Breaking Symmetries in Graph Representation

Michael Codish
Dept. of Computer Science
Ben-Gurion Univ. of the Negev
Beer-Sheva, Israel

Alice Miller and Patrick Prosser
School of Computing Science
University of Glasgow
Scotland

Peter J. Stuckey
Dept. of Comp. Sci. and SE
and NICTA Victoria Laboratory
University of Melbourne, Australia

Abstract

There are many complex combinatorial problems which involve searching for an undirected graph satisfying a certain property. These problems are often highly challenging because of the large number of isomorphic representations of a possible solution. In this paper we introduce novel, effective and compact, symmetry breaking constraints for undirected graph search. While incomplete, these prove highly beneficial in pruning the search for a graph. We illustrate the application of symmetry breaking in graph representation to resolve several open instances in extremal graph theory.

1 Introduction

The canonical graph representation problem is pertinent to a wide range of scientific applications. It is closely related to the graph isomorphism problem, as two graphs are isomorphic if and only if they have the same canonical representation. Examples of applications include data-mining [Washio and Motoda, 2003], mathematical chemistry [Faulon, 1998] and bio-informatics [Gardiner, 2011]. The two problems are poly-time equivalent, and are among the few that are known to be in NP but not known either to be solvable in polynomial time, nor to be NP-complete.

There are a variety of software tools devoted to solving the two problems “in practice”, one of which is nauty, due to McKay [1990]. Nauty is sometimes referred to as the world’s fastest isomorphism testing program. It is also able to produce a canonically-labeled isomorph of a graph to assist in isomorphism testing.

This paper is about constraint problems which involve the search for a graph that satisfies certain properties. For example, consider the problem to determine if there exists an undirected graph with 31 nodes, 81 edges, and which does not contain cycles of length 4 or less. This question arises in “extremal graph theory” [Bollobás, 1978], and its answer is unknown [Garnick et al., 1993]. The search space for problems of this type is enormous, and search may be optimized by restricting it to focus on canonical representations, or to avoid as often as possible isomorphic graphs. The general idea is to “break” symmetries in the search space. However, it is not clear how to apply this idea when searching for a graph.

In this type of problem the graph is a variable, so graph algorithms for canonical representation and isomorphism, as well as tools such as nauty, all of which operate on given graphs, do not apply. This paper provides a solution to this problem.

We assume a setting where testing for the existence of a graph G satisfying a property P is posed as a Boolean constraint \( P(A_G) \) on the variables of the Boolean adjacency matrix \( A_G \) of G. We follow the approach advocated by Crawford et al. [1996], where a predicate, \( \text{sb}(A_G) \), is introduced to break symmetries in the search space. In this way the satisfiability of \( P(A_G) \) is equivalent to that of \( P(A_G) \land \text{sb}(A_G) \).

Ideally, \( \text{sb}(A_G) \) is satisfied by a single member of each equivalence class of \( A_G \) under graph isomorphism, thus drastically restricting the search space for \( P(A_G) \land \text{sb}(A_G) \). However, this is not realistically possible as such a predicate also determines a canonical representation. In practice, it is sufficient that \( \text{sb}(A_G) \) is satisfied by at least one member of the equivalence class of \( A_G \) under isomorphism (typically by more than one) and in this case we say that \( \text{sb} \) is a symmetry breaking predicate. Shlyakhter [2007] notes that the difficulty is to identify a symmetry-breaking predicate which is both effective (rules out a large portion of the search space) and compact (so that checking the additional constraints does not slow down the search).

The presentations in [Crawford et al., 1996; Shlyakhter, 2007] consider symmetry breaking in terms of isomorphism, but focus on different structures such as acyclic digraphs, relations, permutations and functions. We introduce a novel, effective and compact predicate to break symmetries on graph representation. We consider simple graphs (undirected, with no multiple, nor self edges). We demonstrate the effectiveness of our approach through experimentation and resolve several open instances in extremal graph theory.

2 Graphs and their Canonical Representation

Throughout this paper we consider undirected simple graphs without loops or multiple edges. We focus on finite graphs and typically name the n nodes of a graph in the set \( \{1, \ldots, n\} \). We denote the Boolean values true and false by 1 and 0 respectively.

Definition 1 (Graph) A graph \( G = (V, E) \) has nodes \( V = \{1, \ldots, n\} \) and edges \( E \subseteq V \times V \) where \( (x, y) \in E \Rightarrow (y, x) \in E \). The Boolean adjacency matrix, \( A_G \) of \( G \), is
the $n \times n$ symmetric matrix where $A_G[x, y] \leftrightarrow (x, y) \in E$. The $i^{th}$ row of $A$ is denoted by $A[i]$, and $A[i, j]$ denotes the $j^{th}$ element of $A[i]$. The degree of node $u \in V$ is $\text{degree}(u) = \{ (u, v) \mid (u, v) \in E \}$. We denote the minimum and maximum degrees of the nodes in $G$ as $\delta(G)$ and $\Delta(G)$, or $\delta$ and $\Delta$ when the context is clear.

**Example 1** Figure 1 illustrates three graphs with corresponding adjacency matrices.

We use cycle notation to represent permutations. For example, the permutation $(1,2,6)(3,4)$ maps $1$ to $2$, $2$ to $6$, $6$ to $1$, and $3$ to $4$, to $5$, and $5$ to $6$.

**Definition 2 (permuting nodes)** Let $G = (V, E)$ be a graph with $n$ nodes, $A_G$ the adjacency matrix for $G$, and $\pi$ a permutation on $\{1, \ldots, n\}$. Then $\pi(G)$ is the graph represented by permuting the nodes of $G$ using $\pi$. Formally, $\pi(G) = (V, E')$ where $E' = \{ (\pi(x), \pi(y)) \mid (x, y) \in E \}$ and $\pi(A_G)$ is the adjacency matrix of $\pi(G)$.

**Definition 3 (graph isomorphism)** $G$ and $G'$ are isomorphic if there exists a permutation $\pi$ such that $A_G = A_{\pi(G)}$.

**Example 2** The graphs in Figure 1 are isomorphic. We can permute $G_1$ to $G_2$ using $\pi_1 = (2, 8, 5, 9, 4, 7, 3)$ and $G_1$ to $G_3$ using $\pi_2 = (2, 9, 4, 8, 6, 7, 3)$.

**Definition 4 (sequences, lexicographic order)** Let $A$ be matrix and $A[i][j]$ the concatenation of rows $i$ and $j$ (viewed as sequences). The length of a sequence $s$ is denoted $|s|$. We use $\preceq$ to denote the usual lexicographic order on sequences. We extend this notation in the obvious way: for matrices, with $n$ and $m$ rows respectively, $A \preceq B$, if and only if $A[1][a] \cdots A[n] \preceq B[1][b] \cdots B[m]$; and for graphs, $G \preceq G'$ if and only if $A_G \preceq A_{G'}$.

One way to define a canonical representation of a graph is to take the smallest graph (i.e. in the lexicographic order) which is isomorphic to $G$ [Read, 1978]. This is the definition which we adopt throughout the paper.

**Definition 5 (canonical form of a graph)** The canonical form of a graph is the graph with $\text{can}(G) = \min_{\pi \in \Pi} \{ \pi(G) \mid \pi$ is a permutation$\}$. We say that $G$ is canonical if $G = \text{can}(G)$.

**Example 3** Consider the graphs of Figure 1. The graph $G_3$ is the canonical representation of $G_1, G_2$ and $G_3$.

Note that the canonical representation of a graph does not necessarily order the nodes by degree. In Figure 1, the nodes of $G_2$ are ordered by degree: nodes $\{1, 2, 3, 4\}$ are of degree 2, nodes $\{5, 6, 7, 8\}$ are of degree 3 and node 9 is degree 4. But this is not the case for the canonical form, $G_3$.

3 **Symmetry breaking on Representation**

We first consider a symmetry breaking predicate, introduced without proof in [Miller and Prosser, 2012], which constrains the rows of the adjacency matrix to be sorted lexicographically in non-decreasing order.

**Definition 6 (lexicographic symmetry break)** Let $A$ be an $n \times n$ adjacency matrix. We define $\text{sb}_i(A) = A[i] \preceq A[i+1]

Observe the graphs in Figure 1. We have $\text{sb}_2(A_G_1) = \text{false}$, $\text{sb}_2(A_G_2) = \text{false}$, and $\text{sb}_2(A_G_3) = \text{true}$.

Definition 6 is more subtle than might first appear. It defines a symmetry breaking predicate only because for every adjacency matrix $A$, $\text{sb}_i(A')$ is true for at least one of the matrices $A'$ isomorphic to $A$. No such proof is provided in [Miller and Prosser, 2012]. In fact, were we to reverse the order, taking $A[i] \succeq A[i+1]$ instead, it would not define a symmetry breaking constraint. Consider for example any representation of the graph $G$ with 2 nodes and a single edge. Then $A_G[1] \not\preceq A_G[2]$. The subtlety arises because, in contrast to the case of breaking symmetries in matrix problems where rows and columns can be reordered, such as in [Gent et al., 2002; Flener et al., 2002; Frisch et al., 2003], here we need to reorder rows and columns both in the same way. To prove the correctness of Definition 6 it is sufficient to show that $\text{sb}_i(\text{can}(A))$ holds.

**Theorem 1** Let $G$ be a graph. Then $\text{sb}_i(\text{can}(A_G))$ holds.

**Proof:** Let $A$ be canonical and assume to the contrary that $A$ does not satisfy $\text{sb}_i(A)$. Let $i$ be such that $A[i] \not\preceq A[i+1]$. It follows that there is a $j$ such that for every $1 \leq j' < j$, $A[i, j'] = A[i+1, j']$ and $A[i, j] = 1$ and $A[i+1, j] = 0$. Let $A'$ be the matrix obtained by swapping rows $i, i+1$ as well as columns $i, i+1$. We show that $A' \prec A$ in contradiction to $A$ being canonical. Considering that $A[i, j] = 1$, so $i \neq j$ and there are two cases. We detail the case for $i < j$. The other case is similar.

If $i < j$, note that because the $j - 1$ length prefixes of $A[i]$ and $A[i+1]$ are equal, hence $A[i, j - 1] = A[i, j - 1] = A'[i, j - 1]$. Note also that $A'[i', i'] = 0$ for $i' < i$ (the $i$ and $i+1$ elements in $A[i']$ are equal because $A$ is symmetric and $A[i, i'] = A[i+1, i']$). It follows that the first cell to differ in $A$ and $A'$ is $A[i, j] = 1$ and $A'[i, j] = 0$. So $A' \prec A$. Contradiction.

We now proceed to strengthen this notion of symmetry breaking. The following example illustrates a symmetry not captured by $\text{sb}_i(A)$.

**Example 4** Consider the adjacency matrix $A_1$ depicted in Figure 2 for which $\text{sb}_1(A_1) = \text{true}$ as the rows are ordered lexicographically. Observe that $A[2] \preceq A[3]$ independent of whether we swap the nodes (rows and columns) 2 and 3, or not. Adjacency matrix $A_2$ depicted in Figure 2 is the result of this swap and it too satisfies $\text{sb}_1(A_2) = \text{true}$. However, it is “closer” to canonical as $A_2 \preceq A_1$. Indeed $A_2$ is the canonical representative of this graph. Figure 2 highlights that the first 3 elements of rows 2 and 3 are invariant under node swap.

In view of Example 4 we introduce the following definition and then introduce a stronger symmetry breaking constraint.

**Definition 7 (extended lexicographic order)** Let $s$ be a sequence and $I \subseteq \{1, \ldots, |s|\}$. We denote by $\langle s \mid I \rangle$ the sequence obtained from $s$ by simultaneously omitting the elements at positions $I$. For a set of natural numbers $I$ we denote by $\preceq_I$ the order on sequences of length at least $\max(I)$ defined by: $s_1 \preceq_I s_2 \iff (s_1 \mid I) \preceq (s_2 \mid I)$.
Definition 8 (improved lexicographic symmetry break)
Let \( A \) be an \( n \times n \) adjacency matrix. We define

\[
\text{sb}_k^*(A) = \bigwedge_{i < j} A[i] \preceq_{(i,j)} A[j]
\]

Observe that Definition 8 introduces \( O(n^2) \) constraints on lexicographic order whereas Definition 6 introduces only \( O(n) \). This is needed because we lack a “transitivity” like property stating that if \( s_1 \preceq_{(i,j)} s_2 \) and \( s_2 \preceq_{(j,k)} s_3 \) then also \( s_1 \preceq_{(i,k)} s_3 \). But this does not hold as illustrated by the following example.


Interesting, transitivity does hold for rows two apart.

Theorem 2 \( A[i] \preceq_{(i,i+1)} A[i+1] \wedge A[i+1] \preceq_{(i+1,i+2)} A[i+2] \Rightarrow A[i] \preceq_{(i,i+2)} A[i+2] \)

Proof: Assume the premise and adopt the following representation where the boxed elements are at positions \( i \), \( i+1 \) and \( i+2 \).

\[
\begin{align*}
A[i] &= S_1 \begin{bmatrix} 0 & x & y \\ 0 & 0 & 0 \end{bmatrix} T_1 \\
A[i+1] &= S_2 \begin{bmatrix} 0 & x & z \\ 0 & 0 & 0 \end{bmatrix} T_2 \\
A[i+2] &= S_3 \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} T_3
\end{align*}
\]

From the premise by definition of \( \preceq_{(i,j)} \), we have \( S_1 y T_1 \preceq_{(1,2)} S_2 x T_2 \) and \( S_2 x T_2 \preceq_{(2,3)} S_3 y T_3 \). We prove that \( S_1 x T_1 \preceq_{(1,3)} S_3 y T_3 \) which gives the result. There are two cases: either (a) \( S_1 \prec S_3 \) and since \( S_2 \preceq S_3 \) we have \( S_1 \prec S_3 \) and the result holds; or (b) if \( S_1 = S_3 \), then either \( S_2 < S_3 \), and the result follows, or \( S_2 = S_3 \), and it remains to show that \( x T_1 \preceq_{(1,3)} x T_3 \). So we have \( S_1 = S_2 = S_3 \). Suppose \( y \prec z \) then since \( y \preceq y \) we have that \( x \prec z \) and the result holds. Otherwise, \( y = z \). Suppose \( x \prec y \) then clearly \( x \prec z \) and the result holds. So assume \( x = y = z \). Then we have that \( T_1 \preceq T_2 \) and \( T_2 \preceq T_3 \) and hence the result holds.

So we can refine Definition 8.

Corollary 1

\[
\text{sb}_k^*(A) = \bigwedge_{i < j; j \not= z} A[i] \preceq_{(i,j)} A[j]
\]

The following proves that \( \text{sb}_k^* \) is a symmetry-breaking predicate.

Theorem 3 Let \( A \) be a canonical adjacency matrix. Then \( \text{sb}_k^*(A) \) holds.

Proof: Let \( A \) be the canonical adjacency matrix for a graph \( G \) and assume to the contrary that \( A \) does not satisfy \( \text{sb}_k^*(A) \).

That is, there exist \( i \) and \( j \) such that \( i < j \) and \( A[i] \not\preceq_{(i,j)} A[j] \). Let \( \pi \) denote the permutation which swaps nodes \( i \) and \( j \) in \( G \). We show that \( B = A_{\pi(G)} \prec A \).

We denote the \( i \) and \( j \) rows, \( A[i] = S_1(0)S_2(x)S_3 \) and \( A[j] = T_1(x)T_2(0)T_3 \) such that the circled 0 and \( x \) are at positions \( i \) and \( j \) in \( A[i] \) and at positions \( j \) and \( i \) in \( A[j] \). They are circled to correspond to Figure 4(a). Since \( A[i] \not\preceq_{(i,j)} A[j] \), hence \( S = S_1S_2S_3 \) \( \succ T_1T_2T_3 = T \). Since \( S \succ T \), hence \( S \) and \( T \) must have prefixes of the form \( W1 \) and \( W0 \), respectively. Namely, \( W \) is a common prefix followed by a 1 in \( S \) and by a 0 in \( T \). Let \( k \) be the column where this first
### Definition 9 (ordered partition)
Let $G$ be a graph. Then $P = \{P_1, \ldots, P_p\}$ is an ordered partition of the nodes of $G$ if $1 \leq i < j \leq p$ then $v_i \in P_i \wedge v_j \in P_j \Rightarrow v_i < v_j$.

### Definition 10 (partition preserving permutation)
Let $P = \{P_1, \ldots, P_p\}$ be an ordered partition on the nodes of $G$. A permutation $\pi$ on the nodes of $G$ is partition preserving for $P$ if $1 \leq i < j \leq p, \forall v_i \in P_i, \pi(v_i) \in P_{\pi(i)}$.

### Example 6
Consider the graph $G_2$ from Figure 1 and the ordered partition $P = \{\{1, 2, 3, 4\}, \{5, 6, 7, 8\}, \{9\}\}$, which partitions vertices by degree. Then the permutation $\pi = (2, 3, 4)$ is partition preserving for $P$. It maps elements in $P_1$ to other elements in $P_1$ and elsewhere is the identity.

### Definition 11 (canonical partitioned adjacency matrix)
The canonical form of a graph $G$ with respect to an ordered partition $P$ is the graph $can(G, P) = min_{\pi} (\pi(G))$ where $\pi$ is a partition preserving permutation for $P$. We say that $G$ is canonical for $P$ if $G = can(G, P)$.

We can define a symmetry breaking predicate for partitioned graphs as follows:

### Definition 12 (partitioned lexicographic symmetry break)
Let $A$ be an $n \times n$ adjacency matrix and $P = \{P_1, P_2, \ldots, P_p\}$ be an ordered partition. We define

$$\text{sb}_p^\pi(A, P) = \bigwedge_{k=1}^p \bigwedge_{i,j \subseteq P_k, i < j} A[i] \preceq_{\{i,j\}} A[j]$$

### Theorem 4
Let $G$ be a canonical partitioned graph for an ordered partition $P$. Then $\text{sb}_p^\pi(A_G, P)$ holds.

**Proof:** Let $A$ be the canonical adjacency matrix for graph $G$ and assume to the contrary that $A$ does not satisfy $\text{sb}_p^\pi(A, P)$. That is, there exists a partition $P_k$ and $\{i, j\} \subseteq P_k$ with $i < j$ where $A[i] \preceq_{\{i,j\}} A[j]$. We show that $B = A_{\pi(G)} \preceq A$ where $\pi$ swaps $i$ and $j$. Note that $\pi$ is a partition preserving permutation for $P$. The remainder of the proof is essentially identical to that of Theorem 3. 

### 4 Extremal Graph Problems
We apply a constraint-based approach to extremal graph problems and illustrate the advantage of symmetry breaking on the graph representation.

The girth of a graph is the size of the smallest cycle contained in it. Let $\mathcal{G}_k(v)$ denote the set of graphs with $v$ vertices and girth at least $k + 1$. Let $f_k(v)$ denote the maximum number of edges in a graph in $\mathcal{G}_k(v)$. A graph in $\mathcal{G}_k(v)$ with $f_k(v)$ edges is called extremal. The number of non-isomorphic extremal graphs in $\mathcal{G}_k(v)$ is denoted $F_k(v)$. Extremal graph problems are about discovering values of $f_k(v)$ and $F_k(v)$ and about finding witnesses. In [Abajo and Diánez, 2010] the
\begin{align*}
\forall 1 \leq i < j \leq v. \ (A[i,j] \equiv A[j,i] \text{ and } A[i,i] \equiv \text{false}) \tag{1} \\
\forall i,j,k. \ A[i,j] + A[j,k] + A[k,i] < 3 \tag{2} \\
\sum_{1 \leq i < j \leq v} A[i,j] = e \tag{4} \\
\forall 1 \leq i \leq v. \left( \delta \leq \sum_{1 \leq j \leq v} A[i,j] \leq \Delta, \ \min_{1 \leq j \leq v} (\sum_{1 \leq j \leq v} A[i,j]) = \delta \right) \tag{5}
\end{align*}

Figure 5: Basic constraint model for extremal graph problems (no cycles of length 4 or less)

authors attribute the discovery of values \( f_4(v) \) for \( v \leq 24 \) to [Garnick et al., 1993] and for \( 25 \leq v \leq 30 \) to [Garnick and Nieuwjaar, 1992]. In [Garnick et al., 1993] the authors report values of \( F_4(v) \) for \( v \leq 21 \). In [Garnick et al., 1993] and [Wang et al., 2001] the authors apply algorithms to compute lower bounds on \( f_4(v) \) for \( 31 \leq v \leq 200 \). Some of these lower bounds are improved in [Abajo et al., 2010]. Values of \( f_4(v) \) for \( v \leq 30 \), and of \( F_4(v) \) for \( v \leq 21 \) are available as sequences A006856 and A159847 of the On-Line Encyclopedia of Integer Sequences [OEIS, 2010].

Our basic constraint model is depicted in Figure 5 where we assume that \( A \) is a Boolean \( v \times v \) matrix. Constraint (1) states that the graph is simple (symmetric with no self loops), Constraints (2) and (3) express that there are no cycles of length 3 and 4, and Constraint (4), that the number of edges is \( e \). Constraints (2) and (3) are implemented more efficiently. We introduce additional Boolean variables for each triplet of (distinct) vertices \( i,j,k \) with \( i < k \): \( x_{i,j,k} \equiv A[i,j] \land A[j,k] \) represents a length 2 path between \( i \) and \( k \) via \( j \); and \( x_{i,k} \equiv \forall i,j,k \ (j \neq i, j \neq k) \) represents the existence of any length 2 path between \( i \) and \( k \). We then express Constraints (2) and (3) as \( \forall i,j,k. \ A[i,j] + A[j,k] < 2 \) and \( \forall i,k. \ \sum_j x_{i,j,k} < 2 \).

To explain Constraint (5) we recall Propositions 2.6 and 2.7 from [Garnick et al., 1993] which state that for every graph in \( F_4(v) \) with \( e \) edges the minimum and maximum vertex degrees, denoted \( \delta \) and \( \Delta \), satisfy the following equations (assuming \( v \geq 1 \)):

\[
v \geq 1 + \Delta \delta \geq 1 + \delta^2, \quad \text{and} \quad \delta \geq e - f_4(v-1), \quad \text{and} \quad \Delta \geq \left[\frac{2e}{v}\right]
\]

Given values for \( v \) and \( e \) we model the problem separately for each potential pair \((\delta, \Delta)\) introducing Constraint (5). In addition to the above constraints we introduce symmetry breaking constraints \( \mathbf{sb}_\delta \) or \( \mathbf{sb}_\delta^T \).

Example 7 For \( v = 31 \) and \( e = 80 \) the possible \((\delta, \Delta)\) pairs satisfying Equation (\textasteriskcentered\textasteriskcentered) are \{(4, 6), (4, 7), (5, 6)\}. Similarly, for \( v = 31 \) and \( e = 81 \) the single pair is \((5, 6)\).

We describe three experiments to evaluate the impact of different symmetry breaking strategies. Experiments were run using two different constraint solvers, BEE [Metodi and Codish, 2012] and Choco [Laburthe and Jussien, ]. We present the results obtained using BEE which compiles finite domain constraints to CNF and solves them using an underlying SAT solver. Our configuration uses CryptoMiniSat v2.5.1 [Soos, 2010]. BEE performs CNF simplification by applying a constraint-driven technique called equi-propagation [Metodi et al., 2011] and partial evaluation. All experiments are performed on a single core of an Intel(R) Core(TM) i5-2400 3.10GHz CPU with 4GB memory under Linux (Ubuntu lucid, kernel 2.6.32-24-generic). BEE is written in Prolog and run using SWI Prolog v6.0.2 64-bits. All experiments were replicated and verified using the Choco constraint programming toolkit.

4.1 Experiment 1: computing \( f_4(v) \)

Table 1 summarizes the results for a constraint-based approach to compute values of \( f_4(v) \). We compare the computation time for three configurations specified as columns for: no symmetry, and breaking symmetries using \( \mathbf{sb}_\delta \) and \( \mathbf{sb}_\delta^T \).

For smaller instances, \( v \leq 15 \), we apply the constraint model from Figure 5. For larger instances, \( 16 \leq v \leq 24 \), we add an additional constraint to the model. To this end we follow [Garnick et al., 1993] where it is noted that every graph in \( F_4(v) \) with at least 5 vertices contains a \((\Delta, \delta-1)\)-star. In general, an \((m, n)\)-star is a rooted tree, denoted \( S_{m,n} \), with \( m \) children, each of which has \( n \geq 1 \) children, all of which are leaves. So, we add constraints to explicitly embed \( S_{\delta,\delta-1} \) in the adjacency matrix. In this setting, based on Theorem 4 we impose symmetry breaking on the \( m \) clusters of \( n \) leaves of \( S_{m,n} \) as well as to the cluster of nodes not in \( S_{m,n} \). Figure 6(a) illustrates the star \( S_{6,4} \). A \( 31 \times 31 \) adjacency matrix with an embedded \( S_{6,4} \) is depicted as Figure 6(b). Black and
white cells indicate values 1 and 0 respectively, and gray cells indicate unassigned Boolean variables. The last row of the matrix corresponds to the root. Then moving up we find the 6 children of the root, and then its 24 grandchildren. We will later return to explain Figure 6(c). Finally, in the third part of the table, we consider two open instances. Here we also consider a model with the constraints that embed the star.

For each value of \( v \) and each type of symmetry break, we search for a graph with \( f_4(v) \) edges (columns \( e = f_4(v) \)), and show the non-existence of a graph with \( f_4(v) + 1 \) edges (columns \( e = f_4(v) + 1 \)). Examining the three \( e = f_4(v) \) columns, it appears that there is no significant gain in symmetry breaking when the instance is satisfiable and we need only find a single witness. When instances are unsatisfiable (the three \( e = f_4(v) + 1 \) columns) we encounter two types of instances: those which involve search and those which do not. For the later type, unsatisfiability derives from the propagation of the constraints in Equation (\(*\)) and the computation is fast for all three configurations. For the other instances, the solver must explore the entire search space and symmetry breaking is then useful.

The bottom two rows in Table 1 describe our results for two open instances, computing \( f_4(31) \) and \( f_4(32) \). A lower bound of \( f_4(31) \geq 80 \) is given in [Garnick et al., 1993] and a witness (discovered using our model in less than 2 seconds) is depicted as Figure 6(c). It is canonical with respect to a partitioning where the first 24 rows form 6 clusters of size 4 each (the grandchildren), the next 6 rows form a cluster (the children), and the last row is a singleton cluster (the root). With the proof that there is no witness with 81 edges (determined using our model in 40 minutes of CPU time) we conclude that \( f_4(31) = 80 \). Given that \( f_4(31) = 80 \), Equation (\(*\)) implies that \( f_4(32) \leq 85 \) and hence that the lower bound \( f_4(32) \geq 85 \) reported in [Garnick et al., 1993] is the precise value, consequently \( f_4(32) = 85 \). These are both new results.

### 4.2 Experiment 2: computing \( F_4(v) \)

In this experiment we apply a constraint-based approach to compute the number of non-isomorphic extremal graphs with \( v \) vertices. We apply a constraint solver to generate all graphs satisfying the constraint model for \( v \) vertices and \( e = f_4(v) \) edges with corresponding symmetry breaking constraints. We then apply \texttt{nauty} to determine the number of non-isomorphic graphs within this set. The time required to run \texttt{nauty} is negligible and not detailed in our results.

For smaller values, \( v \leq 15 \), we consider the constraint model of Figure 5. Table 2 shows for each value of \( v \) the maximum number of edges \( f_4(v) \), the number of non-isomorphs \( F_4(v) \), and the number of graphs generated (columns \# sols) and computation time (time), for each of the three configurations. Our results are as expected: improving symmetry breaking makes a significant difference. The bottom two rows in Table 2 describe our results for two open instances, computing \( F_4(24) \) and \( F_4(32) \). To obtain these results we first extend, in the following lemma, the observation of Garnik regarding the embedding of a star for the two cases. We then explicitly embed the extended structures in the encoding.

**Proposition 1 (a)** Every extremal graph \( G \) with 24 nodes contains a star \( S_{5,5} \) in which the 5 children of the root have degrees 5, 5, 5, 5, 5 in \( G \). (b) Every extremal graph \( G \) with 32 nodes contains a star \( S_{6,4} \) in which the 6 children of the root have degrees 6, 5, 5, 5, 5, 5 in \( G \).

### 4.3 Experiment 3: computing \( F_5(v) \)

For our final experiment we consider the extremal graphs which contain no cycles of length 5 or less. To this end we extend the basic constraint model of Figure 5 with an additional constraint that states that every sequence of five vertices does not form a cycle, and we consider the optimization problem which computes values of \( f_5(v) \). Table 3 shows our results. To better illustrate the impact of improved lexicographic symmetry breaking we consider also a predicate \( \text{sb}_7^* \) which is like \( \text{sb}_7 \) but only compares consecutive rows of the matrix. It is clear that the much larger \( \text{sb}_7^* \) pays off, reducing computation time considerably for the larger instances.

### 5 Conclusion

Symmetry breaking for graph representations using \( \text{sb}_7 \) is considered also in [Miller and Prosser, 2012] where it is shown to have a considerable impact. However, no formal justification is provided. We address in general terms the application of symmetry breaking to improve the search for an undirected graph satisfying a given property. We formally justify the use of \( \text{sb}_7 \). We also introduce and formally justify the use of \( \text{sb}_7^* \) which is a more powerful symmetry breaking predicate. We demonstrate the impact of symmetry breaking to extremal graph theory where we also close several open instances.
References


