Finding Contractions and Induced Minors in Chordal Graphs via Disjoint Paths

Rémy Belmonte1, Petr A. Golovach2, Pinar Heggernes1, Pim van ’t Hof1, Marcin Kamiński3, and Daniël Paulusma2

1 Department of Informatics, University of Bergen, Norway {remy.belmonte,pinar.heggernes,pim.vanthof}@ii.uib.no
2 School of Engineering and Computing Sciences, Durham University, UK {petr.golovach,daniel.paulusma}@durham.ac.uk
3 Département d’Informatique, Université Libre de Bruxelles, Belgium marcin.kaminski@ulb.ac.be

Abstract. The \textit{k-Disjoint Paths} problem, which takes as input a graph \(G\) and \(k\) pairs of specified vertices \((s_i, t_i)\), asks whether \(G\) contains \(k\) mutually vertex-disjoint paths \(P_i\) such that \(P_i\) connects \(s_i\) and \(t_i\), for \(i = 1, \ldots, k\). We study a natural variant of this problem, where the vertices of \(P_i\) must belong to a specified vertex subset \(U_i\) for \(i = 1, \ldots, k\). In contrast to the original problem, which is polynomial-time solvable for any fixed integer \(k\), we show that this variant is \textit{NP}-complete even for \(k = 2\). On the positive side, we prove that the problem becomes polynomial-time solvable for any fixed integer \(k\) if the input graph is chordal. We use this result to show that, for any fixed graph \(H\), the problems \textit{H-Contractibility} and \textit{H-Induced Minor} can be solved in polynomial time on chordal graphs. These problems are to decide whether an input graph \(G\) contains \(H\) as a contraction or as an induced minor, respectively.

1 Introduction

We study algorithmic problems that involve deciding whether the structure of a graph \(H\) appears as a pattern within the structure of another graph \(G\). Table I shows seven graph containment relations that can be obtained by combining vertex deletions (VD), edge deletions (ED), and edge contractions (EC). For example, a graph \(H\) is an \textit{induced minor} of a graph \(G\) if \(H\) can be obtained from \(G\) by a sequence of graph operations, including vertex deletions and edge contractions, but not including edge deletions. The corresponding decision problem, in which \(G\) and \(H\) form the ordered input pair \((G, H)\), is called \textit{INDUCED MINOR}. The other rows in Table I are to be interpreted similarly.

With the exception of \textit{Graph Isomorphism}, all problems in Table I are known to be \textit{NP}-complete when \(G\) and \(H\) are both input (cf. [15]). In search of

* This work is supported by EPSRC (EP/G043434/1) and Royal Society (JP100692), and by the Research Council of Norway (197548/F20).
Table 1. Seven known containment relations obtained by graph operations. The missing combination “no yes yes” corresponds to the minor relation if we allow an extra operation that removes isolated vertices.

<table>
<thead>
<tr>
<th>Containment Relation</th>
<th>VD</th>
<th>ED</th>
<th>EC</th>
<th>Decision Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>minor</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>MINOR</td>
</tr>
<tr>
<td>induced minor</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>INDUCED MINOR</td>
</tr>
<tr>
<td>contraction</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>CONTRACTIBILITY</td>
</tr>
<tr>
<td>subgraph</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>SUBGRAPH ISOMORPHISM</td>
</tr>
<tr>
<td>induced subgraph</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>INDUCED SUBGRAPH ISOMORPHISM</td>
</tr>
<tr>
<td>spanning subgraph</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>SPANNING SUBGRAPH ISOMORPHISM</td>
</tr>
<tr>
<td>isomorphism</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>GRAPH ISOMORPHISM</td>
</tr>
</tbody>
</table>

tractability, it is common to fix the graph $H$ as a part of the problem definition, and to consider only the graph $G$ as input. We indicate this by adding “$H$-” in front of the names of the decision problems. A celebrated result by Robertson and Seymour [16] states that $H$-MINOR can be solved in cubic time for any fixed graph $H$. The problems $H$-SUBGRAPH ISOMORPHISM, $H$-INDUCED SUBGRAPH ISOMORPHISM, $H$-SPANNING SUBGRAPH ISOMORPHISM and $H$-GRAPH ISOMORPHISM can readily be solved in polynomial time for any fixed graph $H$. In contrast, there exist graphs $H$ such that $H$-INDUCED MINOR and $H$-CONTRACTIBILITY are NP-complete. Although a complete complexity classification of these two problems is still not known, there are many partial results.

Fellows, Kratochvíl, Middendorf, and Pfeiffer [5] gave both polynomial-time solvable and NP-complete cases for the $H$-INDUCED MINOR problem on general input graphs. The smallest known NP-complete case is a graph $H$ on 68 vertices [5]. A number of polynomial-time solvable and NP-complete cases for the $H$-CONTRACTIBILITY problem on general input graphs can be found in a series of papers started by Brouwer and Veldman [3], followed by Levin, Paulusma and Woeginger [13,14], and van ’t Hof et al. [9]. The smallest NP-complete cases are when $H$ is a path or a cycle on 4 vertices [3]. When it comes to input graphs with a particular structure, both $H$-INDUCED MINOR [5] and $H$-CONTRACTIBILITY [11] can be solved in polynomial time on planar graphs for any fixed graph $H$. The same is true when the input graphs are split graphs, as shown by Belmonte, Heggernes, and van ’t Hof [1] and Golovach et al. [8]. Finally, Golovach, Kamiński and Paulusma [7] showed that $H$-CONTRACTIBILITY and $H$-INDUCED MINOR are polynomial-time solvable on chordal graphs whenever $H$ is a fixed tree or a fixed split graph. For any fixed $H$ that is neither a tree nor a split graph, the computational complexity of these two problems on chordal graphs was left open.

In this paper, we show that $H$-CONTRACTIBILITY and $H$-INDUCED MINOR can be solved in polynomial time on chordal graphs, for any fixed graph $H$. In fact, our result implies algorithms with running time $|V_G|^O(|V_H|^2)$ for CONTRACTIBILITY and INDUCED MINOR on input pairs $(G, H)$ where $G$ is chordal. Our result is best possible in the sense that both these problems are NP-complete
already on input pairs \((G, H)\), where \(G\) is a split graph and \(H\) is a threshold graph \([1]\). Moreover, both problems are \(W[1]\)-hard parameterized by \(|V_H|\) when \(G\) and \(H\) are both split graphs \([8]\). This means that it is unlikely that these two problems can be solved in time \(f(|V_H|)|V_G|^{O(1)}\) on input pairs \((G, H)\) that are split graphs, such that the function \(f\) is independent of \(|V_G|\).

Our approach differs from previous approaches \([1,7,8]\), as it is based on an application of the following problem. Here, a terminal pair in a graph \(G = (V, E)\) is a specified pair of vertices \(s_i\) and \(t_i\) called terminals, and the domain of a terminal pair \((s_i, t_i)\) is a specified subset \(U_i \subseteq V\) containing both \(s_i\) and \(t_i\). We say that an \((s_i, t_i)\)-path and an \((s_j, t_j)\)-path are vertex-disjoint if they have no common vertices except the vertices in \(\{s_i, t_i\} \cap \{s_j, t_j\}\).

**Set-Restricted k-Disjoint Paths**

**Instance:** A graph \(G\), terminal pairs \((s_1, t_1), \ldots, (s_k, t_k)\), and domains \(U_1, \ldots, U_k\).

**Question:** Does \(G\) contain \(k\) mutually vertex-disjoint paths \(P_1, \ldots, P_k\) such that \(P_i\) is a path from \(s_i\) to \(t_i\) using only vertices from \(U_i\) for \(i = 1, \ldots, k\)?

Observe that the domains \(U_1, \ldots, U_k\) can have common vertices. Furthermore, if we let every domain contain all vertices of \(G\), we obtain the well-known problem \(k\)-DISJOINT PATHS. This problem can be solved in cubic time for any fixed \(k\), as was shown by Robertson and Seymour \([16]\). In contrast to this result, we show that Set-Restricted \(k\)-Disjoint Paths is \(\text{NP}\)-complete even for \(k = 2\). However, we show that on chordal graphs Set-Restricted \(k\)-Disjoint Paths can be solved in polynomial time for any fixed integer \(k\). If \(k\) is part of the input, then this problem is \(\text{NP}\)-complete for chordal graphs, as Kammer and Tholey \([10]\) showed that its special case Disjoint Paths remains \(\text{NP}\)-complete when restricted to this graph class.

## 2 Preliminaries

Let \(G = (V, E)\) be a graph. If the vertex set and edge set of \(G\) are not specified, then we use \(V_G\) and \(E_G\) to denote these sets, respectively. The number of vertices in an input graph is denoted by \(n\). A subset \(U \subseteq V\) is a clique if there is an edge in \(G\) between any two vertices of \(U\). A vertex is called simplicial if its neighbors form a clique. We write \(G[U]\) to denote the subgraph of \(G\) induced by \(U \subseteq V\), i.e., the graph on vertex set \(U\) and an edge between any two vertices whenever there is an edge between them in \(G\). Two sets \(U, U' \subseteq V\) are called adjacent if there exist vertices \(u \in U\) and \(u' \in U'\) such that \(uu' \in E\).

The edge contraction of an edge \(uv\) in \(G\) deletes the vertices \(u\) and \(v\) from \(G\), and replaces them by a new vertex adjacent to precisely those vertices to which \(u\) or \(v\) were adjacent. A graph \(H\) is a contraction of a graph \(G\) if \(H\) can be obtained by performing a series of edge contractions in \(G\). An \(H\)-witness structure \(W\) is a partition of \(V_G\) into \(|V_H|\) nonempty sets \(W(x)\), one set for each \(x \in V_H\), called \(H\)-witness sets, such that (i) each \(W(x)\) induces a connected subgraph of \(G\),

\(^1\) Threshold graphs form a subset of split graphs, which form a subset of chordal graphs.
and (ii) for all \( x, y \in V_H \) with \( x \neq y \), sets \( W(x) \) and \( W(y) \) are adjacent in \( G \) if and only if \( x \) and \( y \) are adjacent in \( H \). Clearly, \( H \) is a contraction of \( G \) if and only if \( G \) has an \( H \)-witness structure satisfying conditions (i) and (ii); \( H \) can be obtained by contracting each witness set into a single vertex.

A tree decomposition of a graph \( G \) is a pair \((T, \mathcal{X})\), where \( \mathcal{X} \) is a collection of subsets of \( V_G \), called bags, and \( T \) is a tree whose vertices, called nodes, are the sets of \( \mathcal{X} \), such that the following three properties are satisfied. First, \( \bigcup_{X \in \mathcal{X}} X = V_G \). Second, for each edge \( uv \in E_G \), there is a bag \( X \in \mathcal{X} \) with \( u, v \in X \). Third, for each \( x \in V_G \), the set of nodes containing \( x \) forms a subtree of \( T \).

A graph is chordal if it does not contain a chordless cycle on at least four vertices as an induced subgraph. It is easy to see that the class of chordal graphs is closed under edge contractions. Chordal graphs can be recognized in linear time \([17]\). Every chordal graph contains at most \( n \) maximal cliques, and, if it is not a complete graph, at least two non-adjacent simplicial vertices \([4]\). A graph \( G \) is chordal if and only if it has a tree decomposition whose set of bags is exactly the set of maximal cliques of \( G \) \([8]\). Such a tree decomposition is called a clique tree and can be constructed in linear time \([2]\).

### 3 Set-Restricted Disjoint Paths

In this extended abstract some proofs, in particular the proof of the following theorem, are omitted due to limited space.

**Theorem 1.** The SET-RESTRICTED 2-DISJOINT PATHS problem is \( \text{NP} \)-complete.

We now apply dynamic programming to prove that SET-RESTRICTED \( k \)-DISJOINT PATHS can be solved in polynomial time on chordal graphs for any fixed integer \( k \). The first key observation is that the existence of \( k \) disjoint paths is equivalent to the existence of \( k \) disjoint induced, i.e. chordless, paths. The second key observation is that any induced path contains at most two vertices of any clique. Our algorithm can easily be modified to produce the required paths (if they exist).

Kloks \([12]\) showed that every tree decomposition of a graph can be converted in linear time to a nice tree decomposition, such that the size of the largest bag does not increase, and the total size of the tree is linear in the size of the original tree. A tree decomposition \((T, \mathcal{X})\) is nice if \( T \) is a binary tree with root \( X_r \) such that the nodes of \( T \) are of four types: (i) a leaf node \( X \) is a leaf of \( T \) and has size \( |X| = 1 \); (ii) an introduce node \( X \) has one child \( X' \) with \( X = X' \cup \{v\} \) for some vertex \( v \in V_G \); (iii) a forget node \( X \) has one child \( X' \) with \( X = X' \setminus \{v\} \) for some vertex \( v \in V_G \); (iv) a join node \( X \) has two children \( X' \) and \( X'' \) with \( X = X' \cup X'' \). Since each chordal graph \( G \) has a clique tree, it also has a nice tree decomposition with the additional property that each bag is a (not necessary maximal) clique in \( G \).

Now we are ready to describe our algorithm for SET-RESTRICTED \( k \)-DISJOINT PATHS. Let \( k \) be a positive integer, and let \( G \) be a chordal graph with \( k \) pairs of terminal vertices \((s_1, t_1), \ldots, (s_k, t_k)\) that have domains \( U_1, \ldots, U_k \), respectively. If \( G \) is disconnected, we check for each pair of terminals \((s_i, t_i)\) whether \( s_i \) and \( t_i \)
belong to the same connected component. If not, then we return No. Otherwise, we consider each connected component and its set of terminals separately. Hence, we may assume without loss of generality that $G$ is connected. We then proceed as follows.

First, we construct in linear time a nice tree decomposition $(T, \mathcal{X})$ with root $X_r$ for $G$ such that each bag is a clique in $G$. Next, we apply a dynamic programming algorithm over $(T, \mathcal{X})$. We omit the details here and only describe what we store in tables corresponding to the nodes of $T$. For any node $X_i \in V_T$, we denote by $T_i$ the subtree of $T$ with root $X_i$ that is induced by the descendants of $X_i$. We also define the subgraph $G_i = G[\bigcup_{j \in V_{T_i}} X_j]$. For a node $X_i$, the table stores a collection of the records $\mathcal{R} = ((\text{State}_1, R_1), \ldots, (\text{State}_k, R_k))$, where each $\text{State}_j$ can have one of the four values:

- **Not initialized**, **Started from $s$**, **Started from $t$**, and **Completed**.

and $R_1, \ldots, R_k \subseteq X_i$ are ordered sets without common vertices except (possibly) terminals, $R_j \subseteq U_j$, and $0 \leq |R_j| \leq 2$ for $j \in \{1, \ldots, k\}$. These records correspond to the partial solution of **Set-Restricted $k$-Disjoint Paths** for $G_i$ with the following properties.

- If $\text{State}_j = \text{Not initialized}$, then $s_j, t_j \notin V_{G_i}$. If $R_j = \emptyset$, then $(s_j, t_j)$-paths have no vertices in $G_i$ in the partial solution. If $R_j = \langle z \rangle$, then $z$ is the unique vertex of a $(s_j, t_j)$-path in $G_i$ in the partial solution. If $R_j = \langle z_1, z_2 \rangle$, then $z_1, z_2$ are vertices in a $(s_j, t_j)$-path, $z_1$ is the predecessor of $z_2$ in the path, and this path has no other vertices in $G_i$.

- If $\text{State}_j = \text{Started from } s$, then $s_j \in V_{G_i}, t_j \notin V_{G_i}$ and $R_j$ contains either one or two vertices. If $R_j = \langle z \rangle$, then the partial solution contains an $(s_j, z)$-path with the unique vertex $z \in X_i$. If $R_j = \langle z_1, z_2 \rangle$, then the partial solution contains an $(s_j, z_2)$-path such that $z_1$ is the predecessor of $z_2$ with exactly two vertices $z_1, z_2 \in X_i$.

- If $\text{State}_j = \text{Started from } t$, then $s_j \notin V_{G_i}, t_j \in V_{G_i}$ and $R_j$ contains either one or two vertices. If $R_j = \langle z \rangle$, then the partial solution contains a $(z, t_j)$-path with the unique vertex $z \in X_i$. If $R_j = \langle z_1, z_2 \rangle$, then the partial solution contains an $(z_1, t_j)$-path such that $z_2$ is the successor of $z_1$ with exactly two vertices $z_1, z_2 \in X_i$.

- If $\text{State}_j = \text{Completed}$, then $s_j \in V_{G_i}, t_j \in V_{G_i}$. The partial solution in this case contains an $(s_j, t_j)$-path, and $R_j$ is the set of vertices of this path in $X_i$. If $R_j = \langle z_1, z_2 \rangle$, then $z_1$ is the predecessor of $z_2$ in the path.

Observe that since we are solving the decision problem, we do not keep $(s_j, t_j)$-paths or their subpaths themselves. The properties of the algorithm are summarized in the following theorem.

**Theorem 2.** The **Set-Restricted $k$-Disjoint Paths** problem can be solved on chordal graphs in time $n^{O(k)}$. 
For our purposes, we need to generalize Theorem $\text{2}$ from paths to trees. Instead of terminal pairs $(s_i, t_i)$ we speak of terminal sets $S_i$, each contained in a subset $U_i$ called the domain of $S_i$. Moreover, we say that a tree $T_i$ containing $S_i$ is an $S_i$-tree. Then an $S_i$-tree and an $S_j$-tree are vertex-disjoint if they have no common vertices except the vertices in $S_i \cap S_j$. The SET-RESTRICTED $k$-DISJOINT TREES problem takes as input a graph $G$, terminal sets $S_1, \ldots, S_k$, and domains $U_1, \ldots, U_k$, and asks whether $G$ contains mutually vertex-disjoint trees $T_1, \ldots, T_k$ such that $T_i$ is an $S_i$-tree containing only vertices from $U_i$ for $i = 1, \ldots, k$.

**Corollary 1.** The SET-RESTRICTED $k$-DISJOINT TREES problem can be solved on chordal graphs in $n^{O(p)}$ time, where $p = \sum_{i=1}^{k} |S_i|$ is the total size of the terminal sets.

### 4 Contractions and Induced Minors in Chordal Graphs

First we give a structural characterization of chordal graphs that contain a fixed graph $H$ as a contraction. Then we present our polynomial-time algorithm for solving $H$-CONTRACTIBILITY on chordal graphs for any fixed graph $H$, and show how it can be used to solve $H$-INDUCED MINOR as well.

For the statements of the structural results below, let $G$ be a connected chordal graph, let $\mathcal{T}_G$ be a clique tree of $G$, and let $H$ be a graph with $V_H = \{x_1, \ldots, x_k\}$. For a set of vertices $A \subseteq V_G$, we let $G(A)$ denote the induced subgraph of $G$ obtained by recursively deleting simplicial vertices that are not in $A$. Since every leaf in any clique tree contains at least one simplicial vertex, we immediately obtain Lemma $\text{1}$ below. This lemma, in combination with Lemma $\text{2}$, is crucial for the running time of our algorithm.

**Lemma 1.** For any set $A \subseteq V_G$, every clique tree of $G(A)$ has at most $|A|$ leaves.

**Lemma 2.** The graph $H$ is a contraction of $G$ if and only if there is a set $A \subseteq V_G$ such that $|A| = k$ and $H$ is a contraction of $G(A)$.

**Proof.** First suppose that $H$ is a contraction of $G$. Let $\mathcal{W}$ be an $H$-witness structure $\mathcal{W}$ of $G$. For each $i \in \{1, \ldots, k\}$, we choose an arbitrary vertex $a_i \in W(x_i)$, and let $A = \{a_1, \ldots, a_k\}$. Suppose that $G$ has a simplicial vertex $v \notin A$, and assume without loss of generality that $v \in W(x_1)$. Because $v \neq a_1$ and $a_1 \in W(x_1)$, we find that $|W(x_1)| \geq 2$. Hence, $W(x_1)$ contains a vertex $u$ adjacent to $v$. The graph $G'$, obtained from $G$ by deleting $v$, is isomorphic to the graph obtained from $G$ by contracting $uv$, since $v$ is simplicial. Because $u$ and $v$ belong to the same witness set, namely $W(x_1)$, this implies that $H$ is a contraction of $G'$. Using these arguments inductively, we find that $H$ is a contraction of $G(A)$.

Now suppose that $A$ is a subset of $V_G$ with $|A| = k$, and that $H$ is a contraction of $G(A)$. Deleting a simplicial vertex $v$ in a graph is equivalent to contracting an
edge incident with $v$. This means that $G(A)$ is a contraction of $G$. Because $H$ is a contraction of $G(A)$ and contractibility is a transitive relation, we conclude that $H$ is a contraction of $G$ as well. □

For a subtree $T$ of $T_G$, we say that a vertex $v \in V_G$ is an inner vertex for $T$ if $v$ only appears in the maximal cliques of $G$ that are nodes of $T$. By $I(T) \subseteq V_G$ we denote the set of all inner vertices for $T$. For a subset $S \subseteq V_G$, let $T_S$ be the unique minimal subtree of $T_G$ that contains all maximal cliques of $G$ that have at least one vertex of $S$; we say that a vertex $v$ is an inner vertex for $S$ if $v \in I(T_S)$, and we set $I(S) = I(T_S)$. Lemma 4 below provides an alternative and useful structural description of $G$ if it contains $H$ as a contraction. We need the following lemma to prove Lemma 4.

**Lemma 3.** Let $S \subseteq V_G$ and let $T$ be a subgraph of $G$ that is a tree such that $S \subseteq V_T \subseteq I(S)$. Then $K \cap V_T \neq \emptyset$ for each node $K$ of $T_S$.

**Proof.** Let $K$ be a node of $T_S$. If $K \cap S \neq \emptyset$, then clearly $K \cap V_T \neq \emptyset$. Suppose that $K \cap S = \emptyset$. Because $T_S$ is the unique minimal subtree of $T_G$ that contains all maximal cliques of $G$ that have at least one vertex of $S$, we find that $K$ separates two nodes $K_1$ and $K_2$ in $T_S$ for which $K_1 \cap S \neq \emptyset$ and $K_2 \cap S \neq \emptyset$. This means that $K$ separates two vertices $u \in K_1 \cap S$ and $v \in K_2 \cap S$ in $G$. Since $T$ is a tree and $u, v \in V_T$, at least one vertex of $T$ must be in $K$. □

Let $l$ denote the number of leaves in $T_G$; if $T_G$ consists of one node, then we say that this node is a leaf of $T_G$.

**Lemma 4.** The graph $H$ is a contraction of $G$ if and only if there are mutually disjoint nonempty sets of vertices $S_1, \ldots, S_k \subseteq V_G$, each of size at most $l$, such that

1. $V_G \subseteq I(S_1) \cup \ldots \cup I(S_k)$;
2. $V_{T_{S_i}} \cap V_{T_{S_j}} \neq \emptyset$ if and only if $x_i x_j \in E_H$ for $1 \leq i < j \leq k$;
3. $G$ has mutually vertex-disjoint trees $T_1, \ldots, T_k$ with $S_i \subseteq V_{T_i} \subseteq I(S_i)$ for $1 \leq i \leq k$.

**Proof.** First suppose that $H$ is a contraction of $G$. Consider a corresponding $H$-witness structure $W$ of $G$. For $i = 1, \ldots, k$, let $T_i$ be the subgraph of $T_G$ induced by the maximal cliques of $G$ that contain one or more vertices of $W(x_i)$. Because each $W(x_i)$ induces a connected subgraph of $G$, each $T_i$ is connected. This means that $T_i$ is a subtree of $T_G$, i.e., $T_i = T_{W(x_i)}$. We construct $S_i$ as follows. For each leaf $K$ of $T_i$, we choose a vertex of $W(x_i) \cap K$ and include it in the set $S_i$. Because $T_G$ has $l$ leaves, each $T_i$ has at most $l$ leaves. Hence, $|S_i| \leq l$ for $i = 1, \ldots, k$. We now check conditions 1–3 of the lemma.

1. By construction, we have $T_i = T_{S_i}$. All vertices of $W(x_i)$ are inner vertices for $T_i$, so $W(x_i) \subseteq I(T_i) = I(T_{S_i}) = I(S_i)$. Hence, $V_G = \bigcup_{i=1}^k W(x_i) \subseteq \bigcup_{i=1}^k I(S_i)$. 

...
2. Any two vertices \(u, v \in V_G\) are adjacent if and only if there is a maximal clique \(K\) in \(G\) containing \(u\) and \(v\). Hence, two witness sets \(W(x_i)\) and \(W(x_j)\) are adjacent if and only if there is a maximal clique \(K\) in \(G\) such that \(K \cap W(x_i) \neq \emptyset\) and \(K \cap W(x_j) \neq \emptyset\). This means that \(W(x_i)\) and \(W(x_j)\) are adjacent if and only if \(V_{T_i} \cap V_{T_j} \neq \emptyset\). It remains to recall that \(T_i = T_{S_i}\) and \(T_j = T_{S_j}\), and that two witness sets \(W(x_i)\) and \(W(x_j)\) are adjacent if and only if \(x_i, x_j \in E_H\).

3. Every \(G[W(x_i)]\) is a connected graph. Hence, every \(G[W(x_i)]\) contains a spanning tree \(T_i\). Because the sets \(W(x_1), \ldots, W(x_k)\) are mutually disjoint, the trees \(T_1, \ldots, T_k\) are mutually vertex-disjoint. Moreover, as we already deduced, \(S_i \subseteq V_{T_i} = W(x_i) \subseteq I(S_i)\) for \(i = 1, \ldots, k\).

Now suppose that there are mutually disjoint nonempty sets of vertices \(S_1, \ldots, S_k \subseteq V_G\), each of size at most \(l\), that satisfy conditions 1–3 of the lemma. By condition 3, there exist mutually vertex-disjoint trees \(T_1, \ldots, T_k\) with \(S_i \subseteq V_{T_i} \subseteq I(S_i)\) for \(i = 1, \ldots, k\). By condition 1, we have \(V_G \subseteq I(S_1) \cup \ldots \cup I(S_k)\). This means that there is a partition \(X_1, \ldots, X_k\) of \(V_G \setminus \bigcup_{i=1}^{k} V_{T_i}\), where some of the sets \(X_i\) can be empty, such that \(X_i \subseteq I(S_i)\) for \(i = 1, \ldots, k\). Let \(W(x_i) = V_{T_i} \cup X_i\) for \(i = 1, \ldots, k\). We claim that the sets \(W(x_1), \ldots, W(x_k)\) form an \(H\)-witness structure of \(G\). By definition, \(W(x_1), \ldots, W(x_k)\) are mutually disjoint, nonempty, and \(W(x_1) \cup \ldots \cup W(x_k) = V_G\), i.e., they form a partition of \(V_G\). It remains to show that these sets satisfy conditions (i) and (ii) of the definition of an \(H\)-witness structure.

(i) By definition, each tree \(T_i\) is connected. By Lemma 3, each node \(K\) of \(T_{S_i}\) contains a vertex of \(T_i\). By definition, \(X_i \subseteq I(S_i)\), which implies that for each \(v \in X_i\), there is a node \(K\) in \(T_{S_i}\) such that \(v \in K\). Because \(K\) is a clique in \(G\), we then find that \(v\) is adjacent to at least one vertex of \(T_i\). Therefore, each \(W(x_i)\) induces a connected subgraph of \(G\).

(ii) For the forward direction, suppose that \(W(x_i)\) and \(W(x_j)\) are two adjacent witness sets. Then there exist two vertices \(u \in W(x_i)\) and \(v \in W(x_j)\) such that \(uv \in E_G\). Let \(K\) be a maximal clique that contains both \(u\) and \(v\). Because \(u \in W(x_i)\) and \(v \in W(x_j)\), we find that \(K\) is a node of \(T_{S_i}\) and of \(T_{S_j}\), respectively. Hence, \(V_{T_{S_i}} \cap V_{T_{S_j}} \neq \emptyset\), which means that \(x_i, x_j \in E_H\) by 2.

For the reverse direction, let \(x_i\) and \(x_j\) be two adjacent vertices in \(H\). By condition 2, we find that \(V_{T_{S_i}} \cap V_{T_{S_j}} \neq \emptyset\). Hence, there is a node \(K \in V_{T_{S_i}} \cap V_{T_{S_j}}\). By Lemma 3, we deduce that \(K\) contains a vertex \(u \in V_{T_i}\) and a vertex \(v \in V_{T_j}\). Because \(K\) is a clique in \(G\), this means that \(u\) and \(v\) are adjacent. Because \(V_{T_i} \subseteq W(x_i)\) and \(V_{T_j} \subseteq W(x_j)\), we obtain \(u \in W(x_i)\) and \(v \in W(x_j)\), respectively. Hence, \(W(x_i)\) and \(W(x_j)\) are adjacent.

We are now ready to describe our algorithm for \(H\)-contractibility on chordal graphs, for any fixed graph \(H\). Although the presented algorithm solves the decision problem, it can be modified to produce an \(H\)-witness structure if one exists.

**Theorem 3.** For any fixed graph \(H\), the \(H\)-contractibility problem can be solved in polynomial time on chordal graphs.
Proof. Let $G$ be a chordal graph on $n$ vertices and let $H$ be a graph on $k$ vertices with $V_H = \{x_1, \ldots, x_k\}$. If $k > n$ or the number of connected components of $G$ and $H$ are different, then return No. Suppose that $G$ and $H$ have $r > 1$ connected components $G_1, \ldots, G_r$ and $H_1, \ldots, H_r$, respectively. For each permutation $\langle i_1, \ldots, i_r \rangle$ of the ordered set $\{1, \ldots, r\}$, check whether $H_{i_j}$ is a contraction of $G_j$ for every $j \in \{1, \ldots, r\}$. Return Yes if this is the case for some permutation, and No otherwise. Hence, we may assume that $G$ and $H$ are connected.

Construct a clique tree $T_G$ of $G$. If $T_G$ has at least $k + 1$ leaves, then consider each set $A \subseteq V_G$ with $|A| = k$, and continue with $G(A)$ instead of $G$. This is allowed due to Lemma 2. Note that a clique tree of $Kammer$ and Tholey [10] improved the running time of Disjoint Paths 5.

Concluding Remarks

Let $G$ and $H$ be two arbitrary graphs. Then $G$ contains $H$ as an induced minor if and only if $(P_1 \bowtie G)$ contains $(P_1 \bowtie H)$ as a contraction.

Since $P_1 \bowtie G$ is chordal whenever $G$ is chordal, we can combine Lemma 5 with Theorem 3 to obtain the following result, with the same running time as in the proof of Theorem 3.

Corollary 2. For any fixed graph $H$, the $H$-INDUCED MINOR problem can be solved in polynomial time on chordal graphs.

5 Concluding Remarks

Finding Contractions and Induced Minors

Disjoint Paths fixed-parameter tractable on chordal graphs, when parameterized by $k$?

Recall that the problems CONTRACTIBILITY and INDUCED MINOR are $W[1]$-hard for pairs $(G, H)$ that are chordal graphs, when parameterized by $|V_H|$ \[8\]. Is either of these problems fixed-parameter tractable on pairs $(G, H)$ that are interval graphs, which constitute an important subclass of chordal graphs?

References