Efficient Frequent Connected Induced Subgraph Mining in Graphs of Bounded Tree-width

Tamás Horváth[1,2], Keisuke Otaki[3], and Jan Ramon[4]

[1] Dept. of Computer Science III, University of Bonn, Germany
[2] Fraunhofer IAIS, Schloss Birlinghoven, Sankt Augustin, Germany
tamas.horvath@iais.fraunhofer.de
[3] Dept. of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University, Kyoto, Japan
ootaki@iip.ist.i.kyoto-u.ac.jp
[4] Dept. of Computer Science, Katholieke Universiteit Leuven, Belgium
jan.ramon@cs.kuleuven.be

Abstract. We study frequent connected induced subgraph mining, i.e., the problem of listing all connected graphs that are induced subgraph isomorphic to at least a certain number of transaction graphs. We first show that this problem cannot be solved for arbitrary transaction graphs in output polynomial time (if P $\neq$ NP) and then prove that for graphs of bounded tree-width, frequent connected induced subgraph mining is possible in incremental polynomial time by levelwise search. Our algorithm is an adaptation of the technique developed for frequent connected subgraph mining in bounded tree-width graphs. While the adaptation is relatively natural for many steps of the original algorithm, we need entirely different combinatorial arguments to show the correctness and efficiency of the new algorithm. Since induced subgraph isomorphism between bounded tree-width graphs is NP-complete, the positive result of this paper provides another example of efficient pattern mining with respect to computationally intractable pattern matching operators.

1 Introduction

Over the past 15 years substantial research efforts have been devoted toward designing effective frequent graph mining algorithms. Despite the numerous studies in this field of research, the theoretical aspects of the topic are still not well understood. The importance of a better understanding of the complexity aspects of the various graph mining problem settings appears somewhat neglected, which has as negative side effect that most algorithms are limited to some ten thousands transaction graphs only. The goal of this work is to take a step towards a better understanding of the *frequent connected induced subgraph mining* (FCISM) problem, which is the problem of listing all pairwise non-isomorphic connected graphs that are induced subgraph isomorphic to at least t transaction graphs for some $t \in \mathbb{N}$. This problem, as we show, cannot be solved in output polynomial time for arbitrary transaction graphs. For forests, however, it can be solved in incremental polynomial time; this follows e.g. from the results in [5].

As the main result for this work, we generalize the positive result on forests by showing that the FCISM problem can be solved in incremental polynomial time for graphs of *bounded tree-width*. The positive result of this paper is of practical, theoretical, and algorithmic importance. Regarding its practical importance, we mention e.g. the ZINC dataset containing about 16.5 millions molecular graphs: 99.99% of these graphs have tree-width at most 3. Regarding its theoretical importance we note that induced subgraph isomorphism is one of the "persistent" problems that remain NP-complete even for graphs of tree-width 2 [8]. Thus, our result provides an example of the case that *efficient pattern mining is possible even for computationally intractable pattern matching operators.* To the best of our knowledge, there is only one further such example [5].

Finally, it is of algorithmic importance because it shows that the algorithmic paradigm we followed here, and which is used also in [5] for frequent connected subgraph mining in graphs of bounded tree-width, appears sufficiently generic for the design of graph mining algorithms for further pattern matching operators. This paradigm consists of the following main steps: (1) Give a generic levelwise search algorithm and provide conditions guaranteeing its efficiency, (2) choose an efficiently computable complete pattern refinement operator, and (3) show that the otherwise exponential-time dynamic-programming algorithm [8] deciding the underlying pattern matching works in time polynomial in the size of the set of patterns generated by the algorithm so far. When comparing the (sub)steps of this paradigm for ordinary subgraph isomorphism [5] and for induced subgraph isomorphism, one can notice, on the one hand, a number of steps that are (almost) the same for the two problems. On the other hand, however, there are some crucial steps that require entirely different techniques. Thus, for example, the pattern refinement operator and the combinatorial characterization of the necessary information needed to calculate by the pattern matching algorithm become much more complicated for induced subgraph isomorphism, as we will show in Section 4.

The rest of the paper is organized as follows. In the next section we collect all necessary notions. In Section 3 we give a generic levelwise search algorithm and formulate five conditions for the efficiency of this algorithm. In Sections 4 and 5 we prove that all these conditions are fulfilled by the class of bounded tree-width graphs. Finally, in Section 6 we conclude and mention some problems.

2 Preliminaries

In this section we collect and fix all necessary notions and notations used in the paper. Most of the definitions and notations are taken from [5].

Graphs An *undirected graph* is a pair (V, E), where V is a finite set of *vertices* and $E \subseteq \{e \subseteq V : |e| = 2\}$ is a set of *edges*. A *labeled undirected graph* is a triple (V, E, λ), where (V, E) is an undirected graph and λ is the labeling function $\lambda : V \cup E \to \mathbb{N}$; the set of vertices, the set of edges, and the labeling function of a graph G are denoted by $V(G)$, $E(G)$, and λ_G, respectively. Unless otherwise stated, by graphs in this paper we always mean *labeled undirected* graphs.

A *subgraph* of G is a graph G' with $V(G') \subseteq V(G)$, $E(G') \subseteq E(G)$, and $\lambda_{G'}(x) = \lambda_G(x)$ for all $x \in V(G') \cup E(G')$; G' is an *induced subgraph* of G if it is a subgraph of G satisfying $\{u, v\} \in E(G')$ iff $\{u, v\} \in E(G)$ for all $u, v \in V(G')$. For $S \subseteq V(G)$, $G[S]$ denotes the induced subgraph of G with vertex set S. For $v \in V(G)$, $G \ominus v$ denotes $G[V(G) \setminus \{v\}]$.

A *path* connecting two vertices v_1, v_k of a graph G, denoted by P_{v_1,v_k}, is a sequence $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{k-1}, v_k\} \in E(G)$ such that the v_i's are pairwise distinct. A graph is *connected* if there is a path between any pair of its vertices. A *connected component* of a graph G is a maximal subgraph of G that is connected. The set of all connected components of a graph G is denoted by $\mathcal{C}(G)$.

Graph morphisms Two graphs G_1 and G_2 are *isomorphic*, denoted $G_1 \simeq G_2$, if there is a *bijection* $\varphi : V(G_1) \to V(G_2)$ satisfying (i) $\{u, v\} \in E(G_1)$ iff $\{\varphi(u), \varphi(v)\} \in E(G_2)$ for every $u, v \in V(G_1)$, (ii) $\lambda_{G_1}(u) = \lambda_{G_2}(\varphi(u))$ for every $u \in V(G_1)$, and (iii) $\lambda_{G_1}(\{u, v\}) = \lambda_{G_2}(\{\varphi(u), \varphi(v)\})$ for every $\{u, v\} \in E(G_1)$. For G_1 and G_2 we say that G_1 is *subgraph isomorphic* to G_2, denoted $G_1 \preccurlyeq G_2$, if G_1 is isomorphic to a subgraph of G_2; it is *induced subgraph isomorphic* to G_2, denoted $G_1 \preccurlyeq_i G_2$, if it is isomorphic to an induced subgraph of G_2. In what follows, two graphs are regarded the same graph if they are isomorphic.

Tree-width A central notion to this work is *tree-width*, which was reintroduced in [10]. A *tree-decomposition* of a graph G is a pair $(T, \mathcal{X})$, where T is a rooted unordered tree and $\mathcal{X} = (X_z)_{z \in V(T)}$ is a family of subsets of $V(G)$ satisfying (i) $\cup_{z \in V(T)} X_z = V(G)$, (ii) for every $\{u, v\} \in E(G)$, there is a $z \in V(T)$ such that $u, v \in X_z$, and (iii) $X_{z_1} \cap X_{z_3} \subseteq X_{z_2}$ for every $z_1, z_2, z_3 \in V(T)$ such that z_2 is on the path connecting z_1 with z_3 in T. The set X_z associated with a node z of T is called the *bag* of z. The nodes of T will often be referred to as the nodes of the tree-decomposition. The tree-width of $(T, \mathcal{X})$ is $\max_{z \in V(T)} |X_z| - 1$, and the *tree-width* of G, denoted $\mathrm{tw}(G)$, is the minimum tree-width over all tree-decompositions of G. By graphs of bounded tree-width we mean graphs of tree-width at most k, where k is some constant. The following notation will be used many times in what follows. Let G be a graph, $(T, \mathcal{X})$ be a tree-decomposition of G, and $z \in V(T)$. Then $G_{[z]}$ denotes the induced subgraph of G defined by the union of the bags of z's descendants, where z is considered also as a descendant of itself.

We will need a special kind of tree-decomposition. More precisely, a *nice* tree-decomposition of G, denoted $NTD(G)$, is a tree-decomposition $(T, \mathcal{X})$, where T is a rooted binary tree composed of three types of nodes: (i) a *leaf* node has no children, (ii) a *separator* node z has a single child z' with $X_z \subseteq X_{z'}$, and (iii) a *join* node z has two children z_1 and z_2 with $X_z = X_{z_1} \cup X_{z_2}$. It follows from [3] that for graphs of tree-width at most k, where k is some constant, a nice tree-decomposition of tree-width at most k always exists and can be constructed in linear time.

Tree-width is a useful parameter of graphs in algorithmic graph theory, as many NP-hard problems can be solved in polynomial time for graphs of bounded tree-width. However, subgraph isomorphism and induced subgraph isomorphism remain NP-complete even for graphs of tree-width 2 [11, 8].

Listing algorithms A common feature of many listing problems is that the size of the output can be exponential in that of the input. Clearly, for such cases there exists no algorithm enumerating the output in time polynomial in the size of the input. Thus, the size of the output must also be taken into account. The following listing complexity classes are usually distinguished: in the literature (see, e.g., [6]): For some input $\mathcal{I}$, let $\mathcal{O}$ be the output set of some finite cardinality N. Then the elements of $\mathcal{O}$, say $o_1, \ldots, o_N$, are listed with

- *polynomial delay* if the time before printing o_1, the time between printing o_i and o_{i+1} for every $i = 1, \ldots, N-1$, and the time between printing o_N and the termination is bounded by a polynomial of the size of $\mathcal{I}$,
- *incremental polynomial time* if o_1 is printed with polynomial delay, the time between printing o_i and o_{i+1} for every $i = 1, \ldots, N-1$ (resp. the time between printing o_N and the termination) is bounded by a polynomial of the combined size of $\mathcal{I}$ and the set $\{o_1, \ldots, o_i\}$ (resp. $\mathcal{O}$),
- *output polynomial time* (or *polynomial total time*) if $\mathcal{O}$ is printed in time polynomial in the combined size of $\mathcal{I}$ and the *entire* output $\mathcal{O}$.

Clearly, polynomial delay implies incremental polynomial time, which, in turn, implies output polynomial time. Furthermore, in contrast to incremental polynomial time, the delay of an output polynomial time algorithm may be exponential in the size of the input even before printing the first element of the output.

3 Frequent Connected Induced Subgraph Mining

In this section we first define the frequent connected induced subgraph mining problem and, using a simple polynomial reduction, show in Theorem 1 that it cannot be solved in output polynomial time (if P $\neq$ NP). We then give a generic levelwise search algorithm [7] for mining frequent connected induced subgraphs and provide sufficient conditions in Theorem 2 for the efficiency of this algorithm. As a corollary of Theorem 2, we get that the frequent connected induced subgraph mining problem can be solved in incremental polynomial time for *forest* [5] transaction graphs. In the next section we generalize the positive result on forests to graphs of bounded tree-width. We start by defining the pattern mining problem we are interested in.

The Frequent Connected Induced Subgraph Mining (FCISM) Problem: *Given* a class $\mathcal{G}$ of graphs, a transaction database (i.e., multiset) DB of graphs from $\mathcal{G}$, and an integer threshold $t > 0$, *list* the set $\mathcal{O}$ of all distinct frequent connected *induced* subgraphs, that is, all connected graphs that are induced subgraph isomorphic to at least t graphs in DB.

Notice that each isomorphism type (i.e., equivalence class under isomorphism) of $\mathcal{O}$ is a singleton, as isomorphic graphs are not distinguished from each other. The *parameter* of the above problem is the size of DB. Clearly, the size of

[5] In this paper by forests we mean a set of disjoint unrooted and unordered trees.

Algorithm 1 FCISM

Require: transaction database DB of graphs and integer $t > 0$
Ensure: all frequent connected induced subgraphs

```
1: let S_1 ⊆ 𝒢 be the set of frequent graphs consisting of a single labeled vertex
2: for (l := 1; S_l ≠ ∅; l := l + 1) do
3:    C_{l+1} := S_{l+1} := ∅
4:    forall P ∈ S_l do
5:       forall H ∈ ρ(P) ∩ 𝒢 satisfying (i) H ∉ C_{l+1} and (ii) ρ^{-1}(H) ⊆ S_l do
6:          add H to C_{l+1}
7:          if |{G ∈ DB : H ≼_i G}| ≥ t then
8:             print H and add it to S_{l+1}
```

$\mathcal{O}$ can be exponential in that of DB. Thus, in general, the set of all frequent connected induced subgraphs cannot be computed in time polynomial only in the size of DB. The following simple polynomial reduction shows that even output polynomial time enumeration is unlikely.

Theorem 1. *Unless $P = NP$, the FCISM problem cannot be solved in output polynomial time.*

Proof. We prove the claim by a reduction from the NP-complete k-Clique problem. For an unlabeled graph G with n vertices, let DB consist of G and the clique K_n with n vertices. For DB and $t = 2$, the number of frequent connected induced subgraphs is at most n (i.e., all cliques up to size n). Thus, if the FCISM problem could be solved in output polynomial time, we could decide the k-Clique problem in polynomial time by listing first the set $\mathcal{O}$ of all 2-frequent connected induced subgraphs and checking then whether $|\mathcal{O}| \geq k$ or not. □

3.1 A Generic Levelwise Search Mining Algorithm

Our goal in this paper is to show that the FCISM problem can be solved in incremental polynomial time for graphs of *bounded tree-width*. To prove this result, we start by giving a generic algorithm, called FCISM, that lists frequent connected induced subgraphs with levelwise search (see Algorithm 1). The algorithm assumes the transaction graphs to be elements of some graph class $\mathcal{G}$ that is closed under taking subgraphs. Thus, as we are interested in mining frequent connected induced subgraphs, all patterns belong to $\mathcal{G}$ as well.

One of the basic features of the levelwise search algorithms is that the underlying pattern language $\mathcal{L}$ is associated with some, usually naturally defined partial order.Following the common pattern mining terminology (see, e.g., [7]), for a partially ordered pattern language $(\mathcal{L}, \leq)$ we say that a pattern $P_1 \in \mathcal{L}$ is a *generalization* of a pattern $P_2 \in \mathcal{L}$ (or P_2 is a *specialization* of P_1) if $P_1 \leq P_2$; P_1 is a *proper generalization* of P_2 (or P_2 is a *proper specialization* of P_1), denoted by $P_1 < P_2$, if $P_1 \leq P_2$ and $P_1 \neq P_2$. Furthermore P_1 is a *direct generalization*

of P_2 (or P_2 is a *direct specialization* of P_1) if $P_1 < P_2$ and there is no $P_3 \in \mathcal{L}$ with $P_1 < P_3 < P_2$.

In case of the FCISM problem, the underlying pattern language $\mathcal{L}$ is the set of all finite connected (labeled) graphs of $\mathcal{G}$, associated with the natural generalization relation $\leq$ defined as follows: For any $P_1, P_2 \in \mathcal{L}$, $P_1 \leq P_2$ if and only if $P_1 \preccurlyeq_i P_2$. The proofs of the two claims in the proposition below are straightforward.

Proposition 1. *Let $\mathcal{L}$ and $\leq$ be as defined above. Then $(\mathcal{L}, \leq)$ is a partially ordered set. Furthermore, for any $P_1, P_2 \in \mathcal{L}$ it holds that P_1 is a direct generalization of P_2 if and only if*

$$P_1 < P_2 \text{ and } |V(P_1)| = |V(P_2)| - 1 \ . \tag{1}$$

In the main loop of Algorithm 1 (lines (4–8)), the set S_{l+1} of frequent connected induced subgraphs containing $l+1$ *vertices* are calculated from those containing l *vertices*, in accordance with condition (1). In particular, for each frequent pattern $P \in S_l$, we first compute a set $\rho(P) \cap \mathcal{G}$ of graphs, where $\rho(P)$ is a subset of the set of *direct* specializations of P. Clearly, the graphs in $\rho(P)$ are all connected by the choice of $\mathcal{L}$. Notice, however, that $\rho(P)$ cannot be defined as the set of *all* direct specializations of P because its cardinality is exponential in the size of P, even for unlabeled graphs. In Theorem 2 below we will provide sufficient conditions for ρ needed for efficient pattern enumeration.

For each direct specialization $H \in \rho(P) \cap \mathcal{G}$, we check whether it has already been generated during the current iteration (see condition (i) in line 5). If not, we also check for each connected direct generalization of H, denoted by $\rho^{-1}(H)$ in the algorithm, whether it is frequent (condition (ii) in line 5). Here we utilize that frequency is an anti-monotonic interestingness predicate for $(\mathcal{L}, \leq)$. In what follows, candidate patterns generated by Algorithm 1 that satisfy conditions (i) and (ii) in line 5 will be referred to as *strong candidates*. If H is a strong candidate, we add it to the set C_{l+1} of candidate patterns with $l+1$ *vertices* and compute its support count (lines (6–7)). If H is frequent, i.e., it is induced subgraph isomorphic to at least t transaction graphs in DB, we add it to the set S_{l+1} of frequent connected graphs containing $l+1$ *vertices*.

By Theorem 1 above, the FCISM problem cannot be solved in output polynomial time for the general problem setting. If, however, the class $\mathcal{G}$ of transaction graphs and the *refinement operator* ρ satisfy the conditions of Theorem 2 below, the FCISM problem can be solved in incremental polynomial time. To state the theorem, we recall some basic notions for refinement operators (see, e.g., [9]). A refinement operator Ξ for a poset $(\mathcal{L}, \leq)$ is a function $\Xi : \mathcal{L} \to 2^{\mathcal{L}}$ with $\Xi(P_1) \subseteq \{P_2 : P_1 \leq P_2\}$ for all $P_1 \in \mathcal{L}$. That is, $\Xi(P_1)$ is a subset of the set of specializations of P_1.[6] For Ξ, we define the n-th power $\Xi^n : \mathcal{L} \to 2^{\mathcal{L}}$ recursively

[6] We note that refinement operators based on pattern specializations are referred to as *downward* refinement operators in Inductive Logic Programming (ILP). In ILP, however, this terminology has been introduced for the dual poset $(\mathcal{L}, \leq)$ (see, p.226 in [9] for a detailed discussion about the historical background of the controversial terminology).

by

$$\Xi^n(P_1) = \begin{cases} \Xi(P_1) & \text{if } n = 1 \\ \Xi(\Xi^{n-1}(P_1)) & \text{o/w} \end{cases}$$

for all $n \in \mathbb{N}$. Finally, we say that Ξ is *complete*, if for all $P_1 \in \mathcal{L}$, there is some $n \in \mathbb{N}$ with $P_1 \in \Xi^n(\bot)$, where $\bot$ denotes the empty graph. Using the above notions, we can formulate the following generic theorem:

Theorem 2. *Let $\mathcal{G}$ be the class of the transaction graphs, $\mathcal{L}$ be the set of connected graphs in $\mathcal{G}$, and $\rho : \mathcal{L} \to 2^{\mathcal{L}}$ be a refinement operator. If ρ and $\mathcal{G}$ satisfy the conditions below then Algorithm 1 solves the FCISM problem in incremental polynomial time and in incremental polynomial space.*

(i) $\mathcal{G}$ is closed under taking subgraphs.
(ii) The membership problem in $\mathcal{G}$ can be decided in polynomial time.
(iii) ρ is complete and $\rho(P)$ can be computed in time polynomial in the combined size of the input and the set of frequent patterns listed so far by Algorithm 1.
(iv) Isomorphism can be decided in polynomial time for $\mathcal{G}$.
(v) For every $H, G \in \mathcal{G}$ such that H is connected, it can be decided in time polynomial in the combined size of the input and the set of frequent patterns listed so far by Algorithm 1 whether $H \preccurlyeq_i G$.

Proof. It follows directly from the remarks and concepts above.

The following positive result on forests can immediately be obtained by applying the theorem above (see, also, [5]):

Corollary 1. *The FCISM problem can be solved in incremental polynomial time for forest transaction graphs.*

4 Mining Graphs of Bounded Tree-width

In this section we generalize the positive result of Corollary 1 to graphs of bounded tree-width and prove the main result of this paper:

Theorem 3. *The FCISM problem can be solved in incremental polynomial time for graphs of bounded tree-width.*

Before proving this result, we first note that the class of bounded tree-width graphs is not only of theoretic interest, but also of practical relevance. As an example, consider the ZINC dataset[7] consisting of more than 16 million chemical compounds. Regarding the distribution of the molecular graphs with respect to their tree-width, 99.99% of the 16,501,334 molecular graphs in this dataset have tree-width at most 3 and 99, 31% only tree-width at most 2.

To prove Theorem 3, it suffices to show that all conditions of Theorem 2 hold for bounded tree-width graphs. The proof of the claims in the theorem below is shown in [5] for the positive result on frequent connected subgraph mining in graphs of bounded tree-width, as the conditions considered in the theorem are all independent of the underlying pattern matching operator.

[7] We used a commercial version of the ZINC dataset for the tree-width statistics.

Theorem 4. *For the class of graphs of bounded tree-width, conditions (i), (ii), and (iv) of Theorem 2 hold.*

Thus, only conditions (iii) and (v) have to be proven. We first show (iii).

Theorem 5. *For the class of graphs of bounded tree-width there exists a refinement operator ρ satisfying condition (iii) of Theorem 2.*

Proof. For a connected pattern P with $\mathrm{tw}(P) \leq k$, define the refinement $\rho(P)$ of P as follows: A connected graph P' with $\mathrm{tw}(P') \leq k$ is in $\rho(P)$ iff P' has a vertex v with degree at most k such that $P \simeq P' \ominus v$. Notice that this definition is unique, as isomorphic graphs are not distinguished from each other by definition. Clearly, $\rho(P)$ is a subset of the set of direct specializations of P. Utilizing condition (i) of Theorem 2 and the basic fact that every graph of tree-width at most k has a vertex of degree at most k,[8] the completeness of ρ follows directly by induction on the number of vertices.

We now turn to the complexity of computing $\rho(P)$ and show the stronger property that $\rho(P)$ can actually be computed in time polynomial in the size of DB. Since each new vertex v is connected to P by at least one and at most k vertices, for the cardinality of $\rho(P)$ we have

$$|\rho(P)| \leq \sum_{i=1}^{k} |\Lambda|^{i+1} \binom{n}{i} < |\Lambda|^{k+1}(n+1)^k \ ,$$

where Λ is the set of vertex and edge labels used in DB and n is the number of vertices of P. Since k is a constant, $|\rho(P)|$ is polynomial in the size of DB, and hence, as condition (iv) of Theorem 2 holds by Theorem 4, $\rho(P)$ can be computed in time polynomial in the size of DB, as claimed. □

It remains to show for the proof of Theorem 3 that condition (v) also holds. In Section 4.1 we first recall from [4] a dynamic programming algorithm deciding induced subgraph isomorphism for a restricted class of bounded tree-width graphs. Given a connected graph H and a transaction graph G, both of bounded tree-width, this algorithm decides $H \preccurlyeq_i G$ by computing recursively a certain set of tuples representing *partial* induced subgraph isomorphisms between induced subgraphs of H and G. The problem is, however, that for arbitrary bounded tree-width graphs, the number of such partial solutions can be exponential in the size of H. Using the paradigm developed in [5] for frequent connected subgraph mining, we will show that $H \preccurlyeq_i G$ can be decided by computing only a polynomial number of *new* partial solutions and efficiently recovering all missing partials solutions from those calculated for the already generated frequent patterns.

[8] This fact holds trivially if the graph has at most $k+1$ vertices; o/w it has a tree-decomposition of tree-width at most k with a leaf z having a parent z' such that $X_z \not\subseteq X_{z'}$. But then there is a $v \in X_z$ that is not in the bag of any other node in the tree-decomposition and thus, v can be adjacent only to the vertices in $X_z \setminus \{v\}$.

4.1 A Dynamic Programming Algorithm

To make the paper as self-contained as possible, in this section we recall the dynamic programming algorithm from [4] that decides induced subgraph isomorphism for a restricted class of bounded tree-width graphs. The algorithm is based on an efficient algorithm [8] deciding various morphisms between bounded tree-width *and* bounded degree graphs, which, in turn, follows a generic dynamic programming approach designed in [2]. In order to be consistent with [5] on frequent connected subgraph mining in graphs of bounded tree-width, we naturally adapt the notions and notations from Section 4.1 of [5] from subgraph isomorphism to induced subgraph isomorphism.

In what follows, let H and G denote connected graphs with $\mathrm{tw}(H), \mathrm{tw}(G) \leq k$. In fact, as one can easily see, the results of this section hold also for the case that G is not connected. Given H and G, the algorithm in [4] decides whether $H \preccurlyeq_i G$ by computing a nice tree-decomposition $NTD(G)$ of G, traversing $NTD(G)$ in a postorder manner, calculating for each node in the tree-decomposition a set of tuples, called *characteristics*, and by testing whether the root of $NTD(G)$ has a characteristic satisfying a certain condition formulated in Lemma 1 below. More precisely, an *iso-quadruple* of H relative to a node z of $NTD(G)$ is a quadruple $(S, \mathcal{D}, K, \psi)$, where (i) $S \subseteq V(H)$ with $|S| \leq k+1$, (ii) $\mathcal{D} \subseteq \mathcal{C}(H[V(H) \setminus S])$, (iii) $K = H[S \cup V(\mathcal{D})]$, and (iv) $\psi : S \to X_z$ is an *induced* subgraph isomorphism from $H[S]$ to $G[X_z]$. Notice that K is redundant; it is used for keeping the explanation as simple as possible. The set of all iso-quadruples of H relative to a node z of $NTD(G)$ is denoted by $\Gamma(H, z)$.

For a node z in $NTD(G)$, an iso-quadruple $(S, \mathcal{D}, K, \psi) \in \Gamma(H, z)$ is a *z-characteristic* of H if there exists an induced subgraph isomorphism φ from K to $G_{[z]}$ satisfying (i) $\varphi(u) = \psi(u)$ for all $u \in S$ and (ii) $\varphi(v) \notin X_z$ for all $v \in V(\mathcal{D})$. These definitions imply that $\varphi(u) \in X_z$ for all $u \in S$. The set of all z-characteristics of H relative to z is denoted by $\Gamma_{\mathrm{ch}}(H, z)$. Clearly, $\Gamma_{\mathrm{ch}}(H, z) \subseteq \Gamma(H, z)$. The following lemma from [4] provides a characterization of induced subgraph isomorphism in terms of r-characteristics for the root r of $NTD(G)$.

Lemma 1. *Let r be the root of a nice tree-decomposition $NTD(G)$ of G. Then $H \preccurlyeq_i G$ iff there exists $(S, \mathcal{D}, K, \psi) \in \Gamma_{\mathrm{ch}}(H, r)$ with $K = H$.*

Thus, by the lemma above, we need to calculate the characteristics of the root of $NTD(G)$. Lemma 2 below from [4] shows how to compute the set of characteristics for leafs, and how for internal (i.e., separator or join) nodes from the sets of characteristics of their children. This enables the computation of the characteristics for all nodes of $NTD(G)$ by a postorder traversal of $NTD(G)$.

Lemma 2. *Let G, H be connected graphs of bounded tree-width and z be a node in $NTD(G)$. For all $(S, \mathcal{D}, K, \psi) \in \Gamma(H, z)$ it holds that $(S, \mathcal{D}, K, \psi) \in \Gamma_{\mathrm{ch}}(H, z)$ iff one of the following conditions holds:*

LEAF: *z has no children and $\mathcal{D} = \emptyset$.*

SEPARATOR: *z has a single child z' and $\exists(S', \mathcal{D}', K', \psi') \in \Gamma_{\mathrm{ch}}(H, z')$ with*
(S.a) $S = \{v \in S' : \psi'(v) \in X_z\}$,
(S.b) $\mathcal{D}' = \{D' \in \mathcal{C}(H[V(H) \setminus S']) : D'$ *is a subgraph of some* $D \in \mathcal{D}\}$,
(S.c) $\psi(v) = \psi'(v)$ *for every* $v \in S$.
JOIN: *z has two children z_1, z_2 and there exist $(S_1, \mathcal{D}_1, K_1, \psi_1) \in \Gamma_{\mathrm{ch}}(H, z_1)$ and $(S_2, \mathcal{D}_2, K_2, \psi_2) \in \Gamma_{\mathrm{ch}}(H, z_2)$ satisfying*
(J.a) $S_i = \{v \in S : \psi(v) \in X_{z_i}\}$ *for* $i = 1, 2$,
(J.b) the connected components of $\mathcal{D}$ are partitioned into $\mathcal{D}_1$ and $\mathcal{D}_2$, and
(J.c) for $i = 1, 2$, $\psi_i(v) = \psi(v)$ *for every* $v \in S_i$.

As mentioned, Lemma 2 provides a polynomial time algorithm for deciding induced subgraph isomorphism for restricted subclasses of bounded tree-width graphs (e.g., when the degree is also bounded [8] or when the graphs have log-bounded fragmentation [4]). Clearly, the algorithm is exponential for arbitrary bounded tree-width graphs; this follows directly from the negative result in [8].

It is important to stress that almost the same notions and conditions are used for the frequent *connected subgraph* mining (FCSM) problem (cf. Section 4.1 in [5]), where, in contrast to the FCISM problem, ordinary subgraph isomorphism is the underlying pattern matching operator. The only difference is in the definition of iso-quadruples, in particular, in the definition of ψ, in accordance with the semantic difference between the FCSM and FCISM problems. However, as it turns out in Section 4.2 below, we need a different combinatorial arguments to show the positive result for the FCISM problem.

4.2 Feasible Iso-Quadruples

Like in the FCSM problem, the main source of computational intractability of the algorithm based on Lemma 2 is the possibly exponential number of iso-quadruples needed to test. Using the paradigm developed for the FCSM problem [5], in this section we show that for each node of $NTD(G)$, it suffices to check only a polynomial number of iso-quadruples, as we can utilize the characteristics of the frequent patterns computed earlier by Algorithm 1. In order to show this result, we recall some necessary notions from [5]. As for the case of the FCSM problem, for all transaction graphs we fix a nice tree-decomposition computed in a preprocessing step for the entire mining process.

Let H_1, H_2, and G be connected graphs of bounded tree-width, $NTD(G)$ be some fixed nice tree-decomposition of G, and z be a node in $NTD(G)$. For any two $\xi_1 = (S_1, \mathcal{D}_1, K_1, \psi_1) \in \Gamma(H_1, z)$ and $\xi_2 = (S_2, \mathcal{D}_2, K_2, \psi_2) \in \Gamma(H_2, z)$, ξ_1 is *equivalent* to ξ_2, denoted $\xi_1 \equiv \xi_2$, if there is an isomorphism π between K_1 and K_2 such that π is a bijection between S_1 and S_2 and $\psi_1(v) = \psi_2(\pi(v))$ for every $v \in S_1$. The lemma below from [5] shows that it suffices to store only one representative z-characteristic for each equivalence class of the set of z-characteristics and that equivalence between iso-quadruples can be decided in polynomial time.

Lemma 3. *Let G, H_1, and H_2 be connected graphs of tree-width at most k, z be a node in $NTD(G)$, and $\xi_i = (S_i, \mathcal{D}_i, K_i, \psi_i) \in \Gamma(H_i, z)$ $(i = 1, 2)$. Then*

(i) $\xi_1 \in \Gamma_{\mathrm{ch}}(H_1, z)$ *iff* $\xi_2 \in \Gamma_{\mathrm{ch}}(H_2, z)$ *whenever* $\xi_1 \equiv \xi_2$ *and*
(ii) $\xi_1 \equiv \xi_2$ *can be decided in time* $O\left(n^{k+4.5}\right)$.

For a strong candidate pattern H generated by Algorithm 1 (i.e., which satisfies both conditions in line 5), let $\mathcal{F}_H$ denote the set of patterns consisting of H and all frequent patterns listed before H. For a transaction graph G and node z of $NTD(G)$, an iso-quadruple $\xi \in \Gamma(H, z)$ of a strong candidate pattern H is *redundant* if there are $P \in \mathcal{F}_H \setminus \{H\}$ and $\xi' \in \Gamma(P, z)$ with $\xi \equiv \xi'$; otherwise, ξ is *non-redundant*. Finally, $\Gamma_{\mathrm{nr}}(H, z)$ and $\Gamma_{\mathrm{nr,ch}}(H, z)$ denote the set of non-redundant iso-quadruples of H relative to a node z in $TD(G)$ and the set of non-redundant z-characteristics of H, respectively.

Proposition 2 below implies that for a strong candidate pattern H and $\xi \in \Gamma(H, z)$, it has to be tested whether ξ is a z-characteristic of $TD(G)$ only when ξ is non-redundant; otherwise, it suffices to check whether ξ is equivalent to a non-redundant z-characteristic for some frequent pattern $P \in \mathcal{F}_H \setminus \{H\}$ (see [5] for the proof).

Proposition 2. *Let* $\xi \in \Gamma(H, z)$ *for a strong candidate pattern* H*, transaction graph* G*, both of bounded tree-width, and node* z *in* $NTD(G)$*. Then* $\xi \in \Gamma_{\mathrm{ch}}(H, z)$ *iff there exists* $\xi' \in \bigcup_{P \in \mathcal{F}_H} \Gamma_{\mathrm{nr,ch}}(P, z)$ *with* $\xi \equiv \xi'$.

Thus, induced subgraph isomorphism can be decided by using the non-redundant z-characteristics of the frequent patterns only. Instead of non-redundant iso-quadruples, as we will show below, we can use an efficiently computable superset of them, the set of *feasible* iso-quadruples. We first state a lemma that provides a necessary condition of non-redundancy.

Lemma 4. *Let* H*,* G*, and* z *be as in Proposition 2 and* $\xi \in \Gamma_{\mathrm{nr}}(H, z)$ *with* $\xi = (S, \mathcal{D}, K, \psi)$*. Then, for all vertices* $v \in V(H) \setminus V(K)$ *it holds that*

(i) the degree of v *in* H *is at least 2 and*
(ii) v *is a cut vertex in* H.

Proof. The proof of (i) applies a similar argument used for ordinary subgraph isomorphism [5]. In particular, suppose for contradiction that $V(H) \setminus V(K)$ has a vertex v with degree 1 in H. Since, by assumption, H contains at least one edge and is connected, it has no isolated vertices. Let H' be the graph obtained from H by removing v and the (only) edge adjacent to it. Clearly, H' is a connected induced subgraph of H. Since H is a strong candidate pattern, H' is a frequent connected induced subgraph and has therefore already been generated by Algorithm 1. Furthermore, K is an induced subgraph of H' implying $\xi \in \Gamma(H', z)$. But ξ is then redundant for H, contradicting the assumption.

To prove (ii), suppose there is a non-cut vertex $v \in V(H) \setminus V(K)$ of H. Let $H' = H \ominus v$. Since H is connected and v is a non-cut vertex of H, H' is connected. Similarly to the previous case, it holds that H' contains K as an induced subgraph because all edges that have been removed are outside of $E(K)$. Thus, $\xi \in \Gamma(H', z)$ contradicting that it is non-redundant. □

We now show that for any $S \subseteq V(H)$ of constant size, only a constant number of connected components in $H[V(H) \setminus S]$ can fulfill the two conditions of Lemma 4. Although the statement formulated below is similar to the corresponding claim stated for the case of ordinary subgraph isomorphism in [5], the arguments used in the proofs are entirely different, due to the difference between ordinary and induced subgraph isomorphism.

Lemma 5. *Let H be a strong candidate pattern generated by Algorithm 1, $S \subseteq V(H)$ with $|S| \leq k+1$, and $\mathcal{C}_A$ be the set of connected components C from $\mathcal{C}(H[V(H) \setminus S])$ such that for all $v \in C$, v satisfies both conditions of Lemma 4. Then*

$$|\mathcal{C}_A| \leq \binom{k+1}{2} .$$

To show the claim above, we first prove two technical lemmas.

Lemma 6. *Let H, S, and $\mathcal{C}_A$ be as defined in Lemma 5. Then for all $C \in \mathcal{C}_A$, C is connected to S by at least two edges ending in different vertices in S.*

Proof. The claim is straightforward if $|V(C)| = 1$; H has no parallel edges by construction and the only vertex of C for this case must be connected to S by at least two edges, as it is a cut vertex in H.

The proof of the case $|V(C)| > 1$ utilizes the fact that every connected graph has at least two non-cut vertices. More precisely, let u be a non-cut vertex of C. Since u is a cut vertex in H by condition (i) of Lemma 4, it must be the case that u is connected to at least one vertex in S. Thus, C is connected to S by at least two edges, as it has at least two non-cut vertices. Suppose that all non-cut vertices of C are adjacent to the same vertex, say w, in S. Let $u, v \in V(C)$ be different non-cut vertices of C. Since, on the one hand, u is a non-cut vertex of C, and, on the other hand, it is a cut vertex in H by the condition of the lemma, there are two vertices $x, y \in V(H)$ that are disconnected by u (i.e., x and y belong to different connected components of $H \ominus u$). Since u is a non-cut vertex of C, at most one of x and y can belong to C. It can easily be seen for this case that in fact, *exactly* one of x and y, say x, belongs to C. Furthermore, $\{u, w\}$ must be an edge on the path connecting x and y in H, i.e., x and y are connected by a path of the form $P_{x,u} + \{u, w\} + P_{w,y}$, where $P_{x,u}$ is a path in C. Since $C \ominus u$ remains connected, there is a path $P_{x,v}$ connecting x and v in $C \ominus u$. Thus, the path $P_{x,v} + \{v, w\} + P_{w,y}$ connects x and y in $H \ominus u$, contradicting that u disconnects x and y. Hence, all connected components in $\mathcal{C}_A$ are connected to at least two different vertices in S, as stated. □

The second lemma states that each connected component of $\mathcal{C}_A$ "connects" such two vertices of S that are not "connected" by any other component of $\mathcal{C}_A$.

Lemma 7. *Let H, S, and $\mathcal{C}_A$ be as defined in Lemma 5. Then for all connected components $C \in \mathcal{C}_A$, there exist $u', v' \in S$ such that*

(i) $u' \neq v'$ and $\{u, u'\}, \{v, v'\} \in E(H)$ for some $u, v \in V(C)$, and
(ii) for all $C' \in \mathcal{C}_A \setminus \{C\}$, at least one of u' and v' is not adjacent to C'.

Proof. By Lemma 6, for all $C \in \mathcal{C}_A$ there are $u', v' \in S$ satisfying (i). Thus, to show the claim above, suppose for contradiction that there exists a connected component $C \in \mathcal{C}_A$ with the following property: for all $u', v' \in S$ satisfying (i) for C, there is a C' such that *both* u' and v' are adjacent to C'. Let u be the only vertex of C if $|V(C)| = 1$; otherwise let u be a non-cut vertex of C. Since $C \in \mathcal{C}_A$, u is a cut vertex in H by condition and hence, there are $x, y \in V(H)$ such that u disconnects x and y in H. Depending on the number of vertices of C and on the membership of x and y in C, we distinguish the following cases by noting that the case $x, y \in V(C)$ cannot occur by the choice of u:

Case 1. Suppose $|V(C)| = 1$. Then x and y must be connected in H by a path of the form $P_{x,v} + \{v, u\} + \{u, w\} + P_{w,y}$ for some $v, w \in S$ with $v \neq w$, where the length of $P_{x,v}$ and $P_{w,y}$ can be zero. By assumption, v and w are adjacent to some $C' \in \mathcal{C}_A$ and thus, there is a path $P_{v,w}$ in H that does not contain u. But x and y are then connected in H by the path $P_{x,v} + P_{v,w} + P_{w,y}$, contradicting that u disconnects x and y.

Case 2. Suppose $x \in V(C)$. Then x cannot be a non-cut vertex of C, as in this case x must be adjacent to a vertex $x' \in S$, which, in turn, is not adjacent to u. It can be shown in a way similar to the proof of Case 1, that for this case there is a path in H connecting x and y that does not contain u, a contradiction. Thus, as C has at least two non-cut vertices if $|V(C)| > 1$, there is a non-cut vertex $v \in V(C)$ with $v \neq u$. Since u disconnects x and y in H, there is a path of the form $P_{x,u} + \{u, u'\} + P_{u',y}$, where $P_{x,u}$ is a path in C, $\{u, u'\}$ is an edge of H with $u' \in S$, and $P_{u',y}$ is a path connecting u' and y in H. Let $v' \in S$ be a vertex adjacent to v. Notice that $v' \neq u'$, as otherwise the path $P_{x,v} + \{v, u'\} + P_{u',y}$ connects x and y in $H \ominus u$, contradicting that u disconnects x and y; clearly, a path $P_{x,v}$ connecting x and v in $C \ominus u$ always exists, as u is a non-cut vertex of C. Thus, u', v' fulfill condition (i) and hence, u' and v' are connected by a path $P_{u',v'}$ via some connected component $C' \in \mathcal{C}_A$ by assumption. But then x and y are connected by the path $P_{x,v} + \{v, v'\} + P_{v',u'} + P_{u',y}$ in $H \ominus u$, a contradiction.

Case 3. The case of $x, y \notin V(C)$ can be shown in a way similar to the cases above. □

The proof of Lemma 5 follows directly from Lemma 7 and from $|S| \leq k + 1$. Following the paradigm designed for the FCSM problem in [5], we define *feasible* iso-quadruples, a superset of non-redundant iso-quadruples, and formulate in Theorem 6 the main result of this section which states that feasible iso-quadruples can be used correctly to decide induced subgraph isomorphism and that the number of feasible iso-quadruples of a strong candidate pattern is polynomial in the pattern's size. More precisely, for a strong candidate pattern H generated by Algorithm 1 and for a node z in $NTD(G)$ of a transaction graph of bounded tree-width, an iso-quadruple $\xi \in \Gamma(H, z)$ is called *feasible* if it satisfies the conditions of Lemma 4. The sets of feasible iso-quadruples relative to z and feasible z-characteristics are denoted by $\Gamma_{\mathrm{f}}(H, z)$ and $\Gamma_{\mathrm{f,ch}}(H, z)$, respectively.

Theorem 6. *Let H be a strong candidate pattern generated by Algorithm 1 and z be a node of $NTD(G)$ for some transaction graph G with $\mathrm{tw}(G) \leq k$. Then*

(i) $\Gamma_{\mathrm{nr}}(H, z) \subseteq \Gamma_{\mathrm{f}}(H, z)$,
(ii) for all $\xi \in \Gamma(H, z)$, $\xi \in \Gamma_{\mathrm{ch}}(H, z)$ *iff there exist a* $\xi' \in \bigcup_{P \in \mathcal{F}_H} \Gamma_{\mathrm{f,ch}}(P, z)$ *with* $\xi \equiv \xi'$,
(iii) $|\Gamma_{\mathrm{f}}(H, z)| = O\left(|V(H)|^{k+1}\right)$, *and*
(iv) $\Gamma_{\mathrm{f}}(H, z)$ *can be computed in time polynomial in the size of* H.

Proof. The proof of (i) is immediate from the definitions and from Lemma 4. (ii) follows from Proposition 2 and from the fact that $\bigcup_{P \in \mathcal{F}_H} \Gamma_{\mathrm{nr,ch}}(P, z)$ and $\bigcup_{P \in \mathcal{F}_H} \Gamma_{\mathrm{f,ch}}(P, z)$ are equal up to equivalence. To show (iii), let $S \subseteq V(H)$ with $|S| \leq k+1$ and $\mathcal{C}_A$ be the set of connected components as defined in Lemma 5. By definition, for every $\xi = (S, \mathcal{D}, K, \psi) \in \Gamma_{\mathrm{f}}(H, z)$, $\mathcal{D}$ contains all connected components in $\mathcal{C}(H[V(H) \setminus S])$ that are not in $\mathcal{C}_A$. For a fixed subset $S \subseteq V(H)$ with $|S| \leq k+1$ and for a fixed injective function ψ mapping S to the bag X_z of z, the number of possible feasible quadruples is bounded by $2^{|\mathcal{C}_A|}$, which, in turn, is bounded by $2^{\binom{k+1}{2}}$ by Lemma 5. The number of induced subgraph isomorphisms from $H[S]$ to $G[X_z]$ is at most the number of injective functions from S to the bag X_z of z, which is bounded by $(k+1)!$. Since S can be chosen in at most $|V(H)|^{k+1}$ different ways, we have

$$|\Gamma_{\mathrm{f}}(H, z)| \leq 2^{\binom{k+1}{2}} \cdot (k+1)! \cdot |V(H)|^{k+1} \ ,$$

from which we get (iii) by noting that k is a constant. Finally, (iv) holds along the lines in the proof of (iii) above by noting that all cut vertices of H can be found in time $O(V(H) + E(H))$ and it can be decided whether an injective function $\psi : S \to X_z$ is an induced subgraph isomorphism from $H[S]$ to $G[X_z]$ in constant time, as $|S|, |X_z| \leq k+1$. □

5 Deciding Induced Subgraph Isomorphism

In this section we show how to utilize feasible characteristics efficiently for deciding induced subgraph isomorphism. Let H be a strong candidate pattern generated by Algorithm 1 and G be a transaction graph, both of tree-width at most k. Furthermore, let $NTD(G)$ be a nice tree-decomposition of G and r the root of $NTD(G)$. By Lemma 1 and Theorem 6, $H \preccurlyeq_i G$ iff there is a feasible r-characteristic $(S, \mathcal{D}, K, \psi) \in \Gamma_{\mathrm{f,ch}}(H, r)$ with $K = H$. The algorithm deciding $H \preccurlyeq_i G$ assumes that all nodes z in $NTD(G)$ is associated with a set containing all elements of $\Gamma_{\mathrm{f,ch}}(P, z)$, for all frequent patterns $P \in \mathcal{F}_H \setminus \{H\}$. It visits the nodes of $NTD(G)$ in postorder traversal and calculates first $\Gamma_{\mathrm{f}}(P, z)$ for all nodes z visited; this can be done in time polynomial in the size of H by (iv) of Theorem 6. It then computes $\Gamma_{\mathrm{f,ch}}(P, z)$ by testing for all $\xi = (S, \mathcal{D}, K, \psi) \in \Gamma_{\mathrm{f}}(P, z)$ whether ξ is a characteristic. Depending on the type of z, this test can be performed by checking the condition given in the corresponding case below:

LEAF: By (LEAF) of Lemma 2, ξ is a characteristic iff $\mathcal{D} = \emptyset$.

SEPARATOR: Let z' be the child of z in $NTD(G)$ and let $\mathfrak{S}(\xi)$ be the set of all iso-quadruples $\xi' \in \Gamma(H, z')$ that satisfy conditions (S.a)–(S.c) of Lemma 2. Using similar arguments as in the proof of Lemma 16 in [5], one can show that (i) ξ is a characteristic iff $\Gamma_{\mathrm{f,ch}}(H, z') \cap \mathfrak{S}(\xi) \neq \emptyset$ and (ii) $\mathfrak{S}(\xi) \subseteq \Gamma_{\mathrm{f}}(H, z')$ and thus, it can be computed in time polynomial in the size of H.

JOIN: Let z_1 and z_2 be the two children of z in $NTD(G)$. To give the condition for this case, we need a definition. Let $\xi_i = (S_i, \mathcal{D}_i, K_i, \psi_i) \in \Gamma(P_i, z_i)$ for some $P_i \in \bigcup_{P \in \mathcal{F}_H}$ $(i = 1, 2)$. We assume w.l.o.g. that K, K_1, and K_2 are pairwise vertex disjoint. The *join* of ξ_1 and ξ_2 with respect to ξ, denoted $\oplus_\xi(\xi_1, \xi_2)$, is an iso-quadruple $(S', \mathcal{D}_1 \cup \mathcal{D}_2, K', \psi')$ relative to z obtained from $(S_1 \cup S_2, \mathcal{D}_1 \cup \mathcal{D}_2, K_1 \cup K_2, \psi_1 \cup \psi_2)$ by (i) replacing u_1 and u_2 in $S_1 \cup S_2$, $K_1 \cup K_2$, and $\psi_1 \cup \psi_2$ with a new vertex u for all vertex pairs $u_1 \in S_1$ and $u_2 \in S_2$ with $\psi_1(u_1) = \psi_2(u_2)$ and by (ii) connecting in K' all original vertices $u, v \in S'$ with $u \in S_1$ and $v \in S_2$ by an edge labeled by ℓ if the vertices $u', v' \in S$ with $\psi(u') = \psi_1(u)$ and $\psi(v') = \psi_2(v)$ are connected in K by an edge labeled with ℓ. One can check that this definition is in fact an adaptation of conditions (J.a)–(J.c) of Lemma 2. In a way similar to the proof of Lemma 17 in [5], one can show that (i) ξ is a characteristic iff there are $\xi_i = (S_i, \mathcal{D}_i, K_i, \psi_i) \in \bigcup_{P \in \mathcal{F}_H} \Gamma_{\mathrm{f,ch}}(P, z_i)$ for $i = 1, 2$ with $\xi \equiv \oplus_\xi(\xi_1, \xi_2)$ and that (ii) $\oplus_\xi(\xi_1, \xi_2)$ can be computed in time polynomial in the size of ξ, ξ_1, and ξ_2 for any ξ_1, ξ_2, implying that it can be decided in time polynomial in the size of $\mathcal{F}_H$, i.e., in *incremental polynomial time*, whether ξ is a characteristic.

Combining the arguments above with Lemma 1, we get Theorem 7 below for condition (v) of Theorem 2. Together with Theorems 4 and 5, this completes the proof of our main result stated in Theorem 3.

Theorem 7. *For every $H, G \in \mathcal{G}$ such that H is connected, it can be decided in time polynomial in the combined size of the input and the set of frequent patterns listed so far by Algorithm 1 whether $H \preccurlyeq_i G$.*

6 Concluding Remarks

By the main result of this paper, the FCISM problem can be solved in incremental polynomial time for bounded tree-width graphs. The positive results on the FCSM problem in [5] and on the FCISM problem in this work suggest the investigation of further, computationally hard pattern matching operators for bounded tree-width graphs, such as, for example (induced) homeomorphism or (induced) minor embedding. We suspect that the systematic study of these and other pattern matching operators will result in an efficient *parameterized* frequent pattern mining algorithm for graphs of bounded tree-width, with the pattern matching operator as the parameter. Designing such a *generic* pattern mining algorithm is a very challenging project because, as the results in [5] and in this paper

show, different pattern matching operators may require entirely different pattern refinement operators and entirely different combinatorial characterizations of feasible iso-quadruples.

The results of this paper raise some interesting open problems. For example, it is an open question whether the positive result formulated in Theorem 3 can further be strengthened. In particular, can the FCISM problem be solved with *polynomial delay* for bounded tree-width graphs? By setting the frequency threshold t to 1, our main result implies that one can efficiently generate all *distinct* connected induced subgraphs of a bounded tree-width graph. Does this positive result hold for arbitrary graphs as well? Or does the negative result given in Theorem 1 apply even to the special case that the database contains a single (arbitrary) graph and the frequency threshold is set to 1?

Finally we note that we are going to design and implement a practically fast algorithm listing frequent connected induced subgraphs for graphs of tree-width at most 3. For this graph class, motivated practically e.g. by pharmacological molecules (see the statistics with the ZINC dataset in Section 4), there are linear time recognition algorithms [1]. Though the arguments used for join nodes in Section 5 might suggest that we need time quadratic in the size of $\mathcal{F}_H$, one can show that this test can be carried out actually in time only *linear* in it.

References

1. S. Arnborg and A. Proskurowski. Characterization and recognition of partial 3-trees. *SIAM Journal Algebraic Discrete Methods*, 7(2):305–314, 1986.
2. S. Arnborg and A. Proskurowski. Linear time algorithms for NP-hard problems on graphs embedded in k-trees. *Discrete Applied Mathematics*, 23:11–24, 1989.
3. H.L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1–2):1–45, 1998.
4. M. Hajiaghayi and N. Nishimura. Subgraph isomorphism, log-bounded fragmentation, and graphs of (locally) bounded treewidth. *Journal of Computer and System Sciences*, 73(5):755–768, 2007.
5. T. Horváth and J. Ramon. Efficient frequent connected subgraph mining in graphs of bounded tree-width. *Theoretical Computer Science*, 411(31-33):2784–2797, 2010.
6. D.S. Johnson, C.H. Papadimitriou, and M. Yannakakis. On generating all maximal independent sets. *Information Processing Letters*, 27(3):119–123, 1988.
7. H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.
8. J. Matousek and R. Thomas. On the complexity of finding iso- and other morphisms for partial k-trees. *Discrete Mathematics*, 108(1-3):343–364, 1992.
9. S.-H. Nienhuys-Cheng and R. de Wolf, editors. *Foundations of Inductive Logic Programming*, volume 1228 of *Lecture Notes in Computer Science*. Springer, 1997.
10. N. Robertson and P.D. Seymour. Graph minors. II. Algorithmic Aspects of Tree-Width. *Journal of Algorithms*, 7(3):309–322, 1986.
11. M.M. Syslo. The subgraph isomorphism problem for outerplanar graphs. *Theoretical Computer Science*, 17:91–97, 1982.