Testing Isomorphism of Graphs in Polynomial Time

RUI XUE*, SKLOIS, Institute of Information Engineering, CAS. School of Cyber Security, UCAS., China

Given a graph G, the graph $[G]$ obtained by adding, for each pair of vertices of G, a unique vertex adjacent to both vertices is called the binding graph of G. In this work, we show that the class of binding graphs is graph-isomorphism complete and that the stable partitions of binding graphs by the Weisfeiler-Lehman (WL) algorithm produce automorphism partitions.

To test the isomorphism of two graphs G and H, one computes the stable graph of the binding graph $[G \uplus H]$ for the disjoint union graph $G \uplus H$. The automorphism partition reveals the isomorphism of G and H. Because the WL algorithm is a polynomial-time procedure, the claim can be made that the graph-isomorphism problem is in complexity class P.

CCS Concepts: • **Theory of computation** → *Graph algorithms analysis*; • **Mathematics of computing** → **Graph algorithms**.

Additional Key Words and Phrases: binding graphs, graph isomorphism, Weisfeiler-Lehman algorithm

1 INTRODUCTION

A graph consists of a set of vertices and a binary relation on the vertices. An isomorphism between two graphs is a bijection of vertices that conforms to relations. Testing whether two given graphs are isomorphic or not is called the graph-isomorphism problem. This work investigates the graph-isomorphism problem for finite graphs.

Apart from its importance in practical applications such as mathematics, physics, chemistry, biology, networking, and many other areas, the graph-isomorphism problem attracts much attention in the theory of computation due to its specific placement on the computational complexity spectrum.

As is well known, whether complexity classes P and NP are equal is an important open problem in the theory of computer science. The importance of developing an efficient solution, or refuting its possibility, to the graph-isomorphism problem comes from the fact that it is one of two natural problems (the other being the integer factorization problem) that potentially have intermediate complexity, or in other words, they may be neither in P nor in NP-complete (cf. Karp [29], Garey and Johnson [20]).

Given a mapping between the vertices of two graphs, it can be verified in polynomial time whether this mapping is an isomorphism. This means that the graph-isomorphism problem is in NP. After a half-century of research, substantial evidence in the literature hints that the graph isomorphism problem is not NP-hard (cf. [3, 21, 22, 37]). The most advanced result was presented by Babai [3], who showed that a procedure in quasipolynomial time exhibited graph-isomorphism. However, a published and provably efficient algorithm for the graph-isomorphism problem is still unavailable. This work tries to attack this problem and to confirm that it is in P.

The graph-isomorphism problem with respect to some restricted classes of graphs, such as bipartite [10] and chordal graphs [47], has been proven to be as difficult as that with respect to all graphs. These classes are referred to as graph-isomorphism complete. A new class of graphs, called binding graphs (cf. Definition 5.1), is proposed and proven to be graph-isomorphism complete in this work.

Two forms of technique are mainly used in the community to address the graph-isomorphism problem. The first and most natural one is the combinatorial technique, which aims to classify graph vertices according to the structural properties of graphs. That approach was initiated by Morgan [40], as reported in the published literature. The most influential approach of this kind was invented by Weisfeiler and Lehman and is known as the Weisfeiler-Lehman (WL) algorithm [48]. The WL algorithm is such a powerful combinatorial tool that it can distinguish almost all graphs that are not isomorphic (cf. [6], [8]).

The other dominant technique used to address graph-isomorphism applies the theory of groups initiated by Babai ([1], [7]). Luks developed a deep theory of groups with respect to graph-isomorphism and made it influential in the community in a seminal paper [36]. By combining the combinatorial and group theoretical techniques developed there, Babai was able to develop a procedure in quasipolynomial time for the graph-isomorphism problem (cf. [3], [5]), which put the graph-isomorphism problem on the borderline of P.

Almost all practical graph-isomorphism algorithms adopt the approach of vertex partitions with various techniques by using the structural properties of graphs. Although much success has been reported in applications, none of them has been confirmed to be in polynomial time.

In the seminal paper [11], Cai, Fürer, and Immerman showed that a WL-like algorithm, even with higher-dimensional extensions in the symbol k-WL, cannot success in less than exponential time for the *general* graph-isomorphism problem, in the way of coordinates distinguishing two separate graphs. Their results indicate that for any fixed k, it is not enough to decide graph-isomorphism with k-WL in that way.

Although the result by Cai, Fürer, and Immerman eliminates the possibility of distinguishing graphs with the WL algorithm in polynomial time *in general*, it does not refute the possibility of distinguishing graphs of specific classes. As is well known, the WL algorithm (2-WL in the literature) has already succeeded in distinguishing graphs of some classes. The class of trees is such an example ([12], [27]).

This work shows that the stable partition of a binding graph is an automorphism partition (Theorem 8.3). Because the graph isomorphism problem is equivalent to the automorphism partition ([37], cf. Theorem 9.1) and the class of binding graphs is graph-isomorphism complete, this approach provides a procedure for testing graph isomorphism. The fact that the WL algorithm is in polynomial time confirms that the graph-isomorphism problem is in the complexity class P.

In other words, although the WL algorithm is used, the decision procedure in this work is different in mechanism from those to distinguishing two graphs separately by the WL-like algorithm to graph isomorphism adopted previously in the literature. The latter is refuted by the result of CFI in [11] as described above.

The next subsection describes the routines developed in this work, including the techniques involved and the results obtained. Related work is then introduced in the last subsections of this introduction.

1.1 Routines and Contributions

This work is simply an application of the WL algorithm to binding graphs. This section first describes the WL algorithm and some of its properties obtained in this work and then introduces binding graphs and routines to obtain a decision procedure for graph isomorphism.

1.1.1 WL algorithm. The WL algorithm is usually implemented in the literature by iterative refinement of coloring graphs. However, this procedure, though elegant, focuses only on coloring of vertices and is not sufficient or convenient for the development proposed in this study. The colors of edges during the procedure are essentially used in the explorations. For that reason, graphs are implemented following Lehman[1] [32] with minor deviations, as matrices with independent variables as entries.

A variable x_0 is reserved as a special symbol to label non-edges or vertices. When a graph of order n is formed as a matrix $G = (g_{ij})$, the set of vertices is assumed to be the set $[n]$. The entry g_{ii} is the label of vertex i. For $i \neq j$, the entry g_{ij} is the label of edge (i, j) if $g_{ij} \neq x_0$ and is the label of non-edge (i, j) otherwise. A set Var of independent variables is used to label graphs. The number of distinct variables in a graph G is, following Lehman [32], called the dimension of G and denoted by symbol $\dim(G)$.

Given a labeled graph, the WL algorithm proceeds iteratively to refine its labels. In the t-th iteration, it counts the directed walks of length 2 (2-d-walks, cf. Section 2 for exact definition) from one vertex to another in the current graph $G_t := (x_{ij})$. The entry z_{ij} of matrix $G_t \diamond G_t := (z_{ij})$ is the collection (multiset) of all 2-d-walks from vertex i to vertex j in G_t. Note that $i \rightarrow i \rightarrow j$ and $i \rightarrow j \rightarrow j$ are naturally counted as 2-d-walks from i to j.

The entries in $G_t \diamond G_t$ will then be globally replaced by variables from Var, such that two entries z_{uv}, z_{rs} are replaced by the same variable if and only if $z_{uv} = z_{rs}$ for all $u, v, r, s \in [n]$. This substitution is called the equivalent variable substitution (e.v.s. for short). After substitution, a new graph G_{t+1} is obtained, ready for the next iteration.

The iteration proceeds until $\dim(G_t) = \dim(G_{t+1})$ for some t. The graph G_t, denoted as $\mathrm{wl}(G)$, is then the stable graph of G. Because $\dim(G_t) \leq n^2$ for a graph G of order n, the iteration will halt in at most n^2 steps. An upper bound $O(n \log n)$ for the number of iterations was obtained by Lichter, Ponomarenko, and Schweitzer [33].

It should be pointed out that before starting the first iteration in the WL algorithm, an equivalent substitution should be performed only on the labels of vertices (entries on the diagonal) in graph G to obtain a graph G_1, such that the labels of vertices in G_1 do not appear as labels of edges or non-edges of G_1.

The WL algorithm is a refinement procedure in the sense that if $G_t := (x_{ij})$ and $G_{t+1} := (y_{ij})$, then $y_{uv} = y_{rs} \implies x_{uv} = x_{rs}$ for all $u, v, r, s \in [n]$. In this setting, it can be said that G_t is embedded in G_{t+1} in symbol $G_t \succeq G_{t+1}$. If $G_t \succeq G_{t+1}$ and $G_{t+1} \succeq G_t$ for some t, they are said to be equivalent and are denoted as $G_t \approx G_{t+1}$. In this case, G_t is a stable graph. We see that all stable graphs of a graph are equivalent.

The iteration procedure in WL is automorphism-preserving. It can therefore be concluded that the automorphism group $\mathrm{Aut}(G)$ of graph G is the same as $\mathrm{Aut}(\mathrm{wl}(G))$.

If a set C is the collection of all vertices with identical labels in $\mathrm{wl}(G)$, it is called a cell of $\mathrm{wl}(G)$. The collection of all cells is then a partition of vertices $[n]$ and is called a stable partition of G. The automorphism-preserving mechanism of WL guarantees that an automorphism of G sends a vertex from a cell to the same cell. However, a cell might not be a single orbit of $\mathrm{Aut}(G)$ in general, as is well known.

If the label x_{uv} is said the connection between vertex u and vertex v in stable graph $X = (x_{ij})$, one of the properties for stable graphs is that a collection of cells and the connections between them in a stable graph compose a stable graph. This is presented as the Composition Theorem by this author (cf. Theorem 4.3).

Another property refers to individualizing a vertex in a stable graph. For a vertex u, the neighbors of u with identical labels on the edges adjacent to u form a block of vertices in a stable graph. These blocks, together with $\{u\}$, form a partition of vertices, which is called the block partition of vertex u.

---

[1]We cite Lehman [32] rather than Weisfeiler or others, following the suggestions in [32].

When a vertex u is individualized in a stable graph $\mathtt{wl}(G)$ by re-marking the label of u using a variable from Var that does not appear in $\mathtt{wl}(G)$, the stable graph for the individualized graph will possess the cell partition the same as the block partition of u in $\mathtt{wl}(G)$. This is presented here as the Individualization Theorem (cf. Theorem 4.1). To the knowledge of this author, neither Theorem has ever appeared before in the literature.

1.1.2 Binding graphs and their properties. Given a graph G of order n, a new graph can be constructed by, for each pair of vertices, adding a vertex adjacent only to both vertices. The graph obtained is unique up to isomorphism and is therefore called the binding graph of G, with symbol $[G]$. The graph $[G]$ is then of order $n_1 = n(n+1)/2$, and G is its basic graph.

The vertices and edges of G are called basic vertices and basic edges in $[G]$. For each pair of basic vertices u, v, the new added vertex p adjacent to u and v is a binding vertex of both u and v, with symbol $p := u \dot\wedge v$. The vertex p and the edges (u, p) and (v, p) are the binding vertex and the binding edges of u and v respectively. In total, there are $n(n-1)/2$ binding vertices in $[G]$ for a basic graph G of order n.

A binding graph is uniquely determined by the basic graph, up to the renaming of binding vertices. This makes it possible to show that two basic graphs are isomorphic if and only if their binding graphs are also (cf. Theorem 5.2). This enables the claim that the class of binding graphs is graph-isomorphism complete.

A graph Φ will be obtained if all labels other than those of vertices and binding edges are replaced by x_0 from a stable graph $\mathtt{wl}([G])$. The resulting graph is then called the ϕ-graph induced by $\mathtt{wl}([G])$.

It can be shown that the stable graph of ϕ-graph Φ is equivalent to $\mathtt{wl}([G])$ (cf. Theorem 7.1), and hence $\mathrm{Aut}(\Phi) = \mathrm{Aut}(\mathtt{wl}([G])) = \mathrm{Aut}([G])$. The proof of $\mathtt{wl}(\Phi) \approx \mathtt{wl}([G])$ essentially relies on the fact that for each pair of basic vertices u, v and their binding vertex $p = u \dot\wedge v$, the labels on the two binding edges (p, u) and (p, v) in $\mathtt{wl}([G])$ recognize whether (u, v) is a basic edge or a non-edge of G (cf. Lemma 6.1). Moreover, the labels on the binding vertices are in fact equivalent to the labels on the basic edges in graph $\mathtt{wl}([G])$ (cf. Lemma 6.2). More properties are given in [49].

Two similar stable graphs are introduced here as stable graphs $X = (x_{ij})$ and $Y := (y_{ij})$ that have the same collection of labels, such that $x_{uv} = y_{rs}$ if and only if the multiset of 2-d-walks from u to v in X is identical to the multiset of 2-d-walks from r to s in Y.

Similar stable graphs of binding graphs are shown to be isomorphic by induction on the number of basic vertices with the help of the induced ϕ-graph, the Individualization Theorem, and the Composition Theorem (cf. Theorem 8.1). As a result, the stable partition of a binding graph is shown to be the automorphism partition (cf. Theorem 8.3).

It is also shown true in the case of binding graphs that the graph-isomorphism problem is equivalent to computing the orbits of automorphism groups ([37],[10], cf. Theorem 9.1). It is not surprising to form a polynomial-time procedure GI for testing graph isomorphism with binding graphs.

The decision procedure GI is intuitively simple: For two connected simple graphs G_1, G_2 of order n, to construct the binding graph $[G_1 \uplus G_2]$ of the disjoint union $G_1 \uplus G_2$, and to compute the stable graph $\mathtt{wl}([G_1 \uplus G_2])$ with the WL algorithm. The cells of the stable partition are then checked to determine whether some of them are shared by vertices from both graphs. If this is true, then the two graphs are isomorphic; otherwise, they are not (cf. Section 9). The computational complexity of the procedure has been analysed to be $O(n^8 \log n)$.

In summary, a class of binding graphs is proposed and proven to be graph-isomorphism complete. In a binding graph, each basic vertex connects all other basic vertices through binding vertices. The local deviations that each basic vertex experiences during refinement are reflected and transferred to all others by means of binding vertices. As a

result, finer refinements are obtained with the WL algorithm. It is shown that the stable partitions of binding graphs are the automorphism partitions, which leads to a polynomial-time test procedure for graph isomorphism.

1.2 Related Work

The graph isomorphism problem has been extensively studied in the literature. Some studies aim at practical applications, whereas others pursue theoretical explorations. This part does not intend to survey the literature on the graph isomorphism problem. Only the most advanced or closely related works will be mentioned according to the author's knowledge. Some studies may not be fairly treated or cited here due to the author's restricted knowledge.

Practical applications have seen great success with algorithms like those of Nauty [38][39], VF2 [13], Saucy [16], Bliss [28], Conauto [34], Traces [39], and Vsep [44], to name a few. Because the intention of this work is theoretical investigation, their practical advances will not be further addressed.

The most advanced result to date was achieved by Babai [2, 3, 5], who proposed procedures for testing graph isomorphism in quasipolynomial time. In the procedures proposed in these studies, group theoretic and combinatoric techniques are ingeniously developed and combined to achieve a result that pushes the graph-isomorphism problem close to the borderline of P. Grohe, Neuen, and Schweitzer [25] gave a more efficient algorithm for graphs of bounded degree.

The application of group theory to tackle the graph-isomorphism problem has seen considerable success. It was initiated by Babai [1], and the seminal work by Luks [36] made it more powerful and popular, exploring profound results that were later frequently adopted. Readers are referred to Grohe, Neuen, and Schweitzer [24, 26] and to Babai [4] and the references there for details.

The classification of vertices by degrees, walks, and so on in graphs is a natural way of determining graph isomorphism [41]. It is involved, more or less, in most studies of graph isomorphism and was initiated by Morgan [40], as reported in the literature. The most popular approach was proposed by Weisfeiler and Lehman [32, 48] as the so-called Weisfeiler-Lehman (WL) procedure. The classical WL algorithm adopted in this work is referred to as 2-dimensional WL and has been extended to k-dimensional WL [9, 11, 27] for positive integers $k > 2$. Fuhlbrück, Köbler, Ponomarenko, and Verbitsky have explored some important properties of the WL algorithm [17]. Grohe showed that the graph-isomorphism problem can be resolved with k-WL to graphs with a forbidden minor in polynomial time [23]. However, Cai, Fürer, and Immerman showed in a seminal paper [11] that it is not possible to resolve the graph-isomorphism problem with k-WL in polynomial time for general graphs, in the way of coordinates distinguishing two separate graphs. Readers may refer to Fürer [19], Kiefer [30], and Grohe and Neuen [24] and the references there for more information about the WL algorithm.

The study by Rücker and Rücker [42, 43] used walks of any length in graphs as a handle to distinguish vertices (cf. [49]). Tinhofer and Klin [46] extensively discussed stabilization procedures and especially developed the total degree partition [45]. Cvetković, Rowlinson, and Simić initiated a vertex classification approach by eigenvalues and eigenvectors of adjacent matrices, called a star partition [15], and showed that every graph has a star partition (cf. [14]).

2 PRELIMINARIES

The number n is always assumed to be a positive integer in $\mathbb{Z}^+$. Let $[m, n]$ denote the set of nonnegative integers $\{m, m + 1, \ldots, n\}$ and $[n]$ the set $[1, n]$. A sequential partition of $[m, n]$ is a series of sets $[a_1, b_1], \ldots, [a_r, b_r]$ such that $a_1 = m, a_{i+1} = 1 + b_i \leq b_{i+1}, b_r = n$ for all $i \in [r - 1]$.

For a permutation σ over $[n]$, using $j = i^\sigma$ means that σ sends i to j. Matrix A^T is the transpose of matrix A, and $|S|$ is the cardinality of set S.

A multiset is a set that allows an element to appear multiple times in it, which is symbolized by $\{\!\!\{ \cdots \}\!\!\}$. Multisets S_1 and S_2 are identical iff they have the same elements counting multiplicities, which is symbolized by $S_1 \equiv S_2$.

2.0.1   Graphs. In this study, the vertex set of a graph is assumed to be a set of positive integers and graphs are finite and labeled on both vertices and edges.

Following Weisfeiler and Lehman [32, 48], independent variables are adopted here as labels. The variable x_0 is reserved as a special variable to signal a "non-edge" in a graph. This makes it convenient to describe, say, simple graphs, connected graphs, and so on. Let $\mathrm{Var} := \{x, y, z, x_1, x_2, \ldots\}$ be a set of independent variables, with $x_0 \notin \mathrm{Var}$. The notion of graphs is formally given as follows.

Definition 2.1 (Graphs). Let $V \subseteq \mathbb{Z}^+$ be a nonempty set of positive integers. *A graph G over V of order $|V|$ is a* function $G : V \times V \to \{x_0\} \cup \mathrm{Var}$. V is the vertex set of graph G. A vertex $v \in V$ is said to be a neighbor of, or adjacent to vertex u iff $u \neq v$ and $g(u, v) \neq x_0$. For different vertices $u, v \in V$ of a graph G, (u, v) is *an edge* with label $g(u, v)$ if $g(u, v) \neq x_0$, and (u, v) is *a non-edge* with label x_0 if $g(u, v) = x_0$. The variable $g(u, u)$ is the label of vertex u.

A graph G over $[n]$ can be conveniently formed as a square matrix $G := (g_{ij})$ of order n with $g_{ij} := g(i, j)$ for $i, j \in [n]$. The matrix G will be referred to as a graph of order n in this context. In other words, a graph of order n is identical to a square matrix with entries from $\{x_0\} \cup \mathrm{Var}$, of order n. The collection of all variables appearing in G is denoted as $\mathrm{Var}(G)$.

The degree $\deg(u)$ of a vertex u in G is the number of its neighbors. Following Lehman [32], the dimension $\dim(G)$ of graph G is the number of distinct entries in G, that is, $\dim(G) = |\mathrm{Var}(G)|$. It is a straightforward fact that $\dim(G) \leq n^2$ for a graph G of order n.

To transpose two vertices $u, v \in V$ in a graph G is to exchange the names of u, v in G. This is equivalent to exchanging the u's and v's rows and simultaneously exchanging the u's and v's columns in matrix G. That will be referred to as a *row-column transposition* to G. A row-column permutation is a series of row-column transpositions.

A *simple graph* $G = (g_{ij})$ of order n is a graph satisfying $G^\mathsf{T} = G$, $\dim(G) \leq 2$ and $g_{ii} = x_0$ for all $i \in [n]$. A *path* from vertex i_0 to i_t of length t in a graph is a sequence of distinct vertices $i_0, i_1 \ldots, i_t$ such that (i_{k-1}, i_k) is an edge for all $k \in [t]$. A *connected simple graph* is a simple graph such that for each pair of vertices i, j, there is a path from i to j.

A regular graph is a simple graph in which all vertices have the same degree. A regular graph is said to be strongly regular iff there are two integers λ, μ such that for each pair of adjacent vertices, there are λ vertices adjacent to both vertices, and for each pair of nonadjacent vertices, there are μ vertices adjacent to both vertices.

In a graph, we use the notion of a *2-d-walk* to extend that of a conventional directed walk of length 2. For a graph of order n, a 2-d-walk from vertex i to j is just a tuple (i, k, j) for some $k \in [n]$, where k is allowed to be any vertex in $[n]$ and (i, k) and (k, j) are not necessarily edges.

2.0.2   Graph isomorphisms. Given two graphs $G = (g_{ij})$ and $H = (h_{ij})$ of order n, if there is a permutation σ on $[n]$ such that $g_{ij} = h_{i^\sigma j^\sigma}$ for all $i, j \in [n]$, it is said that G is *isomorphic to* H, and this relation is denoted as $G \cong H$. In that case, σ is *an isomorphism* from G to H.

A fact from elementary algebra is that there is an isomorphism σ from G to H if and only if there is a permutation matrix P of order n such that $PHP^\mathsf{T} = G$.

The isomorphisms from a graph G to itself are automorphisms of G. The collection of all automorphisms is then a permutation group, which is called the automorphism group of G and denoted as $\mathrm{Aut}(G)$.

An orbit of group $\mathrm{Aut}(G)$ is a set C of vertices in G satisfying not only $C^\sigma = C$ for all $\sigma \in \mathrm{Aut}(G)$, but also $u^\sigma = v$ with some $\sigma \in \mathrm{Aut}(G)$ for each pair of vertices $u, v \in C$.

For a graph G of order n, a partition $C := \{C_1, \ldots, C_s\}$ of $[n]$ consisting of all orbits of $\mathrm{Aut}(G)$ is called *the automorphism partition* of G. A partition consisting of singletons is called a discrete partition, and a partition consisting of one element is called a unit partition.

2.0.3 Substitution and embedding. The following notions introduced in [32] are defined for matrices as well as for graphs.

Definition 2.2 (Embedding and equivalence). Let $A := (a_{ij})$ and $B := (b_{ij})$ be two matrices of order n.

- If $b_{uv} = b_{st}$ implies $a_{uv} = a_{st}$ for all $u, v, s, t \in [n]$, then matrix A is said to *be embedded* in matrix B. This relation is denoted as $A \succeq B$.
- If $A \succeq B$ and $B \succeq A$, then A is said to be *equivalent* to B, which is denoted as $A \approx B$.

In the case of graphs, the following properties are easy to obtain from the definitions:

Proposition 2.3 ([32]). *For any graphs G, H, and X of order n,*

- $G \succeq H$ *and* $H \succeq X$ *imply* $G \succeq X$.
- $G \succeq H$ *implies* $\dim(G) \leq \dim(H)$, *and* $\mathrm{Aut}(H) \subseteq \mathrm{Aut}(G)$.
- $G \approx H$ *implies* $\dim(G) = \dim(H)$, *and* $\mathrm{Aut}(G) = \mathrm{Aut}(H)$.

The entries of a matrix are often replaced by variables from Var to obtain a graph.

Definition 2.4 (Equivalent variable substitution, or e.v.s.). Given a matrix A of order n, replace the entries of A by variables from Var in such a way that identical entries have the same variables and nonidentical entries have distinct variables. The resulting matrix G will be a graph, equivalent to A. A substitution alike is referred to as *an equivalent variable substitution (abbreviated as e.v.s.).* $G = \mathrm{evs}(A)$ is written to signify this operation.

Note that because $x_0 \notin \mathrm{Var}$, the graph obtained by an equivalent variable substitution will be a labeled complete graph. In addition, the operation evs is defined up to equivalence because the entries in a graph $\mathrm{evs}(A)$ can be various variables from Var.

Remark. Although a graph can be expressed in many ways as various combinations of variables by Definition 2.1, all of these expressions are equivalent in the sense of Definition 2.2. The definition proposed here for graph isomorphism is concerned only with graphs of the same set of labels (called a strict isomorphism in the literature). There is no harm in determining the isomorphism of any two graphs with different sets of labels because they can, if they are isomorphic, be relabeled into the same set of labels by an equivalent variable substitution. For two simple graphs, this can be done by replacing labels on edges in both graphs directly with the same variable, e.g., x, and then to test their isomorphism. In the present decision procedure GI, only the isomorphism of any two simple graphs is considered.

3 WL ALGORITHM AND WL STABLE GRAPHS

This section introduces the WL algorithm, WL stable graphs, and some of their properties. All conclusions in this section can be found in [32], but they are included and proved here for self-containment. It should be stressed that the WL algorithm adopted here is usually called the 2-WL algorithm in the literature.

The WL algorithm has appeared in many previous studies and is presented there mainly in the form of colored graphs. Although that presentation is elegant and succinct, it is not sufficient or convenient for the purpose of this study because colors on edges are neglected in that conceptualization. Information on edges, however, is dominant in the deductions of the results in this work.

The presentation of the WL algorithm here inherits the framework proposed by Weisfeiler and Lehman in [32]. The graphs are described in the form of matrices with independent variables as entries. This is why the current study uses a notion of graphs that deviates from the conventions in Section 2.

For convenience, $x \diamond y$ is used to denote uncommutative multiplication such that $x \diamond y = x' \diamond y'$ iff $x = x'$ and $y = y'$ for variables $x, y, x', y' \in \{x_0\} \cup \text{Var}$.

For a graph $G := (g_{ij})$ of order n, an entry $g_{ij}^{(2)}$ in matrix $G \diamond G := (g_{ij}^{(2)})$ is defined as

$$g_{ij}^{(2)} := \sum_{k=1}^{n} g_{ik} \diamond g_{kj} \tag{1}$$

for $i, j \in [n]$. The matrix $G \diamond G$ is called the diamond product of graph G. To perform an equivalent variable substitution to $G \diamond G$, a graph $G_1 = \text{evs}(G \diamond G)$ is obtained such that $G_1 \approx G \diamond G$ as matrices.

Intuitively, the element $g_{ik} \diamond g_{kj}$ records a 2-d-walk (i, k, j) in G. The element $g_{ij}^{(2)}$, however, records the collection of all possible 2-d-walks from i to j and is equivalent to the multiset $\{g_{ik} \diamond g_{kj} \mid k \in [n]\}$. Readers may notice that (i, i, j) and (i, j, j) are also 2-d-walks from i to j.

Now, several notions are ready to propose and then to use later. A graph G is a *WL stable graph* iff $\text{evs}(G \diamond G) \approx G$. A stable graph always refers to a WL stable graph in the context of this work. A graph $G := (g_{ij})$ of order n is said to be *converse equivalent* if $g_{uv} = g_{rs}$ iff $g_{vu} = g_{sr}$ for all $u, v, r, s \in [n]$ (a notion borrowed from [33]). A graph G *recognizes vertices* iff $g_{ii} \neq g_{uv}$ for all $i, u, v \in [n]$ and $u \neq v$.

With these notions and notations, the WL algorithm can be described as follows: Given a graph G of order n, perform an equivalent variable substitution only on the diagonal entries of G to obtain a graph G_1, such that G_1 recognizes vertices. It then computes $G_{t+1} := \text{evs}(G_t \diamond G_t)$ iteratively for $t \geq 1$ until a stable graph is obtained.

The following conclusions belong to Weisfeiler and Lehman [32]. We show them here for self-containment.

PROPOSITION 3.1 ([32]). *For a graph G, the following hold in the WL algorithm for all $t \geq 1$:*

(1) *Graph G_t recognizes vertices.*

(2) $G_t \geq G_{t+1}$.

(3) $\text{Aut}(G_t) = \text{Aut}(G_{t+1})$.

PROOF. To denote $G_t = (g_{ij})$ and $G_t \diamond G_t := (g_{ij}^{(2)})$ for $i, j \in [n]$. These will be shown one by one.

(1) To show that G_t recognizes vertices by induction on $t \geq 1$, note that G_1 recognizes vertices. Assuming that G_t recognizes vertices, the definition of the diamond product gives

$$g_{uu}^{(2)} = g_{uu} \diamond g_{uu} + \sum_{k \in [n] \setminus \{u\}} g_{uk} \diamond g_{ku}, \qquad g_{rs}^{(2)} = g_{rr} \diamond g_{rs} + g_{rs} \diamond g_{ss} + \sum_{k \in [n] \setminus \{r,s\}} g_{rk} \diamond g_{ks}.$$

Because G_t recognizes vertices, there is a term $g_{uu} \diamond g_{uu}$ in $g_{uu}^{(2)}$, but there is no such term in $g_{rs}^{(2)}$ for $r \neq s$. This means that $g_{uu}^{(2)} \neq g_{rs}^{(2)}$ for all $u, r, s \in [n]$ with $r \neq s$. By definition of e.v.s., $G_{t+1} = \text{evs}(G_t \diamond G_t)$ recognizes vertices. This shows that G_t recognizes vertices for all $t \geq 1$ by induction.

(2) If $g_{uv}^{(2)} = g_{rs}^{(2)}$ for $u, v, r, s \in [n]$, equivalently $\sum_{k=1}^{n} g_{uk} \diamond g_{kv} = \sum_{k=1}^{n} g_{rk} \diamond g_{ks}$ in $G_t \diamond G_t$. This will force $g_{uu} \diamond g_{uv} + g_{uv} \diamond g_{vv} = g_{rr} \diamond g_{rs} + g_{rs} \diamond g_{ss}$ by vertex recognizability of G_t. This in turn further forces $g_{uu} \diamond g_{uv} = g_{rr} \diamond g_{rs}$, again by vertex recognizability. Hence, $g_{uv} = g_{rs}$. This shows that $G_t \geq G_{t+1} = \text{evs}(G_t \diamond G_t)$ for all $t \geq 1$.

(3) Because $G_t \geq G_{t+1}$, it holds that $\text{Aut}(G_{t+1}) \subseteq \text{Aut}(G_t)$ for all $t \geq 1$ by Proposition 2.3. On the other hand, if $\sigma \in \text{Aut}(G_t)$, it then holds that $g_{ij} = g_{i^\sigma j^\sigma}$ for all $i, j \in [n]$. This gives

$$g_{ij}^{(2)} = \sum_{k=1}^{n} g_{ik} \diamond g_{kj} = \sum_{k=1}^{n} g_{i^\sigma k^\sigma} \diamond g_{k^\sigma j^\sigma} = \sum_{k=1}^{n} g_{i^\sigma k} \diamond g_{kj^\sigma} = g_{i^\sigma j^\sigma}^{(2)}, \tag{2}$$

because σ is a permutation on $[n]$. The equation (2) states that $\sigma \in \text{Aut}(G_{t+1})$ because $G_{t+1} = \text{evs}(G_t \diamond G_t)$. It hence holds that $\text{Aut}(G_t) \subseteq \text{Aut}(G_{t+1})$.

This finishes the proof. □

Because $G_t \geq G_{t+1}$ implies that $\dim(G_t) \leq \dim(G_{t+1}) \leq n^2$, there should be a $t_0 \geq 1$ such that $\dim(G_{t_0}) = \dim(G_{t_0+1})$, and hence $G_{t_0} \approx G_{t_0+1}$. The WL stable graph G_{t_0} is then called *the stable graph of G* and denoted as $\text{wl}(G)$, which is unique up to equivalence. The least such number t_0 is usually referred to as the number of iterations of the WL algorithm.

It is obvious that the number of iterations of the WL algorithm cannot exceed n^2 for a graph of order n. A lower bound $O(n)$ of the number of iterations in the WL algorithm was shown by Fürer [18]. An upper bound $O(n \log n)$ for the number of iterations was proved by Lichter, Ponomarenko, and Schweitzer [33]. If the computational complexity of the multiplication of two matrices of order n is assumed to be $O(n^3)$, then the following can be stated:

THEOREM 3.2 ([33]). *The computational complexity of a stable graph of order n is $O(n^4 \log n)$.*

Because G_1 is obtained by an equivalent variable substitution only to the labels on vertices of G, it is easy to see that $\text{Aut}(G_1) = \text{Aut}(G)$ in the WL algorithm. This means that $\text{Aut}(\text{wl}(G)) = \text{Aut}(G)$ by Proposition 3.1.

PROPOSITION 3.3 ([32]). *The following statements hold for two graphs G, H of order n:*

- $\text{Aut}(G) = \text{Aut}(\text{wl}(G))$ *for any graph G.*
- $G \geq H$ *implies* $\text{wl}(G) \geq \text{wl}(H)$.

The second result can be shown similarly to the previous Proposition. The properties in the next proposition play important roles in later discussions.

PROPOSITION 3.4 ([32]). *The followings hold for a stable graph $X := (x_{ij})$ of order n for all $u, v, r, s \in [n]$:*

(1) $x_{uv} = x_{rs}$ *implies* $x_{uu} = x_{rr}$ *and* $x_{vv} = x_{ss}$.
(2) $x_{uv} = x_{rs}$ *if and only if* $x_{vu} = x_{sr}$.
(3) $x_{uu} = x_{vv}$ *iff* $\{x_{uk} \mid k \in [n]\} \equiv \{x_{vk} \mid k \in [n]\}$ *and* $\{x_{ku} \mid k \in [n]\} \equiv \{x_{kv} \mid k \in [n]\}$.
(4) $u^\sigma = v$ *implies* $x_{uu} = x_{vv}$ *for all* $\sigma \in \text{Aut}(X)$.
(5) $\text{wl}(PGP^\top) \approx P\,\text{wl}(G)P^\top$ *for all graphs G and permutation matrices P.*

PROOF. Let $\mathrm{evs}(X \diamond X) := (z_{ij})$ with $i, j \in [n]$. The properties are demonstrated item by item.

(1) By the definition of a stable graph, $x_{uv} = x_{rs}$ implies that $z_{uv} = z_{rs}$. This in turn means that $\sum_{k=1}^{n} x_{uk} \diamond x_{kv} = \sum_{k=1}^{n} x_{rk} \diamond x_{ks}$. Because a stable graph recognizes vertices, this leads to $x_{uu} \diamond x_{uv} + x_{uv} \diamond x_{vv} = x_{rr} \diamond x_{rs} + x_{rs} \diamond x_{ss}$, which forces $x_{uu} \diamond x_{uv} = x_{rr} \diamond x_{rs}$ and $x_{uv} \diamond x_{vv} = x_{rs} \diamond x_{ss}$. Finally, $x_{uu} = x_{rr}$ and $x_{vv} = x_{ss}$ can be obtained by the definition of the diamond product.

(2) Because $x_{uk_1} \diamond x_{k_1 v} = x_{rk_2} \diamond x_{k_2 s} \Leftrightarrow x_{vk_1} \diamond x_{k_1 u} = x_{sk_2} \diamond x_{k_2 r}$ as recordings of 2-d-walks for $u, v, r, s, k_1, k_2 \in [n]$, by the stability of X,

$$x_{uv} = x_{rs} \quad \Longleftrightarrow \quad z_{uv} = z_{rs} \quad \Longleftrightarrow \quad \sum_{k=1}^{n} x_{uk} \diamond x_{kv} = \sum_{k=1}^{n} x_{rk} \diamond x_{ks}$$

$$\Longleftrightarrow \quad \sum_{k=1}^{n} x_{vk} \diamond x_{ku} = \sum_{k=1}^{n} x_{sk} \diamond x_{kr} \quad \Longleftrightarrow \quad z_{vu} = z_{sr} \quad \Longleftrightarrow \quad x_{vu} = x_{sr} \, .$$

Hence, the claim is true.

(3) If $x_{uu} = x_{vv}$ in the stable graph X, it holds that $z_{uu} = z_{vv}$, which means that $\sum_{k=1}^{n} x_{uk} \diamond x_{ku} = \sum_{k=1}^{n} x_{vk} \diamond x_{kv}$. This further implies that $\{ x_{uk} \mid k \in [n] \} \equiv \{ x_{vk} \mid k \in [n] \}$ and $\{ x_{ku} \mid k \in [n] \} \equiv \{ x_{kv} \mid k \in [n] \}$ by definition of the diamond product. The latter, in turn, implies $x_{uu} = x_{vv}$ from the vertex recognizability of X.

(4) If $u^{\sigma} = v$, then $x_{uu} = x_{u^{\sigma} u^{\sigma}} = x_{vv}$ for all $\sigma \in \mathrm{Aut}(X)$ by definition.

(5) The last conclusion is easy to show by arguments similar to those used for Proposition 3.1.

This finishes the proof.   □

In a graph G of order n, *a cell C* of G consists of all vertices with the same label. The set $C := \{C_1, \ldots, C_c\}$ of all cells forms a partition of $[n]$, which is called the cell partition of G. The cell partition of $\mathrm{wl}(G)$ is called *the stable partition* of graph G. Because the stable graph of a graph G is unique up to equivalence, the stable partition of a graph is well defined.

If x_{uv} is stated as *a connection* of u and v in the stable graph X, the first result of Proposition 3.1 indicates that two connections are identical only if they connect the vertices from the same pair of cells. The second result claims that a stable graph respects reverse equivalence. The third, together with the first, states that all connections of a vertex u and all connections of v are identical as multisets iff u and v are in the same cell, and disjoint otherwise.

Another observation is that the stable graph $\mathrm{wl}(G)$ of a simple graph G recognizes the edges of G, in the sense that the labels in $\mathrm{wl}(G)$ on edges of G do not overlap with labels on non-edges of G. This can be concluded from the facts that $G \geq \mathrm{wl}(G)$ and that the labels on edges and non-edges are different in G.

COROLLARY 3.5. *The stable graph $\mathrm{wl}(G) := (m_{ij})$ of a simple graph G recognizes the edges of G, and the vertices in the same cell of $\mathrm{wl}(G)$ have the same degree in G.*

Given a stable graph X of order n, one may perform a series of row-column transpositions, so that vertices with the same label are listed consecutively on the diagonal of the resulting graph. In other words, vertices in the same cell are listed consecutively in the stable graph. We *assume* from now on, without loss of generality from the last conclusion of Proposition 3.4, that a stable graph is always in this form, unless explicitly mentioned in context.

In this setting, the cell partition $\alpha = (\alpha_1, \ldots, \alpha_c)$ of stable graph $X := (x_{ij})$ is a sequential partition of $[n]$. This will induce a block partition $X = (X_{uv}) \, (u, v \in [c])$ in graph X, where each $X_{ij} = (x_{uv})$ is a block with $u \in \alpha_i$ and $v \in \alpha_j$.

The stable graph X is now said to be in the form of *cell block partition*. It is easy to obtain the following conclusions from Proposition 3.4:

COROLLARY 3.6. *For a stable graph $X = (X_{ij})$ $(i, j \in [c])$ in the form of a cell block partition, the following statements hold:*

(1) *If $\{u, v\} \not\equiv \{r, s\}$, then $\mathrm{Var}(X_{uv}) \cap \mathrm{Var}(X_{rs}) = \emptyset$ for all $u, v, r, s \in [c]$,*

(2) *For each block matrix $X_{ij} := (x_{uv})$ with $u \in \alpha_i$ and $v \in \alpha_j$ in X:*

$$\{x_{k\,v_1} \mid k \in \alpha_i\} \equiv \{x_{k\,v_2} \mid k \in \alpha_i\} \quad \text{for all } v_1, v_2 \in \alpha_j , \tag{3}$$

$$\{x_{u_1\,k} \mid k \in \alpha_j\} \equiv \{x_{u_2\,k} \mid k \in \alpha_j\} \quad \text{for all } u_1, u_2 \in \alpha_i . \tag{4}$$

A block satisfying both (3) and (4) is called *an equitable block*. To make this concept more intuitive, let us look at an example.

Example 3.7. Graph G is the adjacency matrix of a simple graph of order 10, whereas G_1 is the graph obtained after making it vertex-recognizable. Graph $G_{t+1} = \mathrm{evs}(G_t \diamond G_t)$ for $t = 1, 2, 3$. Graph G_4 is the stable graph of G, and X is a form of cell block partition of G_4. For readability, variables are used as entries only in G_1, and numbers are used for variables in the rest.

$$G = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix},$$

$$G_1 = \begin{pmatrix} x & y & x_0 & x_0 & y & y & x_0 & x_0 & x_0 & x_0 \\ y & x & y & x_0 & x_0 & x_0 & y & x_0 & x_0 & x_0 \\ x_0 & y & x & y & x_0 & x_0 & x_0 & x_0 & y & x_0 \\ x_0 & x_0 & y & x & y & x_0 & x_0 & y & x_0 & x_0 \\ y & x_0 & x_0 & y & x & x_0 & x_0 & x_0 & x_0 & y \\ y & x_0 & x_0 & x_0 & x_0 & x & x_0 & y & y & x_0 \\ x_0 & y & x_0 & x_0 & x_0 & x_0 & x & x_0 & y & y \\ x_0 & y & x_0 & x_0 & x_0 & x_0 & x & x_0 & y & y \\ x_0 & x_0 & x_0 & y & x_0 & y & x_0 & x & x_0 & y \\ x_0 & x_0 & y & x_0 & x_0 & y & y & x_0 & x & x_0 \\ x_0 & x_0 & x_0 & x_0 & y & x_0 & y & y & x_0 & x \end{pmatrix},$$

$$G_2 = \begin{pmatrix} 1 & 2 & 3 & 3 & 2 & 2 & 3 & 3 & 3 & 3 \\ 2 & 1 & 2 & 3 & 3 & 3 & 2 & 4 & 5 & 3 \\ 3 & 2 & 1 & 2 & 3 & 3 & 5 & 3 & 2 & 4 \\ 3 & 3 & 2 & 1 & 2 & 3 & 4 & 2 & 3 & 5 \\ 2 & 3 & 3 & 2 & 1 & 3 & 3 & 5 & 4 & 2 \\ 2 & 3 & 3 & 3 & 3 & 1 & 3 & 2 & 2 & 3 \\ 3 & 2 & 5 & 4 & 3 & 3 & 1 & 3 & 2 & 2 \\ 3 & 4 & 3 & 2 & 5 & 2 & 3 & 1 & 3 & 2 \\ 3 & 5 & 2 & 3 & 4 & 2 & 2 & 3 & 1 & 3 \\ 3 & 3 & 4 & 5 & 2 & 3 & 2 & 2 & 3 & 1 \end{pmatrix},$$

$$G_3 = \begin{pmatrix} 1 & 2 & 3 & 3 & 2 & 4 & 3 & 5 & 5 & 3 \\ 6 & 7 & 8 & 9 & 10 & 11 & 8 & 12 & 13 & 9 \\ 14 & 8 & 7 & 15 & 9 & 14 & 16 & 9 & 8 & 17 \\ 14 & 9 & 15 & 7 & 8 & 14 & 17 & 8 & 9 & 16 \\ 6 & 10 & 9 & 8 & 7 & 11 & 9 & 13 & 12 & 8 \\ 4 & 5 & 3 & 3 & 5 & 1 & 3 & 2 & 2 & 3 \\ 14 & 8 & 16 & 17 & 9 & 14 & 7 & 9 & 8 & 15 \\ 11 & 12 & 9 & 8 & 13 & 6 & 9 & 7 & 10 & 8 \\ 11 & 13 & 8 & 9 & 12 & 6 & 8 & 10 & 7 & 9 \\ 14 & 9 & 17 & 16 & 8 & 14 & 15 & 8 & 9 & 7 \end{pmatrix},$$

$$G_4 = \begin{pmatrix} 1 & 2 & 3 & 3 & 2 & 4 & 3 & 5 & 5 & 3 \\ 6 & 7 & 8 & 9 & 10 & 11 & 8 & 12 & 13 & 9 \\ 14 & 15 & 16 & 17 & 18 & 14 & 19 & 18 & 15 & 20 \\ 14 & 18 & 17 & 16 & 15 & 14 & 20 & 15 & 18 & 19 \\ 6 & 10 & 9 & 8 & 7 & 11 & 9 & 13 & 12 & 8 \\ 4 & 5 & 3 & 3 & 5 & 1 & 3 & 2 & 2 & 3 \\ 14 & 15 & 19 & 20 & 18 & 14 & 16 & 18 & 15 & 17 \\ 11 & 12 & 9 & 8 & 13 & 6 & 9 & 7 & 10 & 8 \\ 11 & 13 & 8 & 9 & 12 & 6 & 8 & 10 & 7 & 9 \\ 14 & 18 & 20 & 19 & 15 & 14 & 17 & 15 & 18 & 16 \end{pmatrix},$$

$$X = \begin{pmatrix} 1 & 4 & 2 & 2 & 5 & 5 & 3 & 3 & 3 & 3 \\ 4 & 1 & 5 & 5 & 2 & 2 & 3 & 3 & 3 & 3 \\ 6 & 11 & 7 & 10 & 12 & 13 & 8 & 9 & 8 & 9 \\ 6 & 11 & 10 & 7 & 13 & 12 & 9 & 8 & 9 & 8 \\ 11 & 6 & 12 & 13 & 7 & 10 & 9 & 8 & 9 & 8 \\ 11 & 6 & 13 & 12 & 10 & 7 & 8 & 9 & 8 & 9 \\ 14 & 14 & 15 & 18 & 18 & 15 & 16 & 17 & 19 & 20 \\ 14 & 14 & 18 & 15 & 15 & 18 & 17 & 16 & 20 & 19 \\ 14 & 14 & 15 & 18 & 18 & 15 & 19 & 20 & 16 & 17 \\ 14 & 14 & 18 & 15 & 15 & 18 & 20 & 19 & 17 & 16 \end{pmatrix}. \tag{5}$$

We now introduce the notion of similar stable graphs, which will be used later.

Definition 3.8 (Similarity of Stable Graphs). Let $X := (x_{ij})$ and $\bar{X} := (\bar{x}_{ij})$ of order n be two stable graphs with the same sequential cell partition $\alpha := (\alpha_1, \ldots, \alpha_c)$. If the following statements are satisfied, it can be said that X and $\bar{X}$ are similar, which is symbolized by $X \sim \bar{X}$.

- $\sum_{k=1}^n x_{ik} = \sum_{k=1}^n \bar{x}_{ik}$ and $\sum_{k=1}^n x_{kj} = \sum_{k=1}^n \bar{x}_{kj}$ for all $i, j \in [n]$.
- $x_{uv} = \bar{x}_{rs}$ iff $\sum_{k=1}^n x_{uk} \diamond x_{kv} = \sum_{k=1}^n \bar{x}_{rk} \diamond \bar{x}_{ks}$ for all $u, v, r, s \in [n]$.

Assume that $\alpha_i := [a_i, b_i]$ for $i \in [c]$. It can be stated that $\sum_{k=a_j}^{b_j} x_{rk} = \sum_{k=a_j}^{b_j} \bar{x}_{rk}$ and $\sum_{k=a_i}^{b_i} x_{ks} = \sum_{k=a_i}^{b_i} \bar{x}_{ks}$ for all $r \in \alpha_i$ and $s \in \alpha_j$ in two similar stable graphs X and $\bar{X}$ by Proposition 3.4 and Corollary 3.6. Moreover, $\mathrm{Var}(X_{ij}) = \mathrm{Var}(\bar{X}_{ij})$, $\mathrm{Var}(X) = \mathrm{Var}(\bar{X})$. Also, $x_{ii} = \bar{x}_{ii}$ by the second condition of similarity.

It is easy to establish the following conclusion by Proposition 3.4.

PROPOSITION 3.9. *In a stable graph X with sequential cell partition $\alpha = (\alpha_1, \ldots, \alpha_c)$, if $u, v \in \alpha_i$ for some $i \in [c]$, the graph $\bar{X}$ obtained after transposing vertex u with vertex v in X by a row-column transposition is then similar to X.*

Readers may prove that for a strongly regular graph G, after making it vertex-recognizable, the graph G_1 obtained is stable in the sense that $G_1 \approx \text{evs}(G_1 \diamond G_1)$. This means that the stable partition of a strongly regular graph is always a unit partition. This indicates that the stable partition of a graph G is not expected to be the automorphism partition of G in general, because there are many strongly regular graphs with even discrete automorphism partitions. Furthermore, one may easily obtain non-isomorphic similar stable graphs from strongly regular graphs. Which claims that similar stable graphs are not isomorphic in general.

We are going to show in Section 9, however, that similar stable graphs of binding graphs are isomorphic, and that the stable partitions of binding graphs are the automorphism partitions.

4   INDIVIDUALIZATION AND COMPOSITION IN A STABLE GRAPH

To individualize a vertex in a stable graph is to re-mark it with a label that has not appeared in the graph. The stable graph of an individualized graph will have the individualized vertex as a singleton cell. The stable partitions of individualized graphs will be explored in subsection 4.1. It will also be shown in subsection 4.2 that the remaining graph after deleting some cells from a stable graph is still a stable graph.

4.1   Individualization of One Vertex

To individualize a vertex u in a stable graph $X = (x_{ij})$ is to replace the label x_{uu} of vertex u by a variable in $\text{Var} \setminus \text{Var}(X)$. The graph obtained is called the individualized graph of u and is denoted as X_u. The stable partition of an individualized graph can be exploited as described below.

For a vertex u in the stable graph $X = (x_{ij})$ of order n, let $\beta := (\beta_1, \ldots, \beta_t)$ be the partition of $[n]$ such that $x_{ua} = x_{ub}$ and $x_{uk} \neq x_{ua}$ for all $a, b \in \beta_i$ and $k \in [n] \setminus \beta_i$. This partition is then called *the block partition* of u in X. It is straightforward that $[u, u]$ is some β_i in the block partition β of u by vertex recognizability.

For simplicity of presentation, the stable graph $\text{wl}(X_1)$ for $u = 1$ will be inspected. The conclusion shown will be valid for any vertex by the last result of Proposition 3.4.

Before individualizing vertex 1, some row-column permutations were made to $X := (x_{ij})$ with $i, j \in [n]$. Let the sequential cell partition of X be $\alpha = (\alpha_1, \ldots, \alpha_c)$, and let the cell partition form of X be $X := (\check{X}_{ij})$ with $i, j \in [c]$. From Corollary 3.6, it is known that $\text{Var}(\check{X}_{uv}) \cap \text{Var}(\check{X}_{rs}) = \emptyset$ if $\{u, v\} \neq \{r, s\}$ for all $u, v, r, s \in [c]$.

Now the first row of X will be fixed. For cell $\alpha_k := [a, b]$, if there are s distinct labels in the set $\{x_{1a}, \ldots, x_{1b}\}$, row-column transpositions are performed on X between the rows and columns in $[a, b]$ only, to obtain a graph such that the same labels of the first row from column a to b lie consecutively. As a result, the fragment $[a, b]$ is split into a sequential partition $[a_1, b_1], \ldots, [a_s, b_s]$, where each $[a_\ell, b_\ell]$ corresponds to a block of entries with identical variables in the first row of the resulting graph.

After permutations in this way for all cells α_k, a sequential partition $\beta = (\beta_1, \beta_2, \ldots, \beta_t)$ of $[n]$ is thus obtained as a refinement of α and is now the block partition of vertex 1. As an example, the block partition of vertex 1 is $([1, 1], [2, 2], [3, 4], [5, 6], [7, 10])$ in the stable graph X shown in (5).

Because a stable graph in the form of the block partition mentioned above can be obtained through row-column permutations, a stable graph X of this form is said to be in the form of block partition β with respect to vertex 1.

Let $X := (X_{ij})$ $(i, j \in [t])$ be in the form of block partition $\beta = (\beta_1, \ldots, \beta_t)$ with respect to vertex 1, which means that $X_{ij} = (x_{uv})$ for $u \in \beta_i, v \in \beta_j$.

The following statements hold now by the formation of X, vertex recognizability, and the reverse equivalence property:

FACT 1. *The following statements hold in $X := (x_{ij})$ with $i, j \in [n]$:*

(1) $\beta_1 = [1, 1]$ *and* $X_{11} = x_{11}$.
(2) $x_{1\,a_u} = \cdots = x_{1\,b_u}$ *in* $X_{1u} = (x_{1\,a_u}, \ldots, x_{1\,b_u})$ *for all* $\beta_u := [a_u, b_u]$ *and* $u \in [t]$.
(3) $\mathrm{Var}(X_{1i}) \cap \mathrm{Var}(X_{1j}) = \emptyset$ *and* $\mathrm{Var}(X_{i1}) \cap \mathrm{Var}(X_{j1}) = \emptyset$ *for all* $i, j \in [t]$ *and* $\beta_i \neq \beta_j$.
(4) *If* $X_{ii} := (x_{rs})$ *for* $r, s \in \beta_i$, *then* $x_{uu} = x_{vv}$ *for all* $u, v \in \beta_i$.

A more interesting property is the following:

FACT 2. *Each block X_{ij} is equatable for all $i, j \in [t]$.*

Proof of Fact 2. Block X_{1i} and X_{i1} are respectively equatable by Fact 1 for all $i \in [t]$. Let $X_{ij} = (x_{rs})$ for $r \in \beta_i$ and $s \in \beta_j$. We intend to show that $\sum_{k \in \beta_j} x_{ak} = \sum_{k \in \beta_j} x_{bk}$ for all $a, b \in \beta_i$.

Because $x_{a1} = x_{b1}$ in X_{i1} by Fact 1, $\sum_{k=1}^n x_{ak} \diamond x_{k1} = \sum_{k=1}^n x_{bk} \diamond x_{k1}$ by the stability of X. Equivalently,

$$\sum_{\ell \in [t]} \sum_{k \in \beta_\ell} x_{ak} \diamond x_{k1} = \sum_{\ell \in [t]} \sum_{k \in \beta_\ell} x_{bk} \diamond x_{k1} . \tag{6}$$

By the third result in Fact 1, x_{k1} are distinct for k from different β_ℓ. This will force $\sum_{k \in \beta_j} x_{ak} \diamond x_{k1} = \sum_{k \in \beta_j} x_{bk} \diamond x_{k1}$ for all $j \in [t]$. This leads to $\sum_{k \in \beta_j} x_{ak} = \sum_{k \in \beta_j} x_{bk}$ by definition of the diamond product.

Similarly, $\sum_{k \in \beta_i} x_{ka} = \sum_{k \in \beta_i} x_{kb}$ can be obtained for all $a, b \in \beta_j$. It is then shown that X_{ij} is equatable for all $i, j \in [t]$. This finishes the proof of Fact 2.

The next step is to individualize the vertex 1 for the stable graph X. Given $X := (X_{ij})$ in the form of the block partition $\beta = (\beta_1, \ldots, \beta_t)$ with respect to vertex 1, the entry $X_{11} = x_{11}$ in X is replaced with $\check{X}_{11} = x \in \mathrm{Var}\backslash\mathrm{Var}(X)$ to obtain graph X_1. We now construct two graphs $\hat{X}$ and Y, both of order n, based on X as follows.

The graph $\hat{X}$ is the same as X except the entries in the first row and the first column. For blocks X_{1j} and X_{j1} in X, perform equivalent variable substitutions with variables from $\mathrm{Var}\backslash\mathrm{Var}(X)$ to obtain $\hat{X}_{1j}$ and $\hat{X}_{j1}$, respectively, such that the followings are satisfied, for all $i, j, k \in [t]$ and $i \neq j$,

$$\hat{X}_{1j} \approx X_{1j}, \qquad \hat{X}_{j1} \approx X_{j1}, \qquad \mathrm{Var}(\hat{X}_{1i}) \cap \mathrm{Var}(\hat{X}_{1j}) = \mathrm{Var}(\hat{X}_{j1}) \cap \mathrm{Var}(\hat{X}_{i1}) = \mathrm{Var}(\hat{X}_{1j}) \cap \mathrm{Var}(\hat{X}_{k1}) = \emptyset . \tag{7}$$

The graph Y is obtained by performing equivalent variable substitutions with variables from Var on each block X_{ij} in X to obtain Y_{ij}, such that $\mathrm{Var}(Y_{uv}) \cap \mathrm{Var}(Y_{rs}) = \emptyset$ if $\{u, v\} \not\equiv \{r, s\}$ for all $u, v, r, s \in [t]$. The graphs $X_1, \hat{X}$ and Y are shown below.

$$X_1 = \begin{pmatrix} \check{X}_{11} & X_{12} & X_{13} & \cdots & X_{1t} \\ X_{21} & X_{22} & X_{23} & \cdots & X_{2t} \\ X_{31} & X_{32} & X_{33} & \cdots & X_{3t} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \check{X}_{t1} & \check{X}_{t2} & \check{X}_{t3} & \cdots & \check{X}_{tt} \end{pmatrix}, \qquad \hat{X} = \begin{pmatrix} \hat{X}_{11} & \hat{X}_{12} & \hat{X}_{13} & \cdots & \hat{X}_{1t} \\ \hat{X}_{21} & X_{22} & X_{23} & \cdots & X_{2t} \\ \hat{X}_{31} & X_{32} & X_{33} & \cdots & X_{3t} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \hat{X}_{t1} & X_{t2} & X_{t3} & \cdots & X_{tt} \end{pmatrix}, \qquad Y = \begin{pmatrix} Y_{11} & Y_{12} & Y_{13} & \cdots & Y_{1t} \\ Y_{21} & Y_{22} & Y_{23} & \cdots & Y_{2t} \\ Y_{31} & Y_{32} & Y_{33} & \cdots & Y_{3t} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ Y_{t1} & Y_{t2} & Y_{t3} & \cdots & Y_{tt} \end{pmatrix} . \tag{8}$$

It is known that $Y_{ij} \approx X_{ij}$ and that Y_{ij} is equatable by the facts above for all $i, j \in [t]$. The following fact can also be stated:

FACT 3. $\hat{X} = \mathrm{evs}(X_1 \diamond X_1)$ *and* $Y = \mathrm{evs}(\hat{X} \diamond \hat{X})$.

Proof of Fact 3. Remember that $X = (x_{uv})$ for $u, v \in [n]$. To show $\hat{X} = \text{evs}(X_1 \diamond X_1)$, let $X_1 \diamond X_1 := (g_{uv})$ for $u, v \in [n]$. For $u, v \in [2, n]$, we have $g_{uv} = \sum_{k \in [n]} x_{uk} \diamond x_{kv}$. Hence $g_{uv} = g_{rs}$ if and only if $x_{uv} = x_{rs}$ for all $u, v, r, s \in [2, n]$, by stability of X.

For $r \in [n]$, $g_{1r} = x \diamond x_{1r} + \sum_{k \in [2,n]} x_{1k} \diamond x_{kr}$ by definition. It is obvious that $g_{1r} \neq g_{uv}$ for all $u, v \in [2, n]$, because x never appears in g_{uv}. Moreover, if $g_{1r} = g_{1s}$, then $x \diamond x_{1r} + \sum_{k \in [2,n]} x_{1k} \diamond x_{kr} = x \diamond x_{1s} + \sum_{k \in [2,n]} x_{1k} \diamond x_{ks}$. It should hold that $x \diamond x_{1r} = x \diamond x_{1s}$ because $x \notin \text{Var}(X)$. Hence $x_{1r} = x_{1s}$ and both x_{1r} and x_{1s} are in the some block X_{1j} of X.

Similarly, $g_{r1} \neq g_{uv}$ for all $u, v \in [2, n]$. If $g_{r1} = g_{s1}$ then both x_{r1} and x_{s1} are in the some block X_{i1} of X. Also, $g_{1r} \neq g_{s1}$ can be similarly established for $r, s \in [2, n]$. These facts states that $X_1 \diamond X_1 \approx \hat{X}$ by the construction of $\hat{X}$ (cf. (7) and (8)). Hence $\hat{X} = \text{evs}(X_1 \diamond X_1)$.

We now turn to show $Y = \text{evs}(\hat{X} \diamond \hat{X})$. To denote $\hat{X} := (\hat{x}_{uv})$, $\hat{X} \diamond \hat{X} := (f_{uv})$ and $Y := (y_{uv})$ for $u, v \in [n]$. Notice that $\hat{x}_{uv} = x_{uv}$ for all $u, v \in [2, n]$. It then holds in $\hat{X} \diamond \hat{X}$ by definition that, for $u, v \in [2, n]$,

$$f_{11} = \hat{x}_{11} \diamond \hat{x}_{11} + \sum_{k=2}^{n} \hat{x}_{1k} \diamond \hat{x}_{k1}, \qquad\qquad f_{1v} = \hat{x}_{11} \diamond \hat{x}_{1v} + \sum_{k=2}^{n} \hat{x}_{1k} \diamond x_{kv}. \tag{9}$$

$$f_{u1} = \hat{x}_{u1} \diamond \hat{x}_{11} + \sum_{k=2}^{n} x_{uk} \diamond \hat{x}_{k1}, \qquad\qquad f_{uv} = \hat{x}_{u1} \diamond \hat{x}_{1v} + \sum_{k=2}^{n} x_{uk} \diamond x_{kv}. \tag{10}$$

It is obvious that $f_{11} \notin \{f_{uv}, f_{1v}, f_{u1} \mid u, v \in [2, n]\}$ from (9) and (10). We now show that $f_{uv} = f_{rs} \iff y_{uv} = y_{rs}$ for all $u, v, r, s \in [n]$.

"$\Rightarrow$": If $f_{uv} = f_{rs}$ for $u, v, r, s \in [n]$, it holds by (9) and (10) that $\hat{x}_{u1} \diamond \hat{x}_{1v} = \hat{x}_{r1} \diamond \hat{x}_{1s}$ because $\hat{x}_{u1}, \hat{x}_{1v}, \hat{x}_{r1}, \hat{x}_{1s} \notin \text{Var}(X)$. Hence, $\hat{x}_{u1} = \hat{x}_{r1}$ and $\hat{x}_{1v} = \hat{x}_{1s}$. These further imply that $x_{u1} = x_{r1}$ and $x_{1v} = x_{1s}$ by (7). It states that $u, r \in \beta_i$ and $v, s \in \beta_j$ for some $i, j \in [t]$. Both x_{uv} and x_{rs} are hence entries of the same block X_{ij} in X. That further implies $y_{u1} = y_{r1}$ and $y_{1v} = y_{1s}$ by construction of Y.

Moreover, because $\hat{X} \geq \hat{X} \diamond \hat{X}$ by Proposition 3.1, $f_{uv} = f_{rs}$ implies $\hat{x}_{uv} = \hat{x}_{rs}$. It is the same as $x_{uv} = x_{rs}$ for $u, v, r, s \in [2, n]$ in $\hat{X}$, which leads to $y_{uv} = y_{rs}$ because $Y_{ij} \approx X_{ij}$.

"$\Leftarrow$": $y_{uv} = y_{rs}$ only if y_{uv} and y_{rs} are in the same block Y_{ij} by the construction of Y. This means that $u, r \in \beta_i$ and $v, s \in \beta_j$ for some $i, j \in [t]$. These statements further imply that $x_{uv} = x_{rs}$ in X_{ij}, and that $x_{u1} = x_{r1}$ and $x_{1v} = x_{1s}$ in X. Hence $\hat{x}_{u1} = \hat{x}_{r1}$ and $\hat{x}_{1v} = \hat{x}_{1s}$ in $\hat{X}$. By the stability of X, equations (9) and (10), it is possible to obtain: for $u, v \in [2, n]$,

$$f_{uv} = \hat{x}_{u1} \diamond \hat{x}_{1v} + \sum_{k=2}^{n} x_{uk} \diamond x_{kv} = \hat{x}_{u1} \diamond \hat{x}_{1v} - x_{u1} \diamond x_{1v} + \sum_{k=1}^{n} x_{uk} \diamond x_{kv} \tag{11}$$

$$= \hat{x}_{r1} \diamond \hat{x}_{1s} - x_{r1} \diamond x_{1s} + \sum_{k=1}^{n} x_{rk} \diamond x_{ks} = \hat{x}_{r1} \diamond \hat{x}_{1s} + \sum_{k=2}^{n} x_{rk} \diamond x_{ks} = f_{rs}. \tag{12}$$

$$f_{1v} = \hat{x}_{11} \diamond \hat{x}_{1v} + \sum_{k=2}^{n} \hat{x}_{1k} \diamond x_{kv} = \hat{x}_{11} \diamond \hat{x}_{1v} + \sum_{\substack{k \in \beta_\ell \\ \ell \in [2, t]}} \hat{x}_{1k} \diamond x_{kv} \tag{13}$$

$$\overset{*}{=} \hat{x}_{11} \diamond \hat{x}_{1s} + \sum_{\substack{k \in \beta_\ell \\ \ell \in [2, t]}} \hat{x}_{1k} \diamond x_{ks} = \hat{x}_{11} \diamond \hat{x}_{1s} + \sum_{k=2}^{n} \hat{x}_{1k} \diamond x_{ks} = f_{1s}. \tag{14}$$

The equation "$\overset{*}{=}$" in (14) is by $\hat{x}_{1v} = \hat{x}_{1s}$, and $\sum_{k \in \beta_\ell} \hat{x}_{1k} \diamond x_{kv} = \sum_{k \in \beta_\ell} \hat{x}_{1k} \diamond x_{ks}$ for each $\ell \in [2, t]$. The latter is true since all entries $\hat{x}_{1k}$ in the block $\hat{X}_{1\ell}$ are identical, and $\{ x_{kv} \mid k \in \beta_\ell \} \equiv \{ x_{ks} \mid k \in \beta_\ell \}$ by equatability of block $X_{\ell j}$ according to Fact 2 . Similarly, it can be obtained that $f_{u1} = f_{r1}$.

This finishes the proof of Fact 3.

FACT 4. *If $\tilde{Y} = \text{evs}(Y \diamond Y)$, then $\tilde{Y} \approx \text{wl}(X_1)$.*

Proof of Fact 4. We need only to show that $\tilde{Y}$ is stable, and the conclusion is followed by Fact 3.

Let $\tilde{Y} = \text{evs}(Y \diamond Y) := (\tilde{y}_{uv})$ and $\tilde{Y} \diamond \tilde{Y} := (z_{uv})$ for $u, v \in [n]$. By the construction of Y, it is known that, for all $u, v, r, s \in [n]$,

$$\tilde{y}_{uv} = \tilde{y}_{rs} \iff \sum_{k \in [n]} y_{uk} \diamond y_{kv} = \sum_{k \in [n]} y_{rk} \diamond y_{ks}$$

$$\iff \sum_{\ell \in [t], k \in \beta_\ell} y_{uk} \diamond y_{kv} = \sum_{\ell \in [t], k \in \beta_\ell} y_{rk} \diamond y_{ks}$$

$$\iff \sum_{k \in \beta_\ell} y_{uk} \diamond y_{kv} = \sum_{k \in \beta_\ell} y_{rk} \diamond y_{ks}, \quad \text{for each } \ell \in [t]$$

$$\iff \sum_{k \in \beta_\ell} x_{uk} \diamond x_{kv} = \sum_{k \in \beta_\ell} x_{rk} \diamond x_{ks}, \quad \text{for each } \ell \in [t] . \tag{15}$$

The last equivalence above is true because of $Y_{uv} \approx X_{uv}$ (cf. Fact 3). It has been determined that for all $u, v, k_1, k_2 \in [n]$,

$$\tilde{y}_{uk_1} \diamond \tilde{y}_{k_1 v} = \tilde{y}_{rk_2} \diamond \tilde{y}_{k_2 s}$$

$$\iff \left(\sum_{\ell \in [n]} y_{u\ell} \diamond y_{\ell k_1} \right) \diamond \left(\sum_{\ell \in [n]} y_{k_1 \ell} \diamond y_{\ell v} \right) = \left(\sum_{\ell \in [n]} y_{r\ell} \diamond y_{\ell k_2} \right) \diamond \left(\sum_{\ell \in [n]} y_{k_2 \ell} \diamond y_{\ell s} \right) \tag{16}$$

$$\iff \left(\sum_{\substack{\ell \in \beta_i \\ i \in [t]}} y_{u\ell} \diamond y_{\ell k_1} \right) \diamond \left(\sum_{\substack{\ell \in \beta_i \\ i \in [t]}} y_{k_1 \ell} \diamond y_{\ell v} \right) = \left(\sum_{\substack{\ell \in \beta_i \\ i \in [t]}} y_{r\ell} \diamond y_{\ell k_2} \right) \diamond \left(\sum_{\substack{\ell \in \beta_i \\ i \in [t]}} y_{k_2 \ell} \diamond y_{\ell s} \right) \tag{17}$$

$$\iff \sum_{\substack{\ell \in \beta_i \\ i \in [t]}} (x_{u\ell} \diamond x_{\ell k_1}) \diamond \sum_{\substack{\ell \in \beta_i \\ i \in [t]}} (x_{k_1 \ell} \diamond x_{\ell v}) = \sum_{\substack{\ell \in \beta_i \\ i \in [t]}} (x_{r\ell} \diamond x_{\ell k_2}) \diamond \sum_{\substack{\ell \in \beta_i \\ i \in [t]}} (x_{k_2 \ell} \diamond x_{\ell s}) \tag{18}$$

$$\iff y_{uk_1} \diamond y_{k_1 v} = y_{rk_2} \diamond y_{k_2 s} . \tag{19}$$

Where (18) is obtained by (15), whereas (19) is obtained as in the proof of Fact 3. It is thus determined that

$$z_{uv} = z_{rs} \iff \sum_{k=1}^{n} \tilde{y}_{uk} \diamond \tilde{y}_{kv} = \sum_{k=1}^{n} \tilde{y}_{rk} \diamond \tilde{y}_{ks}$$

$$\iff \sum_{k=1}^{n} y_{uk} \diamond y_{kv} = \sum_{k=1}^{n} y_{rk} \diamond y_{ks} \iff \tilde{y}_{uv} = \tilde{y}_{rs} . \tag{20}$$

Which states that $\tilde{Y} \diamond \tilde{Y} \approx \tilde{Y}$. This shows that $\tilde{Y}$ is stable, and hence the claim of Fact 4. is upheld.

FACT 5. *If $\tilde{Y} = \text{evs}(Y \diamond Y) := (\tilde{y}_{ij})$, then $\tilde{y}_{ii} = \tilde{y}_{jj}$ if and only if $i, j \in \beta_\ell$ for some $\ell \in [t]$.*

Proof of Fact 5. By the construction of $Y := (y_{ij})$, $y_{ii} = y_{jj}$ only if $i, j \in \beta_\ell$ for some $\ell \in [t]$. Because $Y \geq \tilde{Y}$, that means $\tilde{y}_{ii} = \tilde{y}_{jj}$ only if $i, j \in \beta_\ell$ for some $\ell \in [t]$.

On the other hand, for each $\ell \in [t]$, if $i, j \in \beta_\ell$, one can obtain that $\sum_{k=1}^n y_{ik} = \sum_{k=1}^n y_{jk}$ and $\sum_{k=1}^n y_{ki} = \sum_{k=1}^n y_{kj}$, and Y respects reverse equivalence by the construction of Y. It has therefore been determined that $\sum_{k=1}^n y_{ik} \diamond y_{ki} = \sum_{k=1}^n y_{jk} \diamond y_{kj}$ for $i, j \in \beta_\ell$. This shows that $\tilde{y}_{ii} = \tilde{y}_{jj}$ if $i, j \in \beta_\ell$ for all $\ell \in [t]$. This completes the proof of Fact 5.

Combine the Fact 3-5, we obtain that the stable partition of X_1 is β. Which supports the following claim.

THEOREM 4.1 (INDIVIDUALIZATION THEOREM). *The stable partition of an individualized graph is the block partition of the individualized vertex in a stable graph.*

In fact, a stable graph after individualizing a vertex from X is equivalent to X itself, if the individualized vertex is in a singleton cell of X. It is not hard to obtain the following claim from the individualization procedure described above.

COROLLARY 4.2. *Two similar stable graphs can be obtained after individualizing the vertex* 1 *from each of two similar stable graphs.*

4.2 The Composition Theorem

The following property states that several cells of a stable graph together with the connections between them compose a stable graph. We call this the Composition Theorem.

THEOREM 4.3 (COMPOSITION THEOREM). *In a stable graph, a number of cells together with the connections between them compose a stable graph.*

PROOF. Assume that $X := (x_{ij})$ is a stable graph of order n with sequential cell partition $\alpha = (\alpha_1, \ldots, \alpha_c)$. Let $\alpha_c = [r + 1, n]$ be the last cell of X. Denote by $\bar{X} := (x_{ij})$ $(i, j \in [r])$ the graph after removing the last $n - r$ rows and the last $n - r$ columns from X. By Corollary 3.6, $x_{ik}, x_{ki} \notin \mathrm{Var}(\bar{X})$ for $k \in [r + 1, n]$ and $i \in [n]$. It will be shown here that $\bar{X}$ is a stable graph.

Denote $\mathrm{evs}(X \diamond X) := (y_{ij})$ and $\mathrm{evs}(\bar{X} \diamond \bar{X}) := (z_{ij})$. We want to show that $\bar{X} \approx \mathrm{evs}(\bar{X} \diamond \bar{X})$. Because $\bar{X} \geq \mathrm{evs}(\bar{X} \diamond \bar{X})$ by Proposition 3.1, we need only show that $\mathrm{evs}(\bar{X} \diamond \bar{X}) \geq \bar{X}$.

For $u, v, s, t \in [r]$, if $x_{uv} = x_{st}$ in $\bar{X}$, then $x_{uv} = x_{st}$ also in X. The stability of X implies that $y_{uv} = y_{st}$, which is equivalent to $\sum_{k=1}^n x_{uk} \diamond x_{kv} = \sum_{k=1}^n x_{sk} \diamond x_{kt}$. This further implies that

$$\sum_{k=1}^r x_{uk} \diamond x_{kv} = \sum_{k=1}^r x_{sk} \diamond x_{kt} \quad \text{and} \quad \sum_{k=r+1}^n x_{uk} \diamond x_{kv} = \sum_{k=r+1}^n x_{sk} \diamond x_{kt}$$

because all $x_{uk}, x_{sk} \notin \mathrm{Var}(\bar{X})$ for $k \in [r + 1, n]$. The first equation above implies that $z_{uv} = z_{st}$. We hence show that $\mathrm{evs}(\bar{X} \diamond \bar{X}) \geq \bar{X}$, supporting the claim that $\bar{X}$ is stable.

We have shown in fact that if a cell and its connections to other cells are removed, the remaining graph is stable by Proposition 3.4. The conclusion of the proposition can be obtained by iteratively removing target cells one by one.   □

It is easy to obtain the following:

COROLLARY 4.4. *The same collections of cells from two similar stable graphs, together with their connections, compose two similar stable graphs.*

5   BINDING GRAPHS AND THEIR PROPERTIES

This section proposes the notion of binding graphs and shows some of their graph isomorphism properties.

Intuitively, given a simple graph G of order n, a graph of order $n_1 := n(n+1)/2$ based on G can be constructed by adding $n(n-1)/2$ new vertices, such that each new added vertex will connect only one unique pair of vertices from G. The graph is uniquely determined by G, modulo the names of the added vertices. The formal definition can be stated as follows.

Definition 5.1 (Binding graphs). A simple graph H over $[n_1]$ with $n_1 := n(n+1)/2$ for some $n > 1$ is called *a binding graph* if, for each pair of vertices $u, v \in [n]$, there exists unique $p \in [n+1, n_1]$ of degree 2 adjacent to both u and v. We write $p := u \wedge v$ and say that vertex p binds vertices u and v. Vertex p is called a binding vertex, and both (u, p) and (v, p) are called binding edges.

Formally, a binding graph $H := (h_{ij})$ is a simple graph with $\text{Var}(H) = \{x_0, x\}$ of order n_1 for some $n > 1$, where $x \in \text{Var}$ is some variable. It holds that for each $p \in [n+1, n_1]$, there exists $u, v \in [n]$ with $u \neq v$ such that $h_{ip} = h_{pi} = x$ for $i \in \{u, v\}$ and $h_{jp} = h_{pj} = x_0$ for all $j \in [n_1] \setminus \{u, v\}$. Moreover, $u \wedge v \neq r \wedge s$ if $\{u, v\} \neq \{r, s\}$ for all $u, v, r, s \in [n]$ with $u \neq v, r \neq s$. The induced graph G of H with vertices $[n]$ is called the *basic graph* of H, and H is *a binding graph* of G. The vertices and edges in G are called *basic vertices* and *basic edges* respectively.

The binding graph of a simple graph G of order n is unique, up to the renaming of binding vertices. The binding graph of G is denoted as $[G]$ in context. It is easy to see from the definition that a binding graph $[G]$ can be efficiently constructed given a simple graph G as the basic graph.

It is assumed in the definition that the first n vertices of binding graphs are basic vertices. This is done for simplicity of presentation in context and is not necessary in general. Moreover, binding graphs can be defined for general graphs as well as simple graphs. However, the definition given here is enough for our purpose. Some properties of binding graphs are established below.

THEOREM 5.2. *For two binding graphs $[G]$ and $[H]$ of order $n_1 := n(n+1)/2$ with $n > 3$, the following properties hold if* $\text{Var}([G]) = \text{Var}([H])$:

(1) *Basic vertices will never share the same orbit with binding vertices in* $\text{Aut}([G])$.
(2) $G \cong H$ *if and only if* $[G] \cong [H]$.
(3) $\text{Aut}([G]) \cong \text{Aut}(G)$.

PROOF. Let graphs $[G] = (g_{ij})$ and $[H] := (h_{ij})$ exist for $i, j \in [n_1]$, both with labels x on edges and with labels x_0 on non-edges and vertices.

(1) For $n > 3$, a basic vertex in $[G]$ has $n - 1$ binding vertices and hence a degree of not less than $n - 1 > 2$, whereas a binding vertex in $[G]$ has degree 2. This indicates that a basic vertex cannot be sent to a binding vertex by any automorphism in $\text{Aut}([G])$. In other words, a basic vertex cannot be in the same orbit as a binding vertex in this case.

(2) "$\Rightarrow$": Let σ be a permutation on $[n]$ and an isomorphism from G to H. This can be extended to a mapping τ on $[n_1]$ as follows: For any $w \in [n_1]$, let

$$w^\tau := \begin{cases} w^\sigma, & \text{if } w \in [n], \\ u^\sigma \wedge v^\sigma, & \text{if } w \in [n+1, n_1] \text{ and } w := u \wedge v \text{ in } [G] \text{ for } u, v \in [n]. \end{cases} \tag{21}$$

Because σ is a permutation on $[n]$, the vertices $u^\sigma \in [n]$ and $v^\sigma \in [n]$ in $[H]$ are respectively unique for any $u, v \in [n]$. By the uniqueness of a binding vertex for a pair of vertices u^σ, v^σ in $[H]$, the mapping τ is hence well-defined and is a bijection on $[n_1]$ by Definition 5.1.

To show that τ is an isomorphism from $[G]$ to $[H]$, $g_{ij} = h_{i^\tau j^\tau}$ is shown for all $i, j \in [n_1]$, case by case, as follows.

- If $i, j \in [n]$, it is known, by (21), that $[n]^\tau = [n]^\sigma = [n]$ and $g_{ij} = h_{i^\sigma j^\sigma} = h_{i^\tau j^\tau}$ for all $i, j \in [n]$, because σ is an isomorphism from G to H.

- If $i \in [n]$ and $j \in [n + 1, n_1]$ with $j = i \dot\wedge u$ in $[G]$ for some $u \in [n]$, then $j^\tau = (i \dot\wedge u)^\tau = i^\sigma \dot\wedge u^\sigma$. That is, vertex j^τ is the binding vertex of vertices $i^\tau = i^\sigma$ and u^σ. It then holds that $h_{i^\tau j^\tau} = x$ by Definition 5.1. We get $g_{ij} = x = h_{i^\sigma \, i^\sigma \dot\wedge u^\sigma} = h_{i^\tau j^\tau}$.

- If $i \in [n]$ and $j \in [n + 1, n_1]$ with $j = u \dot\wedge v$ in $[G]$ for some $u, v \in [n]$ and $u \neq i \neq v$, then $u^\sigma \neq i^\sigma \neq v^\sigma$ because σ is a permutation, which means that $u^\sigma \dot\wedge v^\sigma$ is not a binding vertex of basic vertex i^σ in $[H]$. This leads to $g_{ij} = x_0 = h_{i^\sigma \, u^\sigma \dot\wedge v^\sigma} = h_{i^\tau j^\tau}$ by Definition 5.1 and (21).

- If $i, j \in [n + 1, n_1]$, then $g_{ij} = x_0 = h_{i^\tau j^\tau}$ by Definition 5.1, because $i^\tau, j^\tau \in [n + 1, n_1]$ in this case.

The combination of all the above cases supports the claim that τ is an isomorphism from $[G]$ to $[H]$.

"$\Leftarrow$": On the other hand, let $\tau : [G] \to [H]$ be an isomorphism. By the result in the first item, τ will send a basic vertex in $[G]$ to a basic vertex in $[H]$. The restriction $\tau|_{[n]}$ of τ to basic vertices $[n]$ is then an isomorphism from G to H, which shows that $G \cong H$ in this case.

(3) For a mapping τ in $\mathrm{Aut}([G])$, arguments similar to those in "$\Leftarrow$" above will show that the restriction $\tau|_{[n]}$ of τ to $[n]$ is an automorphism in $\mathrm{Aut}(G)$. To define a mapping ξ from $\mathrm{Aut}([G])$ to $\mathrm{Aut}(G)$ such that $\xi : \tau \mapsto \tau|_{[n]}$. This mapping is then well defined.

- The mapping ξ is surjective: For any $\sigma \in \mathrm{Aut}(G)$, to define τ similarly to (21) as follows.

$$w^\tau := \begin{cases} w^\sigma, & \text{if } w \in [n], \\ u^\sigma \dot\wedge v^\sigma, & \text{if } w \in [n + 1, n_1] \text{ and } w := u \dot\wedge v \text{ in } [G] \text{ for } u, v \in [n].\end{cases} \tag{22}$$

With arguments similar to those for "$\Rightarrow$" above, it can be obtained that $\tau \in \mathrm{Aut}([G])$ and $\sigma = \tau|_{[n]}$.

- The mapping ξ is injective: If $\tau_1, \tau_2 \in \mathrm{Aut}([G])$ with $\tau_1|_{[n]} = \tau_2|_{[n]} := \sigma \in \mathrm{Aut}(G)$, then $u^{\tau_1} = u^{\tau_2} = u^\sigma$ for all $u \in [n]$.

  For each binding vertex $p := u \dot\wedge v$ in $[G]$ for $u, v \in [n]$, it holds that $p^{\tau_1} = (u \dot\wedge v)^{\tau_1} = u^{\tau_1} \dot\wedge v^{\tau_1} = u^\sigma \dot\wedge v^\sigma$ by the uniqueness of the binding vertex for each pair of $u, v \in [n]$. Similarly, it can be obtained that $p^{\tau_2} = u^\sigma \dot\wedge v^\sigma$. This shows that $p^{\tau_1} = p^{\tau_2}$ for each binding vertex in $[n + 1, n_1]$, which in turn establishes that $\tau_1 = \tau_2$.

The mapping ξ also preserves group operations by the first conclusion of the Theorem. Hence, ξ is an isomorphism from $\mathrm{Aut}([G])$ to $\mathrm{Aut}(G)$.

This finishes the proof.   □

The conclusions of the Theorem indicate that the class of binding graphs is graph-isomorphism complete. The proof of the Theorem supports the following conclusion by Corollary 3.5.

COROLLARY 5.3. *For a binding graph $[G]$ of order $n_1 = n(n + 1)/2$ with $n > 3$, the basic vertices will never share the same cell with any binding vertices in the stable partition.*

A cell consisting of only basic vertices will be referred to as *a basic cell*, while a cell consisting of only binding vertices is referred to as *a binding cell*.

The following example addresses the cases of binding graphs and their stable graphs with bais graphs of order 2 and 3. For a basic graph of order 1, its binding graph is undefined, or the binding graph is simply itself if one likes.

Example 5.4. The followings are all binding graphs and their stable graphs for basic graphs of order 2 and 3. We still use numbers for variables in graphs. It is easy to verify that all the stable partitions of the binding graphs listed are automorphism partitions in these cases.

One may find in the following that the stable partitions of $[H_1]$ and $[G_0]$ are unit partitions. These are, in fact, the only cases that basic vertices and binding vertices can be in the same cell of stable partitions for binding graphs (cf. Corollary 5.3).

(1) The binding graphs and their stable graphs of basic graphs with 2 vertices are listed below. Because binding graphs are simple graphs, in total there are 2 non-isomorphic binding graphs in this case. Graph H_0 is the empty graph of order 2; graph H_1 is the graph of one edge.

$$H_0 := \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \qquad [H_0] := \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \qquad \mathtt{wl}([H_0]) := \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \\ 4 & 4 & 5 \end{pmatrix}. \tag{23}$$

$$H_1 := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad [H_1] := \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \qquad \mathtt{wl}([H_1]) := \begin{pmatrix} 1 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 1 \end{pmatrix}. \tag{24}$$

(2) The binding graphs and their stable graphs of basic graphs with 3 vertices are listed below. There are 4 non-isomorphic binding graphs in total in this case. Graph G_0 is the empty graph of order 3; graph G_1 is the graph consisting of a single vertex and an edge; graph G_2 consists of two edges incident to a common vertex; and graph K_3 is the complete graph of order 3.

$$G_0 := \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad [G_0] := \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}, \qquad \mathtt{wl}([G_0]) := \begin{pmatrix} 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 1 & 2 & 3 & 4 & 3 \\ 2 & 2 & 1 & 4 & 3 & 3 \\ 3 & 3 & 4 & 1 & 2 & 2 \\ 3 & 4 & 3 & 2 & 1 & 2 \\ 4 & 3 & 3 & 2 & 2 & 1 \end{pmatrix}. \tag{25}$$

$$G_1 := \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad [G_1] := \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}, \qquad \mathtt{wl}([G_1]) := \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 3 & 4 & 6 & 5 \\ 7 & 7 & 8 & 9 & 10 & 10 \\ 11 & 11 & 12 & 13 & 14 & 14 \\ 15 & 16 & 17 & 18 & 19 & 20 \\ 16 & 15 & 17 & 18 & 20 & 19 \end{pmatrix}. \tag{26}$$

$$G_2 := \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \qquad [G_2] := \begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}, \qquad \mathtt{wl}([G_2]) := \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 3 & 4 & 6 & 5 \\ 7 & 7 & 8 & 9 & 10 & 10 \\ 11 & 11 & 12 & 13 & 14 & 14 \\ 15 & 16 & 17 & 18 & 19 & 20 \\ 16 & 15 & 17 & 18 & 20 & 19 \end{pmatrix}. \tag{27}$$

$$K_3 := \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \qquad [K_3] := \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}, \qquad \mathtt{wl}([K_3]) := \begin{pmatrix} 1 & 2 & 2 & 3 & 3 & 4 \\ 2 & 1 & 2 & 3 & 4 & 3 \\ 2 & 2 & 1 & 4 & 3 & 3 \\ 5 & 5 & 6 & 7 & 8 & 8 \\ 5 & 6 & 5 & 8 & 7 & 8 \\ 6 & 5 & 5 & 8 & 8 & 7 \end{pmatrix}. \tag{28}$$

Readers may find that stable graphs $\mathtt{wl}([G_1])$ and $\mathtt{wl}([G_2])$ are isomorphic, but $G_1 \not\cong G_2$. This is the case because the labels are not coordinated during stabilization of the two graphs. This further indicates that the isomorphism of stable graphs alone cannot determine the isomorphism of the original graphs, but that their automorphism groups are isomorphic.

6 SOME PROPERTIES OF STABLE GRAPHS OF BINDING GRAPHS

In this section, some properties of the stable graphs of binding graphs are exploited for the proof of our main result. In a binding graph $[G]$, there are two kinds of edges: basic edges and binding edges. The next result shows that, in general, $\mathtt{wl}([G])$ recognizes the basic and binding edges of $[G]$ respectively, in the sense that the labels in $\mathtt{wl}([G])$ on basic edges of $[G]$ do not overlap with the labels on binding edges (cf. Example 5.4).

LEMMA 6.1. *For a binding graph $[G]$ of order $n_1 := n(n+1)/2$, the stable graph $\mathtt{wl}([G]) := (m_{ij})$ has the following properties. Let $p = u \wedge v, q = r \wedge s$ for $u, v, r, s \in [n]$ and (u, v) be an edge in $[G]$.*

(1) *If (r, s) is a non-edge in $[G]$, then $\{m_{pu}, m_{up}, m_{pv}, m_{vp}\} \cap \{m_{qr}, m_{rq}, m_{qs}, m_{sq}\} = \emptyset$.*

(2) *If $n > 2$, then $m_{uv} \notin \{m_{qr}, m_{rq}, m_{qs}, m_{sq}\}$ for all $u, v, r, s \in [n]$.*

PROOF. Because $[G]$ is a simple graph, it can be assumed that the labels on its edges are x. To make $[G]$ vertex-recognizable, a graph $G_1 := (g_{ij})$ is obtained such that $g_{ii} = y$ and $g_{ij} = x$ if (i, j) is an edge and $g_{ij} = x_0$ if (i, j) is a non-edge in $[G]$, for all $i, j \in [n_1]$. Let us denote $G_1 \diamond G_1 := (g_{ij}^{(2)})$ and $G_2 = \mathtt{evs}(G_1 \diamond G_1) := (z_{ij})$ for $i, j \in [n_1]$.

(1) Let $p = u \wedge v, q = r \wedge s$ for $u, v, r, s \in [n]$ and (u, v) be an edge and (r, s) a non-edge in $[G]$. This means that $g_{pk} = g_{q\ell} = x_0$ for all $k \in [n_1] \backslash \{u, v, p\}, \ell \in [n_1] \backslash \{r, s, q\}$, and that

$$g_{uu} = g_{rr} = g_{pp} = g_{qq} = y, \quad g_{vu} = g_{pu} = g_{pv} = g_{qr} = g_{qs} = x, \quad g_{rs} = g_{sr} = x_0 \ .$$

It can thus be determined that

$$g_{pu}^{(2)} = \sum_{k \in [n_1]} g_{pk} \diamond g_{ku} = g_{pu} \diamond g_{uu} + g_{pv} \diamond g_{vu} + g_{pp} \diamond g_{pu} + \sum_{k \in [n_1] \backslash \{u,v,p\}} g_{pk} \diamond g_{ku}$$

$$= x \diamond y + x \diamond x + y \diamond x + \sum_{k \in [n_1] \backslash \{u,v,p\}} x_0 \diamond g_{ku} , \tag{29}$$

$$g_{qr}^{(2)} = \sum_{k \in [n_1]} g_{qk} \diamond g_{kr} = g_{qr} \diamond g_{rr} + g_{qs} \diamond g_{sr} + g_{qq} \diamond g_{qr} + \sum_{k \in [n_1] \backslash \{r,s,q\}} g_{qk} \diamond g_{kr}$$

$$= x \diamond y + x \diamond x_0 + y \diamond x + \sum_{k \in [n_1] \backslash \{r,s,q\}} x_0 \diamond g_{ks} \ . \tag{30}$$

It is easy to see from (29) and (30) that $g_{pu}^{(2)} \neq g_{qr}^{(2)}$ because there is no term $x \diamond x$ in $g_{qr}^{(2)}$. Similar arguments will show that $g_{pu}^{(2)} \notin \{g_{qr}^{(2)}, g_{rq}^{(2)}, g_{qs}^{(2)}, g_{sq}^{(2)}\}$, and furthermore that $\{g_{pu}^{(2)}, g_{up}^{(2)}, g_{pv}^{(2)}, g_{vp}^{(2)}\} \cap \{g_{qr}^{(2)}, g_{rq}^{(2)}, g_{qs}^{(2)}, g_{sq}^{(2)}\} = \emptyset$. This is equivalent to, in G_2, $\{z_{pu}, z_{up}, z_{pv}, z_{vp}\} \cap \{z_{qr}, z_{rq}, z_{qs}, z_{sq}\} = \emptyset$. Because $G_2 \succeq \mathtt{wl}([G])$, it can thus be obtained that, in $\mathtt{wl}([G])$, $\{m_{pu}, m_{up}, m_{pv}, m_{vp}\} \cap \{m_{qr}, m_{rq}, m_{qs}, m_{sq}\} = \emptyset$.

(2) Set $U := \{u \wedge k \mid k \in [n] \backslash \{u\}\}$ as the set of u's binding vertices in G_1. It holds that $g_{up} = x$ for all $p \in U$. $g_{uv} = x$ is now true. With arguments similar to those in the last setting, it can be shown that

$$g_{uv}^{(2)} = g_{uu} \diamond g_{uv} + g_{uv} \diamond g_{vv} + \sum_{k \in U} g_{uk} \diamond g_{kv} + \sum_{k \in [n_1] \backslash (\{u,v\} \cup U)} g_{uk} \diamond g_{kv}$$

$$= y \diamond x + x \diamond y + \sum_{k \in U} x \diamond g_{kv} + \sum_{k \in [n_1] \backslash (\{u,v\} \cup U)} g_{uk} \diamond g_{kv} , \tag{31}$$

$$g_{qs}^{(2)} = g_{qq} \diamond g_{qs} + g_{qs} \diamond g_{ss} + g_{qr} \diamond g_{rs} + \sum_{k \in [n_1] \backslash \{q,s,r\}} g_{qk} \diamond g_{ks}$$

$$= y \diamond x + x \diamond y + x \diamond g_{rs} + \left(\sum_{k \in [n_1] \backslash \{q,s,r\}} x_0 \diamond g_{ks} \right) . \tag{32}$$

Because $n > 2$ and $|U| = n - 1$, $g_{uv}^{(2)}$ in (31) has at least $n - 1 \geq 2$ terms in the form of $x \diamond g_{kv}$, whereas $g_{qs}^{(2)}$ in (32) has exactly one term, $x \diamond g_{rs}$. This shows that $g_{uv}^{(2)} \neq g_{qs}^{(2)}$.

Similarly, it can be established that $g_{uv}^{(2)} \notin \{g_{qr}^{(2)}, g_{rq}^{(2)}, g_{qs}^{(2)}, g_{sq}^{(2)}\}$. This means that $z_{uv} \notin \{z_{qr}, z_{rq}, z_{qs}, z_{sq}\}$ in G_2. Finally, $m_{uv} \notin \{m_{qr}, m_{rq}, m_{qs}, m_{sq}\}$ by $G_2 \succeq \mathtt{wl}([G])$.

This finishes the proof.   □

The first conclusion in Lemma 6.1 indicates, in fact, that the labels, in the stable graph, on the binding edges of $[G]$ distinguish whether what they bind are edges or non-edges of G. The second one tells us that, in the stable graph, the labels on basic edges do not overlap with the labels on binding edges of $[G]$ whenever $n > 2$. In other words, the stable graph $\mathtt{wl}([G])$ recognizes the basic and binding edges of $[G]$ respectively.

The next result indicates that, in the stable graph $\mathtt{wl}([G])$, the labels on the edges and non-edges of G are equivalent to the labels on their binding edges, and even to the labels on their binding vertices.

LEMMA 6.2. *In the stable graph* $\mathtt{wl}([G]) := (m_{ij})$ *of order* $n_1 = n(n+1)/2$ *with* $n > 2$, *for basic vertices* $u, v, r, s \in [n]$ *and* $p = u \wedge v, q = r \wedge s$, *the following statements hold:*

$$\{m_{uv}, m_{vu}\} \equiv \{m_{rs}, m_{sr}\} \quad \Longleftrightarrow \quad \{m_{up}, m_{vp}\} \equiv \{m_{rq}, m_{sq}\} \quad \Longleftrightarrow \quad m_{pp} = m_{qq} \ .$$

PROOF. Denote $\mathtt{wl}([G]) \diamond \mathtt{wl}([G]) := (m_{ij}^{(2)})$. By the recognizability of binding edges, it is known that: For all $u, v, r, s \in [n]$,

$$m_{uv}^{(2)} = \sum_{k \in [n_1]} m_{uk} \diamond m_{kv} = m_{up} \diamond m_{pv} + \sum_{k \in [n_1] \setminus \{p\}} m_{uk} \diamond m_{kv}, \tag{33}$$

$$m_{rs}^{(2)} = \sum_{k \in [n_1]} m_{rk} \diamond m_{ks} = m_{rq} \diamond m_{qs} + \sum_{k \in [n_1] \setminus \{q\}} m_{rk} \diamond m_{ks} \ . \tag{34}$$

By Lemma 6.1 and by the uniqueness of the binding vertex of u, v or r, s,

$$\{m_{up} \diamond m_{pv}, m_{rq} \diamond m_{qs}\} \cap \{m_{uk} \diamond m_{kv} \mid k \in [n_1] \setminus \{p\}\} = \emptyset \quad \text{and} \tag{35}$$

$$\{m_{up} \diamond m_{pv}, m_{rq} \diamond m_{qs}\} \cap \{m_{rk} \diamond m_{ks} \mid k \in [n_1] \setminus \{q\}\} = \emptyset \ . \tag{36}$$

"$\Rightarrow$": Assume that $\{m_{uv}, m_{vu}\} \equiv \{m_{rs}, m_{sr}\}$. For the case of $m_{uv} = m_{rs}$, it holds that $m_{uv}^{(2)} = m_{rs}^{(2)}$ by the stability of $\mathtt{wl}([G])$. This further implies that $m_{up} \diamond m_{pv} = m_{rq} \diamond m_{qs}$ from (33)–(36). This means in turn that $(m_{up} = m_{rq}) \wedge (m_{pv} = m_{qs})$. In this case, $\{m_{up}, m_{vp}\} \equiv \{m_{rq}, m_{sq}\}$. The other cases are similar to prove.

If $\{m_{up}, m_{vp}\} \equiv \{m_{rq}, m_{sq}\}$, it can be assumed that $m_{up} = m_{rq}$. We have $m_{uu} = m_{rr}$ and $m_{pp} = m_{qq}$ by Proposition 3.4. Similarly, $m_{pp} = m_{qq}$ can be obtained from $m_{up} = m_{sq}$.

"$\Leftarrow$": By the third conclusion of Proposition 3.4 and Corollary 3.5, $m_{pp} = m_{qq}$ implies that $m_{pu} + m_{pv} = m_{qr} + m_{qs}$ because $\{u, v\}$ and $\{r, s\}$ are respectively the neighbors of p and q in $[G]$. That is, $\{m_{pu}, m_{pv}\} \equiv \{m_{qr}, m_{qs}\}$, and $\{m_{up}, m_{vp}\} \equiv \{m_{rq}, m_{sq}\}$ by the reverse equivalence property.

Assuming that $m_{up} = m_{rq}$ from $\{m_{up}, m_{vp}\} \equiv \{m_{rq}, m_{sq}\}$, this implies that $\sum_{k=1}^{n_1} m_{uk} \diamond m_{kp} = \sum_{k=1}^{n_1} m_{rk} \diamond m_{kq}$. It also implies that $m_{uu} \diamond m_{up} + m_{uv} \diamond m_{vp} = m_{rr} \diamond m_{rq} + m_{rs} \diamond m_{sq}$ by the recognizability of binding edges. This in turn leads to $m_{uv} \diamond m_{vp} = m_{rs} \diamond m_{sq}$ and $m_{uv} = m_{rs}$. Moreover, $m_{vu} = m_{rs}$ by the reverse equivalence property. Hence, $\{m_{uv}, m_{vu}\} \equiv \{m_{rs}, m_{sr}\}$. The other cases are similar to prove.

This completes the proof.   □

Lemma 6.2 implies, in fact, that the stable graph $\mathtt{wl}([G])$ is completely determined by the labels on the binding edges and by the labels on the vertices of $[G]$.

7   THE ϕ-GRAPH INDUCED BY THE STABLE GRAPH OF A BINDING GRAPH

This section defines a graph Φ from $\mathtt{wl}([G])$ by changing the labels on the basic edges and non-edges of $[G]$ to x_0, keeping the labels on the vertices and binding edges only. Remember that the first n vertices in the simple graph $[G]$ are basic vertices.

Formally, given a binding graph $[G] := (g_{ij})$ of order $n_1 = n(n+1)/2$ and a stable graph $\mathtt{wl}([G]) := (m_{ij})$, graph $\Phi := (\phi_{ij})$ can be constructed as follows: For all $i, j \in [n_1]$,

$$\phi_{ij} := \begin{cases} x_0, & \text{if } i \neq j \text{ and } g_{ij} = x_0, \\ x_0, & \text{if } i, j \in [n] \text{ and } g_{ij} = x, \\ m_{ij}, & \text{Otherwise.} \end{cases} \tag{37}$$

Because the stable graph $\mathtt{wl}([G])$ recognizes vertices, the basic and binding edges of $[G]$ respectively, it is easy to see that $\Phi \succeq \mathtt{wl}([G])$. The graph Φ is called the ϕ-graph induced by the stable $\mathtt{wl}([G])$. The following fact can be demonstrated:

THEOREM 7.1. *Given a binding graph $[G]$ and a stable graph $\mathtt{wl}([G])$ of order $n_1 = n(n+1)/2$ with $n > 2$, the ϕ-graph Φ induced by $\mathtt{wl}([G])$ satisfies the following statements:*

- $\mathtt{wl}(\Phi) \approx \mathtt{wl}([G])$.
- $\mathrm{Aut}(\Phi) = \mathrm{Aut}(\mathtt{wl}(\Phi)) = \mathrm{Aut}(\mathtt{wl}([G]) = \mathrm{Aut}([G])$.

PROOF. Only the first conclusion is shown here because the second can be concluded from the first by Proposition 3.3. Because $\Phi \succeq \mathtt{wl}([G])$, then $\mathtt{wl}(\Phi) \succeq \mathtt{wl}([G])$ by Proposition 3.3. It can now be shown that $\mathtt{wl}([G]) \succeq \mathtt{wl}(\Phi)$.

Denote $\mathtt{wl}([G]) := (m_{ij})$, $\Phi := (\phi_{ij})$ and $\Phi \diamond \Phi := (\phi_{ij}^{(2)})$ with $i, j \in [n_1]$. For all $u, v, r, s \in [n]$, if $p := u \wedge v, q := r \wedge s$, by definition of Φ,

$$\phi_{uv}^{(2)} = \phi_{up} \diamond \phi_{pv} + \sum_{k \in [n_1] \setminus \{p\}} \phi_{uk} \diamond \phi_{kv} = m_{up} \diamond m_{pv} + \sum_{k \in [n_1] \setminus \{p\}} \phi_{uk} \diamond \phi_{kv}, \tag{38}$$

$$\phi_{rs}^{(2)} = \phi_{rq} \diamond \phi_{qs} + \sum_{k \in [n_1] \setminus \{q\}} \phi_{rk} \diamond \phi_{ks} = m_{rq} \diamond m_{qs} + \sum_{k \in [n_1] \setminus \{q\}} \phi_{rk} \diamond \phi_{ks}. \tag{39}$$

If $\phi_{uv}^{(2)} = \phi_{rs}^{(2)}$, then it should hold that $m_{up} \diamond m_{pv} = m_{rq} \diamond m_{qs}$ by recognizability of the binding edges in $\mathtt{wl}([G])$ and (38), (39). However, if (u, v) is an edge and (r, s) a non-edge of $[G]$, then $m_{up} \diamond m_{pv} \neq m_{rq} \diamond m_{qs}$ by Lemma 6.1. In other words, if (u, v) is a basic edge in $[G]$ and (r, s) is a non-edge, then $\phi_{uv}^{(2)} \neq \phi_{rs}^{(2)}$. Which claims that, in the graph $\mathrm{evs}(\Phi \diamond \Phi)$, the labels on the basic edges of $[G]$ do not overlap with the labels on the basic non-edges of $[G]$, because each pair of basic vertices possesses a unique binding vertex.

It can therefore be established that $[G] \succeq \mathrm{evs}(\Phi \diamond \Phi)$ by definition of Φ. This leads to $[G] \succeq \mathrm{evs}(\Phi \diamond \Phi) \succeq \mathtt{wl}(\Phi)$. Finally, $\mathtt{wl}([G]) \succeq \mathtt{wl}(\mathtt{wl}(\Phi)) \approx \mathtt{wl}(\Phi)$ by Proposition 3.3. This ends the proof.   □

For two similar stable graphs $X := (x_{ij}), \bar{X} := (\bar{x}_{ij})$ of binding graphs of order $n_1 = n(n+1)/2$ with $n > 2$, it holds that $x_{ii} = \bar{x}_{ii}$ for all $i \in [n_1]$ by definition. That implies that the induced ϕ-graphs Φ and $\bar{\Phi}$ by X and $\bar{X}$ respectively satisfy $\mathrm{Var}(\Phi) = \mathrm{Var}(\bar{\Phi})$ by Lemma 6.2 and definition of similarity. It is thus easy to obtain the following claim from Theorem 7.1.

COROLLARY 7.2. *With notations as above, it then holds that $\Phi \cong \bar{\Phi}$ if and only if $X \cong \bar{X}$.*

8 THE MAIN THEOREM

This section proves the main theorem that the stable partition of a binding graph is the automorphism partition. It can also be shown that similar stable graphs of binding graphs are in fact isomorphic. This is not true in general as indicated at the end of Section 3.

THEOREM 8.1. *Similar stable graphs of binding graphs are isomorphic.*

PROOF. Let $X := \mathtt{wl}([G])$ and $\bar{X} := \mathtt{wl}([\bar{G}])$ be two similar stable graphs of order $n_1 = n(n+1)/2$, where basic graphs G and $\bar{G}$ are of order n. Let $\alpha = (\alpha_1, \ldots, \alpha_c)$ be the sequential cell partition of $X := (x_{ij})$ and $\bar{X} := (\bar{x}_{ij})$. The first n vertices of these must be basic vertices.

The conclusion of the Theorem is proved by induction on n. For $n \le 3$, please see Example 5.4. Assume that the conclusion is true for $n-1$ with $n > 3$. In this case, a cell of X (and of $\bar{X}$) is either a basic cell or a binding cell by Corollary 5.3.

By similarity, $\sum_{k}^{n_1} x_{1k} = \sum_{k=1}^{n_1} \bar{x}_{1k}$. Graphs X and $\bar{X}$ have the same block partition $\beta = (\beta_1, \ldots, \beta_t)$ with respect to their first vertex. The block partition forms of X and $\bar{X}$ are shown in (40).

$$X = \begin{pmatrix} X_{11} & X_{12} & X_{13} & \cdots & X_{1t} \\ X_{21} & X_{22} & X_{23} & \cdots & X_{2t} \\ X_{31} & X_{32} & X_{33} & \cdots & X_{3t} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ X_{t1} & X_{t2} & X_{t3} & \cdots & X_{tt} \end{pmatrix}, \qquad \bar{X} = \begin{pmatrix} \bar{X}_{11} & \bar{X}_{12} & \bar{X}_{13} & \cdots & \bar{X}_{1t} \\ \bar{X}_{21} & \bar{X}_{22} & \bar{X}_{23} & \cdots & \bar{X}_{2t} \\ \bar{X}_{31} & \bar{X}_{32} & \bar{X}_{33} & \cdots & \bar{X}_{3t} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \bar{X}_{t1} & \bar{X}_{t2} & \bar{X}_{t3} & \cdots & \bar{X}_{tt} \end{pmatrix}. \tag{40}$$

Where $X_{ij} = (x_{uv})$ with $u \in \beta_i, v \in \beta_j$, $X_{1j} = \bar{X}_{1j}$ and $X_{j1} = \bar{X}_{j1}$ for all $i, j \in [t]$.

Because vertex 1 has $n-1$ binding vertices, there are $n-1$ labels of binding edges in the first row of X and $\bar{X}$ respectively. Because stable graphs recognize binding edges, the labels of the binding edges of vertex 1 compose several blocks in the first row.

Let $X_{1u} = (x_{1\,p+1}, x_{1\,p+2}, \ldots, x_{1\,p+s}) = (\bar{x}_{1\,p+1}, \bar{x}_{1\,p+2}, \ldots, \bar{x}_{1\,p+s}) = \bar{X}_{1u}$ be a block of labels of the binding edges of vertex 1. Let $p + k := 1 \wedge u_k$ for $k \in [s], u_k \in [n]$ in X, and let $p + k := 1 \wedge v_k$ for $k \in [s], v_k \in [n]$ in $\bar{X}$. The labels $x_{1\,p+k}, x_{u_k\,p+k}$ are then labels on the binding edges of binding vertex $p + k$ in X; the labels $\bar{x}_{1\,p+k}, \bar{x}_{v_k\,p+k}$ are labels on the binding edges of binding vertex $p + k$ in $\bar{X}$ for all $k \in [s]$ respectively. It should hold that

$$x_{1\,p+1} = x_{1\,p+2} = \cdots = x_{1\,p+s} = \bar{x}_{1\,p+1} = \bar{x}_{1\,p+2} = \cdots = \bar{x}_{1\,p+s}, \tag{41}$$

$$x_{u_1\,p+1} = x_{u_2\,p+2} = \cdots = x_{u_s\,p+s} = \bar{x}_{v_1\,p+1} = \bar{x}_{v_2\,p+2} = \cdots = \bar{x}_{v_s\,p+s}. \tag{42}$$

Equation (41) is true because they are in the same block $X_{1u} = \bar{X}_{1u}$, and equation (42) is true by Equation (41), Lemma 6.2, and the similarity of X and $\bar{X}$. The following fact can now be verified:

FACT 6. $x_{u_1\,1} = x_{u_2\,1} = \cdots = x_{u_s\,1} = \bar{x}_{v_1\,1} = \bar{x}_{v_2\,1} = \cdots = \bar{x}_{v_s\,1}$.

Proof of Fact 6. It is shown here only that $x_{u_1\,1} = x_{u_s\,1} = \bar{x}_{v_1\,1}$; the other statements can be proved similarly. Because $x_{u_1\,p+1} = x_{u_s\,p+s} = \bar{x}_{v_1\,p+1}$ by (42),

$$\sum_{k=1}^{n_1} x_{u_1 k} \diamond x_{k\,p+1} = \sum_{k=1}^{n_1} x_{u_s k} \diamond x_{k\,p+s} = \sum_{k=1}^{n_1} \bar{x}_{v_1 k} \diamond \bar{x}_{k\,p+1} \tag{43}$$

by stability and similarity. Because stable graphs recognize binding edges, it can be obtained from (43) that

$$x_{u_1\,1} \diamond x_{1\,p+1} + x_{u_1\,p+1} \diamond x_{p+1\,p+1} = x_{u_s\,1} \diamond x_{1\,p+s} + x_{u_s\,p+s} \diamond x_{p+s\,p+s}$$

$$= \bar{x}_{v_1\,1} \diamond \bar{x}_{1\,p+1} + \bar{x}_{v_1\,p+1} \diamond \bar{x}_{p+1\,p+1}. \tag{44}$$

It has now been established that $x_{u_1\,1} \diamond x_{1\,p+1} = x_{u_s\,1} \diamond x_{1\,p+s} = \bar{x}_{v_1\,1} \diamond \bar{x}_{1\,p+1}$. This leads to $x_{u_1\,1} = x_{u_s\,1} = \bar{x}_{v_1\,1}$ by definition of the diamond product and hence finishes the proof.

The above fact implies that all vertices $u_1, \ldots, u_s$ are in one block β_i in X; all vertices $v_1, \ldots, v_s$ are in one block β_i' in $\bar{X}$. Moreover, because distinct blocks X_{1u} as labels of binding edges of vertex 1 are binding edges of vertices from different binding cells by Lemma 6.2, this implies that $\{u_1, u_2, \ldots, u_s\}$ is in fact a complete block. That shows that $\{u_1, \ldots, u_s\} = \beta_i$ for some $i \in [t]$.

However, because $X_{j1} = \bar{X}_{j1}$ for all $j \in [t]$, it follows that $\beta_i = \{u_1, \ldots, u_s\} = \{v_1, \ldots, v_s\} = \beta_i'$ by Fact 6. This establishes the "only if" part of the following fact.

FACT 7. $X_{1u} = \bar{X}_{1u}$ is a block of labels for the binding edges of vertex 1 in X and $\bar{X}$ respectively, if and only if the corresponding binding vertices bind vertices in the same block β_i in X and $\bar{X}$ respectively.

For the other part of the fact, if $p_1 = 1 \wedge u_1$ and $p_2 = 1 \wedge u_2$ with $u_1, u_2 \in \beta_i$, this means that $x_{u_1\,1} = x_{u_2\,1}$. With arguments similar to those in the proof of Fact 6, one obtains $x_{1p_1} = x_{1p_2}$, which means that they are in the same block X_{1u} of X. By similarity, Fact 7 is thus established.

Let Φ and $\bar{\Phi}$ be ϕ-graphs induced by X and $\bar{X}$ respectively, as in Section 7. To show an isomorphism from Φ to $\bar{\Phi}$, one can individualize the first vertex from X and $\bar{X}$ respectively, as described in Section 4.1.

Two similar stable graphs Z and $\bar{Z}$ are obtained after individualizations by Corollary 4.2 (shown in (45)). For simplicity of presentation, it is assumed that the last $t - r$ blocks $X_{1\,(t-r+1)}, \ldots, X_{1\,t}$ and $\bar{X}_{1\,(t-r+1)}, \ldots, \bar{X}_{1\,t}$ include all the labels of the binding edges of 1 in X and $\bar{X}$ respectively.

With this assumption, the first cell, the last $t - r$ cells, and their connections in Z and $\bar{Z}$ respectively can be removed to obtain two graphs Z_1 and $\bar{Z}_1$, as shown in (46).

$$Z = \begin{pmatrix} Z_{11} & Z_{12} & Z_{13} & \cdots & Z_{1t} \\ Z_{21} & Z_{22} & Z_{23} & \cdots & Z_{2t} \\ Z_{31} & Z_{32} & Z_{33} & \cdots & Z_{3t} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ Z_{t1} & Z_{t2} & Z_{t3} & \cdots & Z_{tt} \end{pmatrix}, \qquad \bar{Z} = \begin{pmatrix} \bar{Z}_{11} & \bar{Z}_{12} & \bar{Z}_{13} & \cdots & \bar{Z}_{1t} \\ \bar{Z}_{21} & \bar{Z}_{22} & \bar{Z}_{23} & \cdots & \bar{Z}_{2t} \\ \bar{Z}_{31} & \bar{Z}_{32} & \bar{Z}_{33} & \cdots & \bar{Z}_{3t} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \bar{Z}_{t1} & \bar{Z}_{t2} & \bar{Z}_{t3} & \cdots & \bar{Z}_{tt} \end{pmatrix}. \tag{45}$$

$$Z_1 = \begin{pmatrix} Z_{22} & Z_{23} & \cdots & Z_{2r} \\ Z_{32} & Z_{33} & \cdots & Z_{3r} \\ \vdots & \vdots & \ddots & \vdots \\ Z_{t2} & Z_{t3} & \cdots & Z_{rr} \end{pmatrix}, \qquad \bar{Z}_1 = \begin{pmatrix} \bar{Z}_{22} & \bar{Z}_{23} & \cdots & \bar{Z}_{2r} \\ \bar{Z}_{32} & \bar{Z}_{33} & \cdots & \bar{Z}_{3r} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{Z}_{t2} & \bar{Z}_{t3} & \cdots & \bar{Z}_{rr} \end{pmatrix}. \tag{46}$$

By the Composition Theorem 4.3 and Corollary 4.4, Z_1 and $\bar{Z}_1$ are similar stable graphs. Therefore, two similar stable graphs Z_1 and $\bar{Z}_1$ have been obtained for binding graphs with $n - 1$ basic vertices, and hence $Z_1 \cong \bar{Z}_1$ by the induction hypothesis.

Let Φ_1 and $\bar{\Phi}_1$ be the ϕ-graphs induced by Z_1 and $\bar{Z}_1$ respectively. There is then an isomorphism σ from Φ_1 to $\bar{\Phi}_1$ by Corollary 7.2. For simplicity of presentation, it is assumed here that the basic vertices in Φ_1 and $\bar{\Phi}_1$ are in $[2, n]$.

Because graphs Z_1 and $\bar{Z}_1$ are stable, it holds that $\mathrm{Var}(Z_{ii}) \cap \mathrm{Var}(Z_{jj}) = \mathrm{Var}(\bar{Z}_{ii}) \cap \mathrm{Var}(\bar{Z}_{jj}) = \emptyset$ for all $i \neq j$ and $i, j \in [2, r]$ in Z_1 and $\bar{Z}_1$. Moreover, because σ preserves labels, the following fact is established.

FACT 8. $\beta_i^\sigma = \beta_i$ *for all $i \in [2, r]$.*

A mapping τ from Φ to $\bar{\Phi}$ can now be defined as follows. For any $w \in [n_1]$, let

$$
w^\tau := \begin{cases}
1, & \text{if } w = 1 , \\
w^\sigma, & \text{if } w \in [2, n] , \\
1 \dot{\wedge} u^\sigma, & \text{if } w = 1 \dot{\wedge} u \text{ in } [G] \text{ and } u \in [2, n] , \\
u^\sigma \dot{\wedge} v^\sigma, & \text{if } w = u \dot{\wedge} v \text{ in } [G] \text{ and } u, v \in [2, n] .
\end{cases}
\tag{47}
$$

The mapping τ is bijective on $[n]$ because σ is bijective on $[2, n]$. This guarantees that τ is bijective on $[n_1]$ by the uniqueness of binding vertices.

To show that this is an isomorphism from Φ to $\bar{\Phi}$, notice that $Z := (z_{ij})$ and $\bar{Z} := (\bar{z}_{ij})$ are in fact refinements of X and $\bar{X}$ respectively. This means that $z_{u_1 v_1} = z_{u_2 v_2} \Rightarrow x_{u_1 v_1} = x_{u_2 v_2}$ and that $\bar{z}_{u_1 v_1} = \bar{z}_{u_2 v_2} \Rightarrow \bar{x}_{u_1 v_1} = \bar{x}_{u_2 v_2}$ for all $u_1, v_1, u_2, v_2 \in [n_1]$. These relations still hold for Z_1 and $\bar{Z}_1$.

From definition (47), τ is identical to σ when restricted to basic vertices in $[2, n]$ and to binding vertices that bind vertices within $[2, n]$. This shows that $\beta_i^\tau = \beta_i^\sigma = \beta_i$ for all $i \in [2, r]$ by Fact 8. Together with the discussions in the last paragraph, it is possible to obtain:

FACT 9. *The mapping τ from Φ to $\bar{\Phi}$ preserves the labels of vertices and of binding edges when it is restricted to basic vertices in $[2, n]$ and to the binding vertices that bind the basic vertices in $[2, n]$.*

Furthermore, because $X_{1j} = \bar{X}_{1j}$ for all $j \in [t]$, $X_{11} = \bar{X}_{11}$ can be obtained. In other words, $x_{11} = \bar{x}_{11} = \bar{x}_{1^\tau 1^\tau}$, because $1^\tau = 1$. Let $p = 1 \dot{\wedge} u \in \beta_j$ be a binding vertex of 1; x_{1p} and x_{up} are then the labels of binding edges $(1, p)$ and (u, p) respectively. If $u \in \beta_i$, then $u^\sigma \in \beta_i$ by Fact 8. It is then true that $x_{1p} = \bar{x}_{1 \, 1 \dot{\wedge} u^\sigma} = \bar{x}_{1^\tau p^\tau}$, and $x_{up} = \bar{x}_{u^\sigma \, 1 \dot{\wedge} u^\sigma} = \bar{x}_{u^\tau p^\tau}$ by Fact 7, equations (42) and (47). It also holds that $x_{p1} = \bar{x}_{p^\tau 1^\tau}$ and $x_{pu} = \bar{x}_{p^\tau u^\tau}$ by reverse equivalence.

These assertions, together with Fact 9, show that the mapping τ preserves the labels of vertices and of binding edges in Φ. Because the labels of non-edges are all x_0 in Φ and $\bar{\Phi}$, τ preserves them already. This shows that τ is an isomorphism from Φ to $\bar{\Phi}$.

According to Corollary 7.2, τ is in fact an isomorphism from X to $\bar{X}$. This finishes the proof.   □

For a bing graph $[G]$ of order $n_1 = n(n+1)/2$ with $n > 3$, basic vertices cannot be in the same cell with any binding vertices by Corollary 5.3. Let $X := \text{wl}([G])$ be a stable graph of such a binding graph $[G]$ with sequential cell partition $\alpha = (\alpha_1, \ldots, \alpha_c)$. If there is more than one vertex in basic cell α_1, vertex 1 in X can be transposed, by a row-column transposition, with another vertex $u \in \alpha_1$ to obtain a graph $\bar{X}$. Graphs X and $\bar{X}$ are then similar by Proposition 3.9.

The proof in Theorem 8.1 states that there is an isomorphism τ from X to $\bar{X}$ such that $1^\tau = 1$. This means τ is an automorphism of X such that $1^\tau = u$. That indicates that the cell α_1 is an orbit of $\text{Aut}(X)$. In fact, this also states that every basic cell in X is an orbit, because any basic cell can be permuted to the first one in X by row-column permutations. This establishes the following (cf. also Example 5.4).

COROLLARY 8.2. *The basic cells of a stable graph $\text{wl}([G])$ are orbits of $\text{Aut}([G])$.*

We are finally able to prove the main theorem as follows.

THEOREM 8.3. *The stable partition of a binding graph is the automorphism partition.*

PROOF. Let $X := \text{wl}([G]) = (x_{ij})$ be a stable graph of a binding graph $[G]$ with order $n_1 = n(n+1)/2$. For the cases of $n = 2, 3$, the discussions in Example 5.4 show the conclusion of Theorem is true. It remains, by Corollary 8.2, to show

that each binding cell is an orbit of group $\text{Aut}(X)$ for $n > 3$, because, in this case, each cell of X is either a basic cell or a binding cell by Corollary 5.3.

Denote $\text{Aut}_w(X) := \{\sigma \mid \sigma \in \text{Aut}(X), w^\sigma = w\}$ as the stabilizer of a basic vertex $w \in [n]$ in $\text{Aut}(X)$. We will first show the following fact.

FACT 10. *For any $w \in [n]$, if both binding vertices $p = w \wedge u_1$ and $q = w \wedge u_2$ of w are in the same cell, then there exists an automorphism $\sigma \in \text{Aut}_w(X)$ such that $u_1^\sigma = u_2$.*

Proof of Fact 10. We will show this fact for the vertex 1 in X, but the conclusion presented is valid for any $w \in [n]$ by the last conclusion of Proposition 3.4.

Let $\beta = (\beta_1, \ldots, \beta_t)$ be the block sequential partition of vertex 1, and let $X = (X_{ij})$ be in the form of a block partition with respect to 1, as in the proof of Theorem 8.1.

Let $p = 1 \wedge u_1$, $q = 1 \wedge u_2$ and $x_{pp} = x_{qq}$. It should hold that $x_{1p} = x_{1q}$ in this case by Lemma 6.2. Both x_{1p}, x_{1q} are hence entries of some X_{1u}, and p, q are in β_u. With the same arguments as in Fact 7 and Fact 8, it follows that u_1, u_2 are both in the same set β_i of basic vertices.

Let vertex 1 be individualized in X and the stable graph Z of X_1 be obtained as in (45). The stable graph Z is a stable graph of a binding graph as a refinement of X. By the Individualization Theorem 4.1 and Corollary 8.2, β_i as a basic cell of Z is an orbit of $\text{Aut}(Z)$, and there is therefore an automorphism $\sigma \in \text{Aut}(Z)$ such that $u_1^\sigma = u_2$ since $u_1, u_2 \in \beta_i$. It should hold that $1^\sigma = 1$, because $\beta_1 = [1, 1]$ is a singleton cell of Z. Because Z is a refinement of X, it holds that $X \geq Z$ and hence $\text{Aut}(Z) \subseteq \text{Aut}(X)$ by Proposition 2.3. This claims $\sigma \in \text{Aut}_1(X) \subseteq \text{Aut}(X)$ with $u_1^\sigma = u_2$. That finishes the proof of Fact 10.

We now show that for two binding vertices $p_1 = u \wedge v$, $q_1 = w \wedge s$ in X for $u, v, w, s \in [n]$, if $x_{p_1 p_1} = x_{q_1 q_1}$, then there is an automorphism $\tau \in \text{Aut}(X)$ such that $p_1^\tau = q_1$.

From $x_{p_1 p_1} = x_{q_1 q_1}$ we get $\{x_{up_1}, x_{vp_1}\} \equiv \{x_{wq_1}, x_{sq_1}\}$ by Lemma 6.2. If $x_{up_1} = x_{wq_1}$ and $x_{vp_1} = x_{sq_1}$, it holds that $x_{uu} = x_{ww}$ and $x_{vv} = x_{ss}$ by Proposition 3.4.

According to Corollary 8.2, there is an automorphism $\sigma \in \text{Aut}(X)$ such that $u^\sigma = w$, denoted as $v_1 := v^\sigma \in [n]$. It should then hold that $x_{v_1 v_1} = x_{vv} = x_{ss}$ by Proposition 3.4.

To denote $q_0 := w \wedge v_1$, it holds that $p_1^\sigma = (u \wedge v)^\sigma = u^\sigma \wedge v^\sigma = w \wedge v_1 = q_0$. This implies that $x_{p_1 p_1} = x_{q_0 q_0}$ and hence $x_{q_1 q_1} = x_{q_0 q_0}$, again by Proposition 3.4. Because q_0 and q_1 are both binding vertices of vertex w in the same binding cell, there is, by Fact 10, an automorphism $\delta \in \text{Aut}_w(X) \subseteq \text{Aut}(X)$ such that $v_1^\delta = s$.

Set $\tau := \sigma\delta \in \text{Aut}(X)$. We have $u^\tau = u^{\sigma\delta} = w^\delta = w$ and $v^\tau = v^{\sigma\delta} = v_1^\delta = s$. It therefore holds that $p_1^\tau = (u \wedge v)^\tau = u^\tau \wedge v^\tau = w \wedge s = q_1$. A similar process will show that there is an automorphism $\tau \in \text{Aut}(X)$ such that $p_1^\tau = q_1$ in the case of $x_{up_1} = x_{sq_1}$ and $x_{vp_1} = x_{wq_1}$.

This shows each binding cell is an orbit of $\text{Aut}(X)$, which finishes the proof.   □

9 DECISION PROCEDURE FOR GRAPH ISOMORPHISM AND COMPLEXITY

It is well known that the graph-isomorphism problem is polynomial-time solvable if and only if it is polynomial-time solvable to connected simple graphs. Moreover, the problem of computing automorphism partitions of graphs is polynomial-time equivalent to the graph isomorphism problem (cf. Mathon [37]).

For two simple graphs $G := (g_{ij})$ and $H := (h_{ij})$ of order n such that $\text{Var}(G) = \text{Var}(H)$, the disjoint graph $G \uplus H = (f_{uv})$ of G and H is a graph of order $2n$ and can be constructed as follows. For $u, v \in [2n]$,

$$f_{uv} := \begin{cases} g_{uv}, & \text{if } u, v \in [n] , \\ h_{u-n\,v-n}, & \text{if } u, v \in [n+1, 2n] , \\ x_0, & \text{otherwise.} \end{cases} \tag{48}$$

If the labels of edges are distinct in simple graphs G and H before the construction of the disjoint union, an equivalent variable substitution should be performed so that the labels of edges in the two graphs are identical. It is easy to see that there is no path in graph $G \uplus H$ from a vertex in $[n]$ to any vertex in $[n+1, 2n]$. This is used in the proof of the equivalence of the automorphism partition and graph-isomorphism testing in the setting of binding graphs, as follow.

THEOREM 9.1. *Two connected simple graphs G and H of order $n > 1$ with $\text{Var}(G) = \text{Var}(H)$ are isomorphic if and only if there is a basic cell C in the stable partition of binding graph $[G \uplus H]$ such that $C \cap [n] \neq \emptyset \neq C \cap [n+1, 2n]$.*

PROOF. Let graphs G, H be two connected simple graphs of order $n > 1$. The order of $G \uplus H$ is $2n \geq 4$. The basic vertices will not share a same cell with any binding vertices in $\text{Aut}([G \uplus H])$ by Corollary 5.3. Let $N_1 := 2n(2n+1)/2 = n(2n+1)$.

"$\Rightarrow$": For an isomorphism σ from G to H over $[n]$, a mapping τ on $[N_1]$ regarding to $[G \uplus H]$ is defined as follows. For $w \in [N_1]$,

$$w^\tau := \begin{cases} n + w^\sigma, & \text{if } w \in [n] , \\ (w-n)^{\sigma^{-1}}, & \text{if } w \in [n+1, 2n] , \\ (n+u^\sigma) \land (n+v^\sigma), & \text{if } w = u \land v \text{ in } [G \uplus H] \text{ and } u, v \in [n] , \\ (n+u^\sigma) \land (v-n)^{\sigma^{-1}}, & \text{if } w = u \land v \text{ in } [G \uplus H], u \in [n] \text{ and } v \in [n+1, 2n] , \\ (u-n)^{\sigma^{-1}} \land (v-n)^{\sigma^{-1}}, & \text{if } w = u \land v \text{ in } [G \uplus H] \text{ and } u, v \in [n+1, 2n] . \end{cases} \tag{49}$$

It is routine to check that τ is an automorphism of $[G \uplus H]$ by the isomorphism of σ, Definition 5.1 and (48). For a basic cell C in the stable partition of $[G \uplus H]$, if $w \in C$ is a vertex from G, then $w \in [n]$ and $w^\tau \in C$, because C is an orbit of $\text{Aut}([G \uplus H])$ by Theorem 8.3. $w^\tau = n + w^\sigma$ by (49) is a vertex from H by (48). That shows $w \in C \cap [n] \neq \emptyset \neq C \cap [n+1, 2n] \ni w^\tau$.

On the other hand, if $w \in C$ is a vertex from H, then $w \in [n+1, 2n]$. Moreover, $w^\tau = (w-n)^{\sigma^{-1}} \in C$ by (49). This means that $w^\tau \in C$ is a vertex from G. This shows, in fact, that each basic cell in the stable partition of $[G \uplus H]$ is shared by vertices from both graphs in this case.

"$\Leftarrow$": Let C be a basic cell in the stable partition of $[G \uplus H]$ such that $u \in [n]$, $v \in [n+1, 2n]$ for some $u, v \in C$. There is an automorphism $\tau \in \text{Aut}([G \uplus H])$ such that $u^\tau = v$ by Theorem 8.3.

For any vertex $\bar{u} \in [n]$, we show that $\bar{u}^\tau \in [n+1, 2n]$ now. Because G is connected, there is then a path $\mathsf{p} := (u_0, u_1, \ldots, u_t)$ in $[G \uplus H]$ such that $u_0 = u, u_t = \bar{u}$ and all $u_k \in [n]$. In other words, $\deg(u_k) > 2n - 1 \geq 3$ for all $k \in [0, t]$ because each basic vertex has at least $2n - 1$ binding vertices. The automorphism τ guarantees that $\mathsf{p}^\tau := (u_0^\tau, u_1^\tau, \ldots, u_t^\tau)$ is a path in $[G \uplus H]$ and $\deg(u_k^\tau) > 3$ for all $k \in [0, t]$. That claims p^τ is a path in $G \uplus H$. Because $u_0^\tau = u^\tau = v \in [n+1, 2n]$ and there is no path from vertices in $[n+1, 2n]$ to vertices in $[n]$ in graph $G \uplus H$, it can therefore be determined that all $u_k^\tau \in [n+1, 2n]$ in the path p^τ. Therefore, $\bar{u}^\tau = u_t^\tau \in [n+1, 2n]$ for $\bar{u} \in [n]$. This shows that $[n]^\tau = [n+1, 2n]$.

Let $\delta := \tau|_{[n]}$ be the restriction of τ on $[n]$ and σ be the mapping such that $w^\sigma := w^\delta - n$ for all $w \in [n]$. It is easy to verify that σ is an isomorphism from G to H. Hence, $G \cong H$.

This finishes the proof.   □

The next step is to formalize a decision procedure GI for graph isomorphisms and to estimate its computational complexity. The purpose here is to show a polynomial-time procedure and by no means to pursue the optimal time complexity.

Procedure GI:

    i. Input: Two connected simple graphs G, H of order $n > 1$ with $\mathrm{Var}(G) = \mathrm{Var}(H)$.

    ii. Construct the disjoint union graph $G \uplus H$ and the binding graph $[G \uplus H]$;

    iii. Compute the stable graph $\mathrm{wl}([G \uplus H])$ by the WL algorithm;

    iv. Form the vertex partition C of $\mathrm{wl}([G \uplus H])$;

    v. Output: $\top$ if there is a basic cell $C \in \mathcal{C}$ that satisfies $C \cap [n] \neq \emptyset \neq C \cap [n+1, 2n]$; $\bot$ otherwise.

The procedure GI outputs $\top$ if and only if $G \cong H$ is guaranteed by Theorem 9.1. The time cost at each step in GI will now be estimated.

1. The construction of binding graph $[G \uplus H]$ over $[N_1]$ from G and H accomplishes the task with at most $O(N_1^2) = O(n^4)$ steps, where $N_1 = n(2n + 1)$.

2. The evaluation of $\mathrm{wl}([G \uplus H])$ costs at most $O(N_1^4 \cdot \log N_1) = O(n^8 \log n)$ steps by Theorem 3.2, assuming that the cost for multiplication of matrices of order N_1 is $O(N_1^3)$.

3. The formation of the cell partition C of $\mathrm{wl}([G \uplus H])$ uses only $O(N_1) = O(n^2)$ steps.

4. The examination of cells from C can be completed in $O(N_1) = O(n^2)$ steps.

In total, the procedure GI consumes at most $O(n^8 \log n)$ steps for two graphs of order n. This supports the following claim:

THEOREM 9.2. *The graph isomorphism problem is solvable in polynomial time.*

10   BRIEF DISCUSSION

This work has proposed the class of binding graphs and shown that the stable partitions of binding graphs by the WL algorithm are automorphism partitions. That leads to a polynomial-time procedure GI for testing graph isomorphisms.

The fact that the graph isomorphism problem is in P will bring a multitude of problems into P (cf., e.g., Booth and Colbourn [10], Mathon [37], Luks [35], Babai [3]) and will solve some open problems in computational complexity classes relevant to the graph isomorphism problem (cf., e.g., Köbler, Schöning, and Torán [31]).

The new class of binding graphs is expected to have more applications in graph theory and even in group theory. It would be interesting to apply other graph isomorphism approaches to the class of binding graphs. Other interesting projects include exploiting the optimal complexity of the graph-isomorphism problem in theory and implementing more efficient programs in practice for graph isomorphisms using the results of this work, if this is possible.

REFERENCES

[1] László Babai. 1979. *Monte Carlo algorithms in graph isomorphism testing.* Technical Report. Dép. Math. et Stat., Univ. de Montréal.

[2] László Babai. 2015. Graph Isomorphism in Quasipolynomial Time. *CoRR* abs/1512.03547 (2015). arXiv:1512.03547 http://arxiv.org/abs/1512.03547

[3] László Babai. 2016. Graph isomorphism in quasipolynomial time [extended abstract]. In *Proceedings of STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, Daniel Wichs and Yishay Mansour (Eds.). ACM, 684–697.

[4] László Babai. 2018. Group, graphs, algorithms: the graph isomorphism problem. In *Proceedings of the International Congress of Mathematicians, ICM 2018*, Boyan Sirakov, Paulo Ney de Souza, and Marcelo Viana (Eds.), Vol. 3. International Congress of Mathematicians 2018, Rio de Janeiro, Brazil, 3303–3320.

[5] László Babai. 2019. Canonical form for graphs in quasipolynomial time: preliminary report. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, Moses Charikar and Edith Cohen (Eds.). ACM, 1237–1246.

[6] László Babai, Paul Erdös, and Stanley M. Selkow. 1980. Random Graph Isomorphism. *SIAM J. Comput.* 9, 3 (1980), 628–635. https://doi.org/10.1137/0209047

[7] László Babai, D. Yu. Grigoryev, and David M. Mount. 1982. Isomorphism of Graphs with Bounded Eigenvalue Multiplicity. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, Harry R. Lewis, Barbara B. Simons, Walter A. Burkhard, and Lawrence H. Landweber (Eds.). ACM, 310–324.

[8] László Babai and Ludek Kucera. 1979. Canonical Labelling of Graphs in Linear Average Time. In *20th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 29-31 October 1979*. IEEE Computer Society, 39–46.

[9] László Babai and Rudolf Mathon. 1980. Talk at the South-East Conference on Combinatorics and Graph Theory. (1980).

[10] Kellogg S. Booth and Charles J. Colbourn. 1979. *Problems polynomially equivalent to graph isomorphism.* Technical report. Computer Science Department, University of Waterloo.

[11] Jin-yi Cai, Martin Fürer, and Neil Immerman. 1992. An optimal lower bound on the number of variables for graph identifications. *Comb.* 12, 4 (1992), 389–410. https://doi.org/10.1007/BF01305232

[12] Charles J. Colbourn and Kellogg S. Booth. 1981. Linear Time Automorphism Algorithms for Trees, Interval Graphs, and Planar Graphs. *SIAM J. Comput.* 10, 1 (1981), 203–225.

[13] Luigi P. Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento. 1999. Performance Evaluation of the VF Graph Matching Algorithm. In *10th International Conference on Image Analysis and Processing (ICIAP 1999), 27-29 September 1999, Venice, Italy*. IEEE Computer Society, 1172–1177. https://doi.org/10.1109/ICIAP.1999.797762

[14] Dragoš Cvetković. 1995. Star partitions and the graph isomorphism problem. *Linear and Multilinear Algebra* 39 (1995), 109–132. https://doi.org/DOI:10.1080/03081089508818383

[15] Dragoš Cvetković, Peter Rowlinson, and Slobodan Simić. 1993. A study of eigenspaces of graphs. *Linear Algebra Appl.* 182 (1993), 45–66. https://doi.org/10.1016/0024-3795(93)90491-6

[16] Paul T. Darga, Mark H. Liffiton, Karem A. Sakallah, and Igor L. Markov. 2004. Exploiting structure in symmetry detection for CNF. In *Proceedings of the 41th Design Automation Conference, DAC 2004, San Diego, CA, USA, June 7-11, 2004*, Sharad Malik, Limor Fix, and Andrew B. Kahng (Eds.). ACM, 530–534.

[17] Frank Fuhlbrück, Johannes Köbler, Ilia Ponomarenko, and Oleg Verbitsky. 2021. The Weisfeiler-Leman algorithm and recognition of graph properties. *Theor. Comput. Sci.* 895 (2021), 96–114. https://doi.org/10.1016/j.tcs.2021.09.033

[18] Martin Fürer. 2001. Weisfeiler-Lehman Refinement Requires at Least a Linear Number of Iterations. In *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings (Lecture Notes in Computer Science)*, Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen (Eds.), Vol. 2076. Springer, 322–333. https://doi.org/10.1007/3-540-48224-5_27

[19] Martin Fürer. 2017. On the Combinatorial Power of the Weisfeiler-Lehman Algorithm. In *Algorithms and Complexity - 10th International Conference, CIAC 2017, Athens, Greece, May 24-26, 2017, Proceedings (Lecture Notes in Computer Science)*, Dimitris Fotakis, Aris Pagourtzis, and Vangelis Th. Paschos (Eds.), Vol. 10236. 260–271.

[20] M. R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman.

[21] O. Goldreich, S. Micali, and A. Wigderson. 1986. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *FOCS86*. IEEE, Toronto, 174–187.

[22] Shafi Goldwasser and Michael Sipser. 1986. Private Coins versus Public Coins in Interactive Proof Systems. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, Juris Hartmanis (Ed.). ACM, 59–68. https://doi.org/10.1145/12130.12137

[23] Martin Grohe. 2017. *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory.* Lecture Notes in Logic, Vol. 47. Cambridge University Press.

[24] Martin Grohe and Daniel Neuen. 2021. Recent advances on the graph isomorphism problem. In *Surveys in Combinatorics, 2021: Invited lectures from the 28th British Combinatorial Conference, Durham, UK, July 5-9, 2021*, Konrad K. Dabrowski, Maximilien Gadouleau, Nicholas Georgiou, Matthew Johnson, George B. Mertzios, and Daniël Paulusma (Eds.). Cambridge University Press, 187–234. https://doi.org/10.1017/9781009036214.006

[25] Martin Grohe, Daniel Neuen, and Pascal Schweitzer. 2018. A Faster Isomorphism Test for Graphs of Small Degree. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, Mikkel Thorup (Ed.). IEEE Computer Society, 89–100. https://doi.org/10.1109/FOCS.2018.00018

[26] Martin Grohe and Pascal Schweitzer. 2020. The graph isomorphism problem. *Commun. ACM* 63, 11 (2020), 128–134. https://doi.org/10.1145/3372123

[27] Neil Immerman and Eric Lander. 1990. Describing graphs: a first-order approach to graph canonization. In *Complexity theory retrospective*, Alan L. Selman (Ed.). Springer, 59–81.

[28] Tommi A. Junttila and Petteri Kaski. 2007. Engineering an Efficient Canonical Labeling Tool for Large and Sparse Graphs. In *Proceedings of the Nine Workshop on Algorithm Engineering and Experiments, ALENEX 2007, New Orleans, Louisiana, USA, January 6, 2007*. SIAM, 135–149.

[29] Richard M. Karp. 1972. Reducibility Among Combinatorial Problems. In *Proceedings of a symposium on the Complexity of Computer Computations (The IBM Research Symposia Series)*, Raymond E. Miller and James W. Thatcher (Eds.). Plenum Press, New York, 85–103.

[30] Sandra Kiefer. 2020. *Power and limits of the Weisfeiler-Leman algorithm.* PhD Thesis. RWTH Aachen University, Frankfurt am Main. https://www.lics.rwth-aachen.de/go/id/nwgi/file/785831/lidx/1/

[31] Johannes Köbler, Uwe Schöning, and Jacobo Torán. 1993. *The Graph Isomorphism Problem: Its Structural Complexity.* Birkhäuser/Springer.

[32] Andrei A. Lehman. 1976. A Construction of a Stationary Graph. In *On Construction and Identification of Graphs (Lecture Notes in Mathematics)*, Boris Weisfeiler (Ed.). Springer-Verlag, Berlin, Berlin, 13–22. With contributions by A. Lehman, G. M. Adelson-Velsky, V. Arlazarov, I. Faragev, A. Uskov, I. Zuev, M. Rosenfeld and B. Weisfeiler.

[33] Moritz Lichter, Ilia Ponomarenko, and Pascal Schweitzer. 2019. Walk refinement, walk logic, and the iteration number of the Weisfeiler-Leman algorithm. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*. IEEE, 1–13. https://doi.org/10.1109/LICS.2019.8785694

[34] José Luis López-Presa, Luis F. Chiroque, and Antonio Fernández Anta. 2014. Novel Techniques to Speed Up the Computation of the Automorphism Group of a Graph. *J. Appl. Math.* 2014 (2014), 1–15.

[35] Eugene Luks. 1993. *Permutation groups and polynomial-time computation*. DIMACS series in Discrete Mathematics and Theoretical Computer Science, Vol. 11. Chapter Groups and Computation, 139–175.

[36] Eugene M. Luks. 1980. Isomorphism of Graphs of Bounded Valence Can Be Tested in Polynomial Time. In *21st Annual Symposium on Foundations of Computer Science, Syracuse, New York, USA, 13-15 October 1980*. IEEE Computer Society, 42–49.

[37] Rudolf Mathon. 1979. A Note on the Graph Isomorphism counting Problem. *Inf. Process. Lett.* 8, 3 (1979), 131–132. https://doi.org/10.1016/0020-0190(79)90004-8

[38] Brendan McKay. 1981. Practical graph isomorphism. *Congressus Numerantium* 30 (1981), 45–87.

[39] Brendan D. McKay and Adolfo Piperno. 2014. Practical graph isomorphism, II. *J. Symb. Comput.* 60 (2014), 94–112.

[40] H.L. Morgan. 1965. The generation of a unique machine description for chemical structures˜ŞA technique developed at chemical abstracts service. *J. Chem. Document.* 5, 2 (1965), 107–113.

[41] Ronald C. Read and Derek G. Corneil. 1977. The graph isomorphism disease. *J. Graph Theory* 1, 4 (1977), 339–363.

[42] Gerta Rücker and Christoph Rücker. 1990. Computer perception of constitutional (topological) symmetry: TOPSYM, a fast algorithm for partitioning atoms and pairwise relations among atoms into equivalence classes. *Journal of chemical information and computer sciences* 30, 2 (1990), 187–191.

[43] Gerta Rücker and Christoph Rücker. 1991. On using the adjacency matrix power method for perception of symmetry and for isomorphism testing of highly intricate graphs. *J. Chem. Inf. Comput. Sci.* 31, 1 (1991), 123–126. https://doi.org/10.1021/ci00001a022

[44] Stoicho D. Stoichev. 2019. New Exact and Heuristic Algorithms for Graph Automorphism Group and Graph Isomorphism. *ACM J. Exp. Algorithmics* 24, 1 (2019), 1.15:1–1.15:27.

[45] Gottfried Tinhofer. 1991. A note on compact graphs. *Discret. Appl. Math.* 30, 2-3 (1991), 253–264. https://doi.org/10.1016/0166-218X(91)90049-3

[46] Gottfried Tinhofer and Mikhail Klin. 1999. Algebraic combinatorics in mathematical chemistry. Methods and algorithms III. Graph invariants and stabilization methods. (1999). https://www.researchgate.net/publication/2271886_Algebraic_Combinatorics_in_Mathematical_Chemistry_Methods_and_Algorithms_III_Gra

[47] Ryuhei Uehara, Seinosuke Toda, and Takayuki Nagoya. 2005. Graph isomorphism completeness for chordal bipartite graphs and strongly chordal graphs. *Discret. Appl. Math.* 145, 3 (2005), 479–482. https://doi.org/10.1016/j.dam.2004.06.008

[48] Boris Weisfeiler and Andrei A. Leman. 1968. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya* 9 (1968), 12–16. https://www.iti.zcu.cz/wl2018/pdf/wl_paper_translation.pdf

[49] Rui Xue. 2022. Description Graphs, Matrix-Power Stabilizations and Graph Isomorphism in Polynomial Time. *arXiv: https://doi.org/10.48550/arXiv.2211.05637* abs/2211.05637 (2022). arXiv:2211.05637 https://doi.org/10.48550/arXiv.2211.05637