algorithms

MDPI

Article

Graph Planarity by Replacing Cliques with Paths †

Patrizio Angelini [1], Peter Eades [2], Seok-Hee Hong [2], Karsten Klein [3], Stephen Kobourov [4], Giuseppe Liotta [5], Alfredo Navarra [6,*] and Alessandra Tappini [5]

1 School of Computer Science, John Cabot University, 00165 Rome, Italy; pangelini@johncabot.edu
2 School of Computer Science, Faculty of Engineering, The University of Sydney, Sydney 2006, Australia; peter.edas@sydney.edu.au (P.E.); seokhee.hong@usyd.edu.au (S.-H.H.)
3 Department of Computer Science, University of Konstanz, 78464 Konstanz, Germany; karsten.klein@uni-konstanz.de
4 Department of Computer Science, University of Arizona, Tucson, AZ 85721, USA; kobourov@cs.arizona.edu
5 Department of Engineering, University of Perugia, 06123 Perugia, Italy; giuseppe.liotta@unipg.it (G.L.); alessandra.tappini@unipg.it (A.T.)
6 Department of Mathematics and Computer Science, University of Perugia, 06123 Perugia, Italy
* Correspondence: alfredo.navarra@unipg.it; Tel.: +39-075-585-5046

† This paper is an extended version of our paper published in proceedings of the 26th International Symposium on Graph Drawing and Network Visualization (GD), Barcelona, Spain, 26–28 September 2018. The work began at the Bertinoro Workshop on Graph Drawing (BWGD), Bertinoro, Italy, 4–9 March 2018.

Received: 3 July 2020; Accepted: 11 August 2020; Published: 13 August 2020

Abstract: This paper introduces and studies the following beyond-planarity problem, which we call h-CLIQUE2PATH PLANARITY. Let G be a simple topological graph whose vertices are partitioned into subsets of size at most h, each inducing a clique. h-CLIQUE2PATH PLANARITY asks whether it is possible to obtain a planar subgraph of G by removing edges from each clique so that the subgraph induced by each subset is a path. We investigate the complexity of this problem in relation to k-planarity. In particular, we prove that h-CLIQUE2PATH PLANARITY is NP-complete even when $h = 4$ and G is a simple 3-plane graph, while it can be solved in linear time when G is a simple 1-plane graph, for any value of h. Our results contribute to the growing fields of hybrid planarity and of graph drawing beyond planarity.

Keywords: planar graphs; k-planarity; NP-hardness; polynomial time reduction; cliques; paths

1. Introduction

A typical problem concerning the visual analysis of real-world networks refers to the creation of occlusions and hairball-like structures in dense subnetworks when node-link diagrams are 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 [1,2]. *Hybrid graph representations* combine different visualization metaphors in order to exploit their strengths and overcome their drawbacks.

The *NodeTrix* model [3] represents a first example of hybrid representation. It combines node-link diagrams with adjacency-matrix representations of the denser subgraphs [3–6]. Inspired by NodeTrix, other hybrid representation models were recently introduced [7–9]. The *ChordLink* model [7] embeds chord diagrams, used for the visualization of dense subgraphs (*clusters*), into a node-link diagram. In a (k, p) *representation* [8], each cluster contains at most k vertices and each vertex can occur at most p times along the boundary of the cluster. In the *intersection-link representations* [9] 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.

Clique-planar drawings are defined in [9] as intersection-link representations in which the objects are isothetic rectangles, and the partition into intersection- and link-edges is given as a part of the input, so that the graph induced by the intersection-edges is composed of a set of vertex-disjoint cliques. The corresponding recognition problem, called CLIQUE-PLANARITY, has been proved NP-complete in general and polynomial-time solvable in restricted cases, for example when the rectangle representing each vertex is given as a part of the input, or when the cliques are arranged on levels according to a hierarchy. In [9], it is also proven that, if a graph is clique-planar, then it admits an intersection-link representation in which all vertices in a same cluster are isothetic unit squares whose upper-left corners are aligned along a line of slope one (see Figure 1a,b).

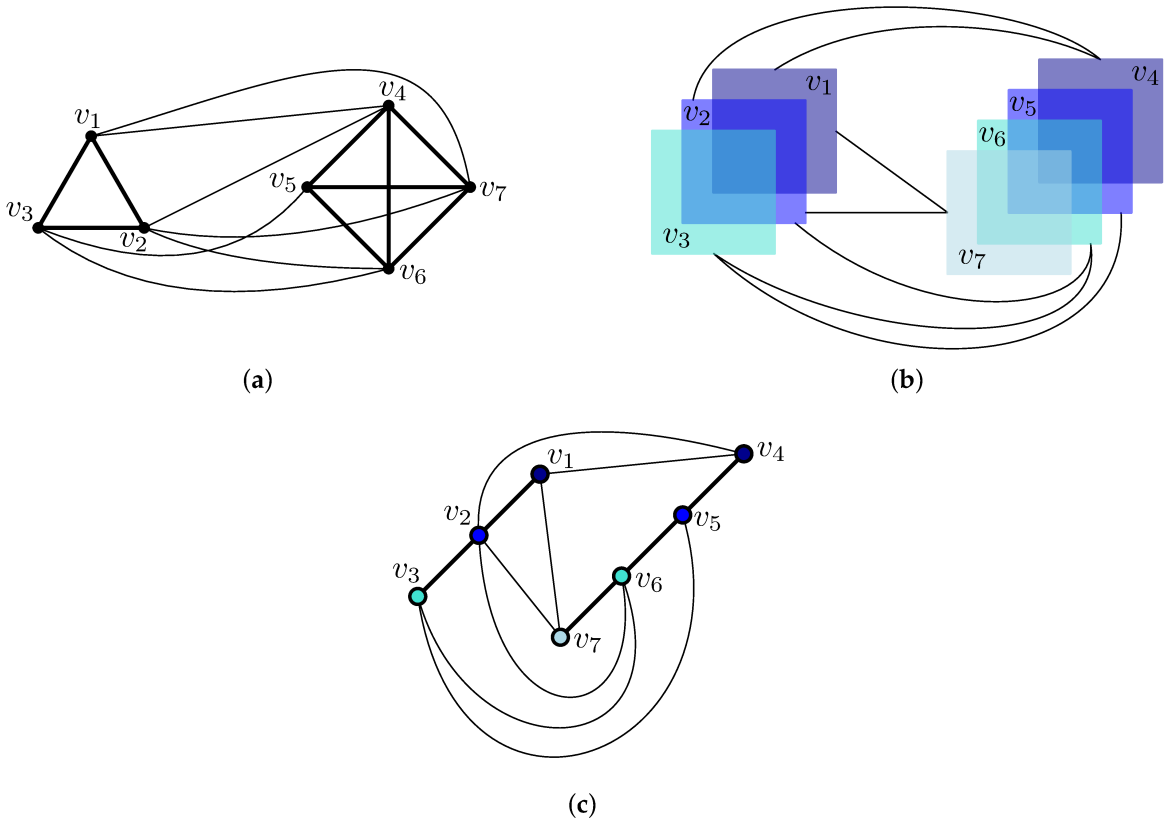

Figure 1. (**a**) A non-planar graph G. Cliques are highlighted with bold edges. (**b**) A clique-planar drawing of G. (**c**) Replacing each clique by a path spanning its vertices. Note that, different from (**a**), in (**c**), the first vertex and the last vertex of each path have only one place to connect to edges, while the interior vertices have two places: this is what makes the problem non-trivial.

Therefore, we can reformulate the CLIQUE-PLANARITY problem in the terminology of *beyond-planarity* [10,11] as follows. Given a graph $G = (V, E)$ and a partition of its vertex set V into subsets $V_1, \ldots, V_m$ 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 $i = 1, \ldots, m$, with a path spanning the vertices of V_i (see Figure 1c).

In this paper, we introduce and study a problem called h-CLIQUE2PATH PLANARITY (for short, h-C2PP), that is a restricted version of CLIQUE-PLANARITY in which the input graph comes with a given embedding and each clique has size at most h. Preliminary results have been presented in [12].

1.1. Our Results

A graph G is *planar* if it admits an embedding in the plane where no two edges cross; this embedding is a *planar embedding* of G. A planar graph with an associated planar embedding is said to be an *embedded planar* graph, or a *plane* graph.

In the version of h-CLIQUE2PATH PLANARITY that we study, the input graph G is a *simple topological graph*. A *topological* graph is embedded in the plane so that each edge is a Jordan arc connecting its end-vertices. A topological graph is *simple* if 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 mutually cross at the same point.

Our main goal is to investigate the complexity of h-C2PP in relation to the well-studied class of *k-planar graphs*, i.e., those that admit a drawing in which each edge has at most k crossings [9,10,13,14]. With a slight abuse of notation, we use the term *embedding* also for non-planar graphs, where we interpret each crossings as a dummy vertex. In particular, a *k-planar* graph together with a k-planar embedding is a *k-plane graph*.

A *geometric graph* is drawn in the plane so that each edge is a straight line segment. The version of h-C2PP in which the input graph G is a geometric graph has been recently studied by Kindermann et al. [15], who called it the *partition spanning forest problem*. They proved that 4-C2PP for geometric graphs is NP-complete, which immediately implies the NP-completeness of 4-C2PP for simple topological graphs.

We strengthen this result by proving that 4-C2PP is NP-complete even for simple topological 3-plane graphs. On the positive side, we prove 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 [15] to solve 3-C2PP for geometric graphs can be easily extended to solve 3-C2PP for any simple topological graph.

1.2. Outline

In Section 2, we further investigate the relationship between h-C2PP and the partition spanning forest problem, that is the problem studied by Kindermann et al. [15]. In Section 3, we prove the NP-completeness of 4-C2PP for simple topological 3-plane graphs. In Section 4, we show that the h-C2PP problem for simple topological 1-plane graphs is linear-time solvable for any value of h. Finally, in Section 5, we provide challenging open problems.

2. Relationship between h-CLIQUE2PAH PLANARITY and the Partition Spanning Forest Problem

The input of the problem studied by Kindermann et al. [15] 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.

The authors proved NP-completeness for the case in which the spanning tree is a path, even when there are at most four vertices with the same color. This result implies that 4-C2PP for geometric graphs is NP-complete. On the other hand, they provided a linear-time algorithm when there exist at most three vertices with the same color, which then extends to 3-C2PP for geometric graphs.

Although not explicitly mentioned in [15], the drawings produced by the reduction used to prove the NP-completeness of 4-C2PP for geometric graphs are 4-planar. We now provide some details about this reduction.

The authors of [15] performed a polynomial-time reduction from PLANAR 3-SATISFIABILITY. The variable gadget (shown in the yellow region of Figure 1) 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. Let T_1 and T_2 be two triangles whose vertices are u, y, z and v, y, z, respectively. They define two faces f_1 and f_2, respectively. Concatenate a triangle T_3 defined as in the variable gadget with f_1 by inserting its crossing-free edge (y, z) 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 by 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 by 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 Figure 2, 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 Figure 2, multiple splittings of a variable X lead to an instance where a triangle has two edges with four crossings.

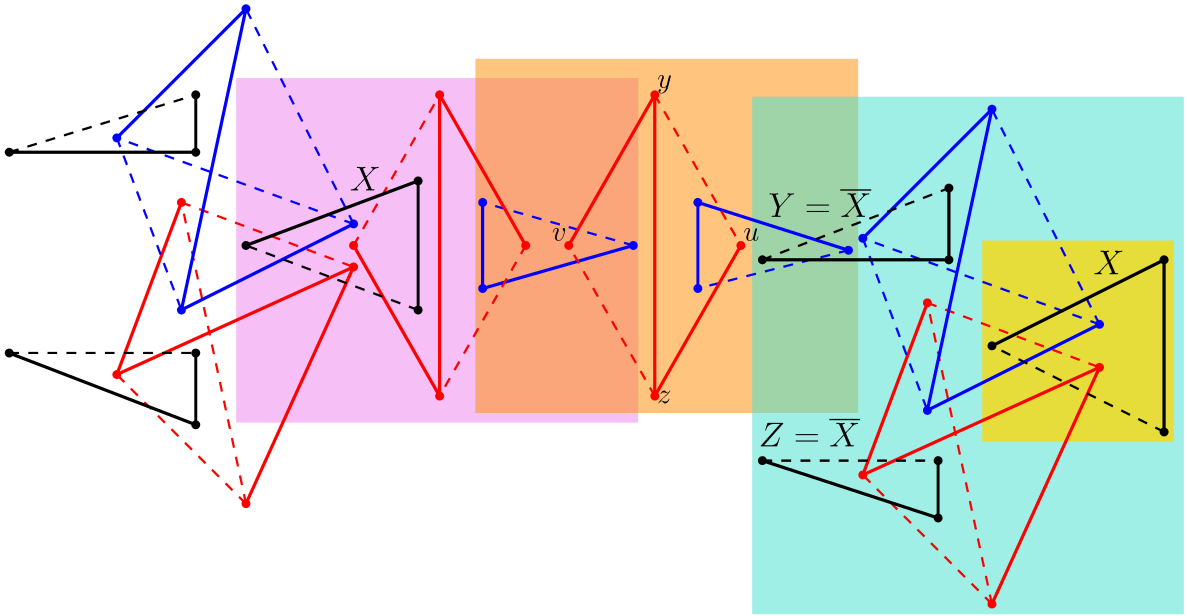

Figure 2. A drawing produced by the reduction in [15]. The yellow region contains a variable gadget, the blue region contains a splitting gadget, the orange region contains a wire gadget, and the violet region contains an inverter gadget.

The NP-completeness of 4-C2PP for geometric graphs implies the NP-completeness of 4-C2PP for simple topological 4-plane graphs. In what follows, we further explore the complexity of 4-C2PP in relation to k-planarity by considering values of $k < 4$. In particular, we prove that the problem remains NP-complete for $k = 3$, while it becomes linear-time solvable for $k = 1$.

3. *NP*-Completeness for Simple Topological 3-Plane Graphs

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

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 [16], 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`. Our reduction is technically different from the one presented in [15], which reduces from PLANAR 3-SATISFIABILITY.

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. In addition, 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 Figure 3a). This gadget contains $2n_x$ 3-cliques $t^x_1, \ldots, t^x_{2n_x}$, forming a ring, so that the left (right) edge of t^x_i only crosses the left (right) edge of t^x_{i-1} and of t^x_{i+1}, for each $i = 1, \ldots, 2n_x$. In addition, gadget G_x contains n_x additional 3-cliques, called $\tau^x_1, \ldots, \tau^x_{n_x}$, so that the right edge of τ^x_j crosses the left edge of t^x_{2j-1} and the right edge of t^x_{2j}, while the left edge of τ^x_j crosses the left edge of t^x_{2j} and the right edge of t^x_{2j-1}.

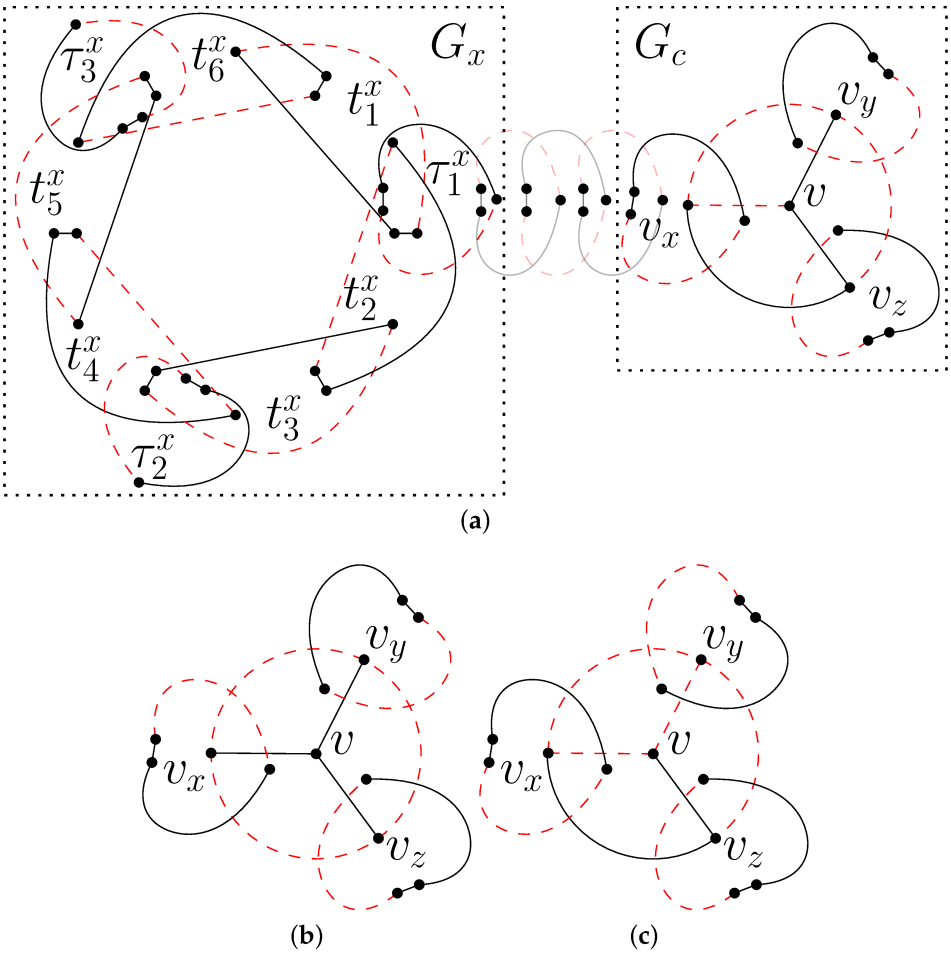

Figure 3. (**a**) The variable gadget G_x for a variable x is represented in the left dotted box. The clause gadget for a clause c is represented in the right dotted box. The chain connecting G_x to G_c is represented with lighter colors. The removed edges are dashed red. (**b**) All variables are `False`. (**c**) At least two variables are `True`.

Then, for each clause c, we construct a simple 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 Figure 3a. 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 jth 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 Figure 3a.

By construction, the resulting simple topological graph G contains cliques of size at most 4, namely one per clause, and hence is a valid instance of 4-C2PP. In addition, 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-planar. 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 have 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.

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 $j = 1, \ldots, n_x$, 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 $t_1^x, \ldots, t_{2n_x}^x$ 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; and (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 Figure 3a), 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.

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. □

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. 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`.

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, and v_z be the vertices of this 4-clique lying inside $t_c(x)$, $t_c(y)$, and $t_c(z)$, respectively; see Figure 3. We assume without loss of generality 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 Figure 3b. 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 Figure 3c. 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 Figure 3a. 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 $j = 1, \ldots, n_x$, we remove the left (right) edge of each 3-clique t_i^x, with $i = 2j$ and $j = 1, \ldots, n_x$, and we remove the right (left) edge of each 3-clique τ_j^x, with $j = 1, \ldots, n_x$. 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 Figure 3a). The statement follows. □

4. *h*-CLIQUE2PAH PLANARITY and 1-Planarity

In this section, we show that, when the given simple topological graph is 1-plane, 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 we can restrict to the case $h \leq 6$, since K_6 is the largest 1-planar complete graph [11].

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 Figure 4). 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 Figure 4a,b). 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 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-planar. 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. □

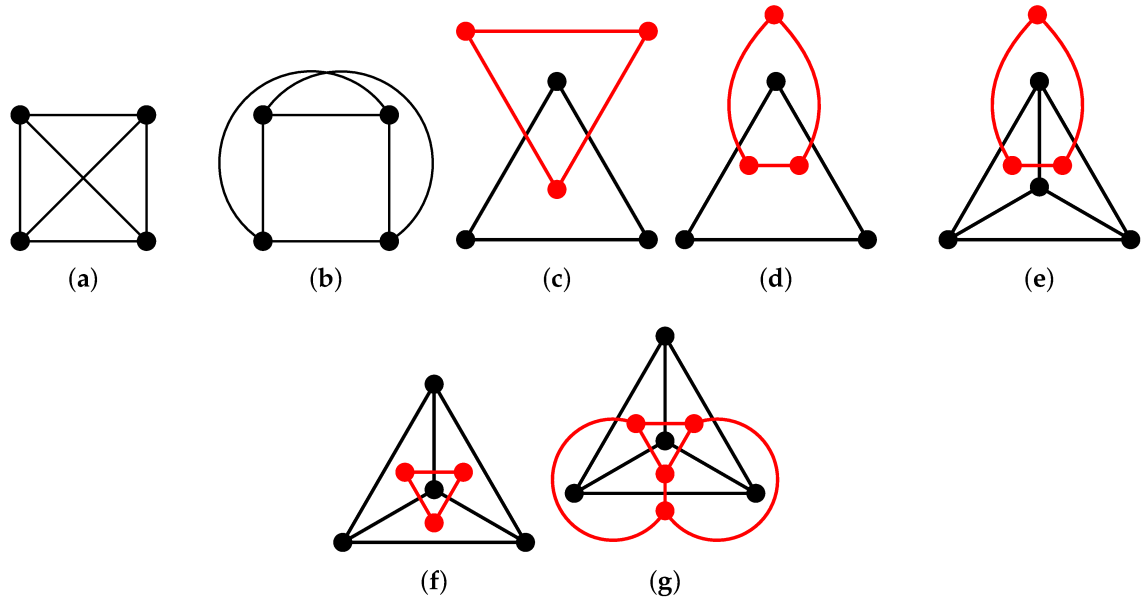

Figure 4. All possible 1-plane graphs involving one or more cliques of type K_3 and K_4 admitting crossings edges. (**a**) and (**b**): two representations of a clique of type K_4; (**c**) and (**d**): two representations of two intersecting cliques of type K_3; (**e**) and (**f**): two representations of a clique of type K_3 intersecting a clique of type K_4; (**g**): two intersecting cliques of type K_4.

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 Figure 4, 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.*

5. Conclusions and Open Problems

We introduce and study 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. [15] 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.

Recall that the version of the h-C2PP problem when the input is an n-vertex abstract graph and $h \in O(n)$ is NP-complete, since it is equivalent to Clique Planarity [9]. What if the input is an abstract graph and h is bounded by a constant or 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.

Finally, another intriguing research direction is to study the h-Clique2Path Planarity problem in the scenario in which the input graph comes without a clustering of its vertex set, but dense portions of the graph are found by an algorithm. While the problem of finding cliques in a graph is NP-complete [17], one could identify dense subgraphs, for example k-cores, in polynomial time [18].

Author Contributions: Conceptualization, All authors; Methodology, All authors; Software, All authors; Validation, All authors; Formal analysis, All authors; Investigation, All authors; Resources, All authors; Data curation, All authors; Writing—original draft preparation, All authors; Writing—review and editing, All authors; Visualization, All authors; Supervision, All authors; Project administration, All authors; Funding acquisition, All authors; All authors have read and agreed to the published version of the manuscript.

Funding: The research was partially supported by: (i) MIUR-DAAD Joint Mobility Program n.57397196 (P.A.); (ii) ARC (Australian Research Council) DP project (S.H.); (iii) Young Scholar Fund/AFF - Univ. Konstanz (K.K.); (iv) NSF grants CCF-1740858 - CCF-1712119 (S.K.); (v) MIUR grant 20174LF3T8 "AHeAD: efficient Algorithms for HArnessing networked Data" (G.L., A.T.); (vi) Dipartimento di Ingegneria dell'Università degli Studi di Perugia, grant RICBA19FM: "Modelli, algoritmi e sistemi per la visualizzazione di grafi e reti" (G.L., A.T.); and (vii) projects "Algorithms and Emergency", "Robot-based computing systems", "Distributed Computing by mobile entities" funded by Fondo Ricerca di Base 2017, 2018, 2019, respectively, University of Perugia (A.N.).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ghoniem, M.; Fekete, J.; Castagliola, P. On the readability of graphs using node-link and matrix-based representations: A controlled experiment and statistical analysis. *Inf. Vis.* **2005**, *4*, 114–135. [CrossRef]
2. Okoe, M.; Jianu, R.; Kobourov, S.G. Node-Link or Adjacency Matrices: Old Question, New Insights. *IEEE Trans. Vis. Comput. Graph.* **2019**, *25*, 2940–2952. [CrossRef] [PubMed]
3. Henry, N.; Fekete, J.; McGuffin, M.J. NodeTrix: A Hybrid Visualization of Social Networks. *IEEE Trans. Vis. Comput. Graph.* **2007**, *13*, 1302–1309. [CrossRef] [PubMed]
4. Da Lozzo, G.; Di Battista, G.; Frati, F.; Patrignani, M. Computing NodeTrix Representations of Clustered Graphs. *J. Graph Algorithms Appl.* **2018**, *22*, 139–176. [CrossRef]
5. Di Giacomo, E.; Liotta, G.; Patrignani, M.; Rutter, I.; Tappini, A. NodeTrix Planarity Testing with Small Clusters. *Algorithmica* **2019**, *81*, 3464–3493. [CrossRef]
6. Yang, X.; Shi, L.; Daianu, M., Tong, H.; Liu, Q.; Thompson, P. Blockwise Human Brain Network Visual Comparison Using NodeTrix Representation. *IEEE Trans. Vis. Comput. Graph.* **2017**, *23*, 181–190. [CrossRef] [PubMed]
7. Angori, L.; Didimo, W.; Montecchiani, F.; Pagliuca, D.; Tappini, A. ChordLink: A New Hybrid Visualization Model. In Proceedings of the Graph Drawing and Network Visualization—27th International Symposium, GD, Prague, Czech Republic, 17–20 September 2019; Volume 11904, pp. 276–290. [CrossRef]
8. Di Giacomo, E.; Lenhart, W.J.; Liotta, G.; Randolph, T.W.; Tappini, A. (k, p)-Planarity: A Relaxation of Hybrid Planarity. In Proceedings of the WALCOM: Algorithms and Computation—13th International Conference, Guwahati, India, 27 February–2 March 2019; pp. 148–159. [CrossRef]
9. Angelini, P.; Da Lozzo, G.; Di Battista, G.; Frati, F.; Patrignani, M.; Rutter, I. Intersection-Link Representations of Graphs. *J. Graph Algorithms Appl.* **2017**, *21*, 731–755. [CrossRef]
10. Didimo, W.; Liotta, G.; Montecchiani, F. A Survey on Graph Drawing Beyond Planarity. *ACM Comput. Surv.* **2019**, *52*, 4:1–4:37. [CrossRef]
11. Kobourov, S.G.; Liotta, G.; Montecchiani, F. An annotated bibliography on 1-planarity. *Comput. Sci. Rev.* **2017**, *25*, 49–67. [CrossRef]

12. Angelini, P.; Eades, P.; Hong, S.; Klein, K.; Kobourov, S.G.; Liotta, G.; Navarra, A.; Tappini, A. Turning Cliques into Paths to Achieve Planarity. In Proceedings of the 26th International Symposium on Graph Drawing and Network Visualization (GD), Barcelona, Spain, 26–28 September 2018; Volume 11282, pp. 67–74.
13. Bekos, M.A.; Kaufmann, M.; Raftopoulou, C.N. On Optimal 2- and 3-Planar Graphs. In Proceedings of the 33rd International Symposium on Computational Geometry, SoCG 2017, Brisbane, Australia, 4–7 July 2017; pp. 16:1–16:16. [CrossRef]
14. Pach, J.; Tóth, G. Graphs Drawn with Few Crossings per Edge. *Combinatorica* **1997**, *17*, 427–439. [CrossRef]
15. Kindermann, P.; Klemz, B.; Rutter, I.; Schnider, P.; Schulz, A. The Partition Spanning Forest Problem. In Proceedings of the 34th European Workshop on Computational Geometry (EuroCG'18), Franconia, Germany, 21–23 March 2018; p. 53.
16. Mulzer, W.; Rote, G. Minimum-weight triangulation is NP-hard. *J. ACM* **2008**, *55*, 1–29. [CrossRef]
17. Karp, R.M. Reducibility Among Combinatorial Problems. In Proceedings of the symposium on the Complexity of Computer Computations, New York, NY, USA, 20–22 March 1972; pp. 85–103. [CrossRef]
18. Batagelj, V.; Zaversnik, M. Fast algorithms for determining (generalized) core groups in social networks. *Adv. Data Anal. Classif.* **2011**, *5*, 129–145. [CrossRef]