Complete Symmetry Breaking Constraints for the Class of Uniquely Hamiltonian Graphs*

Avraham Itzhakov and Michael Codish

Department of Computer Science
Ben-Gurion University of the Negev
Beer-Sheva, Israel
{itzhakoa,mcodish}@cs.bgu.ac.il

Abstract. This paper introduces, for the first time, a complete symmetry breaking constraint of polynomial size for a significant class of graphs: the class of uniquely Hamiltonian graphs. We introduce a canonical form for uniquely Hamiltonian graphs and prove that testing whether a given uniquely Hamiltonian graph is canonical can be performed efficiently. Based on this canonicity test, we construct a complete symmetry breaking constraint of polynomial size which is satisfied only by uniquely Hamiltonian graphs which are canonical. We apply the proposed symmetry breaking constraint to show new results regarding the class of uniquely Hamiltonian graphs. Given that it is unknown if there exist polynomial sized complete symmetry breaking constraints for graphs, this paper makes a first step in the direction of identifying specific classes of graphs for which such constraints do exist.

Keywords: Symmetry breaking · Uniquely Hamiltonian graphs · Isomorph-free graph generation

1 Introduction

Graph search problems are fundamental in graph theory. Such problems include: existence problems, where the goal is to determine whether a simple graph with certain graph properties exists, enumeration problems, which are about finding all solutions modulo graph isomorphism, and extremal problems, where we seek the smallest/largest solution with respect to some target such as the number of edges or vertices in a solution. Solving graph search problems is typically hard due to the enormous search space and the large number of symmetries. For graph search problems, any graph obtained by permuting the vertices of a (non-)solution is also a (non-)solution, which is isomorphic, or "symmetric". Hence, any permutation of the vertices of a graph is a symmetry, and each isomorphism class of graphs consists of either equivalent solutions, or equivalent non-solutions.

The presence of symmetries often causes redundant search effort by revisiting symmetric objects. To optimize the search we aim to restrict it to focus on one graph from each equivalence class. The focus on symmetry elimination has facilitated the solution

* Supported by the Israel Science Foundation, grant 625/17.

of many open instances of combinatorial search problems and graph search problems in particular. For example, the proof that the Ramsey number $R(3,3,4)$ is equal to 30 [5], the solution for the Sudoku minimum number of clues problem [16], and the enumeration of all non-word representable graphs of order 12 [15]. Nevertheless, there is to date no efficient way to avoid all symmetries when searching for a general graph, regardless of the search method.

Constraint programming is a powerful paradigm to solve combinatorial problems. The essence of constraint programming is to use constraints to prune the search space efficiently during search. A constraint based approach to solve graph search problems is to represent an unknown graph as a set of Boolean variables which define its adjacency matrix, and to model the desired graph properties as constraints on these variables. A constraint solver is then applied to find a solution graph which satisfies these constraints.

One common approach to break symmetries in constraint programming is to add symmetry breaking constraints [8, 24, 25] which are satisfied by at least one member of each isomorphism class. A symmetry breaking constraint is called *complete* if it is satisfied by exactly one member of each isomorphism class and *partial* otherwise. A universal measure for the size of a symmetry breaking constraint is the size of its representation in propositional logic. All known techniques to define complete symmetry breaking constraints for graph search problems are based on predicates which are exponential in size. There is no known polynomial size complete symmetry breaking constraint for graph search problems. For this reason, in practice, one often applies partial symmetry breaking constraints [6, 7] which are polynomial in size or, one seeks compact representations for complete symmetry breaking constraints on small graphs [4, 13–15].

In this paper we identify a particular class of graphs for which all symmetries can be broken efficiently. We introduce the notion of symmetry breaking constraints for a particular class of graphs. Given a class C of graphs, a complete (partial) symmetry breaking constraint for that class is satisfied by exactly (at least) one member of each isomorphism class of graphs from C. We focus on the class of *uniquely Hamiltonian* graphs. This is the class of graphs that contain exactly one Hamiltonian cycle. The research community has shown interest in graphs of this type [2, 9, 11, 23] and there remain unresolved questions regarding them.

This paper makes the following contributions. First, we introduce a canonical form for uniquely Hamiltonian graphs which can be tested in polynomial time. Second, we show that the automorphism group and the canonical form of a given uniquely Hamiltonian graph can be computed efficiently if its Hamiltonian cycle is known. Third, using these results, we introduce a complete symmetry breaking constraint of polynomial size for the class of uniquely Hamiltonian graphs. Finally, we use this symmetry breaking constraint and apply a constraint programming approach to show new results regarding uniquely Hamiltonian graphs. We generate all order $n \leq 18$ uniquely Hamiltonian graphs of minimum degree 3, and all order $n \leq 20$ uniquely Hamiltonian graphs of minimum degree 3 and girth at least 4. We determine the, previously unknown, smallest orders for which uniquely Hamiltonian graphs of minimum degree 3 and girths 3 and 4 exist. The contributions of this paper provide a first step to answer a broader

question: are there significant classes of graphs for which complete symmetry breaking constraints of polynomial size exist?

The rest of this paper is structured as follows. Section 2 presents preliminary definitions and notation. Section 3 introduces a canonical form for uniquely Hamiltonian graphs. We show an efficient canonicity test which forms the basis for the construction of a complete symmetry breaking constraint for uniquely Hamiltonian graphs. Section 4 introduces a complete symmetry breaking constraint of polynomial size for the class of uniquely Hamiltonian graphs. Section 5 describes a constraint programming approach which applies the proposed symmetry breaking constraint to generate non-isomorphic uniquely Hamiltonian graphs. This approach is then refined and shown to provide solutions to previously unanswered questions regarding uniquely Hamiltonian graphs. Section 6 describes the current state-of-the-art for generation of uniquely Hamiltonian graphs and compares it to our approach. Based on the results achieved in this paper, we suggest possible improvements to the state-of-the-art approach. Finally, Section 7 concludes.

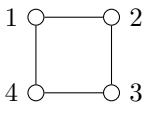
2 Preliminaries

Throughout this paper we consider simple graphs, i.e. undirected graphs with no self loops. The vertex set of a graph G of order n, is denoted $V(G)$ and assumed to be $V = \{1, \ldots, n\}$, its edge set is denoted $E(G)$, and in abuse of notation its adjacency matrix representation is also denoted G. The set of simple graphs on n vertices is denoted $\mathcal{G}_n$. For two graphs H, G, we say that H is a subgraph of G, denoted $H \subseteq G$ if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. The *cycle graph* C_n is the graph with vertices $V = \{1, \ldots, n\}$ ($n \geq 3$) and edges $E = \{\{1,2\}, \{2,3\}, \ldots, \{n-1, n\}, \{n, 1\}\}$. A *Hamiltonian cycle* in a graph $G \in \mathcal{G}_n$ is a permutation $h = [v_1, \ldots, v_n]$ of the vertices such that for every pair $1 \leq i < n$, $\{v_i, v_{i+1}\} \in E(G)$ and also $\{v_n, v_1\} \in E(G)$. Two Hamiltonian cycles h_1, h_2 are considered to be the same if and only if their corresponding edge sets are equal. A graph is called uniquely Hamiltonian if it contains exactly one Hamiltonian cycle. The set of uniquely Hamiltonian graphs of order n is denoted $\mathcal{UH}_n$. An *unknown graph* of order n is represented as an $n \times n$ adjacency matrix of Boolean variables which is symmetric and has the values $false$ (denoted by 0) on the diagonal. We denote by $G_{i,j}$ the element at row i and column j of the adjacency matrix of a (unknown) graph G.

The group of all permutations on $\{1 \ldots n\}$ is denoted S_n. We represent a permutation $\pi \in S_n$ as a sequence of length n where the ith element indicates the value of $\pi(i)$. For example: the permutation $[2, 3, 1] \in S_3$ maps as follows: $\{1 \mapsto 2, 2 \mapsto 3, 3 \mapsto 1\}$. The *inverse permutation* of a permutation $\pi \in S_n$ is the permutation which maps $\pi(i)$ to i, for all $1 \leq i \leq n$. For a permutation π, its inverse permutation is denoted by π^{-1}. Permutations act on graphs and on unknown graphs in the natural way. For a graph $G \in \mathcal{G}_n$ and also for an unknown graph G, viewing G as an adjacency matrix and given a permutation $\pi \in S_n$, then $\pi(G)$ is the adjacency matrix obtained by mapping each element $G_{i,j}$ to $G_{\pi(i),\pi(j)}$ (for $1 \leq i, j \leq n$). The permutation, $\pi(G)$ of G, can equivalently be described as the adjacency matrix obtained by permuting both rows and columns of G using π. Two graphs $G, H \in \mathcal{G}_n$ are *isomorphic*, denoted $G \approx H$ if

there exists a permutation $\pi \in S_n$ such that $G = \pi(H)$. An *automorphism* of a graph $G \in \mathcal{G}_n$ is a permutation $\pi \in S_n$ such that $G = \pi(G)$. The set of all automorphisms of a graph $G \in \mathcal{G}_n$ forms a group which is denoted $Aut(G)$. For a group X, we use the notation $Y \leqslant X$ to indicate that Y is a *subgroup* of X. The *dihedral group* $D_n \leqslant S_n$ is the automorphism group of the cycle graph C_n. The dihedral group consists of $2n$ permutations corresponding to n rotations and n reflections of C_n. Example 1 details the automorphisms in D_4.

Example 1. Below is the cycle graph C_4 and its corresponding set of automorphisms which are the elements of the dihedral group D_4.



$$D_4 = \left\{ \begin{array}{l} [1,2,3,4],[4,3,2,1], \\ [2,3,4,1],[1,4,3,2], \\ [3,4,1,2],[2,1,4,3], \\ [4,1,2,3],[3,2,1,4] \end{array} \right\}$$

An order n graph search problem is a predicate, $\varphi(G)$, on an unknown, order n graph G, which is closed under isomorphism. A solution to $\varphi(G)$ is a satisfying assignment for the variables of G. Given a (non-)solution for a graph search problem, each permutation of its vertices yields a symmetric (non-)solution.

One common way to break all symmetries in graph search problems is to define a symmetry breaking predicate which is satisfied only by canonical representatives of each isomorphism class. A *canonical form* of a graph G, denoted by $Can(G)$, is a graph isomorphic to G such that for every two graphs H, G it holds that $Can(G) = Can(H)$ if and only if $H \approx G$. The graph $Can(G)$ is the canonical representative of the isomorphism class of G.

We consider the lexicographic ordering over graphs defined viewing their adjacency matrices as strings. Because adjacency matrices are symmetric with zeroes on the diagonal, it suffices to focus on the upper triangle parts of the matrices [3]. Let s_1, s_2 be the strings obtained by concatenating the rows of the upper triangular parts of the adjacency matrices of two graphs $G, H \in \mathcal{G}_n$ respectively. Then $G \leq_{lex} H$ if and only if $s_1 \leq_{lex} s_2$. When G, H are unknown graphs, expressed in terms of Boolean variables, then the lexicographic ordering can be viewed as specifying a lexicographic order constraint over these variables. This constraint is true with respect to an assignment for the variables of G, H if $G \leq_{lex} H$ under this assignment.

Example 2. The following depicts an unknown graph G and its permutation $\pi(G)$, for $\pi = [2,1,3,5,4]$, both represented as adjacency matrices of Boolean variables.

$$\mathbf{G} = \begin{bmatrix} 0 & x_1 & x_2 & x_3 & x_4 \\ x_1 & 0 & x_5 & x_6 & x_7 \\ x_2 & x_5 & 0 & x_8 & x_9 \\ x_3 & x_6 & x_8 & 0 & x_{10} \\ x_4 & x_7 & x_9 & x_{10} & 0 \end{bmatrix} \quad \pi(\mathbf{G}) = \begin{bmatrix} 0 & x_1 & x_5 & x_7 & x_6 \\ x_1 & 0 & x_2 & x_4 & x_3 \\ x_5 & x_2 & 0 & x_9 & x_8 \\ x_7 & x_4 & x_9 & 0 & x_{10} \\ x_6 & x_3 & x_8 & x_{10} & 0 \end{bmatrix}$$

The constraint $G \leq_{lex} \pi(G)$ is

$$[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_1, x_5, x_7, x_6, x_2, x_4, x_3, x_9, x_8, x_{10}]$$

and it can be simplified as described by Frisch *et al.* [10] to:

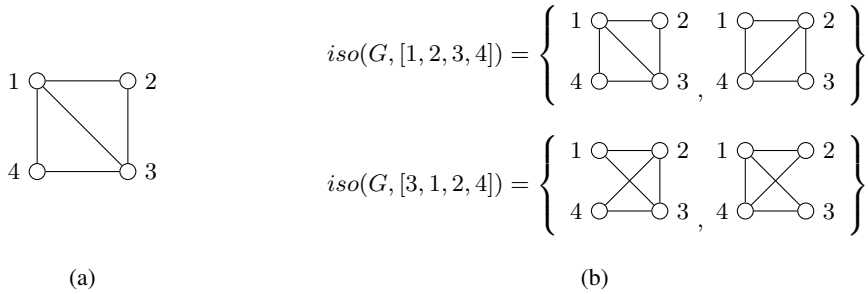
$$[x_2, x_3, x_4, x_8] \leq_{lex} [x_5, x_7, x_6, x_9]$$

Constraint Programming (CP) is a paradigm in which problems are expressed declaratively using constraints over variables (a constraint model). Each variable is associated with a domain of possible values. Each constraint is defined over a subset of variables and limits the combination of values that these variables can be assigned to. The goal of a constraint solver is to find an assignment for the variables which satisfies all the constraints. In constraint programming, constraints are implemented by propagators. The purpose of a propagator, for a constraint, is to remove inconsistent values from domains of variables. That is, values that cannot be part of a solution for this constraint. Constraint solvers typically apply a backtracking search where propagators for the constraints of a problem are repeatedly executed to prune the search space.

3 A Canonical Form of Uniquely Hamiltonian Graphs

In this section we define a canonical form for uniquely Hamiltonian graphs. We introduce an efficient test of canonicity for uniquely Hamiltonian graphs. Furthermore, we show that computing the canonical form and the automorphism group of a given uniquely Hamiltonian graph can be performed efficiently given its Hamiltonian cycle. The proposed canonical form is the basis for the construction of a complete symmetry breaking constraint of polynomial size for uniquely Hamiltonian graph search problems defined in this paper.

Before introducing the main results of this section, we provide some background notation. Given an order n uniquely Hamiltonian graph G and any permutation $h \in S_n$. Denote by $iso(G, h)$ the set of graphs isomorphic to G which contain the Hamiltonian cycle h. Figure 1 depicts (a) the uniquely Hamiltonian graph G and (b) the sets $iso(G, [1, 2, 3, 4])$ and $iso(G, [3, 1, 2, 4])$ which consist of the sets of graphs from the isomorphism class of G which contain respectively the Hamiltonian cycles $[1, 2, 3, 4]$ and $[3, 1, 2, 4]$.



(a) (b)

Fig. 1: (a) the graph G, and (b) the sets $iso(G, [1, 2, 3, 4])$ and $iso(G, [3, 1, 2, 4])$.

In Lemma 1 we prove that the set $iso(G, h)$ is non-empty for any uniquely Hamiltonian graph $G \in \mathcal{UH}_n$ and any permutation $h \in S_n$. In Lemma 2 we prove that if

G_1 and G_2 are isomorphic uniquely Hamiltonian graphs of order n then $iso(G_1, h) = iso(G_2, h)$ for any permutation $h \in S_n$.

Lemma 1. *Let $G \in \mathcal{UH}_n$ be a uniquely Hamiltonian graph and let $h \in S_n$ be a permutation of its vertices. Then $iso(G, h)$ is not empty.*

Proof. Let $G \in \mathcal{UH}_n$ and let h' be the Hamiltonian cycle in G. Let $\pi \in S_n$ be a permutation π which maps h' to h. Namely $\pi \circ h' = h$. The graph $\pi(G)$ is isomorphic to G and thus also uniquely Hamiltonian. By the choice of π it follows that $\pi(G)$ contains the Hamiltonian cycle h. Hence, $\pi(G) \in iso(G, h)$. □

Lemma 2. *Let $G_1, G_2 \in \mathcal{UH}_n$ be isomorphic uniquely Hamiltonian graphs, and let $h \in S_n$ be a permutation. Then $iso(G_1, h) = iso(G_2, h)$.*

Proof. Let G_1, G_2 and h be as in the premise. Let $H \in iso(G_1, h)$. By definition $H \approx G_1$. Hence $H \approx G_2$. H is uniquely Hamiltonian, contains the Hamiltonian cycle h, and isomorphic to G_2. Hence, by definition $H \in iso(G_2, h)$. The proof for the reverse direction is obtained by swapping the roles of G_1 and G_2. □

The following definition introduces a canonical form for a uniquely Hamiltonian graph G as the lexicographically smallest graph isomorphic to G, which contains the Hamiltonian cycle $[1, 2, \ldots, n]$.

Definition 1 (uniquely Hamiltonian graph canonicity). *Let $G \in \mathcal{UH}_n$ be a uniquely Hamiltonian graph of order n. The following is a canonical form of G:*

$$Can_{\mathcal{UH}}(G) \equiv min_{\leq_{lex}} iso(G, [1, 2, \ldots, n])$$

The following theorem proves that Definition 1 is a well defined canonical form.

Theorem 1. *Let $G, H \in \mathcal{UH}_n$ be order n uniquely Hamiltonian graphs. Then,*

$$Can_{\mathcal{UH}}(G) = Can_{\mathcal{UH}}(H) \quad \textit{if and only if} \quad G \approx H.$$

Proof. ($\Leftarrow$) Assume that $G, H \in \mathcal{UH}_n$ and $G \approx H$. The sets $iso(G, [1, 2, \ldots, n])$ and $iso(H, [1, 2, \ldots, n])$ are finite non-empty sets, thus, $Can_{\mathcal{UH}}(G)$, $Can_{\mathcal{UH}}(H)$ exist (Lemma 1). Since $iso(G, [1, 2, \ldots, n]) = iso(H, [1, 2, \ldots, n])$ (Lemma 2) it follows that $Can_{\mathcal{UH}}(G) = Can_{\mathcal{UH}}(H)$. ($\Rightarrow$) Assume that $G, H \in \mathcal{UH}_n$ and $Can_{\mathcal{UH}}(G) = Can_{\mathcal{UH}}(H)$. By definition, $G \approx Can_{\mathcal{UH}}(G)$ and $H \approx Can_{\mathcal{UH}}(H)$. So, $G \approx H$. □

The following theorem is the main result of this section and suggests an efficient test of canonicity for uniquely Hamiltonian graphs with respect to Definition 1.

Theorem 2. *Let $G \in \mathcal{UH}_n$ be a uniquely Hamiltonian graph. Then G is canonical if and only if G contains the Hamiltonian cycle $[1, 2, \ldots, n]$ and $\forall \pi \in D_n$, $G \leq_{lex} \pi(G)$.*

Proof. ($\Leftarrow$) Let $G \in \mathcal{UH}_n$ be a uniquely Hamiltonian graph with the Hamiltonian cycle $[1, 2, \ldots, n]$ and assume that $\forall \pi \in D_n : G \leq_{lex} \pi(G)$. We prove that G is the lexicographically smallest graph in $iso(G, [1, 2, \ldots, n])$ and thus by definition it is canonical. Let $H \in iso(G, [1, 2, \ldots, n])$. So there exists $\sigma \in S_n$ such that $H = \sigma(G)$. By

definition, the edges $\{\{1,2\},\{2,3\},\ldots,\{n-1,n\},\{n,1\}\} \subseteq E(G)$ are mapped to $\{\{\sigma(1),\sigma(2)\},\{\sigma(2),\sigma(3)\},\ldots,\{\sigma(n-1),\sigma(n)\},\{\sigma(n),\sigma(1)\}\} \subseteq E(H)$. Thus, $[\sigma(1),\ldots,\sigma(n)]$ is a Hamiltonian cycle in H. Since $H \in iso(G,[1,2,\ldots,n])$, the only Hamiltonian cycle in H is $[1,2,\ldots,n]$. So, $[\sigma(1),\ldots,\sigma(n)]$ and C_n are the same cycle (consist in the same edges) which implies that σ is an automorphism of the cycle graph C_n. Thus, by definition, $\sigma \in D_n$. We assume that $\forall \pi \in D_n : G \leq_{lex} \pi(G)$. Thus, $G \leq_{lex} \sigma(G) = H$ and we conclude that G is the smallest graph in $iso(G,[1,2,\ldots,n])$.

($\Rightarrow$) Suppose that $G \in \mathcal{UH}_n$ is a canonical uniquely Hamiltonian graph. By definition G is the lexicographically smallest graph, in its isomorphism class, that contains the Hamiltonian cycle $C_n = [1,2,\ldots,n]$. For any $\pi \in D_n$, the graph $\pi(G)$ also contains the Hamiltonian cycle C_n because π is an automorphism of the cycle graph C_n and hence preserves C_n in $\pi(G)$. The graph $\pi(G)$ is isomorphic to G and contains the Hamiltonian cycle C_n. Hence it follows that $G \leq_{lex} \pi(G)$. □

The following corollary provides the basis for the efficient computation of the canonical form of a given uniquely Hamiltonian graph with its Hamiltonian cycle h.

Corollary 1. *Let $G \in \mathcal{UH}_n$ be a uniquely Hamiltonian graph with its Hamiltonian cycle h. Recall that we view h as a permutation. Then,*

$$Can_{\mathcal{UH}}(G) = min_{\leq_{lex}} \{\, \pi(h^{-1}(G)) \,|\, \pi \in D_n \,\}$$

where h^{-1} is the inverse permutation of h. One can view h^{-1} as mapping the cycle h to the cycle $[1\ldots n]$.

Let $G \in \mathcal{UH}_n$ and let h be its Hamiltonian cycle. To compute $Can_{\mathcal{UH}}(G)$, consider the graph $h^{-1}(G)$ which is isomorphic to G and contains the Hamiltonian cycle $[1,2,\ldots,n]$. Then, compute $iso(G,[1,2,\ldots,n])$ by applying on $h^{-1}(G)$ all the permutations in D_n. Observe that $iso(h^{-1}(G),[1,2,\ldots,n]) = iso(G,[1,2,\ldots,n])$. Finally, the lexicographically smallest graph from $iso(G,[1,2,\ldots,n])$ is $Can_{\mathcal{UH}}(G)$. Since the size of $iso(G,[1,2,\ldots,n])$ is bounded by $2n$, finding the lexicographically smallest graph can be done efficiently.

The following observation provides the basis for the efficient computation of the automorphism group of a given uniquely Hamiltonian graph G with its Hamiltonian cycle h.

Observation 1. *Let G be a uniquely Hamiltonian graph with the Hamiltonian cycle h. Then $Aut(G) \leqslant D_n^h$, where D_n^h is the automorphism group of h.*

By definition every automorphism of G preserves the adjacencies of the Hamiltonian cycle h and hence it is an element of D_n^h. Hence, to compute the automorphism group of G it is sufficient to consider only permutations from D_n^h. Observe that each element in D_n^h is of the form $\pi \circ h$ where $\pi \in D_n$. Hence, D_n^h can be computed efficiently from the elements of D_n, and its size is exactly $2n$. It is straightforward to test each permutation from D_n^h to determine if it is an automorphism of G.

In the next section we show how to apply the results of this section to break symmetries in the search for uniquely Hamiltonian graphs.

4 Symmetry Breaking for Uniquely Hamiltonian Graphs

In this section we introduce a complete symmetry breaking constraint of polynomial size for the class of uniquely Hamiltonian graphs. A symmetry breaking constraint ψ is complete for a class of graphs C, if ψ is satisfied by exactly one member of each isomorphism class of graphs from C. If ψ is complete for a class of graphs C then one can view the solutions of ψ, restricted to elements of C, as a set of canonical representatives of C.

We introduce a constraint $sb_{\mathcal{UH}}$ such that for a uniquely Hamiltonian graph G, $sb_{\mathcal{UH}}(G)$ is true if and only if G is canonical with respect to Definition 1. The definition of $sb_{\mathcal{UH}}$ is based on the canonicity test for a given graph, specified in Theorem 2. The constraints comprising $sb_{\mathcal{UH}}(G)$, on the unknown graph G, are derived from the conditions of the canonicity test for a given graph.

The following definition specifies a complete symmetry breaking constraint $sb_{\mathcal{UH}}$ for the class of uniquely Hamiltonian graphs.

Definition 2. *Let G be an unknown order n graph. Then,*

$$sb_{\mathcal{UH}}(G) = (C_n \subseteq G \ \wedge \bigwedge_{\pi \in D_n} G \leq_{lex} \pi(G))$$

The following theorem proves that $sb_{\mathcal{UH}}$ is a complete symmetry breaking constraint of polynomial size.

Theorem 3. *$sb_{\mathcal{UH}}$ is a complete symmetry breaking constraint of polynomial size for uniquely Hamiltonian graphs.*

Proof. The fact that $sb_{\mathcal{UH}}$ is complete follows directly from the construction. For any $G \in \mathcal{UH}_n$, there exists exactly one graph (the graph $Can_{\mathcal{UH}}(G)$) from the isomorphism class of G for which $sb_{\mathcal{UH}}$ is true. The fact that $sb_{\mathcal{UH}}$ is of polynomial size follows because it consists of $2n$ lexicographic order constraints of size $\binom{n}{2}$, and an additional constraint which fixes variables to ensure that the Hamiltonian cycle $[1, \ldots, n]$ is present. □

Example 3. Consider the unknown graph G of order 5 and its corresponding constraint, $sb_{\mathcal{UH}}(G)$:

$$\mathbf{G} = \begin{bmatrix} 0 & x_1 & x_2 & x_3 & x_4 \\ x_1 & 0 & x_5 & x_6 & x_7 \\ x_2 & x_5 & 0 & x_8 & x_9 \\ x_3 & x_6 & x_8 & 0 & x_{10} \\ x_4 & x_7 & x_9 & x_{10} & 0 \end{bmatrix}$$

$$
\begin{aligned}
&(x_1 \wedge x_4 \wedge x_5 \wedge x_8 \wedge x_{10}) \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_4, x_3, x_2, x_1, x_{10}, x_9, x_7, x_8, x_6, x_5] \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_5, x_6, x_7, x_1, x_8, x_9, x_2, x_{10}, x_3, x_4] \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_1, x_7, x_6, x_5, x_4, x_3, x_2, x_{10}, x_9, x_8] \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_8, x_9, x_2, x_5, x_{10}, x_3, x_6, x_4, x_7, x_1] \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_5, x_2, x_9, x_8, x_1, x_7, x_6, x_4, x_3, x_{10}] \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_{10}, x_3, x_6, x_8, x_4, x_7, x_9, x_1, x_2, x_5] \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_8, x_6, x_3, x_{10}, x_5, x_2, x_9, x_1, x_7, x_4] \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_4, x_7, x_9, x_{10}, x_1, x_2, x_3, x_5, x_6, x_8] \wedge \\
&[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}] \leq_{lex} [x_{10}, x_9, x_7, x_4, x_8, x_6, x_3, x_5, x_2, x_1]
\end{aligned}
$$

Simplifying each lex constraint independently and taking into consideration the fact that the variables $x_1, x_4, x_5, x_8, x_{10}$ are fixed to true, yields the following simplified model for $sb_{\mathcal{UH}}(G)$:

$$
\begin{aligned}
&(x_1 \wedge x_4 \wedge x_5 \wedge x_8 \wedge x_{10}) \wedge \\
&[x_2, x_6] \leq_{lex} [x_3, x_9] \wedge \\
&[x_2, x_3, x_6, x_7, x_9] \leq_{lex} [x_6, x_7, x_9, x_2, x_3] \wedge \\
&[x_2, x_3] \leq_{lex} [x_7, x_6] \wedge \\
&[x_2, x_3, x_6, x_7, x_9] \leq_{lex} [x_9, x_2, x_3, x_6, x_7] \wedge \\
&[x_3, x_6] \leq_{lex} [x_9, x_7] \wedge \\
&[x_2, x_3, x_6, x_7, x_9] \leq_{lex} [x_3, x_6, x_7, x_9, x_2] \wedge \\
&[x_2, x_7] \leq_{lex} [x_6, x_9] \wedge \\
&[x_2, x_3, x_6, x_7, x_9] \leq_{lex} [x_7, x_9, x_2, x_3, x_6] \wedge \\
&[x_2, x_3] \leq_{lex} [x_9, x_7]
\end{aligned}
$$

5 Isomorph Free Generation of Uniquely Hamiltonian Graphs

In this section we demonstrate the application of the symmetry breaking constraint introduced in Section 4 to solve graph search problems concerning uniquely Hamiltonian graphs. We focus in particular on enumeration problems where the goal is to enumerate all solutions modulo graph isomorphism, and on extremal problems, where we seek the smallest/largest solution with respect to some target.

Any algorithm for the generation of uniquely Hamiltonian graphs, possibly with additional properties, must deal with two issues:

1. restricting the process to graphs which contain exactly one Hamiltonian graph; and
2. eliminating symmetries among the generated graphs.

Our main contribution is with respect to the second issue. Given a set of uniquely Hamiltonian graphs which contain the cycle C_n, the polynomial sized predicate $sb_{\mathcal{UH}}$ eliminates all symmetries. We address the first issue by defining a propagator which ensures that each generated graph contains exactly one Hamiltonian cycle.

We adopt a constraint based approach combining both issues and show that eliminating all symmetries when searching for uniquely Hamiltonian graphs can be done efficiently. First, we demonstrate this approach for generation of all uniquely Hamiltonian graphs of order $n \leq 13$ modulo graph isomorphism. To this end, we introduce a constraint model for which the satisfying assignments represent the exact set of canonical uniquely Hamiltonian graphs. Then, we demonstrate that by refining this constraint model we can resolve several previously unanswered questions regarding uniquely Hamiltonian graphs. All computations were performed on an Intel Xeon E5-2660 CPU's clocked at 2 GHz, Each instance is run on a single thread.

5.1 The Constraint Model

We present a constraint model for isomorph free generation of order n uniquely Hamiltonian graphs. The model consists of $\binom{n}{2}$ Boolean variables representing the order n unknown graph G (for each $1 \leq i, j \leq n$, the variable $G_{i,j}$ is true if and only if the edge $\{i, j\}$ is present in the solution), and two constraints:

1. $sb_{\mathcal{UH}}(G)$: the symmetry breaking constraint; and
2. $\varphi_{\mathcal{UH}}(G)$: ensures that there is exactly one Hamiltonian cycle in G.

The constraint $sb_{\mathcal{UH}}(G)$ presented in Definition 2 is a conjunction of $2n+1$ constraints: one constraint which fixes the edges of the cycle C_n, and $2n$ lexicographic order constraints corresponding to the permutations in D_n.

The constraint $\varphi_{\mathcal{UH}}(G)$ is implemented by a propagator which forbids the presence of an additional Hamiltonian cycle. We observe that since the symmetry breaking constraint $sb_{\mathcal{UH}}(G)$ fixes the Hamiltonian cycle C_n, it is sufficient to ensure that each edge $e \notin C_n$ is not part of a Hamiltonian cycle.

Algorithm 1 Propagate no additional Hamiltonian cycle

procedure PROPAGATEUH(G)
  input: An unknown graph G which is assumed to contain the Hamiltonian cycle C_n ;
  propagator event: when a graph variable $G_{i,j}$ is fixed to 1
  if $\{i, j\} \in E(C_n)$ **then return**
  else
    for each Hamiltonian path $[v_1, \ldots, v_n]$ in G which involves the edge $\{i, j\}$ **do**
      remove 1 from the domain of the variable G_{v_n, v_1}

Algorithm 1 describes a propagator for the constraint $\varphi_{\mathcal{UH}}(G)$. The propagator is executed whenever a graph variable $G_{i,j}$ is assigned the value 1, which implies that the edge $e = \{i, j\}$ is added to the graph. The propagator iterates over all Hamiltonian paths in G which contain the edge e and checks that each such path $P = [v_1, \ldots, v_n]$

cannot be closed to a Hamiltonian cycle. To ensure this, the propagator removes the value 1 from the domain of the variable G_{v_n,v_1}, so that the edge $\{v_n, v_1\}$ cannot be present in a solution. If the domain of the variable G_{v_n,v_1} consists of the single value 1, then the propagator fails. One can show that Algorithm 1 ensures domain consistency, and that it defines an idempotent propagator (i.e reaches a fix point after a single execution). In theory, the time complexity of Algorithm 1 is equivalent to the complexity of finding all Hamiltonian paths in a graph. The fastest known such algorithm is based on dynamic programming and has time-complexity $O(2^n * n^{O(1)})$ [1, 12] where n is the number of vertices. Despite this worst-case time-complexity, in practice, algorithms for finding Hamiltonian paths can effectively deal with large graphs. In our experiments we consider graphs of relatively small order ($n \leq 22$) and of a very particular form: we always add a single edge to a graph with at most one Hamiltonian cycle. The experiments show that the number of Hamiltonian paths encountered is not an obstacle.

We also found that a slight weakening of Algorithm 1 significantly improves the solving time. Instead of iterating, in the propagator, over all Hamiltonian paths with the edge $\{i, j\}$, we iterate only on those Hamiltonian paths in which the first and second vertices are i and j respectively. This relaxation is justified by the fact that a Hamiltonian cycle which contains an edge $\{i, j\}$ can be represented as a permutation of the form $[i, j, \ldots]$. Hence, when adding a new edge $\{i, j\}$ to the graph, it is sufficient to consider only Hamiltonian paths which start with the vertices i, j in order to verify that there is no Hamiltonian cycle which contains the edge $\{i, j\}$. However, this weakens the consistency level of the propagator because there can be Hamiltonian paths which contain the edge $\{i, j\}$ but do not start with i, j. Hence, the relaxed propagator may not remove all inconsistent values from the domains of the unassigned variables. The following example demonstrates an application of the relaxed propagator.

Example 4. Consider the following partial assignment for an unknown graph G of order 8:

$$\left\{ \begin{array}{l} G_{1,2} = 1, G_{1,4} = 0, G_{1,5} = 0, G_{2,3} = 1, G_{3,4} = 1, G_{4,5} = 1, \\ G_{5,6} = 1, G_{6,7} = 1, G_{7,8} = 1, G_{1,8} = 1, G_{1,6} = 1, G_{3,7} = 1 \end{array} \right\}.$$

The fixed part of the graph G consists of the edges of the cycle graph C_8 and two additional edges $\{1, 6\}, \{3, 7\}$. This partial assignment does not conflict with the constraints $\varphi_{\mathcal{UH}}(G)$ and $sb_{\mathcal{UH}}(G)$. The relaxed propagator, when applied for the assignments $G_{1,6} = 1$ and $G_{3,7} = 1$ does not remove values from the domains of the unassigned variables as there are no Hamiltonian paths which start with $3, 7$ or $1, 6$. Notice however that there are Hamiltonian paths which contain the edges $\{1, 6\}$ and $\{3, 7\}$. For example, $[4, 5, 6, 7, 3, 2, 1, 8]$ and $[5, 4, 3, 2, 1, 6, 7, 8]$. So the edges $\{4, 8\}, \{5, 8\}$ are inconsistent but are not forbidden by the relaxed propagator. Assume that the variable $G_{2,5}$ is now selected for branching and is assigned to the value 1, namely, the edge $\{2, 5\}$ is added to the graph. When triggered, the relaxed propagator finds the Hamiltonian paths $[2, 5, 6, 1, 8, 7, 3, 4]$, $[2, 5, 4, 3, 7, 8, 1, 6]$ and $[2, 5, 4, 3, 7, 6, 1, 8]$ which all begin with the vertices $2, 5$, and ensures that none of these paths can be closed to a Hamiltonian cycle by fixing $G_{2,4} = 0$, $G_{2,6} = 0$ and $G_{2,8} = 0$, thus, forbidding the edges $\{2, 4\}$, $\{2, 6\}$ and $\{2, 8\}$.

Our implementation integrates several heuristics to guide the search. These are inspired by the heuristics for edge selection implemented in the tool generateUHG [11]

for generation of uniquely Hamiltonian graphs. We use a dynamic variable ordering in which the graph variable $G_{i,j}$ selected for branching is the smallest with respect to the lexicographical order on an associated vector defined as follows:

- The first and second entries in the vector are the maximum and minimum values between the degrees of the vertices i and j in G.
- The third and fourth entries in the vector are the maximum and minimum between the sums of the degrees of the neighbors of i and of j which are in the cycle C_n.
- The fifth entry in the vector is the number of common neighbors of i and j in G.

When branching on a variable we first assign it to the value 1. We implemented the constraint model and the search heuristics using the constraint solver Choco 4 [20]. Our implementation is solver independent and can be applied using any other constraint solver.

We first illustrate the application of our approach to generate all canonical uniquely Hamiltonian graphs of order $n \leq 13$. Table 1 details the generation of uniquely Hamiltonian graphs of order $3 \leq n \leq 13$ using Choco. For each value of n we detail the number of uniquely Hamiltonian graphs and the solving time. All running times reported are CPU times and specified in an appropriate unit: (s) seconds or (h) hours. We also detail two statistics regarding the application of the propagator: the number of its applications, and the total number of Hamiltonian paths explored in all applications.

Table 1: Computing order $3 \leq n \leq 13$ uniquely Hamiltonian graphs.

n	solving time	$\mathcal{UH}_n$ graphs	propagator stats	
			ham paths	executions
3	0.01s	1	0	0
4	0.01s	2	0	1
5	0.01s	3	1	4
6	0.01s	12	1	23
7	0.03s	49	19	97
8	0.18s	482	120	746
9	0.21s	6,380	1,249	8,070
10	1.71s	135,252	17,740	154,701
11	55.36s	3,939,509	298,604	4,211,641
12	3,027.15s	166,800,470	9,054,711	174,410,488
13	66.57h	9,739,584,172	337,842,137	10,038,602,346

The results presented in Table 1 are consistent with those reported using the state-of-the-art special purpose uniquely Hamiltonian graph generation tool generateUHG [11]. The statistics regarding the propagator indicate that the potential exponential behavior of Algorithm 1 is not a concern in our setting. In particular, the ratio between the number of applications of the propagator and the total number of Hamiltonian paths encountered is quite small. The generateUHG tool is able to generate all uniquely Hamiltonian graphs for $n \leq 15$. Note that there are more than $9 * 10^{14}$ such graphs. Our general, constraint based approach is not able to enumerate this scale of solutions. The experiment

detailed in Table 1 is a starting point to solve graph search problems related to uniquely Hamiltonian graphs which are based on refinements of the same constraint model.

5.2 Applications

We describe the application of our constraint based approach to answer unresolved questions in the research of uniquely Hamiltonian graphs. These questions relate to the class of uniquely Hamiltonian graphs of minimum degree 3, denoted $\mathcal{UH}3$. In particular, concerning $\mathcal{UH}3$ graphs with specified girth (the length of the smallest cycle in the graph). The identification of $\mathcal{UH}3$ graphs has received much attention in the research of uniquely Hamiltonian graphs. The interest in $\mathcal{UH}3$ graphs with specified girth has emerged from [9] where the author asked whether there exist $\mathcal{UH}3$ graphs of girth greater than 3. A positive answer for this question is given in [22], however that paper shows no concrete examples. The smallest concrete example for such a graph was found by Royle [21] in 2017. Royle provided a $\mathcal{UH}3$ graph of order 18 with girth 5 and also showed that there are no $\mathcal{UH}3$ graphs of smaller order. Goedgebeur *et al.* [11] continued the research on small $\mathcal{UH}3$ graphs with specified girth. They generated all $\mathcal{UH}3$ graphs with girth at least 5 for $n \leq 22$ and found some $\mathcal{UH}3$ graphs of order 20 with girths 3 and 4. They also observed that there are no $\mathcal{UH}3$ graphs with girth 4 for $n \leq 18$ vertices. Motivated by this previous work we ask:

> *What is the smallest positive n such that a $\mathcal{UH}3$ graph with girth $3 \leq k \leq 4$ exists, and how many order n graphs with girth k (up to isomorphism) exist?*

To answer these questions we generate all $\mathcal{UH}3$ graphs of order 18, and all order $n \leq 20$ $\mathcal{UH}3$ graphs with girth at least 4. From these computations we deduce the, previously unknown, smallest orders for which $\mathcal{UH}3$ graphs with girths 3 and 4 exist, and compute all solution graphs of the corresponding orders.

We apply the same constraint programming approach used to generate the results of Table 1 except that the constraint model is extended: (a) to constrain the degree of each vertex to be at least three, and (b) to constrain the girth k of the graph to be such that $3 \leq k \leq 5$. Adding these constraints is straightforward. The degree constraints are specified using sum constraints on the rows of the adjacency matrix. The girth constraints are specified using constraints which ensure that there are no cycles of length less than k. Since k is small these constraints can be implemented efficiently.

Table 2 summarizes the computation of $\mathcal{UH}3$ graphs with girth at least $k \in \{3, 4, 5\}$ as computed by Choco. After the column n which indicates the order, the next three columns detail the number of $\mathcal{UH}3$ graphs of order n with girth at least 3–5 and the time required to compute them. In the table, the symbol "-" indicates that we could not compute this case within 100 days of computation. All running times reported are CPU times and specified in an appropriate unit: (s) seconds, (h) hours or (d) days. The purpose of the fourth column titled "girth ≥ 5" is to verify our results against previously known results. Our results are indeed in complete agreement with those in [11].

From these computations we obtain the following results: There are eleven $\mathcal{UH}3$ graphs of order 18 and nine of them are of girth 3. Previous work show that the smallest $\mathcal{UH}3$ graph has order 18, hence these results lead to the following observation about $\mathcal{UH}3$ graphs with girth 3:

Table 2: Computing $\mathcal{UH}3$ graphs with girth at least 3,4 and 5, respectively.

n	girth ≥ 3		girth ≥ 4		girth ≥ 5	
	$\mathcal{UH}3$ graphs	time	$\mathcal{UH}3$ graphs	time	$\mathcal{UH}3$ graphs	time
18	11	13.80d	2	6.97h	2	0.16h
19	-	-	1	4.28d	1	1.33h
20	-	-	12	91.58d	2	12.01h
21	-	-	-	-	25	5.24d
22	-	-	-	-	33	53.40d

Observation 2. *The smallest $\mathcal{UH}3$ graph with girth 3 is of order 18, and there are exactly nine $\mathcal{UH}3$ graphs of order 18 with girth 3.*

All three $\mathcal{UH}3$ graphs of order $n < 20$ with girth at least 4 have girth 5. There are twelve $\mathcal{UH}3$ graphs of order 20 with girth at least 4, ten of these are of girth 4, and two are of girth 5. These results lead to the following observation about $\mathcal{UH}3$ graphs with girth 4:

Observation 3. *The smallest $\mathcal{UH}3$ graph with girth 4 is of order 20, and there are exactly ten $\mathcal{UH}3$ graphs of order 20 with girth 4.*

Figure 2 depicts a smallest $\mathcal{UH}3$ graph with girth 3, and a smallest $\mathcal{UH}3$ graph with girth 4. These graphs were obtained from our implementation.

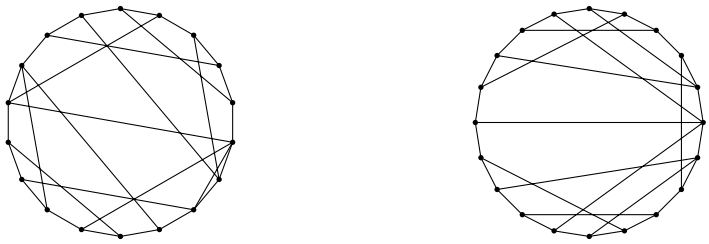


Fig. 2: $\mathcal{UH}3$ graphs of order 18 with girth 3 (left) and of order 20 with girth 4 (right).

6 Related work

The state-of-the-art uniquely Hamiltonian graph generation tool generateUHG [11] is based on the notion of canonical construction path [18]. In this approach, to generate all order n non-isomorphic uniquely Hamiltonian graphs, the algorithm starts with the cycle graph C_n and iteratively adds edges in all possible ways while checking that the graph remains uniquely Hamiltonian. To avoid generating isomorphic graphs, the algorithm only visits graphs that are constructed through a sequence of so called canonical edges. For our purposes it is only important to note that the identification of canonical edges requires the use of calculations of automorphisms and of canonical forms

of graphs during search. In generateUHG, these calculations are performed using the graph isomorphism tool nauty [17, 19] which has exponential behavior in the worst case. Our approach in contrast, breaks all symmetries efficiently using a complete symmetry breaking constraint of polynomial size. Moreover, as we have shown in Section 3 we can compute efficiently the canonical form and the automorphism group of a uniquely Hamiltonian graph if its Hamiltonian cycle is given, which is the case here. Hence, the algorithm of generateUHG can utilize our results to avoid symmetries in a more efficient way. Yet we note that the computation time of our implementation for the task of generating all uniquely Hamiltonian graphs is not as good as generateUHG. For example, in our implementation it took 66.57 hours to generate all uniquely Hamiltonian graphs of order 13 while in generateUHG it took 7.20 hours. Moreover, our implementation is able to generate the complete set of non-isomorphic uniquely Hamiltonian graphs for up to order 13 while generateUHG is able to generate them for up to order 15. We do not expect our general approach to compete with dedicated algorithms for enumeration. However, for specific problems related to uniquely Hamiltonian graphs in which the problem is more constrained as described in Section 5.2, we are able to answer previously unresolved questions which generateUHG could not solve.

7 Conclusion

This paper introduces, for the first time, a complete symmetry breaking constraint of polynomial size for a significant class of graphs: the class of uniquely Hamiltonian graphs. State-of-the-art algorithms for isomorph-free generation of graphs from this class deal with symmetries in the generated graphs using general purpose graph-isomorphism testing tools which have worst-case exponential behavior. Our symmetry breaking method, in contrast, eliminates all symmetries efficiently. We first introduce a canonical form for uniquely Hamiltonian graphs and show an efficient canonicity test. Then we show that the automorphism group and the canonical form of a given uniquely Hamiltonian graph can be computed efficiently given its Hamiltonian cycle. Based on these results we construct a complete symmetry breaking constraint of polynomial size for uniquely Hamiltonian graphs. We demonstrate the application of the proposed symmetry breaking constraint using a constraint based approach and show new results regarding uniquely Hamiltonian graphs.

An important aspect of this paper is future work. To date there is no known complete symmetry breaking constraint of polynomial size for graph search problems in general. Identifying significant classes of graphs for which such symmetry breaking constraints exist is important for two reasons: first, it may help to solve additional hard graph search problems, and second, it may improve our understanding of symmetry in graph search problems in general. This paper takes a step in this direction and can be seen as a first answer, for graph search problems, to the question posed by Walsh [26] who discusses several tractable cases for symmetry breaking: ”Are there other common types of symmetry which occur in practice that are polynomial to break?”

References

1. Bellman, R.: Dynamic programming treatment of the travelling salesman problem. J. ACM **9**(1), 6163 (Jan 1962)
2. Bondy, J., Jackson, B.: Vertices of small degree in uniquely Hamiltonian graphs. Journal of Combinatorial Theory, Series B **74**(2), 265 – 275 (1998)
3. Cameron, R., Colbourn, C., Read, R., Wormald, N.C.: Cataloguing the graphs on 10 vertices. Journal of Graph Theory **9**(4), 551–562 (1985)
4. Codish, M., Ehlers, T., Gange, G., Itzhakov, A., Stuckey, P.J.: Breaking symmetries with lex implications. In: Gallagher, J.P., Sulzmann, M. (eds.) Functional and Logic Programming - 14th International Symposium, FLOPS 2018, Nagoya, Japan, May 9-11, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10818, pp. 182–197. Springer (2018)
5. Codish, M., Frank, M., Itzhakov, A., Miller, A.: Computing the Ramsey number r(4, 3, 3) using abstraction and symmetry breaking. Constraints An Int. J. **21**(3), 375–393 (2016)
6. Codish, M., Gange, G., Itzhakov, A., Stuckey, P.J.: Breaking symmetries in graphs: The nauty way. In: Rueher, M. (ed.) Principles and Practice of Constraint Programming - 22nd International Conference, CP 2016, Toulouse, France, September 5-9, 2016, Proceedings. Lecture Notes in Computer Science, vol. 9892, pp. 157–172. Springer (2016)
7. Codish Michael, Miller Alice, Prosser Patrick, Stuckey Peter J.: Constraints for symmetry breaking in graph representation. Constraints **24**(1), 1–24 (2019)
8. Crawford, J.M., Ginsberg, M.L., Luks, E.M., Roy, A.: Symmetry-breaking predicates for search problems. In: Aiello, L.C., Doyle, J., Shapiro, S.C. (eds.) Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR'96), Cambridge, Massachusetts, USA, November 5-8, 1996. pp. 148–159. Morgan Kaufmann (1996)
9. Fleischner, H.: Uniquely Hamiltonian graphs of minimum degree 4. Journal of Graph Theory **75**(2), 167–177 (2014)
10. Frisch, A.M., Harvey, W.: Constraints for breaking all row and column symmetries in a three-by-two matrix. In: In Proceedings of SymCon03 (2003)
11. Goedgebeur, J., Meersman, B., Zamfirescu, C.T.: Graphs with few Hamiltonian cycles. Mathematics of Computation **89**(322), 965991 (Sep 2019)
12. Held, M., Karp, R.M.: The construction of discrete dynamic programming algorithms. IBM Syst. J. **4**(2), 136–147 (1965)
13. Heule, M.J.H.: Optimal symmetry breaking for graph problems. Mathematics in Computer Science (2019)
14. Itzhakov, A., Codish, M.: Breaking symmetries in graph search with canonizing sets. Constraints pp. 1–18 (2016)
15. Itzhakov, A., Codish, M.: Incremental symmetry breaking constraints for graph search problems. Proceedings of the AAAI Conference on Artificial Intelligence **34**(02), 1536–1543 (Apr 2020)
16. McGuire, G., Tugemann, B., Civario, G.: There is no 16-clue Sudoku: Solving the Sudoku minimum number of clues problem. CoRR **abs/1201.0749** (2012)
17. McKay, B.D.: Practical Graph Isomorphism. Congressus Numerantium **30**, 45–87 (1981)
18. McKay, B.D.: Isomorph-free exhaustive generation. Journal of Algorithms **26**(2), 306–324 (1998)
19. McKay, B.D., Piperno, A.: Practical graph isomorphism, II. Journal of Symbolic Computation **60**, 94–112 (Jan 2014)
20. Prud'homme, C., Fages, J.G., Lorca, X.: Choco Solver Documentation. TASC, INRIA Rennes, LINA CNRS UMR 6241, COSLING S.A.S. (2016), http://www.choco-solver.org

21. Royle, G.: What is the smallest uniquely Hamiltonian graph with minimum degree at least 3? MathOverflow, https://mathoverflow.net/q/255784, URL:https://mathoverflow.net/q/255784 (version: 2020-10-17)
22. Seamone, B.: On uniquely Hamiltonian claw-free and triangle-free graphs. Discussiones Mathematicae Graph Theory **35**(2), 207 – 214 (01 May 2015)
23. Sheehan, J.: Graphs with exactly one Hamiltonian circuit. Journal of Graph Theory **1**(1), 37–43 (1977)
24. Shlyakhter, I.: Generating effective symmetry-breaking predicates for search problems. Discrete Applied Mathematics **155**(12), 1539–1548 (2007)
25. Walsh, T.: General symmetry breaking constraints. In: Principles and Practice of Constraint Programming - CP 2006, 12th International Conference, CP 2006, Nantes, France, September 25-29, 2006, Proceedings. pp. 650–664 (2006)
26. Walsh, T.: Symmetry breaking constraints: Recent results. Proceedings of the AAAI Conference on Artificial Intelligence **26**(1) (Jul 2012)