graph is not unique as the choice of the seed u in line 6 is arbitrary. It is not hard to see that the triangle-CCG$^+$ obtained by Algorithm 1 is indeed a tree due to our assumption that seeds are in general position. Algorithm 1 yields the following result.

Algorithm 1: V-shaped triangle-CCG$^+$ on fixed seeds realizing a tree.

```
    Input: seed set $S$ with at least two seeds
    Output: cover $\mathcal{C}$ of $S$
 1  initialize $L \leftarrow S$
 2  initialize $\mathcal{C} \leftarrow \emptyset$
 3  while $|L| > 2$ do
 4      $\{s, t\} \leftarrow$ closest pair in $L$
 5      $\{\Delta_s, \Delta_t\} \leftarrow$ pair of atomic V-shaped triangles for $\{s, t\}$
 6      choose $u \in \{s, t\}$
 7      add $\Delta_u$ to $\mathcal{C}$
 8      delete $u$ from $L$
 9  end
10  add both atomic triangles for $L = \{s, t\}$ to $\mathcal{C}$
11  return $\mathcal{C}$
```

Theorem 3.2. *Given a set $S \subset \mathbb{R}^1$ of seeds in general position and a tree T with a fixed seed assignment for each vertex, we can decide in $O(n \log n)$ time whether T can be realized as a V-shaped triangle-CCG$^+$ on S.*

Proof. We construct a cover that realizes T edge by edge. We start with T and an empty cover $\mathcal{C}$. Observe that the closest pair of seeds must form an edge of T, otherwise the triangle-CCG$^+$ of $\mathcal{C}$ would not be connected. Furthermore, one of the vertices of this edge must be a leaf since only one of the atomic triangles can grow any further. Thus, we determine the closest pair of seeds $\{s, t\}$ and check whether $\{s, t\}$ is a leaf edge of T. If this is not the case, we answer "no". Otherwise we choose the seed u in line 6 of Algorithm 1 as the leaf vertex of edge $\{s, t\}$. Now Algorithm 1 adds the atomic triangle Δ_u for u to $\mathcal{C}$ and removes u from L in line 8. Accordingly, we delete the leaf u and its incident edge in T. By induction the closest pair in L again corresponds to a leaf edge of the modified tree T and we repeat the above process. If the construction terminates without answering "no" in one of the iterations, it is clear that we have constructed a cover $\mathcal{C}$ that represents T as a triangle-CCG$^+$and we answer "yes".

Finding the closest pair of seeds in line 4 is the time-critical part of Algorithm 1 and can be done by maintaining a priority queue that is initialized with the distances of all neighboring seeds. Every time the closest pair is extracted we perform one deletion and one insertion in $O(\log n)$ time in order to reflect the removal of the seed u. This takes $O(n \log n)$ total time. □

From Theorem 3.2 it is clear that Algorithm 1 can be used to generate *all* trees that can be realized as triangle-CCG$^+$ on S and thus we obtain the following corollary.

Corollary 3.2. *Let $S \subset \mathbb{R}^1$ be a set of seeds in general position, let $\mathcal{T}(S)$ be the set of trees that are realizable on S as CCG$^+$ with homothetic V-shaped triangles as cover elements, and let $\mathcal{M}(S)$ be the set of trees that can be obtained by Algorithm 1. Then $\mathcal{T}(S) = \mathcal{M}(S)$.*

Proof. As all triangle-CCG$^+$'s obtained by Algorithm 1 are trees that are realizable on S it remains to show that $\mathcal{T}(S) \subseteq \mathcal{M}(S)$. So consider any tree $T \in \mathcal{T}(S)$ and its realization as triangle-CCG$^+$. Since covering objects are V-shaped homothetic triangles the y-coordinate of the contact point between two triangles is proportional to the distance of the covered seeds. We sort the triangles in the realization of T in increasing order by their height and impose the same order on the associated seeds. Then we apply Algorithm 1 under the constraint that u (and the triangle Δ_u) in line 6 is chosen according to this ordering of the seeds. It follows that in each iteration the point covered by the lowest remaining triangle in the given triangle-CCG$^+$ of T is a point of the closest pair. Hence the algorithm is able to reconstruct T. □

Remark 3.1. Note that the restriction to V-shaped triangles in Theorem 3.2 and Corollary 3.2 is made only for ease of presentation. The results for V-shaped triangle families can immediately be extended to families of homothetic triangles whose top sides are parallel to the x-axis. Even in the case of families of arbitrary homothetic triangles with a unique bottom vertex analogous results hold. Here, however, the priority queue to select the next pair of neighboring seeds is not based on the seed distances but on the points in time when a sweep line parallel to the top side of the triangles reaches the top edges of the atomic triangles of neighboring seeds. So instead of finding the closest seed pair in line 4 of Algorithm 1, the modified algorithm always picks the pair of seeds whose atomic triangles are reached next by the sweep line (see Figure 17) and otherwise proceeds exactly the same.

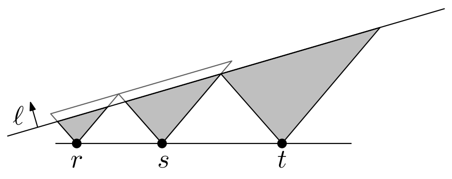


Fig. 17: The sweep line ℓ reaches the top sides of the atomic triangles of $\{s, t\}$ before reaching the top sides of the atomic triangles of $\{r, s\}$ although $|r - s| < |s - t|$.

3.2 Enumeration

The keys to enumerating the CCG's and CCG$^+$'s on sets of point seeds on a line are some of the earlier results on realizability in this section. Again, our results depend on how much information we have about the seeds.

As a consequence of Proposition 3.2, which states that for any tree T there is a seed set on which T is realizable as CCG$^+$, we know that the number of labeled trees that are realizable as CCG$^+$ (or CCG) is the total number of labeled trees. So, we get the following result as a consequence of Cayley's formula for the number of labeled trees.

Proposition 3.3. *The number of labeled trees with n vertices realizable as CCG$^+$ using point seeds on a line is n^{n-2}.*

In our next enumeration result, we establish the link between completely parenthesized factorizations of n natural numbers, which are described by the n-th Catalan number, and labeled n-vertex trees that are realizable as CCG$^+$ if the order of the seeds is given. For a seed set $S \subset \mathbb{R}$, let the *rank* of a seed $s \in S$ be the index of s when S is in sorted order.

Theorem 3.3. *The n-th Catalan number $\binom{2n}{n}/(n+1)$ is a lower bound on the number of labeled $(2n+1)$-vertex trees that can be realized as CCG$^+$ on some seed set $S \subset \mathbb{R}$ such that vertex labels correspond to seed ranks.*

Proof. Recall that in a *full* binary tree every vertex has either two children or no children. We consider the interpretation of the n-th Catalan number as the number of full binary trees with n internal vertices [Knu97]. A full binary tree with n internal vertices has $n+1$ leaves and hence, $2n+1$ vertices in total. We label the vertices with the numbers $1, \dots, 2n+1$ inorder. Given this labeling, we place the corresponding disks recursively, starting with the root, as in the proof of Proposition 3.1. For an example, see Figure 18. The seeds are simply the points where the disks touch the x-axis. Since the inorder labeling has the binary-search-tree property [CLRS09], it is clear that vertex labels correspond to seed ranks. $\square$

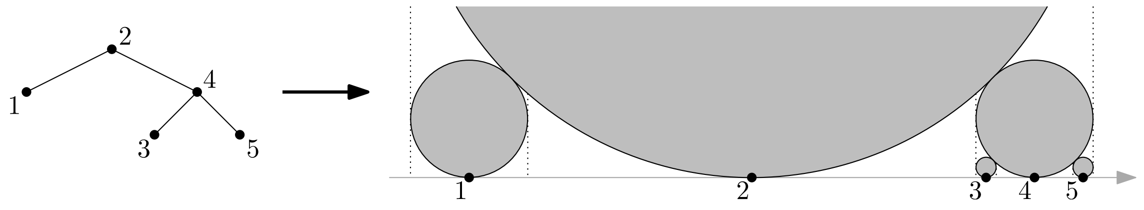


Fig. 18: Every full binary tree can be realized as a CCG$^+$ such that, for each vertex, its inorder label and the rank of the corresponding seed coincide.

The $(2n+1)$-vertex path is not a full binary tree and thus not counted by the n-th Catalan number, but it is still a tree realizable as a CCG$^+$; for example, on the seed set $\{1, 2, \dots, 2n+1\}$. This shows that the n-th Catalan number is indeed only a lower bound as stated in Theorem 3.3.

Finally, we consider a fixed seed set S. In Corollary 3.2 and Remark 3.1 we established that every tree that is realizable as triangle-CCG$^+$ on S can be obtained by Algorithm 1. In each of the $n-2$ steps of this algorithm one of the triangles in the next atomic triangle pair defined by the closest pair of points (for V-shaped triangles) or reached by the sweep line (in the general case) is selected to be in the cover while the other one can grow further. So, we obtain the following result.

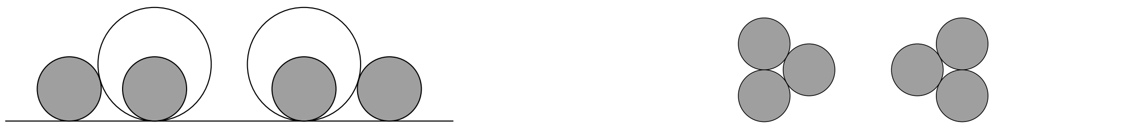
Proposition 3.4. *Given a set $S \subset \mathbb{R}$ of seeds in general position, the number of labeled trees realizable on S as CCG$^+$ with a fixed class of homothetic triangles (with unique bottom vertex) as cover elements is 2^{n-2}.*

4 Seeds are Disks or Triangles in the Plane

In this section, we consider a different class of seeds, namely disks or homothetic triangles in the plane. We cover them with the same kind of objects, that is, the covers for disks are disks and the covers for homothetic triangles are homothetic triangles. If the seeds are not points, the main difference is that the minimal size of each cover is bounded, so the results differ in many cases from those obtained in the previous sections when the seeds were points. We first consider seeds in $\mathbb{R}^2_+$ that are tangent to the x-axis (for triangles the bottom vertex is on the x-axis) and then describe how to translate the results to general seeds in the plane.

4.1 Connectivity

Unlike the connectivity results for points we can neither guarantee the existence of a connected CCG$^+$ for disk seeds tangent to the x-axis nor the existence of a connected CCG for disk seeds in the plane, see Figure 19. For homothetic triangles, however, the situation is different.


(a) Disks on the x-axis without connected CCG$^+$.

(b) Disks in the plane without connected CCG.

Fig. 19: Disk seeds that do not have a connected disk cover. Seeds are drawn in gray, covers in white.

Proposition 4.1. *Every seed set consisting of homothetic triangles touching the x-axis from above has a connected triangle-CCG$^+$.*

Proof. We consider the family of parallel lines induced by the triangle sides opposite to each bottom vertex of the seeds. Among these lines there is a line ℓ that contains all seed triangles in its lower half space; see Figure 20. We cover the seed s_ℓ belonging to ℓ by a big triangle such that the interval on the x-axis between the projections of the two vertices of the top edge along the direction of the respective opposite triangle sides contains all bottom vertices of the seeds, see the dotted projection lines in Figure 20. Then we inflate a covering triangle for each of the seeds until it touches one of the previous triangles. Due to the size of the first covering triangle, each inflated triangle eventually touches another covering triangle. Hence, the CCG$^+$ is connected. □

Our method for computing connected triangle-CCG$^+$'s can be extended to computing connected triangle-CCG's, again for homothetic triangles.

Corollary 4.1. *Every set of homothetic triangles has a connected triangle-CCG.*

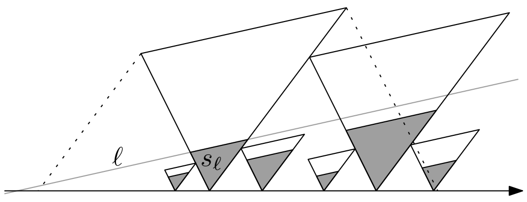


Fig. 20: Any set of homothetic triangles touching the x-axis from above has a connected triangle-CCG$^+$.

Proof. The initial seed to be covered is selected as before and the size of its covering triangle must be large enough such that all other seeds are contained in the (unbounded) region between the two projection lines. This guarantees that inflating covering triangles for the other seeds will always lead to a contact event. □

We cannot extend the result of Proposition 4.1 to higher degrees of connectivity. Assume that the seed triangles are small enough to be conceptually treated as point seeds. Then by arguments similar to those used in Theorem 3.2 only trees or forests can be realized as triangle-CCG$^+$'s on seeds in general position.

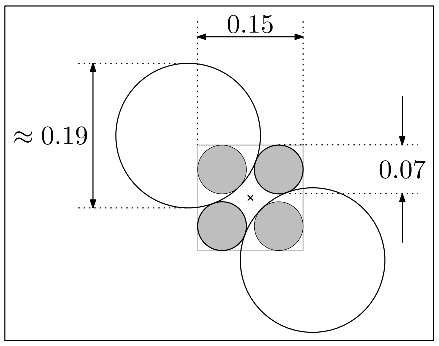
Somewhat surprisingly, the connectivity question turns out to be hard for *disk* seeds.

Theorem 4.1. *Given a set S of disk seeds, it is NP-hard to decide whether there is a connected CCG on S, even if there are only two different seed radii.*

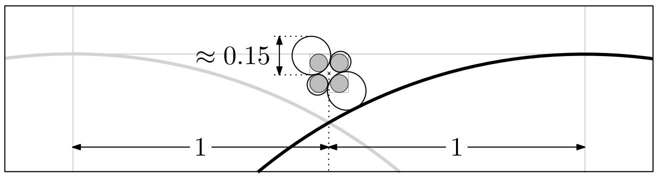
Proof. Our proof is by reduction from the problem PLANAR3SAT, which is NP-hard [Lic82]; it has been introduced in the proof of Theorem 2.3. Given a planar Boolean 3-CNF formula φ, we construct two types of gadgets, one for the variables of φ and one for the clauses of φ. Again we use that the variable-clause graph H_φ can be drawn on a grid of polynomial size with all variable vertices placed on a horizontal line and the clause vertices connecting in a comb-shaped manner from above or below that line [KR92].

First, as an important building block, we define a *stopper* consisting of four congruent disks of diameter 0.07 each of which touches two sides of a 0.15×0.15 square; see Figure 21a. We call the center point of the square the *reference point* of the stopper, see the large cross in Figure 21a. Observe that there are various ways how these four seeds can be covered; Figure 21 depicts some examples. Even in the extreme case (see Figure 21a), however, the diameter of any cover disk is at most ≈ 0.19, that is, roughly 2.75 times that of the seeds.

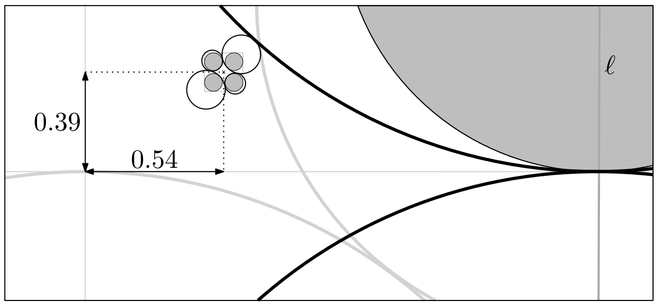
Variable gadgets. For each variable v of φ, we place a set of unit-disk seeds such that their centers lie on the x-axis. The centers of consecutive seed disks have distance 4; see Figure 22. Both below and above the unit-disk seeds, we place stoppers such that their reference points have distance 1.925 from the seed centers. We do the same below and above the midpoints of pairs of consecutive seeds. In this way, the two outer disks in the stoppers touch a horizontal line at distance 2 above or below the x-axis. (Due to their small size, the stoppers are only indicated by their bounding boxes in Figure 22; for an enlarged drawing, see Figure 21b.) There are two obvious connected covers of such a variable gadget:



(a) Dimensions.



(b) In a variable gadget.



(c) At the beginning of a literal wire.

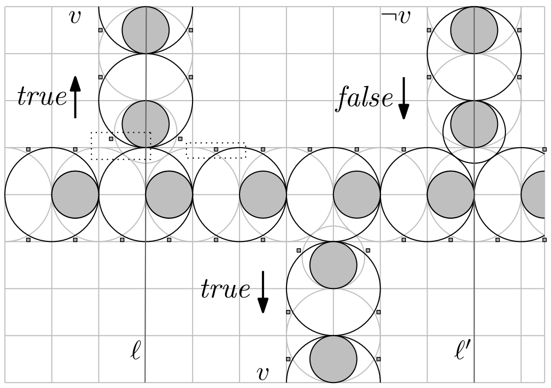
Fig. 21: The stopper.

all stoppers are covered as shown in Figure 21b (or mirrored) and the unit-disk seeds are covered by radius-2 disks all in a left position (as in Figure 22a) or all in a right position (as in Figure 22b). In the left position, seed and cover share their rightmost points; in the right position, they share their leftmost points.

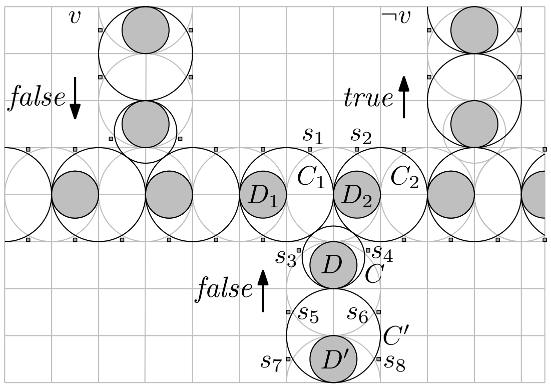
Note that there are more connected covers: the radius-2 disks can be slightly enlarged and rotated around their seeds, alternately clockwise and counterclockwise. By moving the stoppers slightly closer to the x-axis, the slack of the enlarged radius-2 disks can, however, be made arbitrarily small, so that we can safely ignore it. We need this slack in order to make sure that the stoppers have rational coordinates—otherwise our reduction would not be polynomial. Note that it is not possible that some covers are in the left position and some covers are in the right position; this would either cause two covers to overlap or two stoppers to be isolated.

The two (main) positions of the connected covers correspond to the values *true* and *false* of the variable v; see Figures 22a and 22b, respectively. In order to transmit the truth value into the clause gadgets, we attach vertical literal "wires" to the variable gadgets. For a positive literal, the wire is centered on the left vertical tangent (as ℓ in Figure 22a) of a seed of the variable gadget. For a negative literal, on the other hand, the wire is centered on a right vertical tangent (as ℓ' in Figure 22b). The wires consist of unit disk seeds flanked by stoppers as before, where the first seed of each wire is placed with its center at distance 3 from the x-axis. Additionally, we place two stoppers at distance 1.46 from ℓ (or ℓ') and at distance 2.39 above or below the x-axis; see the enlarged view in Figure 21c. In order to form a connected CCG, the covers of literal wires that are *false* are pulled towards the variable gadget and the covers of literals that are *true* are pushed away from the variable gadget as illustrated in Figure 22.

The only places where possibly something could go wrong are the unit-disk seeds



(a) Variable v is *true*; the two dotted boxes mark the position of the enlarged views in Figures 21b and 21c.



(b) Variable v is *false*.

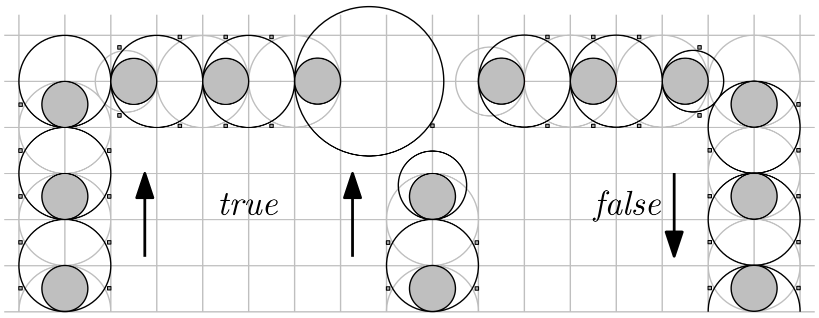
Fig. 22: The variable gadget on a grid with cells of size 2×2.

in the literal wires that are adjacent to the variable gadgets. As an example, consider the disk D in Figure 22b. Clearly, we cannot make the cover C of D smaller if we want to keep the connection with the variable gadget and stoppers s_3 and s_4.

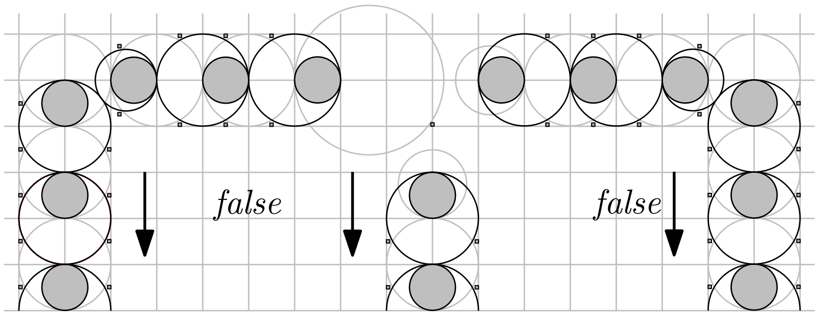
First, suppose that we enlarge C downwards. Obviously, C cannot grow much further before it overlaps the seeds of stoppers s_3 and s_4; it certainly cannot connect to stoppers s_5 and s_6. Hence, these two stoppers must be connected to the cover C' of the disk D' directly below D, which also must connect to the stoppers s_7 and s_8 that are aligned with D'. This exactly defines the target position of C' in the *false* state of the wire. Now suppose that we enlarge C upwards. Again, C cannot grow much before it overlaps the seeds of the stoppers s_3 and s_4. Furthermore, covers C_1 and C_2 must connect to stoppers s_1 and s_2, respectively. So apart from the usual slack of the covers, no structural changes are possible. Hence, C can be enlarged neither downwards nor upwards.

Clause gadgets. The clause gadget is depicted in Figure 23 and combines three literal wires in a comb-like shape with a stopper in the center. The left and right wires make a 90-degree turn, which works similarly as the connection of a literal wire to its variable gadget. (The correctness of the bend construction can be analyzed similarly as above.) The horizontal wire parts, too, consist of unit disk seeds flanked by stoppers—except for the last seed of each wire. The wires stop with the last seed that can be placed left of, right of, or below the central stopper.

Observe that in this construction, covers of wires that are pushed into the clause (transmitting the value *true*) connect to the central stopper if the final covering disk is inflated appropriately (see Figure 23a). Wires, on the other hand, that are pulled away from the clause (transmitting the value *false*) cannot connect to the central stopper. Inflating the final covering disk while keeping the contact to the previous disk in the wire would lead to an overlap with at least one of the two last stoppers of the wire (see Figure 23b). Hence, the central stopper of a clause gadget can be connected to the remaining graph if and only



(a) Two literals are *true* and one of them connects to the central stopper.



(b) All literals are *false* and the central stopper cannot be connected.

Fig. 23: The clause gadget.

if at least one of the wires transmits the value *true*. Note that it suffices if *one* of the literal wires connects to the central stopper.

Further note that the grid distance between the left and the middle vertical wire in Figure 23 is even, whereas the grid distance between the middle and the right vertical wire is odd. Due to the flexible size of the final covering disk of a literal wire, the disk can bridge a distance of one or two grid cells, as needed.

Reduction. In order to obtain a connected CCG, it is necessary to connect the stopper in the center of each clause to at least one literal wire. By construction, this is possible if and only if there is a variable assignment that satisfies all clauses. We further need to ensure that all the variable gadgets, which are placed on a horizontal line, are in the same connected component of the CCG. This can easily be achieved by placing a single disk seed between any two neighboring variable gadgets that can be covered by a disk that touches both neighboring gadgets irrespective of their truth values. (Alternately, we may assume that the variable-clause graph H_φ is connected.) Now it is easy to see that all variable gadgets and all literal wires are in a single connected component. Hence, the whole CCG is connected if and only if each clause gadget is also connected to that component by at least one literal wire. This is the case if and only if each clause is satisfied by the truth value assignment encoded by the variable gadgets.

The variable-clause graph H_φ of φ can be embedded on a grid whose size is quadratic in the size of φ. The number of seeds in our construction is $O(nm)$, where n is the number of variables and m is the number of clauses of φ. Hence, the unit grid on which we place the centers of the large disk seeds has polynomial size. We have taken care that the centers of the disks in the stoppers have rational coordinates with respect to this grid. This ensures that the reduction takes polynomial time. We have used only two different seed radii. □

4.2 Realizability and Enumeration

In the case of disks for seeds and covers, the minimal sizes of the covers given by the sizes of their associated seeds are not an obstacle to generalizing the realizability results of Section 2. Thus, as each point seed set gives rise to disk seed sets, the hardness of the realizability problem in Theorem 2.3 can be generalized to disk seeds. The necessary conditions for realizability in Proposition 2.1 can be adapted to disk seeds: simply take the centers of the disk seeds as the required point seeds.

Regarding enumeration, the bounds obtained for point seeds apply here as well. Clearly every CCG for disk seeds can be considered a CCG for point seeds, so the number g_n (see Section 2.3) remains an upper bound on the number of graphs realizable as disk-CCG's.

5 Open problems

This paper has opened a new field with many interesting questions.

1. We know that every 3-vertex graph can be represented as CCG on any set of three points. We have given an example of six points whose Delaunay triangulation is not representable as a CCG. What about plane geometric graphs with four or five vertices? Do they always have a representation?

2. Does any set of point seeds in convex position have a triangulation that can be represented as CCG?

3. We know that any set of point seeds has a 2-connected CCG. What about 3-connectivity?

4. Is it NP-hard to decide whether a set of disks touching a line has a connected CCG$^+$?

5. Is there an equivalent to Theorem 3.1 for CCG's instead of CCG$^+$'s, that is, can we characterize vertex-labeled trees that have a realization as CCG on a set of seeds on a line such that the realization respects the vertex order prescribed by the labeling?

6. What about other classes of seeds and covers?

References

[ABH$^+$06] Manuel Abellanas, Sergey Bereg, Ferran Hurtado, Alfredo García Olaverri, David Rappaport, and Javier Tejel. Moving coins. *Comput. Geom. Theory Appl.*, 34(1):35–48, 2006.

[AdCC$^+$08] Nieves Atienza, Natalia de Castro, Carmen Cortés, M. Ángeles Garrido, Clara I. Grima, Gregorio Hernández, Alberto Márquez, Auxiliadora Moreno, Martin Nöllenburg, José Ramon Portillo, Pedro Reyes, Jesús Valenzuela, Maria Trinidad Villar, and Alexander Wolff. Cover contact graphs. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *Proc. 15th Int. Symp. Graph Drawing (GD'07)*, volume 4875 of *Lect. Notes Comput. Sci.*, pages 171–182. Springer-Verlag, 2008.

[AdCH$^+$06] Manuel Abellanas, Natalia de Castro, Gregorio Hernández, Alberto Márquez, and Carlos Moreno-Jiménez. Gear system graphs. Manuscript, 2006.

[CCJ90] Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Math.*, 86(1–3):165–177, 1990.

[CDR07] Sergio Cabello, Erik D. Demaine, and Günter Rote. Planar embeddings of graphs with specified edge lengths. *J. Graph Algorithms Appl.*, 11(1):259–276, 2007.

[CG09] Jérémie Chalopin and Daniel Gonçalves. Every planar graph is the intersection graph of segments in the plane. In *Proc. 41st Annu. ACM Symp. Theory Comput. (STOC'09)*, pages 631–638, 2009.

[CLR87] Fan R. K. Chung, Frank Thomson Leighton, and Arnold L. Rosenberg. Embedding graphs in books: A layout problem with applications to VLSI design. *SIAM J. Algebra. Discr.*, 8(1):33–58, 1987.

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press and McGraw-Hill, 3rd edition, 2009.

[CS03] Charles R. Collins and Kenneth Stephenson. A circle packing algorithm. *Comput. Geom. Theory Appl.*, 25(3):233–256, 2003.

[For86] Steven Fortune. A sweepline algorithm for Voronoi diagrams. In *Proc. 2nd Annu. ACM Symp. Comput. Geom. (SoCG'86)*, pages 313–322, 1986.

[GN09] Omer Giménez and Marc Noy. Counting planar graphs and related families of graphs. In Sophie Huczynska, James D. Mitchell, and Colva M. Roney-Dougal, editors, *Surveys in Combinatorics 2009*, volume 365 of *London Math. Soc. Lect. Note Ser.*, pages 169–210. Cambridge Univ. Press, 2009.

[Hal80] William K. Hale. Frequency assignment: Theory and applications. *Proc. IEEE*, 68(12):1497–1514, 1980.

[HJLM93] Frank Harary, Michael S. Jacobson, Marc J. Lipman, and Fred R. McMorris. Abstract sphere-of-influence graphs. *Math. Comput. Model.*, 17(11):77–83, 1993.

[JLM95] Michael S. Jacobson, Marc J. Lipman, and Fred R. McMorris. Trees that are sphere-of-influence graphs. *Appl. Math. Lett.*, 8:89–93, 1995.

[Knu97] Donald E. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching.* Addison-Wesley, 1997.

[Koe36] Paul Koebe. Kontaktprobleme der konformen Abbildung. *Ber. Sächs. Akad. Wiss. Leipzig, Math.-Phys. Klasse*, 88:141–164, 1936.

[KR92] Donald E. Knuth and Arvind Raghunathan. The problem of compatible representatives. *SIAM J. Discr. Math.*, 5(3):422–427, 1992.

[KW02] Michael Kaufmann and Roland Wiese. Embedding vertices at points: Few bends suffice for planar graphs. *J. Graph Algorithms Appl.*, 6(1):115–129, 2002.

[Lic82] David Lichtenstein. Planar formulae and their uses. *SIAM J. Comput.*, 11(2):329–343, 1982.

[Mit92] Joseph S. B. Mitchell. L_1 shortest paths among polygonal obstacles in the plane. *Algorithmica*, 8:55–88, 1992.

[PA95] János Pach and Pankaj K. Agarwal. *Combinatorial Geometry.* John Wiley & Sons, New York, 1995. (Contains a proof of Koebe's theorem.)

[PW01] János Pach and Rephael Wenger. Embedding planar graphs at fixed vertex locations. *Graphs Combinator.*, 17(4):717–728, 2001.

[RT90] Jean-Marc Robert and Godfried T. Toussaint. Computational geometry and facility location. In *Proc. Int. Conf. Oper. Res. Manage. Sci.*, pages B1–B19, Manila, 1990.

[Sac94] Horst Sachs. Coin graphs, polyhedra, and conformal mapping. *Discrete Math.*, 134(1-3):133–138, 1994.

[Thu80] William P. Thurston. *The Geometry and Topology of 3-Manifolds.* Princeton University Notes, 1980.

[Tou88] Godfried T. Toussaint. A graph-theoretical primal sketch. In Godfried T. Toussaint, editor, *Computational Morphology: A Computational Geometric Approach to the Analysis of Form*, pages 229–260. North-Holland, 1988.

[Wel91] Emo Welzl. Smallest enclosing disks (balls and ellipsoids). In H. Maurer, editor, *New Results and New Trends in Computer Science*, volume 555 of *Lect. Notes Comput. Sci.*, pages 359–370. Springer-Verlag, 1991.