Δ_k-Confluent and O_k-Confluent Graphs

Michael J. Pelsmajer
Department of Applied Mathematics
Illinois Institute of Technology
Chicago, Illinois 60616, USA
pelsmajer@iit.edu

Marcus Schaefer
Department of Computer Science
DePaul University
Chicago, Illinois 60604, USA
mschaefer@cs.depaul.edu

Kevin Stern
Department of Computer Science
DePaul University
Chicago, Illinois 60604, USA
kevin.l.stern@gmail.com

June 14, 2007

Abstract

In this paper we extend the concept of Δ-confluence to Δ_k-confluence by allowing more generalized junctions, called Δ_k-junctions. We present an algorithm for recognizing graphs that are Δ_k-confluent. We then generalize Δ_k-confluence to O_k-confluence by allowing non-intersecting chords within a junction, resulting in O_k-junctions. We present an algorithm for recognizing graphs that are O_k-confluent. Finally, we show that the clique problem can be solved in polynomial time for Δ_k-confluent graphs.

1 Introduction

Some years ago Dickerson, Eppstein, Goodrich and Meng introduced the notion of confluent graph drawing, a new graph drawing model that—in a sense—sidesteps the thorny crossing number issue. In a confluent graph drawing, one thinks of curves as if they were train tracks: curves can merge, rather like train tracks merge in a switch. Two vertices in the drawing are connected if one can move from one vertex to the other without making any sharp turns on the track, without passing over any point twice, and without passing through another vertex. Figure 1 shows a confluent drawing of a K_5; each pair of vertices is connected by a smooth curve.

The basic tool in a confluent drawing is a *switch*, that is, a point in which several curves merge smoothly. Figure 2 shows two switches, one of them *simple*, meaning that only two curves merge. Figure 1 uses 15 switches to draw a K_5 (it can be done with fewer).

It is not known how hard it is to recognize whether a graph is confluent or not—that is, whether or not it has a confluent drawing—though a variant

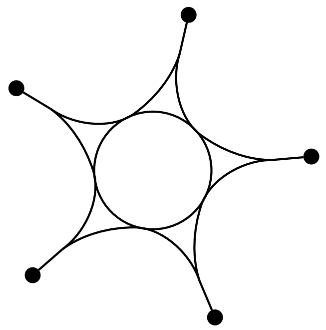

Figure 1: Confluent drawing of a K_5

of the problem does lie in **NP** [2, 6]. Consequently, there has been interest in defining notions of confluency that can be recognized efficiently and capture large classes of graphs. For example, *tree-confluent graphs* are those graphs that have a drawing that can be obtained from a tree by replacing inner vertices by switches [6]. Δ-confluent graphs were defined in [4] as an extension of tree-confluent graphs, admitting Δ-junctions as well as switches. For example, the drawing of K_5 in Figure 1 uses five Δ-junctions; it is not however, a Δ-confluent drawing, since it cannot be obtained from a tree. However, severing the circle in the middle between any two vertices, then removing the four incident simple curves, yields a Δ-confluent drawing of a K_5. The tree-confluent graphs are the bipartite distance-hereditary graphs and the Δ-confluent graphs are the distance-hereditary graphs [6, 2]. Thus, both classes of graphs are well-understood and easily recognizable. In this paper, we investigate two natural generalizations of tree-confluent and Δ-confluent graphs. The first, suggested by Eppstein [3], is to allow junctions with an arbitrary number of ports rather than just three. We call those drawings Δ_∞-confluent; in Section 2 we give an elimination ordering characterization of these graphs, showing, in particular, that they can be recognized in polynomial time. We then take the notion one step further, by allowing non-crossing chords to be drawn within the junctions. This leads us to the notion of O_∞-confluent graphs which also have an elimination ordering characterization. Although that characterization is a bit more complex, it can be used to recognize and draw O_∞-confluent graphs in polynomial time (Section 3). Finally, we show that we can find all maximal cliques of

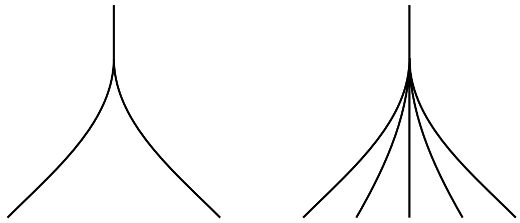

Figure 2: Two switches; the left one is simple.

a Δ_∞-confluent graph in polynomial time. In particular, the CLIQUE problem is polynomial time solvable for Δ_∞-confluent graphs.

2 Δ_k-Confluent Graphs

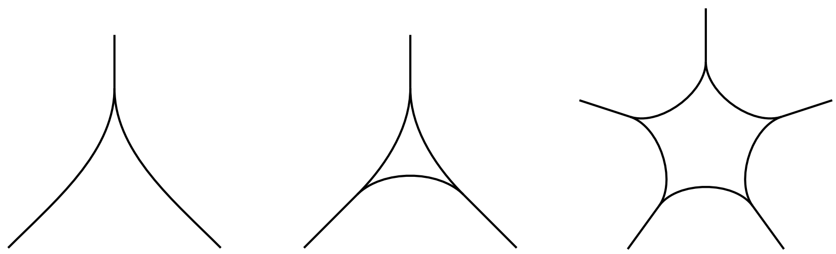

Figure 3: Drawings of a Δ_2-junction, a Δ_3-junction, and a Δ_5-junction.

A *smooth curve* is the image of a differentiable map from the unit interval to the plane. We will be representing graphs with drawings; in such a drawing a *legal curve* is a smooth curve without self-intersections and which does not pass through a point representing a vertex.

A *switch* is a point at which several curves combine and a *simple switch* is a switch in which two curves merge into a single curve; the merging curves are the *tails* of the switch, the merged curve its *head*. A *branch* is a curve not containing any vertices or switches. Using this terminology, we define a Δ_k-junction with $k \geq 3$ to consist of k simple switches, $S_0, \ldots, S_{k-1}$ arranged in a cycle with one tail of S_j leading to a tail of $S_{(j-1) \bmod k}$, and the other tail leading to a tail of $S_{(j+1) \bmod k}$ and all the heads of the switches within the outside region of the junction. We also define a Δ_2-junction to be a simple switch (also called a Λ-junction in [4]). A *port* of a Δ_k-junction is the head of one of its simple switches. A Δ_k-junction has k ports with the exception of Δ_2 which has 3 ports.

We say a port *leads to* another element of the drawing if there is a legal curve between the port and that other element not passing through the junction that the port belongs to. The set of all vertices a port leads to is called its *port-set*.

A Δ_k-*confluent* drawing of a graph G is a drawing which can be obtained from a tree by replacing some of the vertices of the tree with Δ_ℓ-junctions, $2 \leq \ell \leq k$, where we require that the degree of the vertex and the number of ports of the junction are equal. The remaining vertices of the tree (which includes all its leaves) are the *vertices* of the graph G represented by the drawing; there is an edge uv in G if there is a legal curve from u to v in the drawing. The Δ_k-*confluent* graphs are those that admit a Δ_k-confluent drawing. Note that Δ_2-confluent graphs are the tree confluent graphs, Δ_3-confluency is the same as Δ-confluency, and a graph is Δ_∞-*confluent* if it is Δ_k-confluent for some k.

Note that in a Δ_∞-confluent drawing, all port-sets are nonempty, since otherwise the drawing would not have the tree-structure required of a Δ_∞-confluent drawing.

We observe that the use of Δ_k-junctions in a Δ_∞-confluent drawing forces there to be a copy of C_k, that is, a k-cycle as an induced subgraph of whatever graph is represented by the confluent drawing.

Lemma 2.1. *If a Δ_∞-confluent drawing of a graph contains a Δ_k-junction, then the graph contains a copy of C_k.*

Proof. Consider the Δ_k-junction in the drawing. Since port-sets are nonempty, we can pick one vertex from each port-set of the Δ_k-junctions. Because of the tree-structure of the drawing, the only legal curves between those vertices are through the Δ_k-junction, hence this set of vertices induces a C_k in the graph. $\square$

Although the Δ_3-confluent graphs are the same as the Δ_4-confluent graphs (which follows since a Δ_4-junction can be replaced by two 2-switches, see Figure 4), the other Δ_k-confluent graphs form a proper hierarchy. This is demonstrated by the k-cycle, C_k, which is Δ_k-confluent, but not Δ_{k-1}-confluent unless $k = 4$.

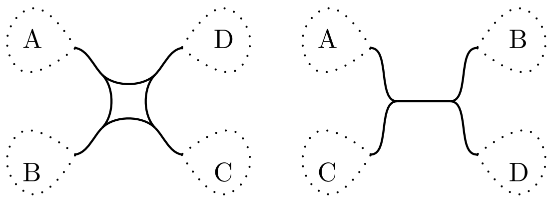

Figure 4: Replacing a Δ_4-junction by two simple switches

Theorem 2.2. *C_k is not Δ_{k-1}-confluent when $k = 3$ or $k \geq 5$.*

Note that we need to exclude the case $k = 4$, since C_4 is tree-confluent as shown in Figure 5.

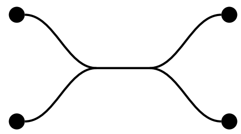

Figure 5: Tree-confluent drawing of C_4

Proof. C_k is not tree-confluent when $k = 3$ (not bipartite) or $k \geq 5$ (not distance-hereditary). In the case where $k = 3$ we are done because the Δ_{k-1}-confluent graphs are simply the tree-confluent graphs. In the case where $k \geq 5$, this implies that a Δ_{k-1}-confluent drawing of C_k must contain a Δ_ℓ-junction with $3 \leq \ell \leq k-1$. By Lemma 2.1, this forces the presence of a copy of C_ℓ as an induced subgraph. $\square$

Note that any induced subgraph of a Δ_k-confluent graph is also Δ_k-confluent (using the induced drawing). Consequently, Theorem 2.2 implies that the Δ_k-confluent graphs are all $(k+1)$-chordal, that is, any cycle of length at least $k+1$ has a chord. In the next section we will see that the Δ_k-confluent graphs are a subset of the $(k+1)$-chordal graphs.

2.1 Recognizing Δ_k-Confluent Graphs

In this section we give an elimination ordering characterization of the Δ_k-confluent graphs.

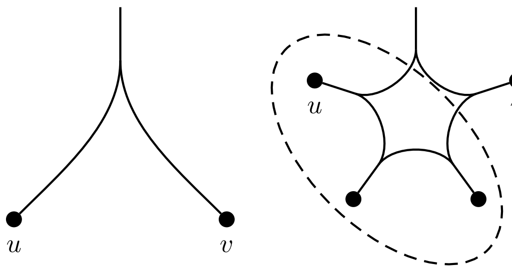

Figure 6: Drawings of: A vertex u such that there is another vertex v with $N(u) = N(v) \neq \emptyset$ *(left)*. A subgraph $P - \{v\}$, where P is a path of length at most $k-2$ between vertices u and v, $N(u) - P = N(v) - P$, and all interior vertices of P have degree two *(right)*.

Theorem 2.3. *A graph is Δ_k-confluent if and only if repeatedly removing*

(i) vertices of degree 1*, and*

(ii) vertices u such that there is another vertex v with $N(u) = N(v) \neq \emptyset$, and

(iii) $P - \{v\}$, where P is a path of length $\ell \leq k-2$ between vertices u and v and $N(u) - P = N(v) - P$, all interior vertices of P have degree two, and u and v are not adjacent if $\ell > 1$,

leads to a graph with a single vertex.

Proof. If G less a vertex of type (i) or (ii) or a subgraph of type (iii) is Δ_k-confluent, then we modify its drawing to get a Δ_k-confluent drawing of G: For (i) we simply draw the removed vertex attached to its original neighbor. For (ii) and (iii), we first modify the Δ_k-confluent drawing near v so that the incoming curves merge via simple switches before reaching v along a single branch.

Then we make v into a port of a new Δ_ℓ-junction, such that $\ell = 2$ in case (ii) (with v at the head) and $\ell - 2$ is the length of P in case (iii). Then we add a new vertex at the head of each of the other ports. This completes a drawing of G when there is a subgraph of type (i), (ii), or (iii). Finally, a graph containing only a single vertex is trivially Δ_k-confluent.

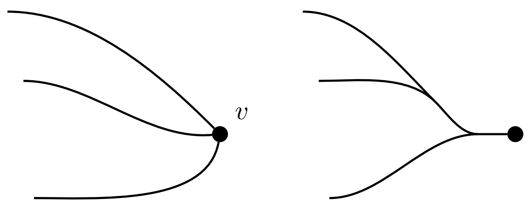

Figure 7: Reducing the number of branches entering a vertex.

To prove the other direction, consider a Δ_k-confluent drawing of G, a graph with more than one vertex. It suffices to show that there exists a reduction of type (*i*), (*ii*), or (*iii*) which produces a (smaller) Δ_k-confluent drawing, since then we can apply induction.

If there is a vertex of degree 1, there is a unique maximal branch to it in the drawing, which must end in a vertex or in a tail of a simple switch. Remove the vertex and the branch from the drawing; this suffices. Thus we may assume that there is no vertex of degree 1.

Consider the tree associated with the drawing; it must have internal vertices. Designate one of its vertices of G to be the root and choose a junction which is farthest from the root in the underlying tree. Given our choice of junction, only the port (p) leading to the root can lead towards another junction, and each of the remaining ports of the junction must lead to a single vertex. If the junction is Δ_ℓ with $\ell \geq 3$, then these $\ell - 1$ vertices form a path P that satisfies (*iii*); its endpoints are in the port-set of the ports before and after p. This gives us a subgraph $P - \{v\}$ of type (*iii*). Replace the junction with a single vertex at p to represent v; this is a Δ_k-confluent graph for $G - (P - \{v\})$, so we are done. If the junction is not Δ_ℓ with $\ell \geq 3$, the junction is Δ_2, and as there are no vertices of degree 1, it is oriented such that we have vertices u, v of type (*ii*). Remove u and the curve from the Δ_2-junction to u; this suffices. $\square$

Remark 2.4. It is easy to quickly detect a substructure of type (*i*), (*ii*), or (*iii*) in a graph, which gives us a polynomial time algorithm for recognizing Δ_k-confluent graphs. The Δ_k-confluent graph recognition algorithm also serves to recognize Δ_∞-confluent graphs by simply running it with $k = |G|$.

3 O_k-Confluent Graphs

One way to generalize Δ-confluent drawings would be to take a drawing of tree and replace some vertices v by a "*k-overpass junction*" with $k = \deg_T(v)$: this has k ports as before, and legal curves can go between *every* two ports. However, this is uninteresting because it's equivalent to Δ-confluency: each k-overpass junction can be replaced by the usual confluent drawing of K_k, which only uses Δ_3-junctions. We will study a different generalization of Δ-confluency in this section.

We begin by demonstrating that cycles with chords but no crossing chords are not Δ_∞-confluent with the exception of $K_4 - e$. Note that cycles with no pairs of crossing chords are precisely the 2-connected outerplanar graphs.

Theorem 3.1. *Any 2-connected outerplanar graph that is Δ_∞-confluent must be a cycle or $K_4 - e$.*

A $K_4 - e$ is indeed Δ_3-confluent, as it has a drawing with one Δ_3-junction and one Δ_2-junction.

Proof. Fix a k-vertex, 2-connected outerplanar graph. The theorem is vacuously true for $k = 3, 4$ since K_4 is not outerplanar, so we can assume that $k \geq 5$.

A tree-confluent graph is bipartite and distance hereditary, which means that its only induced cycles have length 4, and there is no "domino" (an induced graph consisting of two 4-cycles that share one edge) [1]. Since a 2-connected outerplanar graph with at least one chord will have two faces that share exactly one edge, a bipartite 2-connected outerplanar graph will either have a domino or an induced even cycle of length greater than 4. Hence our graph is not tree-confluent, and there must be a Δ_ℓ-junction with $3 \leq \ell$, in a Δ_∞-confluent drawing of it. Since $\ell \geq k$ violates Lemma 2.1, it must be that $\ell < k$.

Let J be a Δ_ℓ-junction with $3 \leq \ell < k$. Suppose that a port p of J contains just one vertex v in its port-set. Since the graph is 2-connected, v is not a cut-vertex. Therefore the part of the drawing that is closer to p than to J contains only one vertex of G; thus it consists of a single branch to v. Then, by the pigeonhole principle, J has a port p with at least two vertices u, v in its port-set. Let q, r be the neighboring ports in J, and let s, t be vertices in the port-set of q, r, respectively. Note that us, sv, vt, tu are all edges, and that there is an s,t-path with $\ell - 3$ internal vertices that does not contain u or v. If $\ell \geq 4$ then we have a subdivision of $K_{2,3}$, contradicting outerplanarity. So $\ell = 3$. We get a $K_{2,3}$-subgraph if either port-set for q or r contains a second vertex, or if the port-set for p contains a third vertex, again contradicting outerplanarity. Now suppose that the graph has a vertex $x \neq u, v, s, t$. Since the port-sets for q, r only contain a single vertex, any curve from x to the junction passes through p. Then any path from x to s, t passes through u or v. By 2-connectivity there must be internally disjoint paths from x to s, t. Then one must contain u, and the other, v, and together they yield a u, v-path through x that does not intersect s or t. However, this gives a subdivision of $K_{2,3}$, so there are only four vertices in the graph. Observe that the drawing yields the graph $K_4 - e$. $\square$

A natural extension of the Δ_k-junction that captures the example above is to allow non-crossing chords within the interior of the junction. Towards that end, we define an *O_k-junction* with $k \geq 3$ to consist of k switches, $S_0, \ldots, S_{k-1}$ arranged in a cycle with one tail of S_j leading to a tail of $S_{(j-1) \bmod k}$, and a second tail leading to a tail of $S_{(j+1) \bmod k}$ and all the heads of the switches within the outside region of the junction. Furthermore, we allow tails of switches to connect within the junction (without using additional switches), forming *chords*, as long as no two of those chords intersect. We define an O_2*-junction* to be a simple switch. A *port* of an O_k-junction is the head of any of its switches. We talk of a port leading to another element of the drawing if there is a legal curve between the port and that other element not passing through the junction that the port belongs to. The set of all vertices a port leads to is again called the *port-set*.

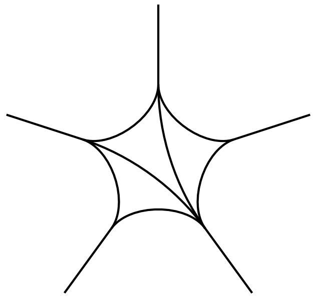

Figure 8: An O_5-junction with two chords.

An O_k*-confluent* drawing is one that is obtained from a tree by replacing some of the vertices of the tree with O_ℓ-junctions, with $2 \leq \ell \leq k$, where we require that the number of ports equals the degree of the vertex. The O_k*-confluent* graphs are those that admit an O_k-confluent drawing. A graph is O_∞*-confluent* if it is O_k-confluent for some k.

Clearly, the Δ_∞-confluent graphs are a subset of the O_∞-confluent graphs, and inclusion is proper by the example above. Moreover, all outerplanar graphs are O_∞-confluent, and 2-connected outerplanar graphs can be drawn using but a single junction. Lastly, all port-sets in an O_∞-confluent drawing are nonempty.

3.1 Recognizing O_k-Confluent Graphs

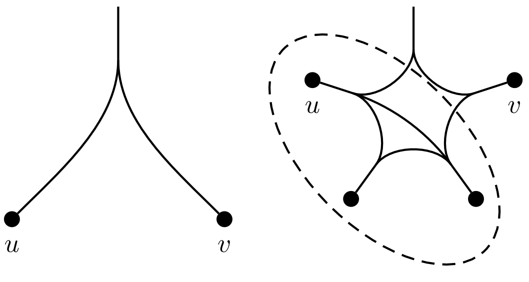

Figure 9: A vertex u such that there is another vertex v with $N(u) = N(v) \neq \emptyset$ *(left)*. A subgraph $J - v$ fulfilling the conditions of the theorem *(right)*.

In this section we give an elimination ordering for recognizing O_k-confluency. It is based on finding sets that separate a junction from the remainder of the graph. We say that a set $X \subseteq V(G)$ *separates* two sets $A, B \subseteq V(G)$ if any path from A to B contains a vertex from X. Note that this definition allows X to overlap with A and B and A and B to be empty.

Theorem 3.2. *A graph is O_k-confluent if and only if repeatedly removing*

(i) vertices of degree 1*, and*

(ii) vertices u such that there is another vertex v with $N(u) = N(v) \neq \emptyset$, and

(iii) subgraphs $J - v$, $v \in V(J)$, with at most $k - 2$ vertices such that

- *$N(v) - J$ separates the vertices in J from the vertices in $G - J$,*
- *for any two $x, y \in N(v) - J$ we have $N(x) \cap J = N(y) \cap J$,*
- *$V(J) \cup \{x\}$ induces a 2-connected outerplanar graph, where $x \in N(v) - J$,*

leads to a graph with a single vertex.

Proof. If is O_k-confluent after removing a vertex of type (i) or (ii) or $J - v$ as in (iii), then we modify its drawing to get a O_k-confluent drawing of G: For (i) we simply draw the removed vertex attached to its original neighbor. For (ii) and (iii) we first modify the O_k-confluent drawing near v so that the incoming curves merge via simple switches before reaching v along a single branch. In case (ii), we make v into the head port of a new O_2-junction and draw u and v at its tails. In case (iii) we consider the graph F induced by $V(J) \cup \{x\}$; by assumption, F is a 2-connected outerplanar graph with at most $(k-2)+2 = k$ vertices. Hence, F can be drawn using a single O_ℓ-junction with $\ell \leq k$, such that each port-set contains exactly one vertex of F; in particular, one port-set will contain x and one will contain a point labeled v (there is also a point labeled v in the drawing of $G - (J - v)$). Now take the drawing of G less $J - v$ as in (iii) and extend its branch leading to v until it smoothly joins the branch to x in the drawing of F, and remove the labels v and x from those points. This completes an O_k-confluent drawing of G when there is a subgraph of type (i), (ii), or (iii). Finally, note that a single vertex graph is O_k-confluent.

To prove the other direction, consider an O_k-confluent drawing of G, a graph with more than one vertex. It suffices to show that there exists a reduction of type (i), (ii), or (iii) which produces a smaller O_k-confluent drawing. As in the proof of Theorem 2.3, we may assume that there are no vertices of degree 1, and we assign a root to the associated tree of the drawing and fix a junction furthest from the root, and we deal with the case when it is just a simple switch (an O_2-junction). Thus we may assume that it is an O_ℓ-junction with $\ell \geq 3$. As before there is a unique port p to the root and each other port leads to exactly one vertex; let V_J be the set of these $\ell - 1$ vertices and let J be the graph induced by V_J. Note that any path from a vertex in J to a vertex in $G - J$ has to pass through a vertex of $N(v) - V_J$.

The vertices in $N(v) - V_J$ all have the same set of neighbors within J because all legal curves to J go through p. If we remove all heads of the junction and label each port $q \neq p$ with the vertex in its port-set, and label p with x, then we obtain an outerplane drawing of the graph induced by $V_J \cup \{x\}$, that satisfies case (iii). Replacing the junction by a single vertex at the root port that represents v produces the desired O_k-confluent drawing of $G - (J - \{v\})$. □

Remark 3.3. As in the case of Δ_k-confluency, the elimination ordering leads to a polynomial-time algorithm for recognizing O_k-confluency. The interesting case here is part (iii); the main idea is that it can be phrased as a search for two vertices u and v, as follows: Since $V(J) \cup \{x\}$ is supposed to induce a 2-connected outerplanar graph, we must have that (1) x has a neighbor $u \neq v$ in J, (2) $|N(u) \cap N(v) \cap J| \leq 1$ (since x, u, v, and two vertices in $N(u) \cap N(v) \cap J$ would give a $K_{2,3}$-subgraph, contradicting outerplanarity), and (3) J is connected. The first and second parts of (iii) imply $N(u) - J \subseteq N(v) - J$ and $N(u) - J \supseteq N(v) - J$ respectively, which yields $N(u) - J = N(v) - J = N(u) \cap N(v) - J$. Thus we may search for two vertices u and v and test $X = N(u) \cap N(v)$ and $X = N(u) \cap N(v) - z$ for each $z \in N(u) \cap N(z)$ to see whether a component of $G - X$ contains u and v, has size at most $k - 1$, and (as J) satisfies the last two properties of (iii). This can be checked in polynomial time.

As earlier, O_∞-confluent graphs can be recognized by running the O_k algorithm with $k = |G|$.

Some initial test-runs with an implementation of the O_∞-confluency recognition algorithm suggest that very dense graphs have a high probability of being O_∞-confluent, while sparser graphs tend to be not O_∞-confluent. O_∞-confluency holds promise of being a useful tool to exhibit the structure of dense graphs visually.

4 Complexity-Theoretic Aspects of Δ_k and O_k-Confluent Graphs

The maximum Δ-confluent subgraph problem is defined in [4] and shown to be **NP**-complete using the Induced Subgraph With Property Π Problem from [5]. This problem is **NP**-complete for any property Π that can be determined in polynomial time, holds for arbitrarily large graphs, does not hold for all graphs, and is hereditary. By extending each of the properties Δ_k-confluency, Δ_∞-confluency, O_k-confluency, and O_∞-confluency to be closed under taking disjoint unions of graphs, each then fulfills property Π. Therefore their corresponding maximum subgraph decision problems are also all **NP**-complete.

Next, we turn our attention to complete subgraphs and demonstrate that finding cliques within Δ_∞-confluent graphs is in **P**.

Lemma 4.1. *A Δ_∞-confluent drawing of K_n consists of n vertices each connected via a branch to a unique port of one of $n-2$ Δ_3-junctions, such that any two Δ_3 junctions have ports such that there is a legal curve connecting those two ports.*

Proof. Consider a Δ_∞ drawing of K_n. There can be no Δ_ℓ junctions for $\ell = 2$ or $\ell > 3$ since then there would be two nonadjacent vertices. So the drawing is a tree T with n leaves where internal vertices are replaced by Δ_3-junctions. Letting k be the number of junctions, the degree-sum formula yields $(3k+n)/2 = |E(T)| = |V(T)| - 1 = k + n - 1$, which gives $k = n - 2$. $\square$

Theorem 4.2. *Finding all maximal cliques within Δ_∞-confluent graphs is in **P**.*

Proof. Fix a Δ_∞-confluent drawing. Consider a maximal set J of Δ_3-junctions such that any two junctions are reachable from one another via a legal curve, and a set of vertices, U, with every port of a junction in J connected either to another junction in J or to exactly one vertex in U via a legal curve. It is easy to see that any set of vertices with such a corresponding set of junctions is a maximal clique and, by the lemma, the set of vertices involved in a maximal clique has a corresponding set of junctions. Then, to find all maximal cliques requires only to find all maximal sets of Δ_3-junctions which are pairwise connected by legal curves. This can be done as follows.

Let T be the corresponding tree to a Δ_∞-confluent drawing of G, and, if there is one, fix a Δ_k-junction with $k \neq 3$. For each port p of the junction, consider the graph drawing beyond p's port with p as a temporary vertex. Recursively find all maximal cliques of that graph, and let S_p be the set of maximal cliques that do not contain p, and let S'_p be the set of maximal cliques that do contain p. For each pair of adjacent ports p, q in the junction, let $S_{\{p,q\}} = \{s \cup t : s \in S_p, t \in S_q\}$. Then the set of maximal cliques in G is the union of all S_p (where p is a port in the junction) and all $S_{\{p,q\}}$ (where p, q are adjacent ports in the junction). In the base case, the drawing consists of Δ_3-junctions only and is a clique itself. $\square$

Corollary 4.3. *The* CLIQUE *problem for Δ_∞-confluent graphs is in **P**.*

References

[1] Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad. *Graph classes: a survey.* SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1999.

[2] Matthew T. Dickerson, David Eppstein, Michael T. Goodrich, and Jeremy Yu Meng. Confluent drawings: Visualizing nonplanar diagrams in a planar way. In *Proc. 11th Int. Symp. Graph Drawing (GD 2003)*, Lecture Notes in Computer Science. Springer-Verlag, 2003.

[3] David Eppstein. Delta-confluent drawings. Slides of a talk given at GD'05. `http://www.ics.uci.edu/~eppstein/pubs/EppGooMen-GD-05.pdf` (accessed May 29th, 2007).

[4] David Eppstein, Michael T. Goodrich, and Jeremy Yu Meng. Delta-confluent drawings. In *Proc. 13th Int. Symp. Graph Drawing (GD 2005)*, Lecture Notes in Computer Science. Springer-Verlag, 2006.

[5] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[6] Peter Hui, Michael J. Pelsmajer, Marcus Schaefer, and Daniel Štefankovič. Train tracks and confluent drawings. *Algorithmica*, 47(4):465–479, 2007.