No PAIN no Gain: More Expressive GNNs with Paths

Caterina Graziani[*]
DIISM, University of Siena
Siena, Italy
caterina.graziani@student.unisi.it

Tamara Drucks[*]
Research Unit Machine Learning, TU Wien
Vienna, Austria
tamara.drucks@tuwien.ac.at

Monica Bianchini
DIISM, University of Siena
Siena, Italy
monica.bianchini@unisi.it

Franco Scarselli
DIISM, University of Siena
Siena, Italy
franco.scarselli@unisi.it

Thomas Gärtner
Research Unit Machine Learning, TU Wien
Vienna, Austria
thomas.gaertner@tuwien.ac.at

Abstract

Motivated by the lack of theoretical investigation into the discriminative power of *paths*, we characterize classes of graphs where paths are sufficient to identify every instance. Our analysis motivates the integration of paths into the learning procedure of graph neural networks in order to enhance their expressiveness. We formally justify the use of paths based on finite-variable counting logic and prove the effectiveness of paths to recognize graph structural features related to cycles and connectivity. We show that paths are able to identify graphs for which higher-order models fail. Building on this, we propose PAth Isomorphism Network (PAIN), a novel graph neural network that replaces the topological neighborhood with paths in the aggregation step of the message-passing procedure. This modification leads to an algorithm that is strictly more expressive than the Weisfeiler-Leman graph isomorphism test, at the cost of a polynomial-time step for every iteration and fixed path length. We support our theoretical findings by empirically evaluating PAIN on synthetic datasets.

1 Introduction

We investigate the discriminative power of *paths* to improve the expressiveness of message-passing graph neural networks (GNNs). The expressive power of GNNs has been characterized by the Weisfeiler-Leman ($1-$WL) color refinement algorithm [33, 26]. $1-$WL is a polynomial-time heuristic for testing graph isomorphism that identifies almost all graphs [6] but systematically fails for some graph instances (see, e.g., Figure 2). This shortcoming can be partially explained by the limitation of $1-$WL to only recognize forests and star graphs [4]. Higher-order extensions of $1-$WL, i.e., $k-$WL (Babai [5], Immerman & Lander [20], [26]), which operate on $k-$tuples of nodes, are more powerful thanks to their ability to recognize additional substructures. However, they are impractical due to their prohibitive complexity and lack of locality. Given these well-studied limitations, research efforts have resulted in several novel graph neural networks which leverage graph substructures to improve expressiveness [31, 35, 9, 10, 8, 11]. Paths, arguably one of the simplest graph substructures,

[*]These authors contributed equally.

have only recently received attention in the context of graph neural networks [24, 2, 21, 32]. Indeed, there has been little investigation into the discriminative power of paths themselves and a lack of characterization of graph structural properties that paths can recognize. In this paper, we propose to fill this gap in the literature by showing how paths can improve the expressive power of GNNs, especially when we consider graphs with cycles or when the task at hand is related to the connectivity of the graph. These structural properties are crucial in many applications ranging from social networks to biochemistry [25, 27]. For instance, in bioinformatics molecules are often represented as graphs. Due to the inability of graph neural networks to count cycles [3], it is possible that two molecules with a different number of (aromatic) cycles will result in identical embeddings (see Figure 2). Based on our theoretical findings, we design PAIN (PAth Isomorphism Network), a provably more powerful GNN which is characterized by PATH-WL, a path-based adaptation of 1-WL. Our main contributions are summarized in the following.

Main contributions. We formally justify the importance of paths by proving that $1-$WL is not able to recognize paths based on finite-variable counting logic (Section 3.1). We identify classes of graphs which are completely characterized by multisets of paths and show that paths can distinguish between pairs of non-isomorphic graphs for which $3-$WL fails (Section 3.2). Based on our theoretical investigation, we propose PATH-WL, a strictly more expressive path-based adaptation of $1-$WL (Section 4). We further propose an extension that enables PATH-WL to count cycles without incurring any computational overhead. We define PAIN, a novel graph neural network model characterized by PATH-WL, and empirically evaluate PAIN on the EXP and SR datasets (Section 5).

2 Preliminaries

Graph theory. We consider undirected graphs. Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be a **graph** with node set $\mathcal{V}$, edges $\mathcal{E}$ and node features $\mathbf{X} \in \mathbb{R}^{n\times d}$. We denote by $\mathbf{x}_v$ the $d-$dimensional feature vector of node $v \in \mathcal{V}$. Let $\mathcal{N}(v)$ be the *neighborhood* of a node $v \in \mathcal{V}$, i.e. the set of all nodes adjacent to v, and be $\delta(v)$ the *degree* of a node $v \in \mathcal{V}$, i.e., the number of neighbors $|\mathcal{N}(v)|$. A **path** $p = (v_1, \dots, v_l)$ of length $l-1$ in a graph G is a sequence of non-repeated nodes connected through edges present in G. A **cycle** $\gamma_l = (v_1, \dots, v_l)$ of length l is a sequence of adjacent and non-repeated nodes in G. The only exception is that the first and the last node are adjacent, i.e., there exists an edge (v_1, v_l). A graph G is **connected** if for any $v, u \in \mathcal{V}$ there exists a path connecting u and v. A graph G is d**-regular** if every node has the same degree d, i.e., $\forall v \in \mathcal{V}\ \delta(v) = d$. A **strongly regular** graph $SR(n, k, \lambda, \mu)$ is a regular graph with n vertices and degree k such that (i) every two adjacent vertices have λ common neighbors, and (ii) every two non-adjacent vertices have μ common neighbors for some integers $\lambda, \mu \geq 0$. A **traceable** graph is a graph with a Hamiltonian path, namely a path that includes all the nodes of the graph. A traceable graph is **homogeneously traceable** if every node of the graph is an endpoint of a Hamiltonian path. A graph is **Hamiltonian** if it contains a Hamiltonian cycle, i.e., a cyle which contains every node in the graph exactly once. Hamiltonian graphs are homogeneously traceable, but the converse is not necessarily true.

Graph neural networks. Message-passing graph neural networks (GNNs) leverage the graph structure and the node features $\mathbf{X}$ to learn a representation vector (or *embedding*) for individual nodes, denoted by $\mathbf{h}_v$, or for the entire graph, denoted by $\mathbf{h}_G$. Each node embedding is updated by aggregating the embeddings of its neighboring nodes.

Definition 2.1. *Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be a graph with node features $\mathbf{X}$. We initialize $\mathbf{h}_v^{(0)} = \mathbf{x}_v$ for all $v \in \mathcal{V}$. The computation scheme of a message-passing graph neural network for iteration $i > 0$ is defined as*

$$\mathbf{h}_v^{(i)} = \mathtt{COMBINE}(\mathbf{h}_v^{(i-1)}, \mathtt{AGGREGATE}\ (\{\!\{\mathbf{h}_u^{(i-1)} \mid u \in \mathcal{N}(v)\}\!\})),$$

where $\mathbf{h}_v^{(i)}$ is the feature vector of node v at the i-th iteration. The GNN output for a node-level learning task after iteration k is given by

$$\mathbf{h}_v = \mathtt{READOUT}(\mathbf{h}_v^{(k)}),$$

and the output for a graph-level learning task given by

$$\mathbf{h}_G = \mathtt{READOUT}(\{\!\{\mathbf{h}_v^{(k)} \mid v \in \mathcal{V}\}\!\})$$

using a suitable `READOUT` *function.*

The expressive power of graph neural networks can be studied in terms of their capability to distinguish two non-isomorphic graphs. Xu et al. [33] showed that with a sufficient number of GNN layers, and injective `COMBINE`, `AGGREGATE`, and `READOUT` functions, the resulting GNN architecture is as expressive as the $1-$dimensional Weisfeiler-Leman ($1-$WL) test.

Weisfeiler-Leman test. The $1-$WL test, also known as *color refinement algorithm*, is a coloring procedure employed to test graph isomorphism [14, 34].

Definition 2.2. *Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be a graph with node features $\mathbf{X}$ and $v \in \mathcal{V}$. Let $\mathbf{c}_v^{(i)} \in \Sigma$ be the color of the node v at iteration i, from a countable set of colors Σ, and let $\mathbf{c}_v^{(0)} = \mathbf{x}_v$. For iteration $i > 0$*

$$\mathbf{c}_v^{(i)} = \mathtt{HASH}\Big(\mathbf{c}_v^{(i-1)}, \{\!\{\mathbf{c}_u^{(i-1)} \mid u \in N(v)\}\!\}\Big),$$

where `HASH` bijectively maps $(\mathbf{c}_v^{(i-1)}, \{\!\{\mathbf{c}_u^{(i-1)} \mid u \in N(v)\}\!\})$ to a unique value in Σ.

The color refinement procedure partitions the nodes of a graph into equivalence classes (i.e., the colors), where equivalent nodes are assigned the same color. The algorithm terminates when the partition does not change between iterations. To test whether two graphs are isomorphic, the $1-$WL test is applied to both graphs. If the colorings of the two graphs differ after the algorithm has terminated, i.e., the graphs have a different number of nodes with the same color, the graphs are non-isomorphic. The algorithm is not conclusive if the partitioning is the same, i.e., the two graphs *can be*, but are not guaranteed to be, isomorphic.

Due to the shortcomings of $1-$WL in failing to distinguish non-isomorphic graphs, Immerman & Lander [20] devised the more powerful extension $k-$WL. Differently from $1-$WL, $k-$WL colors $k-$tuples from $\mathcal{V}^k$ instead of single nodes (see Morris et al. [26] for more details). As both are used in GNN literature, we point out the existence of two variants of the same algorithm, which are the folklore $k-$WL ($k-$FWL) and the oblivious $k-$WL ($k-$OWL). Whenever we mention $k-$WL, we are referring to $k-$OWL, for which the following properties hold:

- $1-$WL is equivalent in expressive power to $2-$WL
- $(k+1)-$WL is more expressive than $k-$WL for any $k \geq 2$.

The existing relation between the $k-$OWL and the $k-$FWL is the following, for any $k > 1$:

$$k\text{–OWL} = (k-1)\text{–FWL}.$$

3 Why and how paths can be helpful

The purpose of this paper is to analyze the inherent expressive power of paths and subsequently integrate them into the message-passing procedure to increase the expressive power of standard GNNs. The first question we aim to answer is: *Why do we focus on paths and not, more generally, on walks?* By definition, walks are sequences of adjacent nodes, and therefore paths are particular walks without node repetitions. Due to the local nature of the message-passing procedure, GNNs are able to encode all the walks up to a certain length for each node. At the same time, GNNs do not encode the identity of a node and therefore are unable to identify repetitions of the same node in a walk. We thus argue that paths, in contrast to walks (see Kriege [22]), do contribute additional information which can be helpful to distinguish between non-isomorphic graphs. In the following, we first formalize this intuition based on finite variable counting logic. Once we have formally established that paths can be useful, we characterize their contribution in terms of cycle-detection and connectivity.

3.1 The inexpressibility of paths in $\mathcal{C}_2$

As a first step to argue for the ability of paths to improve the expressive power of GNNs, we prove that paths are not expressible in the logic of GNNs, that is, $\mathcal{C}_2$. The logic $\mathcal{C}_2$ is a two-variable fragment of first order logic extended with counting quantifiers of the form $\exists^{\geq n} x \phi(x)$. The semantic of the statement is that there are at least n variables satisfying the formula ϕ. Grohe [16] recently showed that the expressiveness of GNNs with global readout can be characterized precisely by $\mathcal{C}_2$. That is, two graphs G and G' are WL-equivalent if and only if they satisfy the same $\mathcal{C}_2$-sentences.

We refer the reader to Appendix A.2 and [16, 12] for more details on this equivalence.

Let $P_l(x, y)$ be the formula stating the existence of a path from x to y of length l and let $E(x, y)$ be an edge connecting the nodes x and y. We define $P_l(x, y)$ recursively as follows:

$$P_1(x, y) \equiv E(x, y)$$

$$P_l(x, y) \equiv \exists z(P_{l-1}(x, z) \wedge E(z, y)).$$

We show that $P_l(x, y)$ with $l \geq 2$ cannot be expressed in $\mathcal{C}_2$. As an immediate consequence, we can derive that $1-$WL, and equivalently GNNs, are not able to recognize or count paths.

Theorem 3.1. *For any $l \geq 2$, $P_l(x, y) \notin \mathcal{C}_2$. That is, the existence of a path of length greater than one between nodes x and y is inexpressible in $\mathcal{C}_2$, the counting logic with two variables.*

Proof sketch. The main idea of the proof is to use Ehrenfeucht-Fraïssé games, where we show that the second player always has a winning strategy for a game with two pebbles and unlimited moves. The full proof can be found in A.2 in the Appendix and closely follows the proof of Proposition 6.15 provided in Chapter 6 of Immerman [19]. $\square$

3.2 The discriminative power of paths

In light of Theorem 3.1, we argue that incorporating *paths* into the learning procedure has the potential to improve the expressive power of GNNs. In the following, we (i) define the relation induced on nodes and graphs by multisets of paths and (ii) show that paths are sufficient to distinguish well-known failure instances for $1-$WL and even for $3-$WL. We further (iii) analyze the ability of paths to distinguish nodes and graphs in relation to cycles and connectivity.

Definition 3.2. *Let $P_v^l := \{\!\{p_v^k \mid k \leq l\}\!\}$ be the multiset of all possible paths starting from node v of length up to l, where $p_v^k = (\mathbf{h}_v, \mathbf{h}_{v_1}, \ldots, \mathbf{h}_{v_k})$ denotes a path of length k starting at node v and $\mathbf{h}_{v_i}$ is the feature of node v_i.*

Note that it is possible to have repeating node features within the same path. The multiset of paths induces an equivalence relation on the starting nodes.

Definition 3.3. *Two nodes $u, v \in \mathcal{V}$ are said to be $\mathcal{P}^l$-equivalent, denoted by $u \sim_{\mathcal{P}^l} v$, if and only if they have the same multiset of paths of length up to l. Formally,*

$$u \sim_{\mathcal{P}^l} v \iff P_v^l = P_u^l.$$

This relation is reflexive, symmetric, and transitive, i.e., it is an equivalence. Moreover, the expressive power of the relation is monotone with respect to the length of the paths, because $P_v^l \subseteq P_v^{l+1}$, for any $l \in \mathbb{N}$. This equivalence relation enables us to directly compare nodes based on their path multisets and can be extended to graphs.

Definition 3.4. *Two graphs $G = (\mathcal{V}, \mathcal{E})$ and $G' = (\mathcal{V}', \mathcal{E}')$ are said to be $\mathcal{P}^l$-equivalent, denoted by $G \sim_{\mathcal{P}^l} G'$, if and only if a bijective mapping associates every node $v \in \mathcal{V}$ to a node $v' \in \mathcal{V}'$ such that $v \sim_{\mathcal{P}^l} v'$.*

The following theorem states that with sufficiently large l, $\mathcal{P}^l$ can always distinguish between two classes of graphs: non-homogeneously traceable graphs and Hamiltonian graphs.

Theorem 3.5. *Let G and H be two graphs of the same order, i.e., $|V(G)| = |V(H)| = n$. G is a non-homogeneously traceable graph and H a Hamiltonian graph. Then,*

$$G \nsim_{\mathcal{P}^{n-1}} H.$$

Please refer to A.4 in the Appendix for the proof. In the following corollary, we provide a construction for pairs of graphs satisfying the hypothesis of Theorem 3.5. Note that such pairs of graphs are indistinguishable by $1-$WL.

Corollary 3.6. *Let γ_n be a cycle of length n. H is a graph of order $2n$ composed of two cycles γ_{n+1} with an edge in common. G is a graph of order $2n$ composed of two cycles γ_n connected by an extra edge. See Figure 1 for a visualization of how H and G are constructed. For any $n \geq 3$, it holds that*

$$G \nsim_{\mathcal{P}^{2n-1}} H \quad \text{and} \quad G \sim_{WL} H.$$

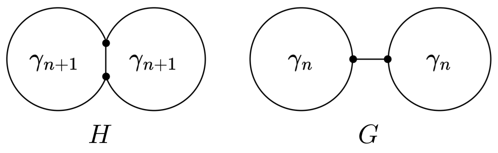


Figure 1: Construction of a pair of two non-isomorphic $1-$WL-equivalent graphs that can be distinguished by the multisets of paths of length up to $2n-1$.

In Figure 2, we present two well-known pairs of non-isomorphic graphs which represent instances of Corollary 3.6. We remark that they are indistinguishable by $1-$WL but we can distinguish them by comparing the multisets of paths associated with each node. In the Figure, $\mathcal{P}^l$ is represented as a box containing the paths associated with a number representing their multiplicity in the multiset. In (a) node v in G has 5 paths of length 3, while u in G' has 3 paths of length 3, and in (b) node v in H has 5 paths of length 5, while node u in H' has 3 paths of length 5. Note that the graphs in Figure 2 have uniform node features (represented as node colors in the picture), and therefore the only difference between the multisets of paths is the multiplicity of paths of length 3 (or 5 respectively).

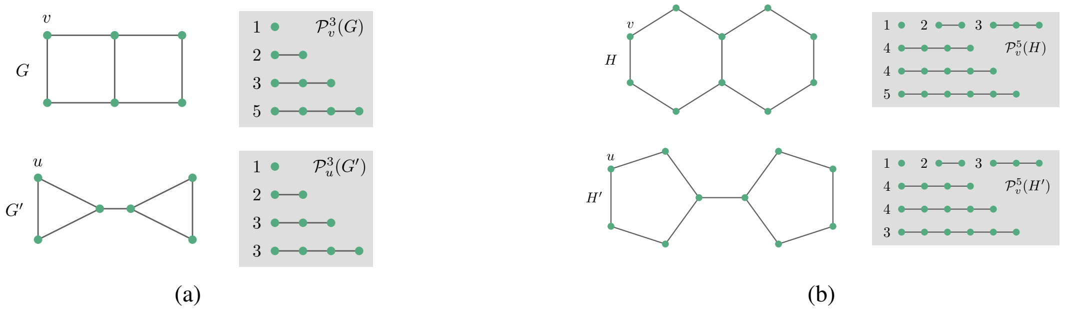


Figure 2: Examples of non-isomorphic graphs for which $1-$WL fails. (a): G and G' are $1-$WL-equivalent, but not $\mathcal{P}^3$-equivalent. (b): *Decalin* (H) and *Bicyclopentyl* (H') are two non-isomorphic molecules where the nodes represent carbon atoms and the edges chemical bonds. H and H' are $1-$WL-equivalent [30], but can be distinguished by $\mathcal{P}^5$.

While Theorem 3.5 exploits the maximum path length to distinguish two graphs, in practice a shorter path is often sufficient, as shown in the example above and summarized in the following proposition.

Proposition 3.7. *The graphs G and G' depicted in Figure 2a and the graphs H and H' in Figure 2b, are $1-$WL-equivalent. We state that*

$$G \nsim_{\mathcal{P}^l} G' \text{ with } l \geq 3 \qquad \text{and} \qquad H \nsim_{\mathcal{P}^l} H' \text{ with } l \geq 5.$$

The proposition extends the example in Figure 2 from nodes to the whole graph. Please refer to Proposition A.6 in the Appendix for more details.

We can further observe that the pairs of graphs in Figure 2 contain cycles of different lengths. In particular, the minimum path length necessary to discriminate the graphs is exactly the size of the minimum cycle. This remark is supported by the following theorem stating that, at the node level, paths can always identify cycles of different lengths, while $1-$WL cannot.

Theorem 3.8 (Cycles)**.** *Let γ_n and γ_m be two cycles of different lengths, with $n > m$. For any $v \in \gamma_n$ and any $u \in \gamma_m$,*

$$v \sim_{WL} u \quad \text{but} \quad v \nsim_{\mathcal{P}^l} u \quad \forall l \geq m.$$

Proof. The left-hand side, i.e., $v \sim_{WL} u$, comes from the fact that every cycle is isomorphic to a connected 2-regular graph (cf. Lemma A.5) and that $1-$WL cannot distinguish regular graphs at the node level. Let u and v be two arbitrary nodes in cycles γ_n and γ_m, respectively, and compute the multiset of paths $\mathcal{P}_v$ and $\mathcal{P}_u$. The longest path from each node in γ_m has length $m-1$ as γ_m is a cycle. Since $n > m$, the multiset of paths from a node in γ_n contains paths of length m so the two multisets $\mathcal{P}^l_v$ and $\mathcal{P}^l_u$ are different for every $l \geq m$. □

Theorem 3.8 can be extended to the class of Hamiltonian graphs.

Corollary 3.9. *Let $\mathcal{H}_n$ and $\mathcal{H}_m$ be two Hamiltonian graphs with n and m vertices, respectively, and $n > m$. For any $v \in \mathcal{H}_n$ and any $u \in \mathcal{H}_m$, it holds*

$$v \nsim_{\mathcal{P}^l} u \quad \forall l \geq m.$$

Proof. By definition, a Hamiltonian graph $\mathcal{H}$ is a graph that contains a cycle γ including all the vertices of the graph. Therefore, the length of the Hamiltonian path in $\mathcal{H}_m$ is $m-1$, which is the longest path in the graph, containing all possible nodes. The same argument holds for $\mathcal{H}_n$, with $n > m$. In particular, we are able to distinguish two graphs $\mathcal{H}_n$ and $\mathcal{H}_m$ by comparing their path multisets, i.e., $\mathcal{P}_v^l \neq \mathcal{P}_u^l \quad \forall l \geq m$. □

Another property of graphs that is closely related to the concept of paths is *connectivity*. In the following we prove that paths can distinguish pairs of regular graphs that are $1-$WL-equivalent but not isomorphic because one is connected and the other is disconnected. This implies that $1-$WL is unable to detect connectivity.

Theorem 3.10 (Connectivity)**.** *Let G be a disconnected graph of n nodes with two connected components which both are d-regular graphs, with $d \geq 2$ and let s be the size of the smallest component. Let G' be a connected d-regular graph on the same number of nodes. We can then distinguish almost every[1] couple of such graphs G, G', whereas $1-$WL cannot. In particular,*

$$G \sim_{WL} G' \quad \textit{but} \quad G \nsim_{\mathcal{P}^s} G'.$$

Proof. Theorem 1 of Robinson & Wormald [29] states that for $d \geq 3$ *almost every* $d-$regular graph is Hamiltonian and from Lemma A.5 we know that every connected $2-$regular graph is Hamiltonian. This, in conjunction with Corollary 3.9, concludes the proof. □

Examples for $d = 2$ and $d = 3$ can be seen in Figure 3.

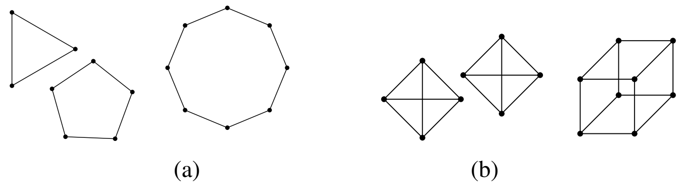


Figure 3: d-regular graphs of the same order which can be distinguished by (a) $\mathcal{P}^3$ and (b) $\mathcal{P}^4$. $1-$WL fails to distinguish the graphs.

Note that it is always possible to construct a connected $d-$regular graph by merging two $d-$regular disconnected components via a *degree-invariant* transformation; see Figure 4 for an example and refer to Definition A.7 in the Appendix for the general case.

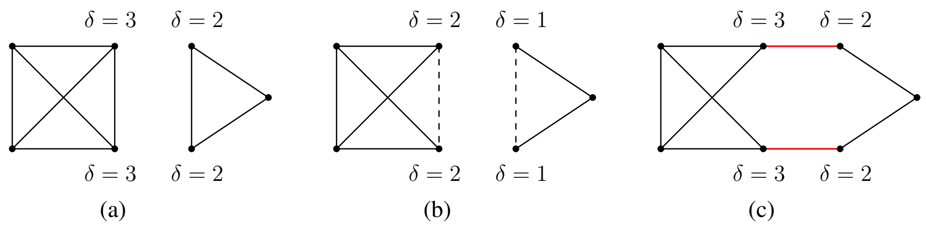


Figure 4: Example of a degree invariant transformation. (a) Two regular graphs of degree 3 and 2 respectively. (b)$-$(c) represent the consecutive steps of the transformation: first remove one edge and then link together the two graphs in such a way that the degree of the nodes is preserved.

[1]Almost all d-regular graphs of order n having a property P means $\lim_{n\to\infty} Pr(P) = 1$; refer to Robinson & Wormald [29] for details.

4 PATH-WL: A path-based Weisfeiler-Leman test

Considering our theoretical investigation into the expressive power of *paths*, we arrive at the general question: *Is the relation induced by paths more powerful than* $1-$*WL in distinguishing* ***any class*** *of graphs?* In the following theorem we answer this question by comparing the multiset of paths up to length l with $1-$WL at iteration l, and we prove the *incomparability* between the two relations, namely there are nodes that can be distinguished by $1-$WL, but not by $\mathcal{P}$ and vice versa.

Theorem 4.1. *$\mathcal{P}^l$-equivalence and WL-equivalence at iteration l are incomparable for any $l \in \mathbb{N}$.*

Proof sketch. For the proof, it is sufficient to find two pairs of nodes such that for the first pair, $1-$WL fails, while the multiset of paths is able to discriminate them, whereas for the second pair, the converse holds. We refer to Theorem A.8 in the Appendix for the complete proof. □

Remark 4.2. *Note that the pair of graphs presented in Theorem A.8 also serves as a counterexample for Theorem 3.3 in Michel et al. [24]. We refer to Appendix A.3 for more details.*

Given that we cannot compare the multiset of paths up to length l with the $l-$ th iteration of $1-$WL, we design a novel coloring algorithm for graphs, denoted by PATH-WL, which iterates over paths. We prove that PATH-WL is more powerful than the standard Weisfeiler-Lehman test.

Definition 4.3 (PATH-WL). *Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be a graph with node features $\mathbf{X}$ and $v \in \mathcal{V}$. Let $\mathbf{c}_v^{(i)} \in \Sigma$ be the color of the node v at iteration i, from a countable set of colors Σ. The initial color of the node v corresponds to the node feature, that is $\mathbf{c}_v^{(0)} := \mathbf{x}_v$. Let $\mathcal{P}$ be the multiset of colored paths, namely sequences of colors corresponding to the nodes of the paths. Let `HASH` be an injective function encoding every P_v^l with $\mathbf{c}_v \in \Sigma$. Then, the updating procedure is defined as*

$$\mathbf{c}_v^{(i)} = \mathtt{HASH}(\mathcal{P}_v^{l\,(i-1)}).$$

For example, $\mathcal{P}_v^{l\,(i-1)}$ will contain $(\mathbf{c}_v^{(i-1)})$ as path of 0 length and $(\mathbf{c}_v^{(i-1)}, \mathbf{c}_u^{(i-1)})$ for any $u \in N(v)$ as path of length 1 and so on.

Theorem 4.4. PATH-WL *is more expressive than* $1-$*WL.*

Proof. The proof consists of two parts: First, we demonstrate that PATH-WL is as expressive as $1-$WL. Then it is enough to show the existence of two nodes u and v that PATH-WL can distinguish, but such that $u \nsim_{WL} v$. Refer to Theorem A.9 in the Appendix for the full proof. □

PATH-WL is a generalization of the multiset of paths $\mathcal{P}$ to an iterative procedure. Please find some intuition on the importance of the iterations for improving expressiveness in Appendix A.4.

PATH-WL with MNS. PATH-WL can be made more expressive by marking, within a path, the neighbors of the starting node, which we refer to as *marking neighbors strategy* (MNS). Marking means to add a special color to all the nodes along the path which are neighbors of the starting node. In this way, two nodes with the same color for the standard PATH-WL will receive different colors if one is a neighbor of the starting node and the other is not. The expressive power of a test can also be described in terms of its ability to count substructures in the graph and PATH-WL with MNS can count cycles.

Theorem 4.5. PATH-WL *with MNS can count cycles.*

Proof sketch. We denote a marked node with an additional $*$, e.g., v^* denotes that v is a neighbor of the starting node of the path. For instance, triangles are exactly represented by paths of the form (v_1, v_2^*, v_3^*). Since the colors are updated based on the multiplicities of the paths, we can thus count triangles by considering the number of all paths of length 2 with two marked neighbors. The complete proof can be found in the Theorem A.10 in the Appendix. □

In addition to increasing the expressive power of PATH-WL, MNS is also of practical relevance as it can reduce the path length needed to distinguish two graphs. For instance, we need path multisets

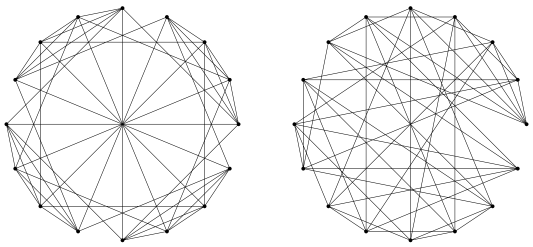


Figure 5: *Rook*'s 4x4 graph (*left*) and the *Shrikhande* (*right*) graph. They cannot be distinguished by $3-$WL but by path multisets of length up to 7. With marking neighbors, we can reduce the length of the paths to 4.

with paths up to length 7 to distinguish the smallest pair of non-isomorphic SR graphs (*Rook* and *Shrikhande*, cf. Figure 5), but paths of length up to 4 are sufficient if we mark neighbors, thus almost halving the required path length.

5 Experimental Analysis

To empirically evaluate our theoretical findings, we design PAIN (PATH ISOMORPHISM NETWORK), a graph neural network architecture with expressive power equivalent to PATH-WL, as specified in Definition 4.3. We perform experiments on two publicly available synthetic datasets for GNN expressiveness, (i) EXP [1] and (ii) SR [7].

The message-passing scheme for PAIN is defined in the following.

Definition 5.1 (PAIN)**.** *Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be a graph with node features $\mathbf{X}$. We initialize $\mathbf{h}_v^{(0)} = \mathbf{x}_v$ for all $v \in \mathcal{V}$. Let P_v^l be the multiset of paths up to length l starting at node v. The computation scheme at layer $i > 0$ is defined as*

$$\mathbf{h}_v^{(i)} = \mathtt{COMBINE}(\mathbf{h}_v^{(i-1)}, \mathtt{AGGREGATE}(\{\!\{\mathbf{z}_p^{(i-1)} \mid p \in P_v^l\}\!\}))$$

$$\mathbf{z}_p^{(i-1)} = f(p) = f((\mathbf{h}_v^{(i-1)}, \dots, \mathbf{h}_{v_l}^{(i-1)})),$$

where $\mathbf{h}_v^{(i)}$ is the feature vector of node v and $\mathbf{z}_p^{(i)}$ is the feature vector of the path p at the i-th layer. As we regard paths as ordered sequences, we use a function f operating on sequences to embed the path p into the embedding $\mathbf{z}_p$.

The output of PAIN *for a graph-level learning task after layer k is given by*

$$\mathbf{h}_G = \mathtt{READOUT}(\{\!\{\mathbf{h}_v^{(k)} \mid v \in \mathcal{V}\}\!\})$$

using a permutation-invariant `READOUT` function.

The functions `COMBINE`, `AGGREGATE` and `READOUT` are set to be the sum. The function f is modeled by an LSTM [18] with two layers. The choice is motivated by the universal approximation capabilities of LSTMs which can approximate any function on sequences [17]. In particular, in order to gain maximal expressive power, the function approximated by the LSTM must be injective and such that `AGGREGATE` (i.e., the sum) is injective over multisets (see Xu et al. [33, Lemma 5]).

Datasets. EXP contains 600 pairs of graphs representing propositional formulas. All non-isomorphic pairs of graphs are constructed such that $1-$WL fails, while $3-$WL can distinguish them. SR comprises pairs of strongly regular graphs ranging from 16 to 40 nodes. An instance of this dataset is visualized in Figure 5. $3-$WL fails to distinguish strongly regular graphs [15].

Experimental setup. For both datasets, we closely follow the experimental setup of Michel et al. [24]. We use our untrained PAIN to compute 16-dimensional embeddings. We consider two graphs as isomorphic if the Euclidean distance between their embeddings is below $\epsilon = 1e - 5$. We use Euclidean normalization on the input for the LSTM and use summation as a global pooling method to obtain a graph-level representation. Analogous to Michel et al. [24], for SR we restrict the path

Table 1: Results on the EXP dataset for paths of length 5. The values in the right column indicate the number of undistinguished non-isomorphic pairs of graphs.

Model	EXP ↓
GIN [33]	600
$3-$WL [23]	0
PAIN	0

length to 4 due to computational considerations, and use path length 5 for EXP. All presented results are repeated over 5 seeds. For SR, we mark neighbors by adding a constant value of 1.0 to all the neighbors of the starting node within a path. We use PyTorch [28] (version 1.13.1) with Python 3.10.9 and conduct our experiments on a single PC with an RTX-3080 GPU and Intel Core i9-11900KF CPU.

Results. For EXP, we obtain a **0% failure rate** with path length 5, which is consistent with the results in Michel et al. [24]. Please note that we, however, do not use distance encoding as proposed in Michel et al. [24] and could thus verify that the multiplicities of paths of length 5 are sufficient to distinguish all non-isomorphic pairs of graphs in the EXP dataset (see Table 1). For SR, we experimentally confirm that we can distinguish *Rook* and *Shrikhande* with path length 7 but we drastically reduce the length from 7 to 4 using MNS. With marking neighbors, PAIN is able to distinguish more than 50% of all SR graph pairs (cf. Figure 6). In general, our results are comparable with Michel et al. [24] with distance encoding. For SR(29,14,67), we obtain significantly better results of around 40% failure rate in comparison to their 80% failure rate.

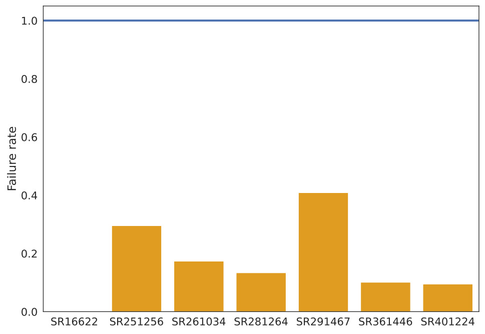


Figure 6: Results on SR for path length 4 and marked neighbors. The blue line indicates the failure rate of $3-$WL.

Time complexity. The time complexity of enumerating all possible paths of length at most l for some fixed l for one node in a graph G can be computed in $\mathcal{O}(D^l)$ using depth-first-search, where D denotes the maximum vertex degree in G. For a graph of order n, this yields an overall time complexity of $\mathcal{O}(nD^l)$ to compute all simple paths up to length l. Note that marking neighbors does not incur additional time complexity, as neighboring vertices are contained in paths of length one. Thus, one forward pass of PAIN with L layers has time complexity $\mathcal{O}(LnD^l)$.

6 Conclusion and Future Work

We investigated the discriminative power of paths to distinguish between non-isomorphic graph instances. Based on the equivalence between the counting logic fragment $\mathcal{C}_2$ and $1-$WL, we formally justified that $1-$WL is not able to recognize paths. We proposed PATH-WL, which is strictly more expressive than $1-$WL, and designed PAIN, a graph neural network with expressive power equivalent to PATH-WL. We could support our theoretical findings by performing experiments on the synthetic datasets EXP and SR. For future work, we aim to extend the investigation to other types of graphs, such as directed or dynamic graphs. We further want to generalize the presented coloring algorithm PATH-WL to $k-$PATH-WL. Given the promising experimental results on strongly regular graphs, we plan to further formalize the power of paths in distinguishing this particular class and also extend the investigation to $k-$isoregular graphs for which $k-$WL fails for every k. As the computational cost of our approach is strongly dependent on the chosen path length l, we plan to characterize graph classes for which it is proven that a reasonably low, fixed path length l is sufficient to be maximally expressive, or for which we can bound the number of paths (e.g., graphs with bounded tree-width).

Acknowledgements

This research has been partially supported by the Italian MUR PRIN 2022 project "MEDICA" (2022RNTYWZ) and by the Next Generation EU program project PNRR ECS00000017 - "THE - Tuscany Health Ecosystem" - Spoke 3 - CUP I53C22000780001. We further thank Fabian Jogl for sharing his precious experience with debugging, Sagar Malhotra for introducing us to the magic Ehrenfeucht-Fraïssé games, Giuseppe Alessio D'Inverno, Marco Tanfoni, Patrick Indri, Veronica Lachi, David Penz, Max Thiessen, and Pascal Welke for the useful discussions, insightful suggestions, and constant support.

References

[1] Ralph Abboud, İsmail İlkan Ceylan, Martin Grohe, and Thomas Lukasiewicz. The surprising power of graph neural networks with random node initialization. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pp. 2112–2118, 8 2021.

[2] Ralph Abboud, Radoslav Dimitrov, and Ismail Ilkan Ceylan. Shortest path networks for graph property prediction. In *Learning on Graphs Conference*, pp. 5–1. PMLR, 2022.

[3] Vikraman Arvind, Johannes Köbler, Gaurav Rattan, and Oleg Verbitsky. On the power of color refinement. In *Fundamentals of Computation Theory: 20th International Symposium, FCT 2015, Gdańsk, Poland, August 17-19, 2015, Proceedings 20*, pp. 339–350. Springer, 2015.

[4] Vikraman Arvind, Frank Fuhlbrück, Johannes Köbler, and Oleg Verbitsky. On weisfeiler-leman invariance: Subgraph counts and related graph properties. *Journal of Computer and System Sciences*, 113:42–59, 2020.

[5] László Babai. Lectures on graph isomorphism. *University of Toronto, Department of Computer Science. Mimeographed lecture notes*, 3:15, 1979.

[6] László Babai, Paul Erdos, and Stanley M Selkow. Random graph isomorphism. *SIaM Journal on computing*, 9(3):628–635, 1980.

[7] Muhammet Balcilar, Pierre Héroux, Benoit Gauzere, Pascal Vasseur, Sébastien Adam, and Paul Honeine. Breaking the limits of message passing graph neural networks. In *International Conference on Machine Learning*, pp. 599–608. PMLR, 2021.

[8] Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan, Michael M Bronstein, and Haggai Maron. Equivariant subgraph aggregation networks. In *International Conference on Learning Representations*, 2021.

[9] Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Lio, Guido F Montufar, and Michael Bronstein. Weisfeiler and lehman go cellular: Cw networks. *Advances in Neural Information Processing Systems*, 34:2625–2640, 2021.

[10] Cristian Bodnar, Fabrizio Frasca, Yuguang Wang, Nina Otter, Guido F Montufar, Pietro Lio, and Michael Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks. In *International Conference on Machine Learning*, pp. 1026–1037. PMLR, 2021.

[11] Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):657–668, 2022.

[12] Jin-Yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992.

[13] Giuseppe Alessio D'Inverno, Monica Bianchini, Maria Lucia Sampoli, and Franco Scarselli. On the approximation capability of gnns in node classification/regression tasks, 2023.

[14] Scott Fortin. The graph isomorphism problem. 1996.

[15] Frank Fuhlbrück, Johannes Köbler, and Oleg Verbitsky. Local wl invariance and hidden shades of regularity. *Discrete Applied Mathematics*, 305:191–198, 2021.

[16] Martin Grohe. The logic of graph neural networks. In *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pp. 1–17. IEEE, 2021.

[17] Barbara Hammer. On the approximation capability of recurrent neural networks. *Neurocomputing*, 31(1-4):107–123, 2000.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[19] Neil Immerman. *Descriptive complexity*. Springer Science & Business Media, 2012.

[20] Neil Immerman and Eric Lander. *Describing graphs: A first-order approach to graph canonization*. Springer, 1990.

[21] Lecheng Kong, Yixin Chen, and Muhan Zhang. Geodesic graph neural network for efficient graph representation learning. *Advances in Neural Information Processing Systems*, 35:5896–5909, 2022.

[22] Nils Morten Kriege. Weisfeiler and leman go walking: Random walk kernels revisited. In *Advances in Neural Information Processing Systems*, 2022.

[23] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *Advances in neural information processing systems*, 32, 2019.

[24] Gaspard Michel, Giannis Nikolentzos, Johannes F Lutzeyer, and Michalis Vazirgiannis. Path neural networks: Expressive and accurate graph neural networks. In *International Conference on Machine Learning*, pp. 24737–24755. PMLR, 2023.

[25] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[26] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 4602–4609, 2019.

[27] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45(2): 167–256, 2003.

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019.

[29] Robert W. Robinson and Nicholas C. Wormald. Almost all regular graphs are hamiltonian. *Random Structures & Algorithms*, 5(2):363–374, 1994.

[30] Ryoma Sato. A survey on the expressive power of graph neural networks. *arXiv preprint arXiv:2003.04078*, 2020.

[31] Erik Thiede, Wenda Zhou, and Risi Kondor. Autobahn: Automorphism-based graph neural nets. *Advances in Neural Information Processing Systems*, 34:29922–29934, 2021.

[32] Quang Truong and Peter Chin. Generalizing topological graph neural networks with paths. *arXiv preprint arXiv:2308.06838*, 2023.

[33] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[34] Viktor N Zemlyachenko, Nickolay M Korneenko, and Regina I Tyshkevich. Graph isomorphism problem. *Journal of Soviet Mathematics*, 29:1426–1481, 1985.

[35] Muhan Zhang and Pan Li. Nested graph neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 15734–15747, 2021.

A Appendix

A.1 Additional preliminaries

A relation R is an **equivalence** if and only if it is reflexive, symmetric, and transitive. Let R_1, R_2 be two relations and let $\geq$ be an ordering between them. If it holds that $R_1 \ngeq R_2$ and $R_1 \nleq R_2$, the ordering relation is partial, and R_1 and R_2 are said to be **incomparable**.

Graph theory. A **tree** is a connected acyclic graph. A **rooted tree** is a tree in which one vertex is chosen as the root. In the following, we will always consider trees as rooted.

Unfolding tree equivalence. Another equivalent way of testing the isomorphism of two graphs is comparing the *unfolding trees* (UT) rooted at their nodes.

Definition A.1 (Unfolding Tree). *The* ***unfolding tree*** $\mathbf{UT}_v^l$ *in graph* $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ *of node* $v \in \mathcal{V}$ ***up to depth*** $l \in \mathbb{N}_0$ *is defined as*

$$\mathbf{UT}_v^l = \begin{cases} Tree(\mathbf{x}_v, \emptyset) & \textit{if } l = 0 \\ Tree\big(\mathbf{x}_v, \mathbf{UT}_{N(v)}^{l-1}\big) & \textit{if } l > 0\,, \end{cases}$$

where $Tree(\mathbf{x}_v, \emptyset)$ *is a tree consisting of node* v *with feature* $\mathbf{x}_v$*.* $Tree\big(\mathbf{x}_v, \mathbf{UT}_{N(v)}^{l-1}\big)$ *is the tree consisting of the root node* v *and subtrees* $\mathbf{UT}_{N(v)}^{l-1} = \{\!\{\mathbf{UT}_u^{l-1} \mid u \in N(v)\}\!\}$ *of depth* $l-1$*. The* ***unfolding tree*** *of* v *is defined as* $\mathbf{UT}_v = \lim_{l\to\infty} \mathbf{UT}_v^l$*.*

D'Inverno et al. [13] and Kriege [22] showed that the unfolding tree and 1-WL are equivalent for testing the isomorphism of two graphs, i.e., the colors of the nodes after i iterations of 1-WL are the same if and only if the unfolding trees of depth i are isomorphic.

A.2 Proofs

The counting logic $\mathcal{C}_2$ is a two-variable fragment of first order logic (FOL) extended with counting quantifiers of the form $\exists^{\geq n} x \phi(x)$, which state that there are at least n x satisfying the formula ϕ. We can interpret those formulas in a graph by substituting the nodes in the graph to the variables in the formulas. Consider, for example, a graph G which has at least four nodes of degree three. Then, $G \models \Phi$ where

$$\Phi \equiv \exists x_1 \exists x_2 \exists x_3 \exists x_4 \; (Deg_3(x_1) \wedge Deg_3(x_2) \wedge Deg_3(x_3) \wedge Deg_3(x_4)).$$

The same formula can be expressed with just one variable in counting logic: $\exists^{\geq 4} x \, Deg_3(x)$.

Let $P_l(x, y)$ be the formula stating the existence of a path of length l from x to y. If the graph G satisfies the formula $P_l(x, y)$, i.e., $G \vDash P_l(x, y)$, then there exists a substitution of the two variables, e.g., v/x and u/y, such that there is an actual path from node v to node u of length l in G. Our aim is to prove that $P_l(x, y)$ cannot be expressed in the logic $\mathcal{C}_2$. This would mean that 1-WL (and equivalently GNNs) cannot encode the information of a path starting at that node in the colors of the nodes.

Theorem A.2. $P_l(x, y) \notin \mathcal{C}_2$*, for any* $l \geq 2$*.*

Proof. This proof is an adaptation from FOL to $\mathcal{C}$ of the proof for Proposition 6.15 presented in Immerman [19], Chapter 6. First, we recursively define the formula $P_l(x, y)$ stating that there exists a path from x to y of length at most l.

$$P_1(x, y) \equiv (x = y) \vee E(x, y)$$

$$P_l(x, y) \equiv (\exists z)(P_{l-1}(x, z) \wedge E(z, y)).$$

We claim that we need *exactly* three variables to define $P_{l+1}(x, y)$.

First, we observe that three variables are enough to define $P_l(x, y)$ as we can reuse the bound variables to redefine $P_l(x, y)$, as shown in in the following:

$$P_{l-1}(x, z) \equiv (\exists y)(P_{l-2}(x, y) \land E(y, z))$$

$$P_{l-2}(z, y) \equiv (\exists x)(P_{l-3}(z, x) \land E(x, y)).$$

What remains is to prove that three variables are necessary to define $P_l(x, y)$ in $\mathcal{C}_2$.

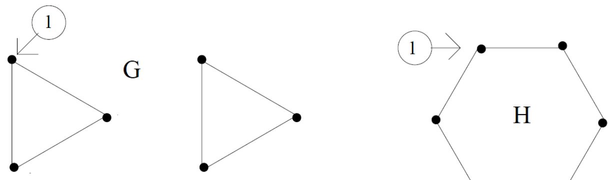


Figure 7: First move of the game with two pebbles.

In order to do that, we need the following intermediate result which will help us conclude the proof.

Proposition A.3. *The two graphs G and H in Figure 7 are such that $G \sim^2 H$. That is, Duplicator has always a winning strategy for the game with two pebbles for any possible number of moves.*

Proof. We prove the statement using the adaptation of the Ehrenfeucht-Fraïssé games for counting logic. The initial configuration has all the pebbles off the board. Spoiler first picks a subset of nodes in one graph and Duplicator responds with a subset of the same cardinality in the other graph. In the first move, it makes no difference which size of the subset Spoiler chooses, because wherever he puts the first pebble, Duplicator can easily win, putting her pebble on one random node. Suppose that the situation after the first move is the one depicted in Figure 7. For the second move, Spoiler chooses a subset of nodes in G (where Duplicator will be forced to play). There are several ways in which Spoiler can choose the subset but there are just two smart strategies: to choose a subset of nodes that obliges Duplicator to play on an adjacent node or on a non-adjacent one with respect to the first placed pebble. That is, Spoiler can choose the left triangle (adjacent strategy) or the right one (non-adjacent strategy). Duplicator can preserve the partial isomorphism in both cases, as can be seen in Figures 8 and 9.

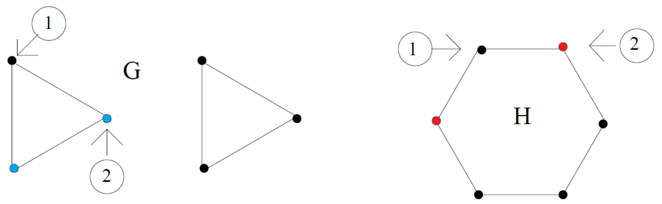


Figure 8: Second move of the game for the *adjacent strategy*. In blue is the subset chosen by Spoiler and in red the reply of Duplicator [19].

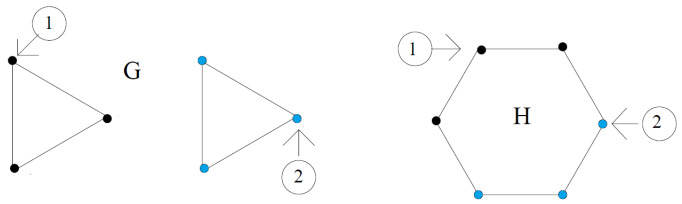


Figure 9: Second move of the game for the *non-adjacent strategy* [19].

□

Given Proposition A.3, we can conclude that the connectivity property is not expressible using two variables, no matter what the quantifier rank is. In particular, this property can be translated to the following formula

$$G = (\mathcal{V}, \mathcal{E}, \mathbf{X}) \text{ is connected } \iff \forall u, v \in \mathcal{V} \; \exists l \in \mathbb{N} \text{ s.t. } G \models P_l(u, v).$$

Suppose by contradiction that paths can be expressed by two variables, then connectivity with two variables would also be expressible, but this contradicts Proposition A.3 and concludes the current proof. □

See Immerman [19] for the definition of Ehrenfeucht-Fraïssé Games and Cai et al. [12] for the extension to counting quantifiers.

Theorem A.4. *Let G and H be two graphs of the same order $|G| = |H| = n$. G is a non-homogeneously traceable graph and H a Hamiltonian graph. Then,*

$$G \nsim_{\mathcal{P}^{n-1}} H.$$

Proof. By definition, a non-homogeneously traceable graph is a traceable graph such that at least one node in the graph is not an ending point of a Hamiltonian path. Formally, there exists a node $v \in G$ such that $p_v^{n-1} \notin \mathcal{P}_v^{n-1}(G)$, where $n-1$ is the length of the Hamiltonian path. A Hamiltonian graph is a graph with a Hamiltonian cycle. Any node in the Hamiltonian cycle is an ending point of a Hamiltonian path. Hence, for any node $u \in H$, $p_u^{n-1} \in \mathcal{P}_v^{n-1}(H)$. The conclusion follows. □

Lemma A.5. *Each connected 2-regular graph of n vertices is isomorphic to a cycle γ_n.*

Proof. We prove the statement above by induction on the number of vertices. The smallest connected 2-regular graph is a triangle, hence the induction base is $n = 3$. Suppose that $G' = (V', E')$ is a connected 2-regular graph with $n+1$ vertices. Consider one vertex $v \in V'$, which, by definition, has degree two and therefore two neighbors denoted by v_1 and v_2. v_1 and v_2 each have another neighbor u_1 and u_2 respectively, with $u_1 \neq u_2 \neq v$, and we thus exclude that they are connected to each other. Indeed, if this were the case, G' would have a disconnected component with 3 vertices. Let G be the graph obtained by removing vertex v from G' and connecting (v_1, v_2) with a new edge (See Figure 10). The resulting G is connected, 2-regular with n vertices hence, for the inductive hypothesis, it is isomorphic to a cycle on n vertices γ_n. If G is a cycle, then adding back the node v and connecting it to v_1 and v_2 as well as deleting the edge (v_1, v_2) is equivalent to adding a path of length two to the cycle. In this way, we obtain a cycle of length $n+1$, which is isomorphic to G'. □

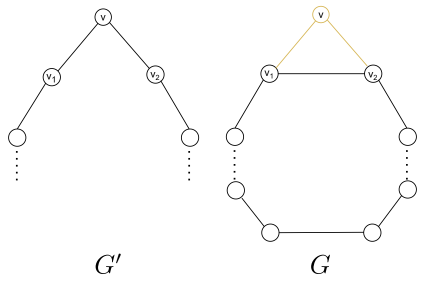


Figure 10: Sketch for the proof of Lemma A.5.

Proposition A.6. *Referring to the graphs in Figure 2,*

$$G \nsim_{\mathcal{P}^l} G' \text{ with } l \geq 3 \qquad \text{and} \qquad H \nsim_{\mathcal{P}^l} H' \text{ with } l \geq 5.$$

Proof. Let $G = (\mathcal{V}, \mathcal{E})$ and $G' = (\mathcal{V}', \mathcal{E}')$ be the graphs in Figure 2a. The two highlighted nodes, $v \in \mathcal{V}$ and $u \in \mathcal{V}'$, have different multisets of paths of length at most 3, i.e., $\mathcal{P}_v^3 \neq \mathcal{P}_u^3$. However, it is well known that $\mathbf{UT}_v = \mathbf{UT}_u$. We first prove that $\mathcal{P}_v^l \neq \mathcal{P}_u^l \quad \forall l \geq 3$.

This is a direct consequence of the following general fact

$$\mathcal{P}_v^k \neq \mathcal{P}_u^k \implies \mathcal{P}_v^l \neq \mathcal{P}_u^l \quad \forall l \geq k,$$

because $\mathcal{P}^k \subseteq \mathcal{P}^l$. The missing step of the proof consists of extending the equivalence from the nodes to the whole graphs G and G'. This can be trivially done given the strong symmetry that characterizes the two graphs. Indeed, consider assigning a different color to each different multiset of paths starting at each node. This will result in two colors (or equivalently, two partitions) for each of the two graphs. The first color is assigned to four of the nodes and the second color to the remaining two nodes. In the graph G, four nodes have the same color **c** as v (by symmetry), and at the same time, in the graph G', four nodes have the same color **c'**, as u. From the first part of the proof we know that $\mathbf{c} \neq \mathbf{c'}$ and this leads us to the conclusion. A similar argument holds for the graphs H and H' in Figure 2b. □

Definition A.7 (Degree-invariant transformation)**.** *Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be a graph. Let $T : \mathcal{E} \to \mathcal{V} \times \mathcal{V}$ be a transformation on the edges of G, such as the deletion and/or the creation of edges. The transformation is said to be degree-invariant if changing the edges does not change the degree of the nodes.*

See Figure 11 for an example of the transformation process which preserves the degree of the nodes.

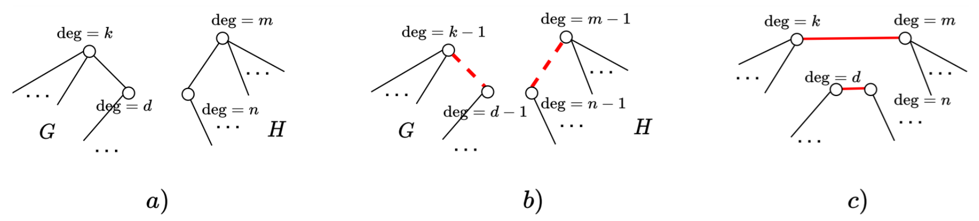


Figure 11: Example of a degree-invariant transformation. (a) The two graphs G and H can be any two graphs, with at least one edge. (b)−(c) represent the two steps of the transformation: first remove one edge and then link together the two structures in such a way that the degree of the nodes is preserved.

Theorem A.8. *$\mathcal{P}^l$-equivalence and WL-equivalence at iteration l are incomparable for any $l \in \mathbb{N}$.*

Proof. We prove the following equivalent formulation of the statement. There exist nodes u, v such that for some $k \in \mathbb{N}$

$$u \nsim_{\mathcal{P}^k} v \quad \text{and} \quad u \sim_{UT^k} v.$$

and there exist nodes u', v' such that for some $i \in \mathbb{N}$

$$u' \sim_{\mathcal{P}^i} v' \quad \text{and} \quad u' \nsim_{UT^i} v',$$

In order to prove the theorem it suffices to show two examples: (i) one example, where $\mathcal{P}$ is able to discriminate between two nodes but UT/WL fails, and (ii) a second example where the converse holds. For the first example, we can choose any graph class described before, or the famous instances shown in Figure 2. For the other direction, we refer to Figure 12. The highlighted nodes in the figure have different unfolding trees (hence, $u \nsim_{UT^3} v$) but they are indistinguishable by the multiset of paths, at least for length $l = 3$.

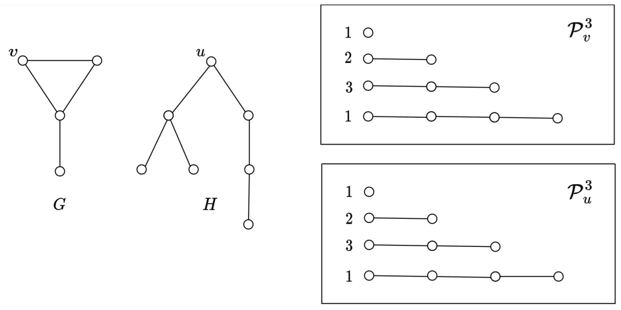


Figure 12: These two graphs serve as a counterexample for paths being more discriminative than unfolding trees (and thus 1-WL).

□

Theorem A.9. PATH-WL *is more expressive than 1-WL.*

Proof. The proof of the theorem consists of two parts: (i) we demonstrate that PATH-WL is as expressive as 1-WL, and (ii) we prove that PATH-WL is more expressive than 1-WL. For (i), we prove that for any length $l > 0$, for any nodes u, v and any iteration $i \in \mathbb{N}$,

$$u \nsim_{WL^{(i)}} v \Rightarrow u \nsim_{(\mathcal{P}^l)^{(i)}} v.$$

By the definition of the two coloring algorithms, this corresponds to

$$\texttt{HASH}\Big(\mathbf{c}_v^{(i)}, \{\!\{\mathbf{c}_w^{(i)} \mid w \in N(v)\}\!\}\Big) \neq \texttt{HASH}\Big(\mathbf{c}_u^{(i)}, \{\!\{\mathbf{c}_w^{(i)} \mid w \in N(u)\}\!\}\Big) \Rightarrow \texttt{HASH}(\mathcal{P}_v^{l\,(i)}) \neq \texttt{HASH}(\mathcal{P}_u^{l\,(i)})$$

Let α be the left-hand side of the implication. Due to the injectivity of the HASH function it is sufficient to show that α implies $\mathcal{P}_v^{l\,(i)} \neq \mathcal{P}_u^{l\,(i)}$. α is true if $\mathbf{c}_v^{(i)} \neq \mathbf{c}_u^{(i)}$ or if $\{\!\{\mathbf{c}_w^{(i)} \mid w \in N(v)\}\!\} \neq \{\!\{\mathbf{c}_w^{(i)} \mid w \in N(u)\}\!\}$, or both. Given that every element of $\mathcal{P}_v^{l\,(i)}$ is a sequence whose first element is $\mathbf{c}_v^{(i)}$, i.e.,

$$\mathcal{P}_v^{l\,(i)} := \{\!\{(\mathbf{c}_v^{(i)})\}\!\} \cup \{\!\{(\mathbf{c}_v^{(i)}, \mathbf{c}_w^{(i)})_{w\in N(v)}\}\!\} \cup \cdots \cup \{\!\{(\mathbf{c}_v^{(i)}, \mathbf{c}_w^{(i)}, \ldots, \mathbf{c}_y^{(i)})_{w\in N(v)\,\wedge\, y=\pi_l(p_v^l)^2}\}\!\},$$

we can simply conclude that if $\mathbf{c}_v^{(i)} \neq \mathbf{c}_u^{(i)}$ then $\mathcal{P}_v^{l\,(i)} \neq \mathcal{P}_u^{l\,(i)}$. Suppose that $\mathbf{c}_v^{(i)} = \mathbf{c}_u^{(i)}$. Then, $\{\!\{\mathbf{c}_w^{(i)} \mid w \in N(v)\}\!\} \neq \{\!\{\mathbf{c}_w^{(i)} \mid w \in N(u)\}\!\}$ implies $\{\!\{(\mathbf{c}_v^{(i)}, \mathbf{c}_w^{(i)})_{w\in N(v)}\}\!\} \neq \{\!\{(\mathbf{c}_u^{(i)}, \mathbf{c}_w^{(i)})_{w\in N(u)}\}\!\}$. Due to the fact that $\mathcal{P}^{l\,(i)}$ is a multiset of sequences of heterogeneous length, the fact that paths of length one are different is enough to conclude that $\mathcal{P}_v^{l\,(i)} \neq \mathcal{P}_u^{l\,(i)}$.

For (ii), we prove that PATH-WL is more expressive than 1-WL. This is accomplished by showing an instance of non-isomorphic graphs which 1-WL fails to distinguish but PATH-WL is able to distinguish (see Figure 13 for one such example). □

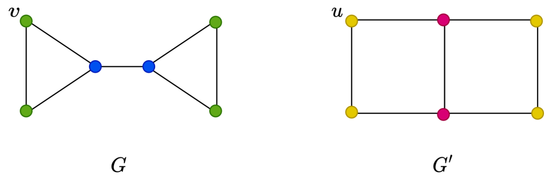


Figure 13: The coloring after one iteration of PATH-WL is enough to distinguish the two non-isomorphic graphs that 1-WL cannot distinguish. Note that the partition is the same but the colors of the nodes are different.

Theorem A.10. PATH-WL *with MNS can count cycles.*

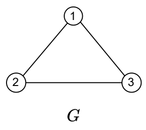
Proof. We identify a marked node with the symbol * and the multiset of marked paths with $\mathcal{P}^*$. Let G be a graph with n cycles of size s and G' be a graph with m cycles of the same size. Let v be a node belonging to a cycle. In $\mathcal{P}_v^*$, every cycle is identified by a path where the last node is a marked node. Therefore, even if the multiplicity of the paths of length $s-1$ is identical, the paths of the form $(v_1, \ldots, v_n^*)$ will be represented with a different multiplicity in the two graphs. □

A.3 Path-based Unfolding Tree

In the following, we introduce a novel type of unfolding tree and the equivalence induced on the nodes of a graph by this tree. We move from a simple consideration: every path in the unfolding tree of a node corresponds to a walk in the original graph, starting at that node. Therefore, we restrict our attention to paths in the original graph, branching from the vertex v, and we define a particular unfolding tree, based on these paths, that we call $\mathbf{PT}_v$. As paths are particular walks, $\mathbf{PT}_v$ is a subtree of the unfolding tree $\mathbf{UT}_v$, where each path in the tree is also a path in the original graph. In other words, along a single branch of the tree, no repetitions of nodes are allowed. Figure 14 depicts the difference between the two kinds of trees.

In the following, we state the formal definition of the *path-based* unfolding tree.

[2] $\pi_j(p_v^l)$ denotes the $j-$th node of path p_v^l



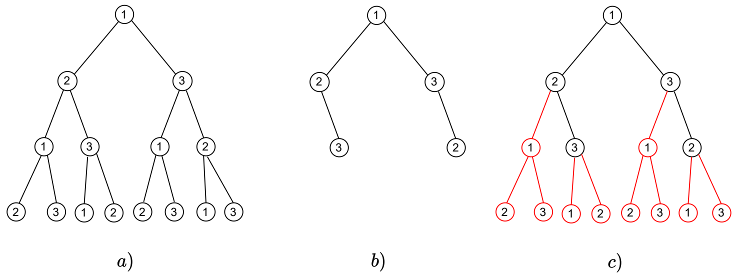


Figure 14: (a) Unfolding tree up to depth 3 of graph G for node 1 (i.e., $\mathbf{UT}_1^3$). (b) Path-based unfolding tree up to depth 3 of G for node 1 (i.e., $\mathbf{PT}_1^3$). Each branch of the tree is indeed a path in G. (c) In red, the difference between $\mathbf{UT}_1^3$ and $\mathbf{PT}_1^3$. In the *path-based* one, we stop unfolding the graph when we encounter an already-seen node.

Definition A.11 (Path-based unfolding tree)**.** *The* ***Path-based unfolding tree*** $\mathbf{PT}_v^l$ *in graph* $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ *of node* $v \in \mathcal{V}$ ***up to depth*** $l \in \mathbb{N}_0$ *is defined as*

$$\mathbf{PT}_v^l = \begin{cases} Tree(\mathbf{x}_v, \emptyset) & \text{if } l = 0 \\ Tree\big(\mathbf{x}_v, \overline{\mathbf{PT}}_{N(v)}^{l-1}\big) & \text{if } l > 0\,, \end{cases}$$

where $Tree(\mathbf{x}_v, \emptyset)$ *is a tree constituted of node* v *with feature* $\mathbf{x}_v$*.* $Tree\big(\mathbf{x}_v, \overline{\mathbf{PT}}_{N(v)}^{l-1}\big)$ *is the tree consisting of the root node* v *and subtrees* $\overline{\mathbf{PT}}_{N(v)}^{l-1} = \{\!\{\overline{\mathbf{PT}}_u^{l-1} \,|\, u \in N(v)\}\!\}$*.*
In particular, $\forall u \in N(v)$, $\overline{\mathbf{PT}}_u^{l-1}$ *is the tree obtained from* $\mathbf{PT}_u^{l-1}$ *by cutting every subtree rooted at* v*. Formally, we define the operator* $\texttt{CUT}_v : \mathcal{PT} \to \mathcal{C}$ *where* $\mathcal{PT}$ *is the domain of path-based unfolding trees and* $\mathcal{C}$ *is the set of cut trees. Then,* $\forall u \in N(v)$, $\overline{\mathbf{PT}}_u^{l-1} := \texttt{CUT}_v(\mathbf{PT}_u^{l-1})$ *where* $\texttt{CUT}_v$ *takes as input a path-based unfolding tree and returns the tree without all the occurrences of subtrees rooted at node* v*. Figure 15 provides a deeper insight into how the* $\texttt{CUT}_v$ *operator works.*

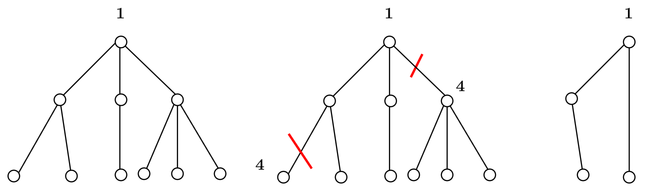


Figure 15: (a) $\mathbf{PT}_1^2$ (b) How the $\texttt{CUT}_4$ operator works in $\mathbf{PT}_1^2$ with respect to the vertex 4. In red, the cuts are shown. (c) The result of $\texttt{CUT}_4(\mathbf{PT}_1^2)$.

Remark A.12. *Note that, for* $l > 0$, $\mathbf{PT}_v^l \neq Tree\big(\mathbf{x}_v, \mathbf{PT}_{N(v)}^{l-1}\big)$*. Indeed, for every* $u \in N(v)$, $\mathbf{PT}_u^{l-1}$ *is a path-based tree rooted at* u *and contains the node* v *(trivially* $v \in N(N(v)) := \{x \in N(y) \,|\, y \in N(v)\}$*). Hence,* $Tree\big(\mathbf{x}_v, \mathbf{PT}_{N(v)}^{l-1}\big) \notin \mathcal{PT}$ *because it contains the repetition of node* v*.*

Remark A.13. $\mathbf{PT}_v^l$ *might be actually of depth* $k < l$*, in case no paths of length* l *are possible from* v *(see, e.g., Figure 14b)). Furthermore, in case* $|G| < \infty$, $\mathbf{PT}_v$ *is always a* ***finite*** *tree, that is* $\mathbf{PT}_v = \lim_{l\to\infty} \mathbf{PT}_v^l = \mathbf{PT}_v^L$ *where* L *is the length of the longest path from* v*.*

The equivalence induced by the path-based tree, the *PT-equivalence*, is defined in the following.

Definition A.14. *Two nodes $u, v \in \mathcal{V}$ are said to be **PT-equivalent**, noted by $u \sim_{PT} v$, if and only if they have the same path-based unfolding tree. Formally,*

$$u \sim_{PT} v \iff \mathbf{PT}_u = \mathbf{PT}_v.$$

Consequently, the same equivalence can be defined for graphs.

Definition A.15. *Two graphs $G = (\mathcal{V}, \mathcal{E})$ and $G' = (\mathcal{V}', \mathcal{E}')$ are said to be **PT-equivalent**, noted by $G \sim_{PT} G'$, if and only if there exists a bijection $b : \mathcal{V} \to \mathcal{V}'$ such that $\forall v \in \mathcal{V},\ v \sim_{PT} b(v)$.*

The previous definition can be rephrased for two graphs of the same cardinality and each node in $\mathcal{V}$ having a correspondent PT-equivalent node in $\mathcal{V}'$, then $G \sim_{PT} G'$.

With the following theorem, we state the existing relation between the WL/UT-equivalence and the PT-equivalence, given that $\forall v \in \mathcal{V},\ \mathbf{PT}^l_v \subseteq \mathbf{UT}^l_v$.

Theorem A.16. *PT-equivalence and UT-equivalence are incomparable.*

Proof. The proof consists of two considerations. The first one is that $\mathbf{PT}^l$ induces a stronger relation with respect to just the multiset of paths $\mathcal{P}^l$. Indeed, there exists a surjective function that maps the tree to the multiset of paths. This means that from the tree we can reconstruct the multiset of paths uniquely. It is not possible to reconstruct the tree from the multiset of paths, because the tree contains the topological information that is missing in the multiset. As different multisets correspond to different trees, Figure 2 serves as an example for proving one direction of the theorem. For the second example, which also serves as a counterexample for Theorem 3.3 in Michel et al. [24] see Figure 16. Note that the $\mathcal{AP}$-Tree in Michel et al. [24] is equivalent to the path-based unfolding tree $\mathbf{PT}$ that we contemporaneously defined. $\square$

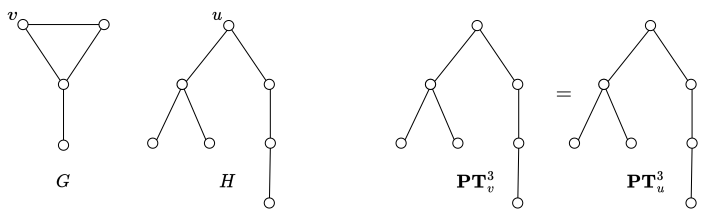


Figure 16: Counterexample showing that $\mathbf{PT}$-equivalence is not more expressive than 1-WL. Indeed $v \nsim_{WL} u$ but the path-based unfolding trees for u and v are identical.

A.4 Importance of iterations

In the following, we want to provide some intuition on the importance of the iterations for improving expressiveness. For instance, consider the two graphs in Figure 12. The two highlighted nodes v and u are distinguishable by 1-WL but they are $\mathcal{P}^3$-equivalent. Hence, the output of the first iteration of PATH-WL will result in the same color, see the upper part of Figure 17. If we perform another iteration, we can distinguish the two nodes, see Figure 17, bottom. Notably, iterations alone are already quite powerful because they assign colors to the nodes, reducing the path length needed to discriminate between non-isomorphic graphs (see Figure 18).

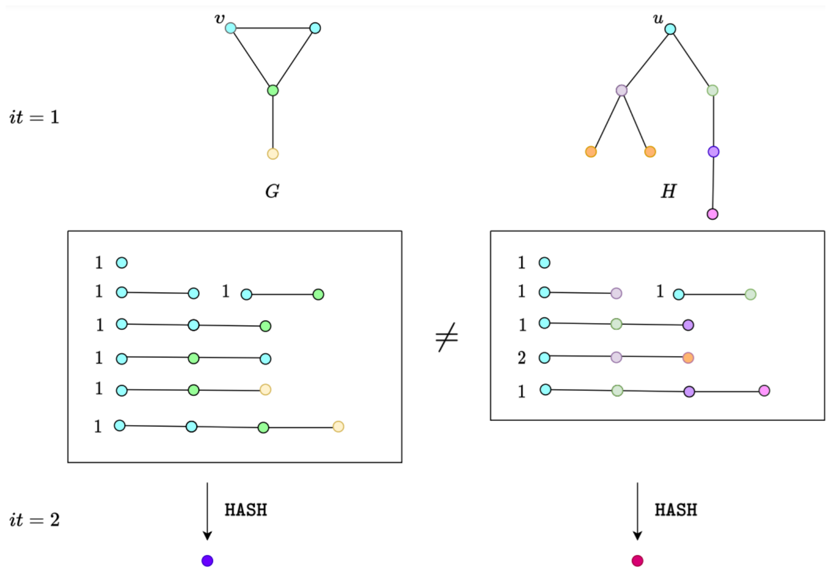


Figure 17: The two nodes v and u are indistinguishable by $\mathcal{P}^3$, indeed they have the same color after iteration 1. But computing the paths with colored nodes allows us to distinguish v and u.

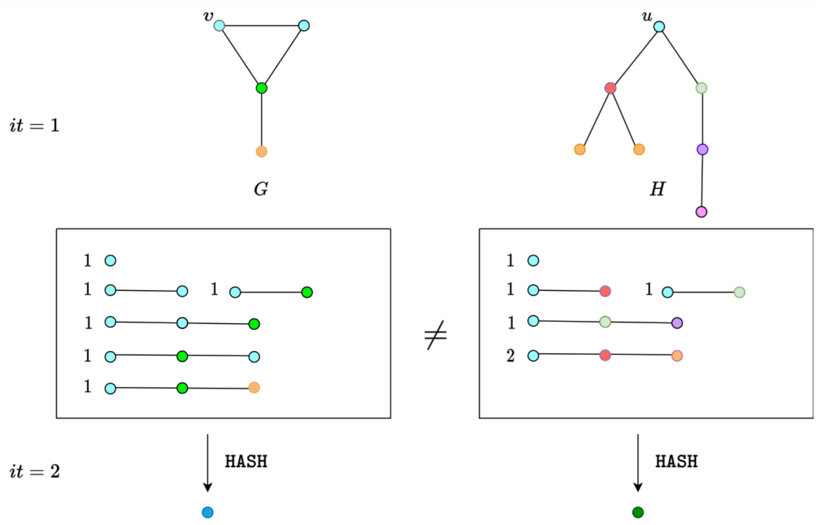


Figure 18: The two graphs are colored from the first iteration of PATH-WL with paths of length 2. The nodes u and v are not distinguishable, but the multiset of paths contains different paths. The output of the second iteration will result in different colors.