Mining graphs to discover new theorems in mathematics

Christian Desrosiers DESROS@CS.UMN.EDU
University of Minnesota (DTC), 117 Pleasant Street SE, Minneapolis MN 55455 USA

Philippe Galinier PHILIPPE.GALINIER@POLYMTL.CA
Alain Hertz ALAIN.HERTZ@GERAD.CA
Ecole Polytechnique de Montreal, C.P. 6079 succ. Centre-ville, Montreal, QC H3C 3A7 CANADA

Pierre Hansen PIERRE.HANSEN@GERAD.CA
HEC Montreal, 3000, ch. Cote-Sainte-Catherine, Montreal, QC H3T 2A7 CANADA

Abstract

This paper introduces a new data mining problem of characterizing a class of graphs with a set of forbidden subgraphs. Efficient methods for this problem are presented, and the potential of these methods illustrated on the task of discovering novel and significant results in the field of graph theory.

One of the main problems of data mining is the extraction of patterns that characterize a set of graphs sharing a common property. A well-known example of this problem is the task of finding the subgraphs occurring frequently in a set of graphs (Nijssen & Kok, 2004; Yan & Han, 2002), which has several applications in the fields of bioinformatics, drug design, computer vision, and the Web. Instead of describing a set of related graphs using co-occurring subgraphs, it may be more appropriate to characterize these graphs with subgraphs that are not allowed to occur. These forbidden subgraphs can represent underlying constraints or inhibitors that block a given property from being expressed in a set graphs, and can be used to produce a compact description of these graphs.

The work presented in this abstract makes three contributions. First, it introduces a new data mining problem of characterizing a class of related graphs with a set of forbidden subgraphs, that has several potential applications. It also presents efficient methods to find such characterizations in an automated fashion, as well as necessary or sufficient conditions to have a characterization. Finally, it illustrates the potential of these methods in the discovery of novel and significant results in the field of graph theory.

Presented at *ILP-MLG-SRL*, Leuven, Belgium, 2009.

1. Forbidden subgraph characterization

Denote by $\mathcal{G}$ the space containing all graphs and let $\mathcal{H} \subset \mathcal{G}$ be a set of forbidden subgraphs. We say that a graph $G \in \mathcal{G}$ is $\mathcal{H}$-free if there is no graph of $\mathcal{H}$ isomorphic to one of its subgraphs, and write $\mathcal{G}_{\mathcal{H}}$ the set of $\mathcal{H}$-free graphs. Note that we do not impose restrictions on the type of subgraphs, i.e. they can be either node or edge induced, depending on the application. In all cases, we use the notation $G \subseteq H$ to denote that G is contained in H, i.e. is isomorphic to a subgraph of H.

Consider a class of graphs $\mathcal{C} \subset \mathcal{G}$ that we want to characterize. This class can either be described explicitly, or implicitly with a predicate $c : \mathcal{G} \rightarrow \{\text{true}, \text{false}\}$ such that $c(G) = \text{true}$ only if $G \in \mathcal{C}$. A *forbidden subgraph characterization* (FSC) of $\mathcal{C}$ is a set of graphs $\mathcal{H}$ such that $\mathcal{G}_{\mathcal{H}} = \mathcal{C}$. FSCs play a key role in graph theory, being at the center of many important results of that field (Chudnovsky et al., 2006), and allowing the development of efficient algorithms to recognize the graphs of a given class (Faudree et al., 1997).

For a class $\mathcal{C}$ to have an FSC, it is well-known that this class should be hereditary (Greenwell et al., 1973), i.e. if G is in $\mathcal{C}$ then so are all its subgraphs. However, such classes are rare and we might instead be interested in finding weaker relations that partially characterize graph classes. These relations come in two forms: *sufficient* conditions (SFSC) and *necessary* conditions (NFSC). Sufficient conditions can be expressed as: if a graph G is $\mathcal{H}$-free, then it belongs to $\mathcal{C}$. Therefore, an SFSC is such that $\mathcal{G}_{\mathcal{H}} \subseteq \mathcal{C}$. On the other hand, necessary conditions can be expressed as follows: if a graph G is in $\mathcal{C}$ then it is $\mathcal{H}$-free. This implies $\mathcal{G}_{\mathcal{H}} \supseteq \mathcal{C}$ for NSFCs.

1.1. Usefulness of characterizations

While a class $\mathcal{C}$ can have many SFSCs and NFSCs, these may not be equally interesting. To find useful characterizations, we define two partial orders that measure their *complexity* and *tightness*. Evaluating the *absolute* complexity of a characterization $\mathcal{H}$ can be subjective, e.g., a clique has a maximum number of edges but is rather simple to describe. Instead, we use a *relative* measure of complexity: $\mathcal{H}$ is simpler than $\mathcal{H}'$ if $\mathcal{H} \neq \mathcal{H}'$ and, for each $H \in \mathcal{H}$, there is a $H' \in \mathcal{H}'$ such that $H \subseteq H'$. On the other hand, the tightness of a characterization evaluates how well it describes $\mathcal{C}$, and is evaluated differently for SFSCs and NFSCs. Let $\mathcal{H}$, $\mathcal{H}'$ be two SFSCs of $\mathcal{C}$, we say that $\mathcal{H}$ is tighter than $\mathcal{H}'$ if $\mathcal{G}_{\mathcal{H}'} \subset \mathcal{G}_{\mathcal{H}}$. Likewise, we say that an NFSC $\mathcal{H}$ is tighter than another one $\mathcal{H}'$ if $\mathcal{G}_{\mathcal{H}} \subset \mathcal{G}_{\mathcal{H}'}$. In both cases, $\mathcal{H}$ is maximally tight if $\mathcal{G}_{\mathcal{H}} = \mathcal{C}$. The concepts of tightness and complexity are related: a tighter SFSC can generally be obtained by increasing its complexity.

1.2. Finding sufficient conditions

An SFSC of a class $\mathcal{C}$ can be written as G is $\mathcal{H}$-free $\Rightarrow G \in \mathcal{C}$. This is logically equivalent to $G \in \overline{\mathcal{C}} \Rightarrow \exists H \in \mathcal{H}$ s.t. $H \subseteq G$, where $\overline{\mathcal{C}} = \mathcal{G} \backslash \mathcal{C}$ is the complement of class $\mathcal{C}$. Thus, the problem of finding an SFSC with a single forbidden subgraph H can be formulated as finding a set $\mathcal{H} = \{H\}$ such that $\forall G \in \overline{\mathcal{C}},\ H \subseteq G$. The forbidden subgraph H is therefore a common subgraph of the graphs in $\overline{\mathcal{C}}$. Furthermore, suppose we have two SFSCs H, H' such that $H \subset H'$. It can be shown that $\mathcal{G}_{\{H\}} \subset \mathcal{G}_{\{H'\}} \subseteq \mathcal{C}$. As a consequence the SFSC that offers the tightest characterization of $\mathcal{C}$ is a common subgraph of $\overline{\mathcal{C}}$ which is maximal w.r.t. inclusion. This principle serves as the main idea of our first method, detailed in Algorithm 1. This algorithm computes a set $\mathcal{L}$ of representative graphs of $\overline{\mathcal{C}}$. At each iteration i, it finds a maximum common subgraph of the graphs in $\mathcal{L}$, i.e. a graph H in maxCS($\mathcal{L}$). It then tries to find a representative graph G of $\overline{\mathcal{C}}$ that does not contain H. If such a graph exists, it is added to $\mathcal{L}$ and the process is repeated. Otherwise, H is an SFSC of $\mathcal{C}$.

Algorithm 1 Single graph SFSC algorithm

```
Input: A graph class $\mathcal{C}$
Output: A single graph SFSC $\mathcal{H}$ of $\mathcal{C}$
Choose any $H_0$ in $\overline{\mathcal{C}}$.
Let $\mathcal{L}_0 \leftarrow \{H_0\}$, and $i \leftarrow 0$.
while $\exists G_i \in \overline{\mathcal{C}} \cap \mathcal{G}_{\{H_i\}}$ do
  Let $\mathcal{L}_{i+1} \leftarrow \mathcal{L}_i \cup \{G_i\}$.
  Choose $H_{i+1}$ in maxCS($\mathcal{L}_{i+1}$).
  Let $i \leftarrow i + 1$.
end while
return $\mathcal{H} = \{H_i\}$.
```

Note that it is always possible to find an SFSC for any given class of graph $\mathcal{C}$. For instance, letting H be an empty graph, we have $\mathcal{G}_{\{H\}} = \emptyset \subset \mathcal{C}$ and therefore H is a SFSC of $\mathcal{C}$. However, this SFSC may not be a tight description of $\mathcal{C}$. As mentioned before, it is possible to increase the tightness of this SFSC by increasing its complexity, i.e. allowing it to have more than one forbidden subgraph. To find SFSCs involving an arbitrary number K of forbidden subgraphs, we proceed as follows. We first use Procedure 2 to find the $K-1$ smallest graphs $\mathcal{M}$ of $\overline{\mathcal{C}}$. Then, we find a single graph SFSC H of $\mathcal{C} \cap \mathcal{G}_{\mathcal{M}}$. The K graph SFSC of $\mathcal{C}$ is simply $\mathcal{H} = \mathcal{M} \cup \{H\}$.

Procedure 2 findMin($\mathcal{C}$,L)

```
Input: A graph class $\mathcal{C}$ and an integer $L > 0$
Output: At most $L$ of the smallest graphs of $\overline{\mathcal{C}}$
Let $\mathcal{M}_0 \leftarrow \emptyset$ and $l \leftarrow 0$.
while $\overline{\mathcal{C}} \cap \mathcal{G}_{\mathcal{M}_l} \neq \emptyset$ and $l < L$ do
  Choose $H_l \in \overline{\mathcal{C}} \cap \mathcal{G}_{\mathcal{M}_l}$ with minimum order.
  Let $\mathcal{M}_{l+1} \leftarrow \mathcal{M}_l \cup \{H_l\}$.
  Let $l \leftarrow l + 1$.
end while
return $\mathcal{M} = \mathcal{M}_l$.
```

1.3. Finding necessary conditions

As opposed to finding SFSCs, the task of finding an NFSC of a class $\mathcal{C}$ corresponds to finding a graph H such that, for all $G \in \mathcal{C}$, $H \not\subseteq G$. Thus, H is a graph that is not a subgraph of any graph of $\mathcal{C}$. Moreover, let H and H' be two NFSCs such that $H \subset H'$. We can show that $\mathcal{C} \subseteq \mathcal{G}_{\{H\}} \subset \mathcal{G}_{\{H'\}}$. Therefore, the tightest NFSC is a graph H which is minimal w.r.t. inclusion. Our single graph NFSC method, detailed in Algorithm 3, is based on this idea. Let $\overline{\mathcal{G}}_{\mathcal{H}} = \mathcal{G} \setminus \mathcal{G}_{\mathcal{H}}$ be the set of graphs containing at least one graph of $\mathcal{H}$, and denote minNS($\mathcal{L}$) the set of minimal graphs that are not subgraphs of any graph of $\mathcal{L}$. Starting with an empty graph H, we find at each iteration a representative graph G of $\mathcal{C}$ that contains H. If no such graph exists, then H is an NFSC of $\mathcal{C}$. Otherwise, we add G to $\mathcal{L}$ and find a better forbidden subgraph H as one of the graphs of minNS($\mathcal{L}$). Since $\mathcal{C}$ may not have an NSFC, we have to limit the number of nodes of H to a value N, otherwise the algorithm may never terminate. In practice, N is selected as the highest integer for which the algorithm terminates in a reasonable amount of time.

As it was the case for sufficient conditions, we can sometimes improve the tightness of an NFSC by allowing it to have more than one subgraph. It can be shown that the NFSC $\mathcal{H}$ of a class $\mathcal{C}$ with maximum tightness is simply the minimal set containing all in-

Algorithm 3 Single graph NFSC algorithm

```
Input: A graph class $\mathcal{C}$ and an integer $N > 0$
Output: A single graph NFSC $\mathcal{H}$ of $\mathcal{C}$ having at most $N$ nodes
Let $H_0$ be the empty graph, $\mathcal{L}_0 \leftarrow \emptyset$, and $i \leftarrow 0$.
while $\exists G_i \in \mathcal{C} \cap \overline{\mathcal{G}}_{\{H_i\}}$ and $|V(H_i)| \leq N$ do
    Let $\mathcal{L}_{i+1} \leftarrow \mathcal{L}_i \cup \{G_i\}$.
    Choose $H_{i+1}$ in minNS($\mathcal{L}_{i+1}$).
    Let $i \leftarrow i + 1$
end while
return $\mathcal{H} = \{H_i\}$.
```

dividual graphs that are, by themselves, NFSCs. Although not presented in this paper, the algorithm to find NFSCs involving multiple forbidden subgraphs is based on this principle.

1.4. Link to Version Space

The tasks of finding SFSCs and NFSCs are related, in many respect, to the problem of learning a description of a class of patterns (Kramer et al., 2001). In this problem, we have a space $\mathcal{P}$ of patterns partially ordered under a relation of generality. For any two patterns X, Y of $\mathcal{P}$, we denote by $X \prec Y$ the fact that X is more general than Y (or equivalently, Y is more specific than X), and write the minimal and maximal patterns of a set $\mathcal{Q} \subset \mathcal{P}$ as $\min(\mathcal{Q})$ and $\max(\mathcal{Q})$. The problem consists in learning a description of the patterns, called *Version Space* (VS), consistent with a given set of constraints $c : \mathcal{P} \rightarrow \{\text{true}, \text{false}\}$ that should be either monotone or anti-monotone w.r.t. generality.

Denote by $\mathcal{Q}$ the set of patterns satisfying these constraints, the VS description of $\mathcal{Q}$ is given by the *generality* and *specificity* borders, i.e. $\mathcal{B}^+ = \max(\mathcal{Q})$ and $\mathcal{B}^- = \min(\mathcal{Q})$, such that

$$\mathcal{Q} \;=\; \{Y \in \mathcal{P} \mid \exists X \in \mathcal{B}^+, \exists Z \in \mathcal{B}^- \text{ s.t. } X \preceq Y \preceq Z\}.$$

The VS of the pattern class is typically obtained using a levelwise approach, where patterns of increasing specificity are generated and then tested against the constraints. See, e.g., (Kramer et al., 2001) for details.

In the context of characterizing a class $\mathcal{C}$ with SFSCs or NFSCs, the order of generality between two graphs G and H corresponds to the subgraph relation, i.e. $G \prec H \Leftrightarrow G \subset H$, and $\mathcal{C}$ is hereditary if the constraint $c(G) \equiv (G \in \mathcal{C})$ is anti-monotone. In this context, it can be showed that the unique tightest FSC of $\mathcal{C}$ is the generality border $\mathcal{B}^+$ of the graphs satisfying c, i.e. $\max(\overline{\mathcal{C}})$. Moreover, when such FSC does not exist, the set of the tightest single graph SFSCs corresponds to the specificity border $\mathcal{B}^-$ of the patterns G satisfying the anti-monotone constraint $\text{support}(G, \overline{\mathcal{C}}) = |\overline{\mathcal{C}}|$, i.e. G should be a subgraph of every graph of $\overline{\mathcal{C}}$. On the other hand, the set of the tightest single graph NFSCs is equivalent to the generality border $\mathcal{B}^+$ of the patterns H consistent with the constraint $\text{support}(H, \mathcal{C}) = 0$ imposing the support of H in $\mathcal{C}$, i.e. the number of graphs of $\mathcal{C}$ containing H, to be 0.

There are, however, important differences between our approach and VS. Thus, while VS builds a possibly infinite border, our approach focuses on finding a single tightest NSFC or SFSC by generating a limited number of representative graphs. Also, our approach avoids the need to define impracticable constraints such as $\text{support}(G, \overline{\mathcal{C}}) = |\overline{\mathcal{C}}|$, where $\overline{\mathcal{C}}$ is most likely infinite. Finally, while it is possible to define the VS patterns as sets of graphs, instead of single graphs, our approach provides a simple way to find tighter SFSCs that have more than one forbidden subgraph.

2. Automated conjecture generation

In recent years, mathematicians in the field of graph theory have turned to computers to find some very important results. A famous illustration of this is the proof to the four color conjecture, which was done in large part by computers (Robertson et al., 1997). Although automated methods have already been proposed for generating conjectures in graph theory, such as Graffiti (Fajtlowicz, 1988) and AutoGraphiX (Caporossi & Hansen, 2000), these methods focus on finding relations defined on graph attributes known as invariants, that hold for every graph in a given class $\mathcal{C}$. As addressed in (Hansen et al., 2005), automated methods for finding other types of results have not been explored. Following this observation, we use our methods to generate conjectures on FSCs.

2.1. Computational approach

A problem with Algorithms 1 and 3 is that they may need to explore an infinite set of graphs. To overcome this problem, we limit the search space to graphs having at most N nodes. As a consequence, our algorithms are no longer guaranteed to find SFSCs or NFSCs. Yet, we can still use these algorithms to generate conjectures based on the hypothesis that if these results are true for graphs of N or less nodes, they must be true for all graphs.

To find the representative graphs of $\overline{\mathcal{C}}$ and $\mathcal{C}$, or the graphs of maxCS($\mathcal{L}$) and minNS($\mathcal{L}$), we use a generate and test approach. In this approach, graphs are enumerated up to N nodes using an algorithm such as GenG (McKay, 1998) and then tested for membership in these classes. While enumerating graphs up to $N=10$ nodes can be done quite rapidly, testing membership usually requires to solve a complex problem for a large number of graphs. For instance, testing if a graph G is perfect requires to compute its chromatic number $\chi(G)$ and its clique number $\omega(G)$, two NP-hard problems.

Table 1 illustrates the time complexity of this approach for finding an FSC for the class of perfect graphs. For N between 7 and 10, this table give the the number of membership to $\overline{\mathcal{C}}$ and subgraph isomorphism tests, and the CPU time required to find an FSC of at most N nodes, using a 2.2 GHz Intel dual core processor with 2Gb of RAM.

N	CPU (sec)	$\overline{\mathcal{C}}$ tests	Subgraph iso.
7	0.1	1,302	43
8	0.8	14,899	1,754
9	19.9	289,567	52,044
10	1381.5	12,583,000	4,996,063

Table 1. Computational statistics for the class of *perfect* graphs.

2.2. Experimental results

We use our algorithms to find results related to the concepts of independence, domination and irredundance of graphs.

An *independent* set S of a graph G is a set of pairwise non-adjacent nodes of G. The *independent domination number* of G, denoted $i(G)$, is the minimum cardinality of a maximal independent set of G. Furthermore, a *dominating set* T is a set of nodes such that each node of $V(G)\setminus T$ is adjacent to at least one node of T. The *domination number* of G, written $\gamma(G)$, is the minimum cardinality of a dominating set of G. Moreover, let $X \subseteq V$, a node $x \in X$ is irredundant in X if it is isolated in X or if it has a private neighbor, i.e. a node $y \in V\setminus X$ such that x is the only node of X adjacent to y. The set X is irredundant if all its nodes are irredundant. We denote $ir(G)$ the minimum cardinality of a maximal irredundant set of G. It is well known in graph theory that the following relations hold for all graphs $G \in \mathcal{G}$ (Haynes et al., 1998): $ir(G) \leq \gamma(G) \leq i(G)$.

Figure 1 illustrates the single graph SFSC algorithm on the class of graphs G satisfying $\gamma(G)=i(G)$. The graph H_1 found by our algorithm correspond to the well-known SFSC originally presented in (Allan & Laskar, 1978). Our methods also led to the discovery of novel and significant results. For instance, Figure 2 describes a run of our multiple SFSC algorithm for the class of graphs G such that $ir(G)=\gamma(G)$, setting the desired number of forbidden subgraphs to $K=2$. The SFSC conjectured by the algorithm, i.e. $\mathcal{H}=\{M,H_3\}$, is a novel result that strengthens a previous SFSC proposed in (Favaron, 1986). Likewise, Figure 3 shows a run of our single graph NFSC algorithm for the class of graphs G that satisfy $ir(G) < \gamma(G)$, where a novel NFSC $\mathcal{H}=\{H_3\}$ was conjectured. Note that there results were discovered right-away by our methods, without having to adjust any parameter. The proofs of these results can be found in (Desrosiers et al., 2007).

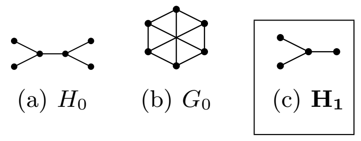

Figure 1. Single graph SFSC for the class of graphs G satisfying $\gamma(G) = i(G)$.

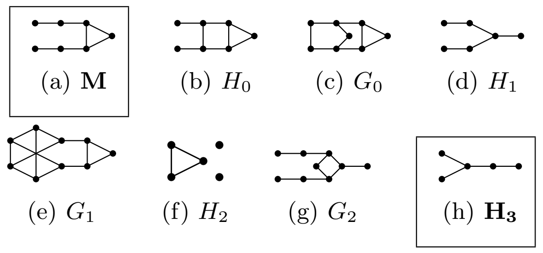

Figure 2. Two graph SFSC for the class of graphs G satisfying $ir(G) = \gamma(G)$.

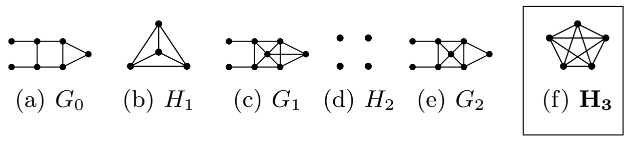

Figure 3. Single graph NFSC for the class of graphs G satisfying $ir(G) < \gamma(G)$.

3. Conclusion

To summarize, we have introduced a novel problem of describing a class of graphs using forbidden subgraphs. This problem, which contrasts the typical characterizations based on recurrent subgraphs, has several potential uses in data analysis, including the discovery of underlying constraints and the design of efficient classification methods. Moreover, we have presented several methods to find sufficient and necessary conditions to characterize a class $\mathcal{C}$, and have illustrated the potential of these methods in the discovery of novel and significant results in the field of graph theory.

References

Allan, R., & Laskar, R. (1978). On domination and independent domination numbers of a graph. *Discrete Mathematics*, *23*, 73–76.

Caporossi, G., & Hansen, P. (2000). Variable neighborhood for extremal graphs: 1-the system autographix. *Discrete Mathematics*, *212*, 29–44.

Chudnovsky, M., Robertson, N., Seymour, P., & Thomas, R. (2006). The strong perfect graph theorem. *Annals of Mathematics*, *164*, 51–229.

Desrosiers, C., Galinier, P., Hansen, P., & Hertz, A. (2007). *Automated generation of conjectures on forbidden subgraph characterization* (Technical Report G-2007-48). Les Cahiers du GERAD.

Fajtlowicz, S. (1988). On conjectures of Graffiti. *Discrete Mathematics*, *72*, 113–118.

Faudree, R., Flandrin, E., & Ryjacek, Z. (1997). Claw-free graphs – A survey. *Discrete Mathematics, Journal of graph theory*, *169*, 87–147.

Favaron, O. (1986). Stability, domination and irredundance in a graph. *Journal of Graph Theory*, *10*, 429–438.

Greenwell, D. L., Hemminger, R., & Klerlein, J. (1973). Forbidden subgraphs. *Proc. of the Fourth Southeastern Conf. on Combinatorics, Graph Theory and Computing* (pp. 389–394). Congressus Numerantium.

Hansen, P., Aouchiche, M., Caporossi, G., Mélot, H., & Stevanovic, D. (2005). What forms do interesting conjectures have in graph theory? *Graphs and Discovery, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 231–252.

Haynes, T. W., Hedetniemi, S., & Slater, P. (1998). *Fundamentals of domination in graphs.* Marcel Dekker.

Kramer, S., De Raedt, L., & Helma, C. (2001). Molecular feature mining in hiv data. *KDD '01: Proc. of the 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining* (pp. 136–143). New York, NY, USA: ACM.

McKay, B. (1998). Isomorph-free exhaustive generation. *Journal of Algorithms*, *26*, 306–324.

Nijssen, S., & Kok, J. N. (2004). The gaston tool for frequent subgraph mining. *Proc. of the Int. Workshop on Graph-Based Tools (Grabats 2004)* (pp. 281–285). Rome, Italy: Elsevier.

Robertson, N., Sanders, D., Seymour, P., & Thomas, R. (1997). The four-colour theorem. *Journal of Combinatorial Theory*, *70*, 2–44.

Yan, X., & Han, J. (2002). gSpan: Graph-based substructure pattern mining. *ICDM '02: Proc. of the 2002 IEEE Int. Conf. on Data Mining (ICDM'02)* (pp. 721–724). Washington, DC, USA: IEEE Computer Society.