Turning Cliques into Paths to Achieve Planarity*

Patrizio Angelini[1], Peter Eades[2], Seok-Hee Hong[2], Karsten Klein[3], Stephen Kobourov[4], Giuseppe Liotta[5], Alfredo Navarra[5], and Alessandra Tappini[5]

[1] University of Tübingen, Germany angelini@informatik.uni-tuebingen.de
[2] The University of Sydney, Australia {peter.eades,seokhee.hong}@usyd.edu.au
[3] University of Konstanz, Germany karsten.klein@uni-konstanz.de
[4] University of Arizona, USA kobourov@cs.arizona.edu
[5] University of Perugia, Italy {name.surname}@unipg.it

Abstract. Motivated by hybrid graph representations, we introduce and study the following beyond-planarity problem, which we call h-Clique2Path Planarity: Given a graph G, whose vertices are partitioned into subsets of size at most h, each inducing a clique, remove edges from each clique so that the subgraph induced by each subset is a path, in such a way that the resulting subgraph of G is planar. We study this problem when G is a simple topological graph, and establish its complexity in relation to k-planarity. We prove that h-Clique2Path Planarity is NP-complete even when $h = 4$ and G is a 3-plane simple topological graph, while it can be solved in linear time, for any h, when G is 1-plane.

1 Hybrid Representations

A common problem in the visual analysis of real-world networks is that dense subnetworks create occlusions and hairball-like structures in node-link diagrams generated by standard layout algorithms, e.g., force-directed methods. On the other hand, different representations, such as adjacency matrices, are well suited for dense graphs but make neighbor identification and path-tracing more difficult [6, 10]. *Hybrid graph representations* combine different representation metaphors in order to exploit their strengths and overcome their drawbacks.

The first example of hybrid representation was the *NodeTrix* model [12], which combines node-link diagrams with adjacency-matrix representations of the denser subgraphs [3, 4, 12, 13]. Another example of hybrid representations are *intersection-link representations* [1]. In this model vertices are geometric objects and edges are either intersections between objects (*intersection edges*), or crossing-free Jordan arcs attaching at their boundary (*link edges*). Different types of objects determine different intersection-link representations.

In [1], *clique-planar* drawings are defined as intersection-link representations in which the objects are isothetic rectangles, and the partition into intersection-

* Research initiated at the Bertinoro Workshop on Graph Drawing (BWGD) 2018.

and link-edges is given in the input, so that the graph induced by the intersection-edges is composed of a set of vertex-disjoint cliques. The corresponding recognition problem is called Clique-planarity, and it has been proved NP-complete in general and polynomial-time solvable in restricted cases.

We study Clique-planarity when all cliques have bounded size. As proved in [1], the Clique-planarity problem can be reformulated in the terminology of *beyond-planarity* [5, 8], as follows. Given a graph $G = (V, E)$ and a partition of its vertex set V into subsets $V_1, \dots, V_m$ such that the subgraph of G induced by each subset V_i is a clique, the goal is to compute a planar subgraph $G' = (V, E')$ of G by replacing the clique induced by V_i, for each $i = 1, \dots, m$, with a path spanning the vertices of V_i. We call h-Clique2Path Planarity (for short, h-C2PP) the version of this problem in which each clique has size at most h; see Fig. 3.

We remark that the version of h-C2PP in which the input graph G is a *geometric graph*, i.e., it is drawn in the plane with straight-line edges, has been recently studied by Kindermann et al. [7] in a different context. The input of their problem is a set of colored points in the plane, and the goal is to decide whether there exist straight-line spanning trees, one for each same-colored point subset, that do not cross each other. Since edges are straight-line, their drawings are determined by the positions of the points, and hence each same-colored point subset can in fact be seen as a straight-line drawing of a clique, from which edges have to be removed so that each clique becomes a tree and the drawing becomes planar. They proved NP-completeness for the case in which the spanning tree must be a path, even when there are at most 4 vertices with the same color. This implies that 4-C2PP for geometric graphs in NP-complete. On the other hand, they provided a linear-time algorithm when there exist at most 3 vertices with the same color, which then extends to 3-C2PP for geometric graphs.

In this paper, we study the version of h-C2PP in which the input graph G is a *simple topological graph*, that is, it is embedded in the plane so that each edge is a Jordan arc connecting its end-vertices; by simple we mean that a Jordan arc does not pass through any vertex, and does not intersect any arc more than once (either with a proper crossing or sharing a common end-vertex); finally, no three arcs pass through the same point. Our main goal is to study the complexity of this problem in relation to the well-studied class of *k-planar graphs*, i.e., those that admit drawings in which each edge has at most k crossings [1, 2, 5, 11].

We observe that the NP-completeness of 4-C2PP for geometric graphs already implies the NP-completeness of 4-C2PP for simple topological graphs; also, though not explicitly mentioned in [7], it is possible to show that the instances produced by that reduction are 4-plane (see Appendix A). We strengthen this result by proving in Section 2 that 4-C2PP is NP-complete even for simple topological 3-plane graphs. On the positive side, we prove in Section 3 that the h-C2PP problem for simple topological 1-plane graphs can be solved in linear time for any value of h. We finally remark that the 2-SAT formulation used in [7] to solve 3-C2PP for geometric graphs can be easily extended to solve 3-C2PP for any simple topological graph. Due to space limitations, some proofs are in the Appendix; the corresponding claims are marked with [*].

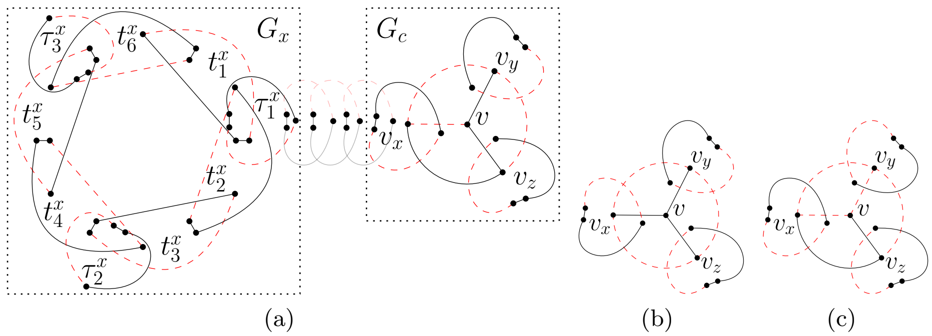


Fig. 1: (a) The variable gadget G_x for a variable x is represented in the left dotted box. The clause gadget for a clause c is represented in the right dotted box. The chain connecting G_x to G_c is represented with lighter colors. The removed edges are dashed red. (b) All variables are `False`. (c) At least two variables are `True`.

2 *NP*-completeness for simple topological 3-plane graphs

In this section we prove that the k-C2PP problem remains NP-complete for $k = 4$ even when the input simple topological graph is 3-plane.

Since the planarity of a simple topological graph can be checked in linear time, the h-C2PP problem for simple topological k-plane graphs belongs to NP for all values of h and k. In the following, we prove the NP-hardness by means of a reduction from the Planar Positive 1-in-3-SAT problem. In this version of the Satisfiability problem, which is known to be NP-complete [9], each variable appears only with its positive literal, each clause has at most three variables, the graph obtained by connecting each variable with all the clauses it belongs to is planar, and the goal is to find a truth assignment in such a way that, for each clause, exactly one of its three variables is set to `True`.

For each 3-clique we use in the reduction, there is a *base edge*, which is crossing-free in the constructed topological graph, while the other two edges always have crossings. We call *left* (*right*) the edge that follows (precedes) the base edge in the clockwise order of the edges along the 3-clique. Also, if an edge e of a clique does not belong to the path replacing the clique, we say that e is *removed*, and that all the crossings involving e in G are *resolved*.

For each variable x, let n_x be the number of clauses containing x. We construct a simple topological graph gadget G_x for x, called *variable gadget*; see the left dotted box in Fig. 1a. This gadget contains $2n_x + 2$ 3-cliques $t_1^x, \ldots, t_{2n_x+2}^x$, forming a ring, so that the left (right) edge of t_i^x only crosses the left (right) edge of t_{i-1}^x and of t_{i+1}^x, for each $i = 1, \ldots, 2n_x + 2$. Also, gadget G_x contains n_x additional 3-cliques, called $\tau_1^x, \ldots, \tau_{n_x}^x$, so that the right edge of τ_j^x crosses the left edge of t_{2j-1}^x and the right edge of t_{2j}^x, while the left edge of τ_j^x crosses the left edge of t_{2j}^x and the right edge of t_{2j-1}^x.

Then, for each clause c, we construct a topological graph gadget G_c, called *clause gadget*, which is composed of a planar drawing of a 4-clique, together with three 3-cliques whose left and right edges cross the edges of the 4-clique as in the right dotted box in Fig. 1a. In particular, observe that the right (left) edge of each 3-clique crosses exactly one (two) edges of the 4-clique.

Every 3-clique in G_c corresponds to one of the three variables of c. Let x be one of such variables; assuming that c is the j-th clause that contains x according to the order of the clauses in the given formula, we connect the 3-clique corresponding to x in the clause gadget G_c to the 3-clique τ_j^x of the variable gadget G_x of x by a chain of 3-cliques of odd length, as in Fig. 1a.

By construction, the simple topological graph G resulting from the construction above contains cliques of size at most 4, namely one per clause, and hence is a valid instance of 4-C2PP. Also, by collapsing each variable and clause gadget into a vertex, and each chain connecting them into an edge, the resulting graph G' preserves the planarity of the PLANAR POSITIVE 1-IN-3-SAT instance. This implies that the only crossings for each edge of G are with other edges in the gadget it belongs to and, possibly, with the edges of the 3-cliques of a chain. Hence, G is 3-plane. Namely, each base edge is crossing-free; each internal edge of a 4-clique has one crossing; each external edge of a 4-clique has two crossings, and the same is true for the left and right edges of each 3-clique in a chain; finally, the left and right edges of each 3-clique in either a variable or a clause gadget has three crossings.

In the following we prove the equivalence between the original instance of PLANAR POSITIVE 1-IN-3-SAT and the constructed instance G of 4-C2PP. For this, we first give a lemma stating that variable gadgets correctly represent the behavior of a variable; indeed they can assume one out of two possible states in any solution for 4-C2PP. The proof of the next lemma is in Appendix B.

Lemma 1. [*] *Let G_x be the variable gadget for a variable x in G. Then, in any solution for* 4-C2PP*, either the left edge of each* 3*-clique τ_j^x, with $j = 1, \dots, n_x$, is removed, or the right edge of each* 3*-clique τ_j^x is removed.*

Given Lemma 1, we can associate the truth value of a variable x with the fact that either the left or the right edge of each 3-clique τ_j^x in the variable gadget G_x of G is removed. We use this association to prove the following theorem.

Theorem 1. [*] *The* 4-C2PP *problem is NP-complete, even for* 3*-plane graphs.*

Proof (sketch). Given an instance of PLANAR POSITIVE 1-IN-3-SAT, we construct an instance G of 4-C2PP in linear time as described above. We prove one direction of the equivalence between the two problems. The other direction follows a similar reasoning. Suppose that there exists a solution for 4-C2PP, i.e., a set of edges of G whose removal resolves all crossings. By Lemma 1, for each variable x either the left or the right edge of each 3-clique τ_j^x in gadget G_x is removed. We assign `True` (`False`) to x if the right (left) edge is removed.

We first claim that for each clause c that contains variable x, the right (left) edge of the 3-clique $t_c(x)$ of the clause gadget G_c corresponding to x is removed if and only if the right (left) edge of each 3-clique τ_j^x is removed. Consider the chain that connects $t_c(x)$ with a 3-clique τ_j^x of G_x. For any two consecutive

3-cliques along the chain the left edge of one 3-clique and the right edge of the other 3-clique must be removed. Since the chain has odd length, the truth value of G_x is transferred to the 3-clique $t_c(x)$ of G_c and thus the claim follows.

Consider now a clause c with variables x, y, and z. Let $t_c(x)$, $t_c(y)$, and $t_c(z)$ be the 3-cliques of the clause gadget G_c of c corresponding to x, y, and z, respectively. Let v be the central vertex of the 4-clique of G_c, and let v_x, v_y, v_z be the vertices of this 4-clique lying inside $t_c(x)$, $t_c(y)$, and $t_c(z)$ (see Fig. 1). Assume that v_x, v_y, and v_z appear in this clockwise order around v. We now show that, for exactly one of $t_c(x)$, $t_c(y)$, and $t_c(z)$ the right edge is removed, which implies that exactly one of x, y, and z is `True` and hence the instance of Planar Positive 1-in-3-SAT is positive.

Assume that for each of $t_c(x)$, $t_c(y)$, and $t_c(z)$ the left edge is removed (i.e., all the three variables are set to `False`), as in Fig. 1b. Hence, the crossings between the right edges of the three 3-cliques and the three edges of triangle (v_x, v_y, v_z) are not resolved. All edges of this triangle should be removed, which is not possible since the remaining edges of the 4-clique do not form a path. Assume now that for at least two of the 3-cliques, say $t_c(x)$ and $t_c(y)$, the right edge is removed (i.e., x and y are set to `True`), as in Fig. 1c. Since each edge of triangle (v_x, v_y, v) is crossed by the left edge of one of $t_c(x)$ and $t_c(y)$, by construction, these crossings are not resolved. Hence, all edges of (v_x, v_y, v) should be removed, which is not possible since the remaining edges of the 4-clique do not form a path of length 4. Finally, assume that for exactly one of the 3-cliques, say $t_c(x)$, the right edge is removed (i.e., x is the only one set to `True`), as in Fig. 1a. By removing edges (v, v_x), (v_x, v_y), and (v_y, v_z), all crossings are resolved; the remaining edges of the 4-clique form a path of length 4, as desired. ☐

3 *h*-Clique2Path Planarity and 1-Planarity

In this section we show that, when the given simple topological graph is 1-plane, problem h-C2PP can be solved in linear time in the size of the input, for any h. We consider all possible simple topological 1-plane cliques and show that the problem can be solved using only local tests, each requiring constant time. Note that $h \leq 6$, since K_6 is the largest 1-planar complete graph [8].

Simple topological 1-plane graphs containing cliques with at most four vertices that cross each other can be constructed but it is easy to enumerate all these graphs (up to symmetry); see Fig. 2. Note that such graphs involve at most two cliques and that if K_4 has a crossing, combining it with any other clique would violate 1-planarity; see Fig. 2a and 2b. The next lemma accounts for cliques with five or six vertices.

Lemma 2. *There exists no* 1*-plane simple topological graph that contains two cliques, one of which with at least five vertices, whose edges cross each other.*

Proof. Consider a simple 1-plane graph G that contains two disjoint cliques K and H, with five and three vertices, respectively. Let K' be the simple plane topological graph obtained from K by replacing each crossing with a dummy

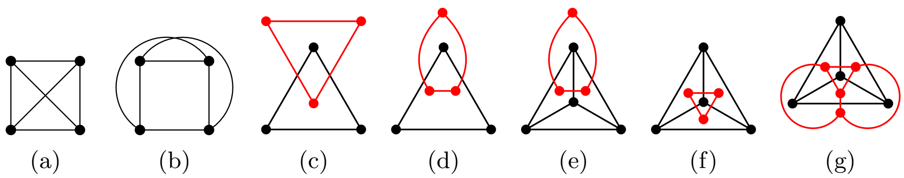


Fig. 2: All 1-plane graphs involving one or more cliques of type K_3 and K_4.

vertex. By 1-planarity, every face of K' is a triangle and contains at most one dummy vertex. Suppose, for a contradiction, that there exists a crossing between an edge of K and an edge of H in G. Then there would exist at least a vertex v of H inside a face f of K' and at least one outside f. Since H is a triangle, there must have been two edges that connect vertices inside f to vertices outside f. If f contains one dummy vertex, then two of its edges are not crossed by edges of H, as otherwise G would not be 1-plane. Hence, both the edges that connect vertices inside f to vertices outside f cross the other edge of f, a contradiction. If f contains no dummy vertices, then each edge of f admits one crossing. Let u be the vertex of f that is incident to the two edges crossed by edges of H. Since u has degree 4 in K, it is not possible to draw the third edge of H so that it crosses only one edge of K, which completes the proof. □

Combining the previous discussion with Lemma 2, we conclude that, for each subgraph of the input graph G that consists either of a combination of at most two cliques of size at most 4, as in Fig. 2, or of a single clique not crossing any other clique, the crossings involving this subgraph (possibly with other edges not belonging to cliques) can only be resolved by removing its edges, which can be checked in constant time. In the next theorem, n denotes the number of vertices.

Theorem 2. *h-C2PP is $O(n)$-time solvable for simple topological 1-plane graphs.*

4 Open Problems

We studied the h-Clique2Path Planarity problem for simple topological k-plane graphs; we proved that this problem is NP-complete for $h = 4$ and $k = 3$, while it is solvable in linear time for every value of h, when $k = 1$. The natural open question is: what is the complexity for simple topological 2-plane graphs?

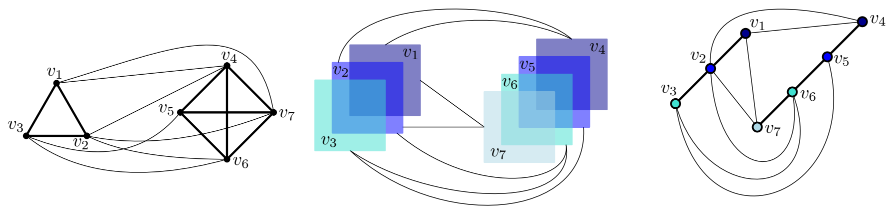
Kindermann et al. [7] recently proved that problem 4-C2PP is NP-complete for geometric 4-plane graphs. It would be interesting to study this geometric version of the problem for 2-plane and 3-plane graphs.

Finally, note that the version of the h-C2PP problem when the input is an abstract graph (which is equivalent to Clique Planarity [1]) is NP-complete when $h \in O(n)$. What if h is bounded by a constant or a sublinear function? We remark that, for $h = 3$, this version of the problem is equivalent to Clustered Planarity, when restricted to instances in which the graph induced by each cluster consists of three isolated vertices.

References

1. Patrizio Angelini, Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, Maurizio Patrignani, and Ignaz Rutter. Intersection-link representations of graphs. *J. Graph Algorithms Appl.*, 21(4):731–755, 2017.
2. Michael A. Bekos, Michael Kaufmann, and Chrysanthi N. Raftopoulou. On optimal 2- and 3-planar graphs. In *33rd International Symposium on Computational Geometry, SoCG 2017, July 4-7, 2017, Brisbane, Australia*, pages 16:1–16:16, 2017.
3. Giordano Da Lozzo, Giuseppe Di Battista, Fabrizio Frati, and Maurizio Patrignani. Computing nodetrix representations of clustered graphs. *J. Graph Algorithms Appl.*, 22(2):139–176, 2018.
4. Emilio Di Giacomo, Giuseppe Liotta, Maurizio Patrignani, and Alessandra Tappini. Nodetrix planarity testing with small clusters. In Fabrizio Frati and Kwan-Liu Ma, editors, *Graph Drawing and Network Visualization - 25th International Symposium, GD 2017, Boston, MA, USA, September 25-27, 2017, Revised Selected Papers*, volume 10692 of *Lecture Notes in Computer Science*, pages 479–491. Springer, 2017.
5. Walter Didimo, Giuseppe Liotta, and Fabrizio Montecchiani. A survey on graph drawing beyond planarity. *CoRR*, abs/1804.07257, 2018.
6. Mohammad Ghoniem, Jean-Daniel Fekete, and Philippe Castagliola. On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. *Information Visualization*, 4(2):114–135, 2005.
7. Philipp Kindermann, Boris Klemz, Ignaz Rutter, Patrick Schnider, and André Schulz. The partition spanning forest problem. In Wolfgang Mulzer, editor, *Proc. 34th European Workshop on Computational Geometry (EuroCG'18)*. Berlin, 2018. To appear.
8. Stephen G. Kobourov, Giuseppe Liotta, and Fabrizio Montecchiani. An annotated bibliography on 1-planarity. *Computer Science Review*, 25:49–67, 2017.
9. Wolfgang Mulzer and Günter Rote. Minimum-weight triangulation is np-hard. *J. ACM*, 55(2):11:1–11:29, 2008.
10. Mershack Okoe, Radu Jianu, and Stephen G. Kobourov. Revisited experimental comparison of node-link and matrix representations. In *Graph Drawing and Network Visualization - 25th International Symposium, GD 2017, Boston, MA, USA, September 25-27, 2017, Revised Selected Papers*, pages 287–302, 2017.
11. János Pach and Géza Tóth. Graphs drawn with few crossings per edge. *Combinatorica*, 17(3):427–439, 1997.
12. Nathalie Henry Riche, Jean-Daniel Fekete, and Michael J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Trans. Vis. Comput. Graph.*, 13(6):1302–1309, 2007.
13. X. Yang, L. Shi, Daianu, H. M., Tong, Q. Liu, and P. Thompson. Blockwise human brain network visual comparison using nodetrix representation. *IEEE Trans. Vis. Comput. Graph.*, 2017.

Appendix



(a) (b) (c)

Fig. 3: (a) A non-planar graph G. Cliques are highlighted with bold edges. (b) A clique-planar drawing of G. (c) Replacing each clique with a path spanning its vertices. Note that differently from (a), in (c) the first vertex and the last vertex of each path have only one place to connect to edges, while the interior vertices have two places: This is what makes the problem non-trivial.

A Omitted Details About the Reduction in [7]

In this section, we show that the instances produced by the reduction in [7] are 4-plane in general.

The variable gadget consists of a triangle X whose edges are x, x_l and x_r. Edge x is crossing-free and the truth value of X is encoded according to which edge among x_l and x_r is crossing-free. Given a pair of triangles T_1 and T_2 whose vertices are u, y, z and v, y, z, they define two faces f_1 is and f_2 respectively. Concatenate a triangle T_3 defined as in the variable gadget with f_1 by inserting its crossing-free edge inside f_1 and by crossing the other two edges of T_3 with (u, y) and (u, z), respectively. Now, concatenate another triangle T_4 defined as in the variable gadget with f_2. If the crossing-free edge of T_4 is inside f_2, the gadget composed of T_1, T_2, T_3 and T_4 is the wire gadget; if the crossing-free edge of T_4 is outside f_2, the gadget composed of T_1, T_2, T_3 and T_4 is the inverter gadget. The splitting gadget consists of three variable gadgets X, Y and Z, and two 4-cliques, concatenated as illustrated inside the blue region in Fig. 4, where the yellow region contains a variable gadget, the orange region contains a wire gadget and the violet region contains an inverter gadget. As shown in Fig. 4, multiple splittings of a variable X lead to an instance where a triangle has two edges with four crossings.

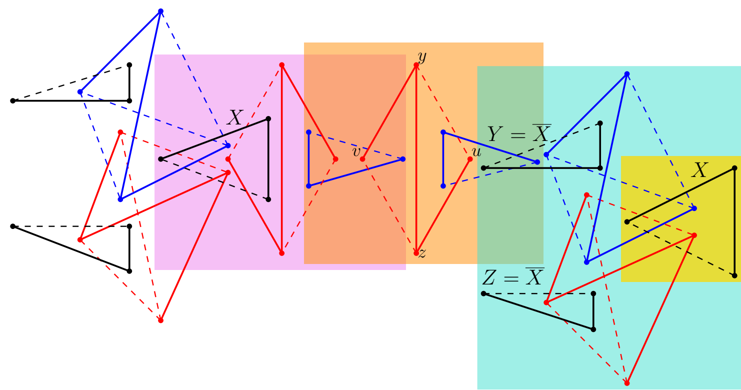


Fig. 4: An instance given by the reduction in [7]. The yellow region contains a variable gadget, the blue region contains a splitting gadget, the orange region contains a wire gadget and the violet region contains an inverter gadget.

B Proof of Lemma 1

Lemma 1. [*] *Let G_x be the variable gadget for a variable x in G. Then, in any solution for* 4-C2PP*, either the left edge of each* 3*-clique τ_j^x, with $j = 1, \dots, n_x$, is removed, or the right edge of each* 3*-clique τ_j^x is removed.*

Proof. We first consider the possible removals of edges in $t_1^x, \dots, t_{2n_x+2}^x$ and claim that, in any solution for 4-C2PP, one of the two following conditions are satisfied: (i) for each 3-clique t_i^x, if i is odd, then the left edge is removed, while if i is even the right edge is removed; (ii) for each 3-clique t_i^x, if i is odd, then the right edge is removed, while if i is even the left edge is removed. Note that this claim is sufficient to prove the statement; in fact, if condition (i) holds (as in Fig. 1a), then the right edge of each 3-clique τ_j^x must be removed, in order to resolve its crossings with the right edge of t_{2j-1}^x and with the left edge of t_{2j}^x, while if condition (ii) holds, then the left edge of each 3-clique τ_j^x must be removed, in order to resolve its crossings with the left edge of t_{2j-1}^x and with the right edge of t_{2j}^x.

In order to prove the claim, we consider the possible removals of edges of t_1^x. Suppose first that the base edge of t_1^x is removed. Thus, the crossings between the left (right) edge of t_1^x and the left (right) edge of t_2^x are not resolved; this implies that they have to be resolved by removing both the left and the right edge of t_2^x, which is not possible. If the right edge of t_1^x is removed, then the crossing between the right edges of t_1^x and t_2^x is resolved, while the one between their left edges is not. Hence, the left edge of t_2^x must be removed. By iterating this argument we conclude that the right (left) edge of each t_i^x with i odd (even) is removed. Symmetrically, we can prove that, if the left edge of t_1^x is removed, then the left (right) edge of each t_i^x with i odd (even) is removed. This concludes the proof of the lemma. □

C Proof of Theorem 1

Theorem 1. [*] *The* 4-C2PP *problem is NP-complete, even for* 3*-plane graphs.*

Proof. Given an instance of PLANAR POSITIVE 1-IN-3-SAT, we construct an instance G of 4-C2PP in linear time as described above. We prove their equivalence.

Suppose first that there exists a solution for 4-C2PP, i.e., a set of edges of G whose removal resolves all crossings. By Lemma 1, for each variable x either the left or the right edge of each 3-clique τ_j^x in the variable gadget G_x is removed. If the right edge is removed, we assign value `True` to variable x, otherwise we assign `False`.

In order to prove that this assignment results in a solution for the given formula of PLANAR POSITIVE 1-IN-3-SAT, we first show that, for each clause c that contains variable x, the right (left) edge of the 3-clique $t_c(x)$ of the clause gadget G_c corresponding to x is removed if and only if the right (left) edge of each 3-clique τ_j^x is removed. Namely, consider the chain that connects $t_c(x)$ with a 3-clique τ_j^x of G_x. Note that, for any two consecutive 3-cliques along the chain, the left edge of one 3-clique and the right edge of the other 3-clique must be removed. Since the chain has odd length, the right (left) edge of $t_c(x)$ is removed if and only if the right (left) edge of τ_j^x is removed, that is, the truth value of G_x is transferred to the 3-clique $t_c(x)$ of G_c.

Finally, consider any clause c, composed of variables x, y, and z. Let $t_c(x)$, $t_c(y)$, and $t_c(z)$ be the three 3-cliques of the clause gadget G_c of c corresponding to x, y, and z, respectively; also, let v be the central vertex of the 4-clique of G_c, and let v_x, v_y, v_z be the vertices of this 4-clique lying inside $t_c(x)$, $t_c(y)$, and $t_c(z)$, respectively; see Fig. 1. We assume w.l.o.g. that v_x, v_y, and v_z appear in this clockwise order around v. As discussed above, the left or the right edge of $t_c(x)$ (of $t_c(y)$; of $t_c(z)$) is removed depending on whether the left or the right edge of each τ_j^x (of each τ_j^y; of each τ_j^z) is removed. We show that, for exactly one of $t_c(x)$, $t_c(y)$, and $t_c(z)$ the right edge is removed, which then implies that exactly one of x, y, and z is `True`, and hence the instance of PLANAR POSITIVE 1-IN-3-SAT is positive.

Suppose first that for each of $t_c(x)$, $t_c(y)$, and $t_c(z)$ the left edge is removed (and hence all the three variables are set to `False`), as in Fig. 1b. This implies that the crossings between the right edges of the three 3-cliques and the three edges of triangle (v_x, v_y, v_z) are not resolved. Hence, all the edges of this triangle should be removed, which is not possible since the remaining edges of the 4-clique do not form a path. Suppose now that for at least two of $t_c(x)$, $t_c(y)$, and $t_c(z)$, say $t_c(x)$ and $t_c(y)$, the right edge is removed (and hence x and y are set to `True`), as in Fig. 1c. Since each edge of triangle (v_x, v_y, v) is crossed by the left edge of one of $t_c(x)$ and $t_c(y)$, by construction, these crossings are not resolved. Hence, all the edges of (v_x, v_y, v) should be removed, which is not possible since the remaining edges of the 4-clique do not form a path of length 4. Suppose finally that for exactly one of $t_c(x)$, $t_c(y)$, and $t_c(z)$, say $t_c(x)$, the right edge is removed (and hence x is the only one to be set to `True`), as in Fig. 1a. Then, by removing edges (v, v_x), (v_x, v_y), and (v_y, v_z), all the crossings are resolved and the remaining edges of the 4-clique form a path of length 4, as desired.

The proof of the other direction is analogous. Namely, suppose that there exists a truth assignment that assigns a `True` value to exactly one variable in each clause. Then, for each variable x that is set to `True` (to `False`), we remove the right (left) edge of each 3-clique t_i^x, with $i = 2j-1$ and $j = 1, \dots, n_x+1$, we remove the left (right) edge of each 3-clique t_i^x, with $i = 2j$ and $j = 1, \dots, n_x+1$, and we remove the right (left) edge of each 3-clique τ_j^x, with $j = 1, \dots, n_x$. Then, we remove the left or right edge of each 3-clique in a chain so that for any two consecutive 3-cliques, one of them has been removed the left edge and the other one the right edge. This ensures that, for each clause c, the right edge of exactly one of the three 3-cliques that belong to the clause gadget G_c has been removed, say the one corresponding to variable x, while for the other two 3-cliques the left edge has been removed. Hence, we can resolve all crossings by removing edges (v, v_x), (v_x, v_y), and (v_y, v_z), as discussed above; see Fig. 1a. The statement follows. $\square$