Graph Contraction Pattern Matching for Graphs of Bounded Treewidth

Takashi Yamada and Takayoshi Shoudai

Department of Informatics, Kyushu University, Japan
{takashi.yamada,shoudai}@inf.kyushu-u.ac.jp

Abstract. A *graph contraction pattern* is a triplet $h = (V, E, U)$ where (V, E) is a connected graph and U is a distinguished subset of V. The graph contraction pattern matching problem is defined as follows. Given a graph contraction pattern $h = (V, E, U)$ and a graph G, can G be transformed to (V, E) by edge contractions so that for any $v \in V \setminus U$, only one vertex in G can be mapped to v? We show that this problem is solvable in polynomial time if (1) (V, E) is of bounded treewidth, (2) U is an independent set of (V, E), and (3) all vertices in U are of bounded degree.

1 Introduction

Large amount of data having graph structures, such as map data, CAD, biomolecular, chemical molecules, the World Wide Web, are stored in databases. Almost chemical compounds stored in the NCI chemical dataset[1] are known to be expressed by outerplanar graphs. Horváth et al. [2] proposed an efficient frequent subgraph mining algorithm for a dataset expressed by outerplanar graphs. We have been developing general graph patterns with structured variables which can be replaced with arbitrary connected graphs, in order to represent expressive patterns appearing in a given dataset of graphs. In [7], we introduced a general concept of block-preserving graph patterns and presented a frequent graph pattern mining algorithm on outerplanar graphs.

Toward a graph mining on more general classes of graphs, Horváth and Ramon proposed a frequent subgraph mining algorithm for a dataset of graphs of bounded treewidth [3]. Some NP-completeness problems on graphs are solvable in polynomial time if the input can be restricted to graphs of bounded treewidth. From a practical viewpoint, 99.97% of $250,251$ chemical compounds in the NCI chemical dataset are expressed by graphs of treewidth at most 3 [3]. We proposed a graph pattern of bounded treewidth and a polynomial time pattern matching algorithm on the graph patterns [6]. On the other hand, the pattern matching problem on graph patterns is computationally expensive unless a graph pattern is essentially 2-connected [4]. In this paper, in order to discover more than 2-connected graph patterns in a target dataset, we define a new graph pattern

[1] http://cactus.nci.nih.gov.

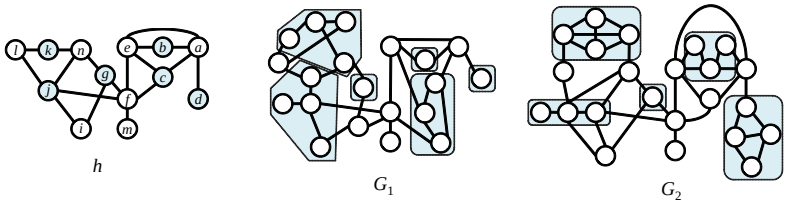

Fig. 1. Let $h = (V, E, U)$ be a graph contraction pattern, where (V, E) is a connected graph of treewidth 2 and $U = \{b, c, d, g, j, k\}$. h is a common pattern of G_1 and G_2

expression, called a *graph contraction pattern*, by using a concept of edge contractions on connected graphs.

H-contractibility problem takes as input two graphs G and H, and asks whether G can be transformed to H by edge contractions. Based on the H-contractibility problem, we define a *graph contraction pattern* as a triplet $h = (V, E, U)$ where (V, E) is a connected graph and U is a distinguished subset of V. The graph contraction pattern matching problem is defined as follows. Given a graph contraction pattern $h = (V, E, U)$ and a graph G, can G be transformed to (V, E) by edge contractions so that for any $v \in V \setminus U$, only one vertex in G can be mapped to v? In Fig. 1, G_1 is transformed to (V, E) by edge contractions so that only one vertex in G_1 is mapped to each vertex in $V \setminus U$. G_2 is also transformed to (V, E) in such a way. In this paper, we show that this problem is solvable in polynomial time if (1) (V, E) is of bounded treewidth, (2) U is an independent set of (V, E), and (3) all vertices in U are of bounded degree.

2 Preliminaries

2.1 Normalized tree decomposition

All graphs in this paper are simple and loopless. For a graph G, we denote by $V(G)$ and $E(G)$ the vertex set and the edge set of G, respectively. We denote by $N(v)$ the set of neighbors of a vertex v. For $U_1, U_2 \subseteq V(G)$, we say that U_1 and U_2 are *adjacent* if there is an edge $\{v, w\}$ such that $v \in U_1$ and $w \in U_2$. For $U \subseteq V(G)$, we denote by $G[U]$ the subgraph of G induced by U.

A *tree-decomposition* of a graph G is a 2-tuple $(T, \mathcal{X})$ where T is a tree and $\mathcal{X} = \{\mathcal{X}(\alpha) \mid \mathcal{X}(\alpha) \subseteq V(G) \text{ for all } \alpha \in V(T)\}$ that satisfies the following three conditions. (1) $\bigcup_{\alpha \in V(T)} \mathcal{X}(\alpha) = V(G)$, (2) $\forall v, w \in V(G)$ $[\{v, w\} \in E(G) \Rightarrow \exists \alpha \in V(T)$ $[\{v, w\} \subseteq \mathcal{X}(\alpha)]]$, and (3) $\forall \alpha, \beta, \gamma \in V(T)$ $[\beta$ is on the path from α to γ in $T \Rightarrow \mathcal{X}(\alpha) \cap \mathcal{X}(\gamma) \subseteq \mathcal{X}(\beta)]$. The *width* of a tree-decomposition $(T, \mathcal{X})$ is $\max_{\alpha \in V(T)} |\mathcal{X}(\alpha)| - 1$. The *treewidth* of a graph G is the minimum width over all possible tree-decompositions of G. We say that a tree-decomposition of a graph G is *optimal* if its width equals to the treewidth of G.

Thereafter, to distinguish from a vertex of graph G, we call a vertex of T a node. Below we assume that the tree T of a tree-decomposition $(T, \mathcal{X})$ is a rooted tree by specifying a node of T. For a tree-decomposition $(T, \mathcal{X})$, we denote by

$T^{\downarrow a}$ the maximal subtree rooted at a node $\alpha \in V(T)$, by $\mathcal{X}(T^{\downarrow\alpha})$ the union of elements of nodes of $T^{\downarrow\alpha}$, i.e., $\mathcal{X}(T^{\downarrow\alpha}) = \bigcup_{\beta \in V(T^{\downarrow\alpha})} \mathcal{X}(\beta)$.

A tree-decomposition $(T, \mathcal{X})$ is *smooth* if $\forall\{\alpha, \beta\} \in E(T)$ $[|\mathcal{X}(\alpha) \setminus \mathcal{X}(\beta)| = |\mathcal{X}(\beta) \setminus \mathcal{X}(\alpha)| = 1]$. A tree-decomposition $(T, \mathcal{X})$ has *subtree connected characteristic* if $\forall\{\alpha, \beta\} \in E(T)$ [β is a child of α and $G[\mathcal{X}(T^{\downarrow\beta}) \setminus \mathcal{X}(\alpha)]$ is connected]. A tree-decomposition $(T, \mathcal{X})$ is *normalized* if it satisfies the following three conditions. (1) $(T, \mathcal{X})$ is optimal, (2) $(T, \mathcal{X})$ is smooth, and (3) T is a rooted tree and $(T, \mathcal{X})$ has subtree connected characteristic. Nagoya et al. [5] gave a polynomial time algorithm for constructing a normalized tree-decomposition from any tree-decomposition.

Theorem 1. [5] *A normalized tree-decomposition of G of treewidth k is obtained from any optimal tree-decomposition of G in $O(kn^2)$ time, where $n = |V(G)|$.*

2.2 Graph contraction pattern

Let G and H be connected graphs. An *H-witness structure* $\mathcal{W} = \{W(u) | u \in V(H)\}$ is a partition of $V(G)$ satisfying the following conditions. (1) $\forall W(u) \in \mathcal{W}$ $[G[W(u)]$ is connected], and (2) $\forall u, u' \in V(H)[\{u, u'\} \in E(H) \Leftrightarrow \exists\{v, w\} \in E(G)[v \in W(u), w \in W(u')]]$. We call each set $W(u) \in \mathcal{W}$ the *H-witness set* of u. When G has an H-witness structure, G can be transformed to H by contracting each of H-witness sets into one vertex by edge contractions.

A *graph contraction pattern* h (*GC-pattern*, for short) is a triplet $h = (V, E, U)$ where (V, E) is a connected graph and U is a subset of V. We denote by $V(h)$ and $E(h)$ the vertex set and the edge set of h, respectively. And for $V' \subseteq V(h)$, we denote by $h[V']$ the graph contraction subpattern (*GC*-subpattern, for short) induced by V', i.e., $h[V'] = (V', E(h) \cap \{\{v, w\} | v, w \in V'\}, U \cap V')$.

We say that a *GC*-pattern h *matches* a graph G if G has a $(V(h), E(h))$-witness structure $\mathcal{W} = \{W(u) | u \in V(h)\}$ satisfying $\forall v \in V(h) \setminus U$ $[|W(v)| = 1]$. We call an element of U a *contractible vertex*, and an element of $V(h) \setminus U$ an *uncontractible vertex*. And for a *GC*-pattern h, a $(V(h), E(h))$-witness structure of h is called an *h-witness structure*.

2.3 Main Results

The *graph contraction pattern matching problem* is to decide whether or not a given *GC*-pattern h matches a given graph G. Our main result is the following theorem.

Theorem 2. Given a *GC*-pattern $h = (V, E, U)$ and a graph G, the *GC*-pattern matching problem is solvable in polynomial time if h satisfies the following three conditions. (1) (V, E) is of bounded treewidth, (2) U is an independent set of (V, E), and (3) all vertices in U are of bounded degree.

We show a polynomial time algorithm that solve this problem in the next section. Our algorithm is based on the idea of the graph isomorphism algorithm for a graph with bounded treewidth in [5]. A *GC*-pattern matching problem becomes intractable if it does not satisfy the condition of Theorem 2.

Theorem 3. *For inputs G and h, the GC-pattern matching problem is NP-complete if the degree of a vertex in U is not bounded by a fixed constant.*

Moreover, we can show the time complexities of this problem in the cases whether or not each condition of Theorem 2 is satisfied. We summarize them in the next table. 'T' means that the condition is satisfied and 'F' means not satisfied. And '*' may be either 'T' or 'F'. The class GI is the set of problems with a polynomial-time reduction to the graph isomorphism problem.

U is an independent set of $(V(h), E(h))$	F [1]	*	T	
All vertices in U are of bounded degree	*	F [this paper]	T	
$(V(h), E(h))$ is of a bounded treewidth	*	*	F	T
Time Complexity	NP-complete		GI-hard	P [this paper]

3 A Pattern Matching Algorithm for *GC*-Patterns

We assume that a given GC-pattern h satisfies the conditions of Theorem 2. And let $(T, \mathcal{X})$ be a normalized tree-decomposition of $(V(h), E(h))$.

In our algorithm, we construct a whole witness structure from a union of partial witness structures. So we need the following definition.

Definition 1. For two GC-subpatterns h_1, h_2 of h, let $\mathcal{W}_1$ and $\mathcal{W}_2$ be h_1- and h_2-witness structures, respectively Then, we say that $\mathcal{W}_1$ *does not contradict* $\mathcal{W}_2$ if it satisfies the following conditions. (1) $W_1(v) = W_2(v)$ for any $v \in V(h_1) \cap V(h_2)$, (2) for any $u \in V(h_i)$ and $v \in V(h_2)$, $\{u, v\} \in E(h) \Leftrightarrow W_1(u)$ and $W_2(v)$ are adjacent.

And, we construct a partial witness structure from an injection.

Definition 2. Let dom be a set of contractible vertices, all of those neighbors, and some uncontractible vertices of h. Then for an injection $\psi : dom \to V(G)$, we construct a *ψ-structure* $\mathcal{W}_\psi = \{W_\psi(v) | v \in dom\}$ as follows. (1) $W_\psi(v) = \{\psi(v)\}$ if v is uncontractible, (2) otherwise, $W_\psi(v)$ is the vertex set of the connected component in $G[V(G) \setminus \psi(N(v))]$ that includes $\psi(v)$.

If ψ is suitable, then $\mathcal{W}_\psi$ becomes one of $h[dom]$-witness structures.

Definition 3. For a node $\alpha \in T$, let $dom(\alpha)$ be a vertex set $\mathcal{X}(\alpha) \cup \{N(v) | v \in \mathcal{X}(\alpha) \cap U\}$. And, we say that a mapping $\psi : dom(\alpha) \to V(G)$ is a *node mapping* of α if $\mathcal{W}_\psi$ is an $h[dom(\alpha)]$-witness structure.

Let α' be the parent node of α. Then $h[\mathcal{X}(T^{\downarrow\alpha}) \setminus \mathcal{X}(\alpha')]$ is connected. We define $D_{\alpha,\psi}$ corresponding to $h[\mathcal{X}(T^{\downarrow\alpha}) \setminus \mathcal{X}(\alpha')]$ as follows. $D_{\alpha,\psi}$ is the connected component in the graph obtained from G by removing $W_\psi(h)$ for each $h \in \mathcal{X}(\alpha) \cap \mathcal{X}(\alpha')$ that includes $W_\psi(v)$ where $v \in \mathcal{X}(\alpha) \setminus \mathcal{X}(\alpha')$. Moreover we define $ISO(\alpha)$ as the set of all node mappings ψ satisfying the following condition. $G[V(D_{\alpha,\psi}) \cup \bigcup_{v \in dom(\alpha)} W_\psi(v)]$ has an $h[\mathcal{X}(T^{\downarrow\alpha}) \cup dom(\alpha)]$-witness structure such that does not contradict $\mathcal{W}_\psi$. From these definitions, we can easily see the following lemma.

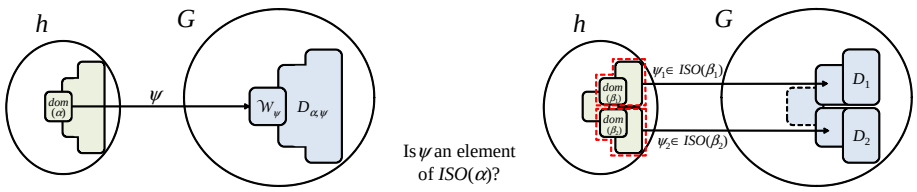

Fig. 2. The algorithm incrementally decides whether a node mapping ψ is an element of $ISO(\alpha)$ or not.

Lemma 1 A GC-pattern h matches a graph G if and only if $ISO(r_T) \neq \emptyset$ where r_T is the root of T.

Let $\beta_1, \ldots, \beta_m$ be the children of α. Then the GC-pattern obtained from $h[\mathcal{X}(T^{\downarrow\alpha}) \setminus \mathcal{X}(\alpha')]$ by removing $\mathcal{X}(\alpha)$ has m connected components. Similarly, the graph obtained from $D_{\alpha,\psi}$ by removing $W_\psi(v)$ for each $v \in \mathcal{X}(\alpha)$ has m' connected components D_i $(i = 1, \ldots, m')$. If $m \neq m'$, G has no h-witness structure that does not contradict $\mathcal{W}_\psi$. Afterwards, we assume that $m = m'$. For each β_i, we assume that there is a node mapping $\psi_{\beta_i} \in ISO(\beta_i)$ such that $\mathcal{W}_{\psi_{\beta_i}}$ does not contradict $\mathcal{W}_\psi$. Let $\mathcal{W}_{\beta_i}$ be a witness structure that makes ψ_{β_i} an element of $ISO(\beta_i)$. Then we construct a witness structure $\mathcal{W}$ as the union of $\mathcal{W}_\psi$ and each $\mathcal{W}_{\beta_i}$. Then, the constructed $\mathcal{W}$ makes ψ an element of $ISO(\alpha)$ (Fig. 1).

Lemma 2 $\psi \in ISO(\alpha)$ if and only if there is an injection from $\beta_1, \ldots, \beta_m$ to $D_1, \ldots, D_m$ satisfying the following condition. If β_i is mapped to D_j, there is a node mapping ψ_{β_i} of β_i such that $\mathcal{W}_{\psi_\beta}$ does not contradict $\mathcal{W}_\psi$ and $(D_{\beta_i,\psi_i} = D_j) \wedge (\psi_{\beta_i} \in ISO(\beta_i))$.

For deciding whether $\psi \in ISO(\alpha)$ or not, we construct a bipartite graph defined as follows and solve the perfect matching problem on the bipartite graph.

Definition 4. For node α, let B be the set of all children of α. And let C be the set of all connected components of the graph obtained from $D_{\alpha,\psi}$ by removing $W_\psi(h)$ for each $h \in \mathcal{X}(\alpha)$. And, let $E = \{\{\beta, D\} | (\beta \in B) \wedge (D \in C) \wedge (\exists \psi_\beta \in ISO(\beta, G)[(\mathcal{W}_{\psi_\beta}$ does not contradict $\mathcal{W}_\psi) \wedge (D = D_{\beta,\psi_\beta})]\}$. Then we define a bipartite graph $Q(\alpha, \psi) = (B, C, E)$.

Lemma 3 The bipartite graph $Q(\alpha, \psi)$ has a perfect matching if and only if $\psi \in ISO(\alpha)$.

We give a formal description of our algorithm in Fig. 3. The correctness of our algorithm is shown from Lemmas 1 and 3. Our algorithm runs in $O(N^{k(d+1)+1.5})$ time, where N is the number of vertices of G and d is the maximum degree of the vertices of U. Then we can show Theorem 2

Algorithm CONTRACTION_PATTERN_MATCHING;
Input: A GC-pattern h, a graph G,
a normalized tree decomposition $(T, \mathcal{X})$ of $(V(h), E(h))$;
begin
forall $\alpha \in V(T)$ **do** $ISO(\alpha) := \emptyset$; // *Initialization.*
while $\exists \alpha \in V(T)$ $[ISO(\beta) \neq \emptyset$ for all children β of $\alpha]$ **do** // *Main Processes.*
forall node mappings ψ of α **do**
if NODE_MAPPING_EXTENSION$(h, (T, \mathcal{X}), \alpha, G, \psi)$ returns *yes* **then**
$ISO(\alpha) := ISO(\alpha) \cup \{\psi\}$;
if $ISO(r_T) \neq \emptyset$ **then return** *yes* **else return** *no* // *Decision.*
end.

Procedure NODE_MAPPING_EXTENSION$(h, (T, \mathcal{X}), \alpha, G, \psi)$;
Input: a GC-pattern h, a normalized tree-decomposition $(T, \mathcal{X})$,
a node $\alpha \in V(T)$, a graph G, and a node mapping ψ of α;
begin
Construct a bipartite graph $Q(\alpha, \psi)$ from h, $(T, \mathcal{X})$, α, G, and ψ;
if $Q(\alpha, \psi)$ has a perfect matching **then return** *yes* **else return** *no*
end;

Fig. 3. GC-pattern matching algorithm.

References

1. A.E. Brouwer and H.J. Veldman. Contractibility and NP-completeness, *Journal of Graph Theory*, **11**, pp.71–79, 1987.
2. T. Horváth, J. Ramon, and S. Wrobel. Frequent subgraph mining in outerplanar graphs, *Data Mining and Knowledge Discovery* **21(3)**, pp.472–508, 2010.
3. T. Horváth and J. Ramon. Efficient Frequent Connected Subgraph Mining in Graphs of Bounded Tree-Width, *Theoretical Computer Science* **411(31–33)**, pp.2784–2797, 2010.
4. T. Miyahara, T. Shoudai, T. Uchida, K. Takahashi, and H. Ueda. Polynomial time matching algorithms for tree-like structured patterns in knowledge discovery, *Proc. PKDD 2000, LNAI* **1805**, pages 5–16, 2000.
5. T. Nagoya, S. Tani, and S. Toda. A Polynomial-Time Algorithm for Counting Graph Isomorphisms among Partial k-trees, *IEICE Trans. on Information and Systems* **J85-D1(5)**, pp.424–435, 2002, (in Japanese).
6. T. Yamada and T. Shoudai. Efficient Pattern Matching on Graph Patterns of Bounded Treewidth, *Electronic Notes in Discrete Mathematics*, to appear, 2011.
7. H. Yamasaki, Y. Sasaki, T. Shoudai, T. Uchida, and Y. Suzuki. Learning block-preserving graph patterns and its application to data mining, *Machine Learning* **76(1)**, pp.137–173, 2009.