The Expressive Power of Path-Based Graph Neural Networks

Caterina Graziani [*1] **Tamara Drucks** [*2] **Fabian Jogl** [2 3] **Monica Bianchini** [1] **Franco Scarselli** [1]
Thomas Gärtner [2]

Abstract

We systematically investigate the expressive power of path-based graph neural networks. While it has been shown that path-based graph neural networks can achieve strong empirical results, an investigation into their expressive power is lacking. Therefore, we propose PATH-WL, a general class of color refinement algorithms based on paths and shortest path distance information. We show that PATH-WL is incomparable to a wide range of expressive graph neural networks, can count cycles, and achieves strong empirical results on the notoriously difficult family of strongly regular graphs. Our theoretical results indicate that PATH-WL forms a new hierarchy of highly expressive graph neural networks.

1. Introduction

We investigate the discriminative power of *paths* to increase the expressivity of graph neural networks (GNNs). The expressive power of a GNN commonly refers to its ability to compute different embeddings for non-isomorphic graphs. The most common GNN, the message-passing graph neural network, has been shown to be at most as expressive as the Weisfeiler-Leman (1-WL) color refinement algorithm (Xu et al., 2019; Morris et al., 2019). 1-WL is a polynomial-time heuristic for testing graph isomorphism that identifies almost all graphs (Babai et al., 1980) but systematically fails for some graph instances (see Figure 1 for an example). This shortcoming can be partially explained by the limitation of 1-WL to recognize and count only simple substructures such as star graphs (Arvind et al., 2020). The higher-order extension of 1-WL is k-WL (Babai, 1979; Immerman & Lander, 1990), which operates on k-tuples of nodes and is more powerful due to its ability to count additional substructures. However, k-WL is impractical due to its prohibitive complexity and lack of locality (Morris et al., 2019). To address these well-studied limitations, several novel GNNs that leverage graph substructures to improve expressivity have been recently proposed (Geerts, 2020; Thiede et al., 2021; Zhang & Li, 2021; Bodnar et al., 2021a;b; Bevilacqua et al., 2022; Bouritsas et al., 2022). Please refer to Papp & Wattenhofer (2022) for an extensive comparison of the expressive power of many such GNN extensions.

Related work. Paths are arguably one of the simplest graph substructures. Despite this, paths have only recently received attention in the context of GNNs. They can be broadly categorized into GNNs which incorporate shortest path information (Abboud et al., 2022; Kong et al., 2022; Li et al., 2020; Ding et al., 2023; Ying et al., 2021; Nikolentzos et al., 2020; Feng et al., 2022; Zhang et al., 2023) and GNNs which aggregate or sample from the set of all paths (Sun et al., 2022; Truong & Chin, 2023; Michel et al., 2023). Please find a more comprehensive related work discussion in Appendix A. While Michel et al. (2023) proved that aggregating shortest paths *alone* is less expressive than 1-WL, they achieve strong empirical results with the incorporation of shortest path distance information. However, a precise characterization of the expressive power of path-based GNNs with distance information is lacking. In this paper, we fill the existing gap in the literature and show that path-based GNNs with shortest path distances form a novel class of highly expressive graph neural networks.

Main contributions. We propose PATH-WL, a general class of color refinement algorithms based on paths and shortest path distance information. PATH-WL is an iterative procedure that performs message passing on all paths up to a certain length. We prove that PATH-WL is strictly more expressive than 1-WL and characterize graph classes that can be distinguished by PATH-WL. We show that PATH-WL can count cycles of arbitrary length and is incomparable to the k-WL algorithm as well as several other expressive GNNs. We design PAIN, a GNN with expressive power equivalent to PATH-WL, and empirically verify our theoretical results.

[*]Equal contribution [1]Department of Information Engineering and Mathematics, University of Siena, Siena, Italy [2]RUML, TU Wien, Vienna, Austria [3]CAIML, TU Wien, Vienna, Austria. Correspondence to: Caterina Graziani <caterina.graziani@student.unisi.it>, Tamara Drucks <tamara.drucks@tuwien.ac.at >.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

2. Preliminaries

Graph theory. We consider undirected graphs. Let $G = (V, E)$ be a **graph** with node set V and edges E. We refer to the number of nodes $|V|$ in G as the *order* of G. Let $N(v)$ be the *neighborhood* of a node $v \in V$, i.e. the set of all nodes adjacent to v, and $\delta(v)$ the *degree* of a node $v \in V$, i.e., the number of neighbors $|N(v)|$. For a graph G, we denote with Δ the maximum node degree of the graph. We define a node coloring as a function $X : V \to \Sigma$ with an arbitrary codomain Σ that we call *set of colors*. We denote a graph endowed with a node coloring by $G = (V, E, X)$ and $X(v)$ by x_v for $v \in V$. In the context of neural networks, we refer to the node coloring as the *node features*. A **path** $p = (v, v_1, \ldots, v_\ell)$ of length ℓ in a graph G is a sequence of non-repeated nodes connected through edges present in G. A **cycle** $C_\ell = (v_1, \ldots, v_\ell)$ of length ℓ is a sequence of adjacent and non-repeated nodes in G with the additional condition 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 $u, v \in V$ there exists a path connecting u and v. A graph G is d**-regular** if every node has the same degree d, i.e., $\forall v \in V\ \delta(v) = d$. A **strongly regular** graph $SR(n, k, \lambda, \mu)$ is a regular graph with n nodes and degree k such that (i) every two adjacent nodes have λ common neighbors, and (ii) every two non-adjacent nodes 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 includes a Hamiltonian cycle, i.e., a cycle that contains every node in the graph exactly once. Hamiltonian graphs are homogeneously traceable, but the converse is not necessarily true. For two graphs G and F, we write $sub(F, G)$ to denote the **number of subgraphs** of G isomorphic to F. Similarly, let $sub(F, G, u)$ be the number of subgraphs in G which include the node u and are isomorphic to F.

Weisfeiler-Leman test. The Weisfeiler-Leman test (1-WL), also known as *color refinement algorithm*, is a coloring procedure employed to test graph isomorphism (Fortin, 1996; Zemlyachenko et al., 1985).

Definition 2.1. Let $G = (V, E, X)$ be a graph with node coloring X and let Σ be a set of colors. Let $c_v^{(i)} \in \Sigma$ be the color of the node v at iteration i and let $c_v^{(0)} = x_v$. The Weisfeiler-Leman test updates the color of node v at each iteration $i > 0$ as follows:

$$c_v^{(i)} = \mathsf{HASH}\left(c_v^{(i-1)}, \left\{\!\!\left\{ c_u^{(i-1)} \mid u \in N(v) \right\}\!\!\right\}\right),$$

where HASH bijectively maps its input to a color from Σ.

1-WL partitions the nodes of a graph into equivalence classes, where equivalent nodes are assigned the same color. The algorithm terminates with a *stable* coloring when the partitioning does not change between iterations. To test whether two graphs are isomorphic, 1-WL is applied to both graphs. If the stable coloring of the two graphs differs, 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 stable coloring is the same, i.e., the two graphs *may be* isomorphic, but are not guaranteed to be. Immerman & Lander (1990) devised the more powerful extension k-WL, which colors k-tuples from V^k instead of single nodes (see Morris et al., 2019).

Given a coloring algorithm T, nodes u and v are called T*-equivalent*, denoted by $u \equiv_T v$, if they result in the same color after the termination of the algorithm T. Similarly, graphs G and H are T-equivalent if every node $v \in G$ is bijectively mapped to a node $u \in H$ s.t. $u \equiv_T v$. We denote the partitioning in equivalence classes induced by T with $p(T)$. Let T_1 and T_2 be two coloring algorithms. We write $T_1 \preceq T_2$ if T_2 is not less expressive than T_1, that is, every class in $p(T_1)$ is a union of classes in $p(T_2)$. We write $T_1 \prec T_2$ if T_2 is strictly more expressive than T_1 and $T_1 \equiv T_2$ iff $T_1 \preceq T_2$ and $T_2 \preceq T_1$.

Graph neural networks. Message-passing GNNs leverage the graph structure and the node features to learn a representation vector (or *embedding*). The specific task determines whether this embedding is learned for each node (*node-level task*) or for the entire graph (*graph-level task*).

Definition 2.2. Let $G = (V, E, X)$ be a graph with node features X. We initialize $h_v^{(0)} = x_v$ for all $v \in V$. The computation scheme of a message-passing graph neural network for iteration $i > 0$ is defined as

$$h_v^{(i)} = \mathsf{COMB}\left(h_v^{(i-1)}, \mathsf{AGG}\left(\left\{\!\!\left\{ h_u^{(i-1)} \mid u \in N(v) \right\}\!\!\right\}\right)\right),$$

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

$$h_v = \mathsf{READOUT}\left(h_v^{(k)}\right),$$

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

$$h_G = \mathsf{READOUT}\left(\left\{\!\!\left\{ h_v^{(k)} \mid v \in V \right\}\!\!\right\}\right).$$

The expressive power of message-passing graph neural networks can be studied in terms of their capability to distinguish non-isomorphic graphs. Xu et al. (2019) showed that with a sufficient number of GNN layers and injective COMB, AGG, and READOUT functions, the resulting GNN architecture is as expressive as 1-WL.

3. Path-WL: A Path-Based WL Test

In this section, we propose PATH-WL, a generalized class of color refinement algorithms to analyze the expressive power of path-based graph neural networks. The main difference between PATH-WL and 1-WL is that instead of aggregating neighbors, PATH-WL aggregates multisets of paths. Furthermore, higher-order variants of PATH-WL employ a *distance encoding*: for every node within a path of length ℓ, we concatenate the node color with the shortest path distance to the starting node. Note that we only add the shortest path distance to every node in the path that is at most d ℓ hops away from the starting node.

We define path multisets with distance encoding as follows.

Definition 3.1. Let $P_v^{`}$ be the set of all paths of length up to $\ell$ starting at node $v$. Based on $P_v^{`}$, we define the path multiset with distance encoding as

$$d\text{-}P_v^{`} := \quad \mathsf{c}_v, \eta^d_{vv} \quad , \quad \mathsf{c}_{v_1}, \eta^d_{vv_1} \quad , \ldots , \quad \mathsf{c}_{v_k}, \eta^d_{vv_k} \quad j\,(v, v_1, \ldots, v_k)\ 2\ P_v^{`}, k \quad \ell \quad ,$$

where c_v is the color of node v and $\eta^d_{vv_i}$ is the shortest path distance from v to v_i if the shortest path distance is less or equal to d and $;$ otherwise.

Now we can introduce the iterative color refinement algorithm PATH-WL.

Definition 3.2 (PATH-WL)**.** Given a graph $G = (V, E, \mathsf{X})$, a set of colors Σ and an injective function HASH, d-PATH-WL$^{`}$ refines node colors as follows. The initial color of node v 2 V corresponds to the node coloring, that is $\mathsf{c}_v^{(0)} = \mathsf{x}_v$. The color of node v 2 V at iteration $i > 0$ is updated as

$$\mathsf{c}_v^{(i)} = \mathsf{HASH} \quad d\text{-}P_v^{`;(i \ \ 1)} \quad .$$

We denote d-PATH-WL$^{`}$ after $i$ iterations by $d$-PATH-WL$^{`;(i)}$. To keep notation simple, we sometimes omit the distance d, the path length ℓ, or the number of iterations i if not relevant in the discussed context. Here, we highlight two variants of d-PATH-WL with interesting theoretical properties:

0-PATH-WL. The simplest variant of PATH-WL is 0-PATH-WL, which incorporates no additional distance information. Despite its simplicity, 0-PATH-WL is strictly more expressive than 1-WL and a single iteration can distinguish a variety of 1-WL-equivalent graph families (see Section 4.3). As an example, consider the graphs G and G^0 in Figure 2. With uniform node features, these two graphs are 1-WL-equivalent. However, the first iteration of 0-PATH-WL can already distinguish these graphs, as shown in the following proposition.

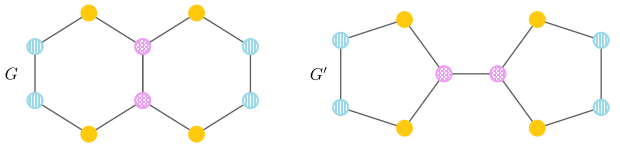

Figure 1: Stable graph coloring after the first iteration of the Weisfeiler-Leman algorithm.

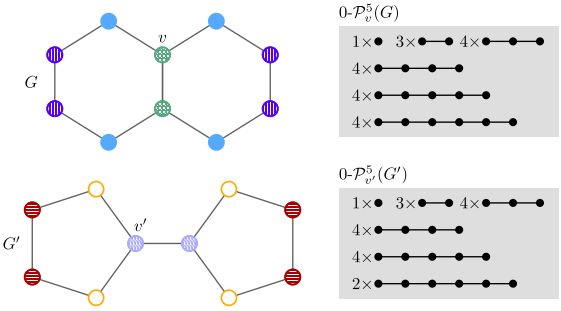

Figure 2: Example of non-isomorphic graphs for which 1-WL fails, but PATH-WL can distinguish them in the first iteration. In the grey boxes, we visualize the path multisets up to length 5 for nodes v and v^0.

Proposition 3.3. *Let G, G^0 be the graphs in Figure 1. For every ℓ 5, G 0-PATH-WL$^{`;(1)}$ G^0, while G 1 WL G^0.*

Proof sketch. To show G 1-WL G^0, refer to Figure 1, where the nodes of the graphs are colored according to 1-WL. The partitioning of the nodes and the node colors (i.e., 1-WL equivalence classes) are the same in the two graphs, thus G and G^0 are 1-WL-equivalent. Figure 2 shows the graph coloring after the first iteration of 0-PATH-WL with path length five. The nodes in the two graphs have different colors, which implies that they are not 0-PATH-WL-equivalent. On the right-hand side of the figure, inside the grey boxes, we represent the multiset of paths of length up to five for two nodes, v in G and v^0 in G^0. Due to the different multiplicities of paths of length five, we can conclude that v 0-PATH-WL v^0. Indeed, 0-P^5_v contains four paths of length five, while 0-$P^5_{v^0}$ contains only two paths of length five. The complete proof can be found in Appendix C.1. □

1-PATH-WL. Distance encoding with $d = 1$ allows 1-PATH-WL to count cycles at the node level and suffices to prove that 1-PATH-WL is not contained within the k-WL hierarchy. Furthermore, this implies that 1-PATH-WL is not bounded in expressivity by several other powerful GNNs such as subgraph GNNs (You et al., 2021; Bevilacqua et al., 2022; Qian et al., 2022) or Local 2-GNNs (Morris et al., 2020b; 2022; Zhang et al., 2023; Frasca et al., 2022). See Section 4.1 and Section 4.2 for more details.

The expressive power of d-PATH-WL$^{\ell}$ is monotonically non-decreasing with respect to the path length ℓ and shortest path distance d ℓ.

Proposition 3.4. *For every d' d 0 and ℓ' ℓ 1, it holds that*

$$d\text{-PATH-WL}^{\ell} \preceq d'\text{-PATH-WL}^{\ell},$$

and

$$d\text{-PATH-WL}^{\ell} \preceq d\text{-PATH-WL}^{\ell'}.$$

Refer to the Appendix C.2 for the proof of Proposition 3.4. Note that to ensure a monotonic increase of the expressive power with respect to the path length ℓ, it is important to consider the multiset of *all* paths *up to* length ℓ. Indeed, shorter paths can be crucial to distinguish two graphs, while longer paths can be identical.

Time complexity. The time complexity of enumerating all possible paths of length at most ℓ for one node in a graph G can be computed in $O(\Delta^{\ell})$ using depth-first-search. For a graph of order n, this yields an overall time complexity of $O(n\Delta^{\ell})$ to compute all paths up to length ℓ. Note that we can compute the distance encoding with minimal computational overhead, as shortest paths are a subset of all paths. Overall, PATH-WL with i iterations and path length ℓ has time complexity $O(in\Delta^{\ell})$. Thus, the complexity of PATH-WL is bounded as a function of the maximum degree of the graph, whereas k-WL and other higher-order extensions are bounded as a function of the order of the graph (Morris et al., 2019; Bouritsas et al., 2022; Maron et al., 2019b; Bodnar et al., 2021a). This means that PATH-WL can be efficient on sparse graphs such as molecular graphs, since many real-world molecular datasets have an average degree of less than three (Wale & Karypis, 2006; Morris et al., 2020a; Wallach et al., 2015). In practice, it is often not necessary to compute *all* paths and a small path length often suffices for maximal expressivity (see Section 5).

We further point out that the least expressive variant, 0-PATH-WL, is at least as expressive as pathNN, the path-based GNN proposed by Michel et al. (2023).

Remark 3.5. For every ℓ 1, it holds that 0-PATH-WL$^{\ell;(\cdot)}$ $\preceq$ pathNN with path length ℓ.

Both, PATH-WL and pathNN pre-compute all paths of length up to ℓ. In contrast to PATH-WL, pathNN does not aggregate all paths of length up to ℓ at the same time, but instead first aggregates paths of length one, then paths of length two, and so on. This limits the number of iterations to at most ℓ whereas PATH-WL is not limited in the number of iterations. As we compare the two architectures with respect to the same length ℓ, this constrains the maximum number of layers of pathNN to be equal to ℓ. Conversely, in PATH-WL the length of the paths and the number of iterations are independent. This is a strength of PATH-WL, since increasing the number of iterations can reduce the path length needed for improving the expressive power (see Section 5 and Appendix C.2). This can lead to a significant decrease in runtime as it only scales linearly with the number of iterations but exponentially with the path length. We refer to the Appendix C.3 for a more in-depth discussion on Remark 3.5.

4. The Discriminative Power of Paths

In this section, we present our main results on the expressive power of path-based graph neural networks. For this, we analyze path-based graph neural networks within the mathematical framework of d-PATH-WL. First, in Section 4.1 we show how d-PATH-WL relates to k-WL. Then, in Section 4.2 we analyze its ability to count cycles. Finally, in Section 4.3 we characterize which graph families can be distinguished by d-PATH-WL in only a single iteration. We provide a summary of our key theoretical results in Table 1.

4.1. Relation to the k-WL Hierarchy

We first show that for every path length and shortest path distance, d-PATH-WL is more expressive than 1-WL (cf. Proposition 4.1 and Remark 4.2). Next, we prove that 1-PATH-WL is incomparable to k-WL (cf. Theorem 4.3).

Proposition 4.1. *For every $\ell > 1$, every d 0, d-PATH-WL$^{\ell}$ is strictly more expressive than 1-WL.*

Proof sketch. The proof consists of two parts: first, we prove that 0-PATH-WL is always at least as expressive as 1-WL by leveraging the injectivity of the HASH function. Then, we provide an example of two graphs G, G' that 0-PATH-WL can distinguish, but such that G $\not\equiv_{\text{1-WL}}$ G' (e.g., see Figures 1 and 2). The conclusion of the proof follows from the monotonicity of the expressive power of d-PATH-WL (cf. Proposition 3.4). See Proposition C.4 in the Appendix for the complete proof. □

Remark 4.2. Note that for path length one it holds that d-PATH-WL^{1} 1-WL, for every d 0.

Proposition 4.1 states that even 0-PATH-WL^{2}, which simply aggregates path multisets in each iteration, is more expressive than the standard Weisfeiler-Leman test. This implies that any graph neural network with expressive power greater than or equal to 0-PATH-WL is more expressive than the entire class of message-passing graph neural networks since they are limited by 1-WL.

We show a stronger result on the relation to the k-WL hierarchy, for d-PATH-WL with d 1.

Theorem 4.3. *For every $d \geq 1$, $k \geq 3$, d-PATH-WL and k-WL are incomparable. Equivalently, the following holds:*

(1) for every $k \geq 3$ there exists a path length ℓ such that d-PATH-WL$^{\ell}$ $\not\equiv$ k-WL;

(2) for every $\ell \geq 1$, there exists a k such that k-WL $\not\equiv$ d-PATH-WL$^{\ell}$.

Proof sketch. For the proof, we construct pairs of non-isomorphic graphs which are (1) k-WL-equivalent but can be distinguished by d-PATH-WL with sufficient path length, and (2) d-PATH-WL-equivalent but can be distinguished by 3-WL. The first direction builds on the fact that d-PATH-WL with $d \geq 1$ can distinguish between graphs with different cycle counts for any cycle length ℓ (see Corollary 4.6) whereas k-WL fails to do so for cycles of length $\ell \geq \frac{k^2}{2}$ (Neuen, 2024, Theorem 1.3). For the other direction, we show that for any fixed length ℓ, we can always construct two graphs of treewidth two that cannot be distinguished by d-PATH-WL$^{\ell}$ but can be distinguished by 3-WL (Kiefer & Neuen, 2022). We refer to the Appendix C.6 for the full proof. □

Theorem 4.3 states that for every k, there exist pairs of non-isomorphic graphs that d-PATH-WL can distinguish whereas k-WL fails and vice-versa. This shows that the two hierarchies are *incomparable* and that d-PATH-WL with $d \geq 1$ thus forms a novel hierarchy of expressive color refinement algorithms. We point out that the proof of Theorem 4.3 provides an *upper bound* on the required path length; a shorter path length is often sufficient (see Appendix C.4). Note that this result depends on the ability to count cycles, which is guaranteed by incorporating neighborhood information, as shown in the following section.

4.2. Counting Cycles

The expressive power of a test can also be described in terms of its ability to count substructures in the graph. Similar to Chen et al. (2020) and Huang et al. (2022), we define the counting power of a test by its ability to distinguish between graphs with different substructure counts. We first show that 0-PATH-WL can distinguish between cycles of different lengths at node level.

Proposition 4.4. *Let C_n and C_m be two cycle graphs of different order, with $n > m$. For any $v \in C_n$ and any $u \in C_m$,*

$$v \equiv_{1\text{-WL}} u \quad \text{but} \quad v \not\equiv_{0\text{-PATH-WL}^{\ell};(1)} u \quad \forall \ell \geq m.$$

Proof. The left-hand side, i.e., $v \equiv_{1\text{-WL}} u$, comes from the fact that every cycle is a connected 2-regular graph (cf. Lemma C.7) and that 1-WL cannot distinguish d-regular graphs for any d at the node level. Let u and v be two arbitrary nodes in cycles C_n and C_m, respectively, and compute the first iteration of 0-PATH-WL for u and v. This consists of computing the multisets of paths $0\text{-}P_u$ and $0\text{-}P_v$. Note that the longest path from each node in C_m has length $m - 1$ as C_m is a cycle. Since $n > m$, the multiset of paths from a node in C_n contains paths of length m so the two multisets $0\text{-}P_v^{\ell}$ and $0\text{-}P_u^{\ell}$ are different for every $\ell \geq m$. □

Our next theorem states that with the incorporation of neighborhood information, i.e., d-PATH-WL with $d \geq 1$, we can count cycles of arbitrary length at node level:

Theorem 4.5 (Node-level cycle counting)**.** *Let $sub(C^{\ell}, G, v) \neq sub(C^{\ell}, H, u)$ for some graphs G, H, nodes u, v and cycle C^{ℓ}. Then,*

$$u \not\equiv_{1\text{-PATH-WL}^{\ell}} v.$$

Proof sketch. Given a node v, its neighbors v_i are identified by pairs $(\mathtt{c}_{v_i}, 1)$ (see Definition 3.1). Thus, triangles on nodes v, v_1, v_2 are represented by paths of length two such as $((\mathtt{c}_v, 0), (\mathtt{c}_{v_1}, 1), (\mathtt{c}_{v_2}, 1))$. These can be counted with 1-PATH-WL$^{2;(1)}$. In general, cycles correspond to paths where the last node is a neighbor of the starting node. The full proof can be found in Theorem C.8 in the Appendix. □

Note that being able to count cycles at node level is stronger than counting cycles at graph level. Indeed, node-level counting implies graph-level counting, but the opposite is not true. As a corollary of Theorem 4.5, 1-PATH-WL can distinguish between two graphs with different cycle counts:

Corollary 4.6 (Graph-level cycle counting)**.** *Let $sub(C^{\ell}, G) \neq sub(C^{\ell}, H)$ for some graphs G, H and cycle C^{ℓ}. Then, $G \not\equiv_{1\text{-PATH-WL}^{\ell}} H$.*

While Corollary 4.6 is of great interest from a theoretical point of view (Arvind et al., 2020; Chen et al., 2020; Huang et al., 2022), this result is also of practical relevance as it can reduce the path length needed to distinguish two graphs. For instance, we can distinguish the smallest pair of non-isomorphic strongly regular graphs *Rook* and *Shrikhande*, cf. Figure 3, with multisets of paths up to length seven (which is in line with Arvind et al. (2020, Fig. 7)), but paths of length up to four are sufficient if we use neighbor information, thus almost halving the required path length. This reduces the runtime by a factor of 200.

We further combine Corollary 4.6 with the analysis of Zhang et al. (2024) that investigates whether different GNNs can count cycles. This allows us to prove that PATH-WL is not bounded in expressivity by Subgraph GNNs (You et al., 2021; Bevilacqua et al., 2022; Qian et al., 2022), Local 2-GNNs (Morris et al., 2020b; 2022; Zhang et al., 2023; Frasca et al., 2022), and Folklore k-GNNs (Maron et al., 2019a; Zhang et al., 2023; Feng et al., 2023).

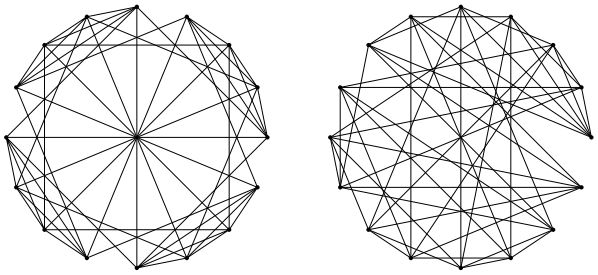

Figure 3: *Rook*'s 4x4 graph (*left*) and *Shrikhande* (*right*). They cannot be distinguished by 3-WL but by 0-PATH-WL$^{7;(1)}$ and 1-PATH-WL$^{4;(1)}$.

Corollary 4.7. *For every $k \geq 2$, there exists a path length ℓ such that 1-PATH-WL$^{\ell;(1)}$ $\not\equiv$ SubgraphGNN, Local 2-GNN, Folklore k-GNN.*

Please find more details in Corollary C.9 in the Appendix.

4.3. One Iteration Is Almost All You Need

In this section, we investigate graph families that can be distinguished by 0-PATH-WL with only *one* iteration. Formally, let $\mathcal{G}$ and $\mathcal{H}$ be two disjoint graph families. We say that a coloring algorithm T can distinguish $\mathcal{G}$ and $\mathcal{H}$, if for every pair of graphs $G \in \mathcal{G}$ and $H \in \mathcal{H}$ it holds that $G \not\equiv_T H$. Note that all results in this section generalize to d-PATH-WL with $d \geq 0$ as well as to pathNN (Michel et al., 2023).

Theorem 4.8. *There exists a path length ℓ such that 0-PATH-WL$^{\ell;(1)}$ can distinguish the following pairs of infinite graph families:*

(1) Hamiltonian graphs of different orders at node and graph level,

(2) Hamiltonian graphs and non-homogeneously traceable graphs at graph level, and

(3) almost all connected d-regular graphs and disconnected graphs with d-regular connected components at graph level.

Proof sketch. For the proof, it is sufficient to show that path multisets can distinguish the graph families, as this is equivalent to 0-PATH-WL with one iteration. Please refer to Section C.5.1 in the Appendix for the full proof. □

Note that the graph families in Theorem 4.8 (2) & (3) contain graph classes which are 1-WL-equivalent. For (2), consider the construction in Corollary 4.9. For (3), it is well known that 1-WL is not able to distinguish d-regular graphs for any d. Examples for $d = 2$ and $d = 3$ can be found in Figure 5.

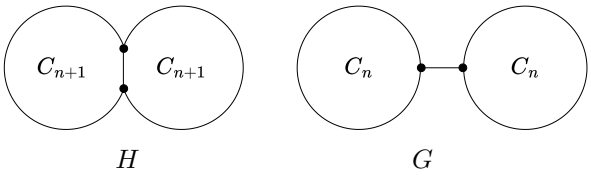

Figure 4: Two non-isomorphic 1-WL-equivalent graphs that can be distinguished by 0-PATH-WL$^{2n\ 1;(1)}$.

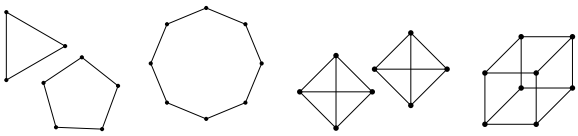

Figure 5: Graphs that are 1-WL-equivalent but can be distinguished by 0-PATH-WL$^{\ell;(1)}$ with (a) $\ell = 3$ and (b) $\ell = 4$. The required length ℓ corresponds to the order of the smallest connected component.

Corollary 4.9. *Let H be a graph of order $2n$ composed of two cycles C_{n+1} with an edge in common. Let G be a graph of order $2n$ composed of two cycles C_n connected by an extra edge. For any $n \geq 3$, it holds that*

$$G \not\equiv_{\text{0-PATH-WL}^{2n\ 1;(1)}} H \quad \textit{and} \quad G \equiv_{\text{1-WL}} H.$$

See Figure 4 for an illustration of how H and G are constructed. Note that the graphs in Figure 2 represent an instance of this graph construction for $n = 5$.

We have shown that even the simplest version of PATH-WL, 0-PATH-WL with one iteration, can already distinguish between graph families which are 1-WL-equivalent. However, we prove in the following theorem that 0-PATH-WL with one iteration is not more expressive than 1-WL.

Theorem 4.10. *1-WL-equivalence at iteration ℓ and 0-PATH-WL$^{\ell;(1)}$-eqivalence are incomparable for every $\ell \geq 3$.*

Proof sketch. For the proof, it suffices to show that there exists a pair of nodes such that (i) 0-PATH-WL$^{\ell;(1)}$ is able to distinguish them, while 1-WL with ℓ iterations fails and (ii) vice-versa. For (i), please refer to Corollary 4.9 for an example. For (ii), please see the counterexample in Figure 6. See Theorem C.14 in the Appendix for the full proof. □

Note that the graphs in Figure 6 provide a counterexample for a recent result of Michel et al. (2023, Theorem 3.3). In contrast, 0-PATH-WL with iterations is able to distinguish the graphs. Indeed, already path length three and only two iterations are sufficient.

Table 1: Summary of our theoretical results.

Theoretical Result	Requirement	Reference
0-PATH-WL` ≡ 1-WL	$\ell = 1$	Rem. 4.2
0-PATH-WL` ≺ 1-WL	$\ell > 1$	Thm. 4.1
0-PATH-WL`;(1) incomparable to 1-WL(`)	$\ell \geq 3$	Thm. 4.10
0-PATH-WL`;(`) ≡ pathNN(`)	$\ell \geq 1$	Rem. 3.5
d-PATH-WL` ≺ $d'$-PATH-WL`	$\ell \geq 1, d' \geq d \geq 0$	Prop. 3.4
d-PATH-WL` ≺ $d$-PATH-WL`$'$	$\ell' \geq \ell \geq 1, d \geq 0$	Prop. 3.4
1-PATH-WL` can count n-cycles	$\ell \geq n \geq 1$	Thm. 4.5
1-PATH-WL` ⊴ $k$-WL $k$-WL ⋬ 1-PATH-WL`	$\ell \geq \frac{k^2}{2}, k \geq 3$ $\ell \geq 1, k \geq 3$	Thm. 4.3 (1) Thm. 4.3 (2)
1-PATH-WL` ⊴ SubgraphGNN 1-PATH-WL` ⊴ Local 2-GNN 1-PATH-WL` ⊴ Folklore 2-GNN	$\ell \geq 7$	Cor. 4.7
1-PATH-WL` ⊴ Folklore k-GNN	$\ell \geq \frac{k^2}{2}, k \geq 3$	Cor. 4.7

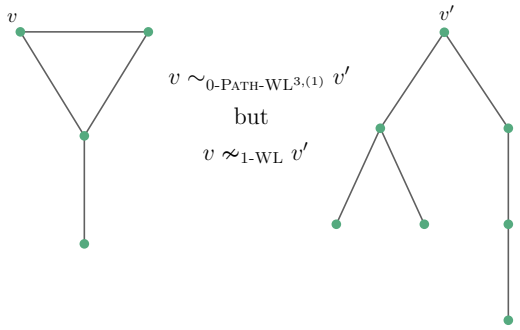

Figure 6: Counterexample that shows that path multisets alone are not more expressive than 1-WL.

5. Experimental Analysis

To empirically evaluate our findings, we design PAIN (PATH ISOMORPHISM NETWORK), a family of GNN architectures with expressive power equivalent to PATH-WL (cf. Remark 5.1). We evaluate PAIN on three datasets designed for studying the expressivity of GNNs and on two real-world benchmark datasets. We provide additional information about our experiments in Appendix D.

Remark 5.1. We say that PAIN has expressive power equivalent to PATH-WL if there exists a set of weights that allows PAIN to match the expressivity of PATH-WL.

The PAIN Family. Let $G = (V, E, \mathbf{X})$ be a graph with node features $\mathbf{X}$. Each GNN in the PAIN family has $n \geq 1$ layers, uses paths of length up to $\ell \geq 1$, and distances up to $d \leq \ell$. Analogously to PATH-WL, for a fixed distance d, we denote such a GNN as d-PAIN. PAIN computes an embedding $\mathbf{h}_v^{(i)}$ for each node $v \in V$ in each layer $i \in \{1, \dots, n\}$. We initialize each node embedding as the node features $\mathbf{h}_v^{(0)} = \mathbf{x}_v$. The embeddings are updated iteratively as

$$\mathbf{h}_v^{(i)} = \mathsf{AGG}\left(\left\{\!\!\left\{ \mathbf{z}_p^{(i-1)} \mid p \in d\text{-}P_v^{`} \right\}\!\!\right\}\right)^{\infty}$$

where $\mathbf{z}_p^{(i-1)}$ is the embedding of path p of length $k \leq \ell$ defined as

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

In order to gain maximal expressive power, the function f must be injective over sequences and AGG must be injective over multisets. For f we select an LSTM (Hochreiter & Schmidhuber, 1997) as they can approximate any function on sequences (Hammer, 2000). In most experiments we use the sum for AGG as it allows the representation of injective functions over multisets (Xu et al., 2019, Lemma 5). To get a graph-level prediction, we pool all node representations in the final layer and apply a multi-layer-perceptron. The runtime complexity of PAIN is equivalent to that of PATH-WL (see Section 3, par. Time Complexity). For additional investigation on the runtime of PAIN see Appendix D.

Datasets. To study the expressivity of the PAIN family, we use the synthetic datasets EXP (Abboud et al., 2021), SR (Balcilar et al., 2021) and CSL (Murphy et al., 2019). EXP contains 600 non-isomorphic pairs of graphs representing propositional formulas. Each pair of graphs in this dataset cannot be distinguished by 1-WL but can be distinguished by 3-WL. With SR we refer to the same subset of strongly regular graphs used by (Michel et al., 2023). Each pair of graphs in this dataset cannot be distinguished by 3-WL, as 3-WL fails to distinguish strongly regular graphs. An instance of this dataset is visualized in Figure 3. CSL contains 150 graphs with 41 nodes belonging to 10 different isomorphism classes that are 1-WL-equivalent. Our experiments

on EXP and SR evaluate to what degree untrained PAIN can distinguish graphs. For CSL, we train PAIN to predict the isomorphism classes. Additionally, we perform an ablation study to investigate the impact of iterations, path length, and distance on expressivity. For real-world evaluation, we use ZINC (Gómez-Bombarelli et al., 2018; Sterling & Irwin, 2015) and OGBG-MOLHIV (Hu et al., 2020; Wu et al., 2018). Both datasets contain graphs that represent small molecules and for both the task is to make graph-level predictions. ZINC contains 12 000 graphs and we perform regression to predict the solubility of each molecule. OGBG-MOLHIV contains 41 127 graphs and we classify whether these molecules inhibit HIV replication (Wu et al., 2018).

Experimental setup. For EXP and SR, we closely follow the experimental setup of Michel et al. (2023). We use an untrained PAIN with a two-layer LSTM to compute 16-dimensional embeddings. We use Euclidean normalization on the input for the LSTM and consider two representations the same if the Euclidean distance is below $\epsilon = 10^{-5}$. Analogous to Michel et al. (2023), for SR we restrict the path length to four due to computational considerations and use path length five for EXP. All presented results are repeated over five seeds. We use a one-layer 0-PAIN for EXP and a one-layer 1-PAIN for SR. For CSL, we perform stratified 5-fold cross validation with a 3:1:1 split. We train a small PAIN model with an embedding dimension of 16. We train 500 epochs with a fixed learning rate of 10^{-5}. We report the test set accuracy in the epoch with the highest validation performance and average this test set accuracy over all cross-validation splits. We perform ablations for different values of the path length $\ell \in \{1, \dots, 6\}$, number of layers $n \in \{1, 2\}$, and distance encoding depth $d \in \{0, 1, \ell\}$. As both real-world datasets contain edge features, we extend PAIN accordingly (see Appendix D). On ZINC we train a five-layer 1-PAIN with path length three and embedding dimension 128. As common on ZINC, we train with an initial learning rate of 10^{-3} that we halve whenever the validation metric does not increase for 20 epochs. The training stops after the learning rate dips below 10^{-5} or after 1 000 epochs. We train the model 10 times and report the average test set mean absolute error (MAE) in the epoch with the lowest validation error. On OGBG-MOLHIV it is folklore knowledge that smaller networks perform better. We thus train a one-layer 1-PAIN with path length two, which is the same path length as used by Michel et al. (2023). To further avoid over-fitting we use a dropout rate of 0.5. As is common on this dataset we train with a fixed learning rate 10^{-3} for a fixed number of 100 epochs. We train the model ten times and report the average test set ROC-AUC score in the epoch with the highest validation score.

Results. On EXP, untrained 0-PAIN with path length five can distinguish all graphs (Table 3), which is consistent with the results in Michel et al. (2023). Note that we do not use distance encoding as proposed in Michel et al. (2023) and could thus verify that the multiplicities of paths of length five are sufficient. On SR, untrained 1-PAIN is able to distinguish more than 50% of all graph pairs (cf. Figure 7). In general, our results are comparable with Michel et al. (2023) with distance encoding. For SR(29,14,6,7) we obtain significantly better results of around 40% failure rate in comparison to the 80% failure rate of pathNN.

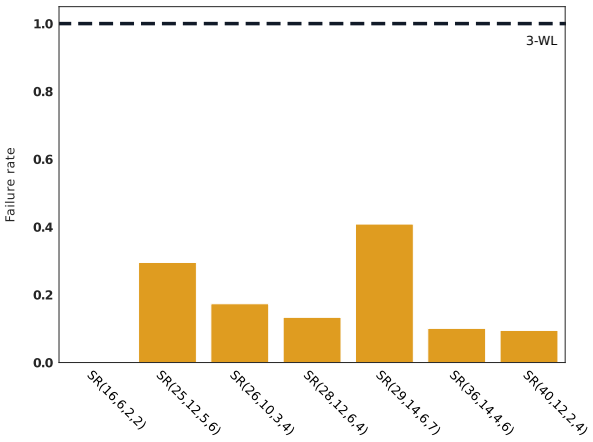

Figure 7: Results on SR for path length $\ell = 4$ and distance encoding $d = 1$. The dashed line indicates the failure rate of 3-WL.

Table 2 shows the results on the CSL dataset, which suggest that the expressivity increases with the number of layers, the path length, and the depth of the distance encoding. Most importantly, increasing the depth d gives a strong boost in accuracy without increasing the time complexity. For example for a single layer and path length $\ell = 5$, increasing the distance encoding depth from 0 to ℓ improves the accuracy from 50% to 100%. What is especially interesting is that no 0-PAIN model achieves an accuracy of over 50% for CSL which indicates that the distance encoding is crucial for the expressive power required for this dataset. Finally, we can see that increasing the number of layers gives a strong boost in predictive performance for short path lengths, especially for $\ell = 2$. This is important, as increasing the number of layers increases the runtime only by a constant factor compared to an exponential increase with ℓ. Table 4 and Table 5 show the results on ZINC and OGBG-MOLHIV, respectively. We can see that in both cases 1-PAIN outperforms classical message-passing GNNs, whereas alternative approaches such as CIN or pathNN still outperform 1-PAIN. This could be due to the fact that PAIN aggregates *all* paths up to length ℓ in each iteration, potentially resulting in a more complicated learning task. In contrast, pathNN increases the length of the paths layer-wise and thus needs to extract information from fewer paths at each layer.

Table 2: Mean and standard deviation of the accuracy (″) on CSL.

ℓ	1 Layer 0-PAIN		1-PAIN		2-PAIN		2 Layers 0-PAIN		1-PAIN		2-PAIN	
2	12	4	20	0	20	0	12	4	64	8	70	9
3	18	4	40	0	50	0	20	0	47	6	64	4
4	29	5	54	5	90	0	32	3	64	5	90	0
5	50	0	59	1	100	0	46	5	67	2	100	0
6	50	0	90	0	100	0	46	5	90	0	100	0

Table 3: Pairs of graphs in EXP that cannot be distinguished by the given models. **Bold** marks the strongest result.

Model	EXP #
GIN (Xu et al., 2019)	600
3-WL (Maron et al., 2019a)	**0**
pathNN (Michel et al., 2023)	**0**
PAIN (ours)	**0**

Table 4: Mean and standard deviation of the MAE on ZINC. GIN, GCN, and GAT experiments were conducted by Dwivedi et al. (2023). All other GNNs were benchmarked by the cited authors. **Bold** marks the strongest architecture.

Model	MAE (#)	
GIN (Xu et al., 2019)	0.387	0.015
GCN (Kipf & Welling, 2017)	0.278	0.003
GAT (Velickovic et al., 2018)	0.384	0.007
CIN (Bodnar et al., 2021a)	**0.079**	**0.006**
ESAN (Bevilacqua et al., 2022)	0.102	0.003
pathNN (Michel et al., 2023)	0.090	0.004
PAIN (ours)	0.148	0.003

Table 5: Mean and standard deviation of the ROC-AUC score on OGBG-MOLHIV. GIN and GCN experiments were conducted by Hu et al. (2020). All other GNNs were benchmarked by the cited authors. **Bold** marks the strongest architecture.

Model	ROC-AUC (″)	
GIN (Xu et al., 2019)	75.58	1.40
GCN (Kipf & Welling, 2017)	76.06	0.97
ESAN (Bevilacqua et al., 2022)	78.00	1.42
CIN (Bodnar et al., 2021a)	**80.94**	**0.57**
GSN (Bouritsas et al., 2022)	77.99	1.00
Graphormer (Ying et al., 2021)	80.51	0.53
pathNN (Michel et al., 2023)	79.17	1.09
PAIN (ours)	78.50	1.4

6. Conclusion and Future Work

In this paper we investigated the discriminative power of paths to distinguish between non-isomorphic graph instances. For this, we proposed PATH-WL, a general class of color refinement algorithms that allows us to investigate the expressivity of path-based graph neural networks. We proved that PATH-WL is incomparable in expressivity to several other powerful GNNs such as subgraph GNNs or k-GNNs. Furthermore, PATH-WL is able to count cycles which is important for learning tasks on molecular structures. All of this indicates that path-based GNNs form a novel class of highly expressive graph neural networks.

Limitations. For general graphs, the enumeration of all paths can be intractable. Thus, PATH-WL might not be well suited for dense graphs. Our experimental results also suggest that, while maximally expressive in theory, considering *all* paths does not outperform state of the art for real-world benchmark datasets. This could be due to the large number of paths that PAIN aggregates in each layer. It might be necessary to adapt PAIN to better select relevant information from all possible paths.

Future work. As the computational cost of our approach is strongly dependent on the chosen path length, we plan to characterize graph classes for which a reasonably low path length is sufficient for maximal expressivity. Furthermore, we intend to investigate approaches that only require the computation of a subset of all paths of a certain length.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

ACKNOWLEDGEMENTS

This research has been partially supported by the Italian MUR PRIN 2022 project "MEDICA" (2022RNTYWZ), the Next Generation EU program project PNRR ECS00000017 - "THE - Tuscany Health Ecosystem" - Spoke 3 - CUP I53C22000780001, and the Vienna Science and Technology Fund (WWTF) project ICT22-059. We thank Silvia Beddar-Wiesing and Alice Moallemy-Oureh for their essential contributions to the brainstorming process and for their valuable comments in every phase of this project. We further express our gratitude to Sagar Malhotra, Giuseppe Alessio D'Inverno, Marco Tanfoni, Patrick Indri, Veronica Lachi, and David Penz for the useful discussions, insightful suggestions, and constant support. We especially thank Max Thiessen and Pascal Welke for providing fundamental hints about the proofs and for their helpful feedback on the paper. Lastly, we are grateful to the anonymous reviewers who significantly improved our work with their suggestions.

References

Abboud, R., Ceylan, I. I., Grohe, M., and Lukasiewicz, T. The surprising power of graph neural networks with random node initialization. In *International Joint Conference on Artificial Intelligence*, pp. 2112–2118, 2021.

Abboud, R., Dimitrov, R., and Ceylan, I. I. Shortest path networks for graph property prediction. In *Learning on Graphs Conference*, pp. 5–1. PMLR, 2022.

Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., Ver Steeg, G., and Galstyan, A. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *International Conference on Machine Learning*, pp. 21–29. PMLR, 2019.

Arvind, V., Fuhlbrück, F., Köbler, J., and Verbitsky, O. On Weisfeiler-Leman invariance: Subgraph counts and related graph properties. *Journal of Computer and System Sciences*, 113:42–59, 2020.

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

Babai, L., Erdos, P., and Selkow, S. M. Random graph isomorphism. *SIAM Journal on computing*, 9(3):628–635, 1980.

Balakrishnan, V. *Schaum's Outline of Graph Theory: Including Hundreds of Solved Problems*. McGraw Hill Professional, 1997.

Balcilar, M., Héroux, P., Gauzere, B., Vasseur, P., Adam, S., and Honeine, P. Breaking the limits of message passing graph neural networks. In *International Conference on Machine Learning*, pp. 599–608. PMLR, 2021.

Bevilacqua, B., Frasca, F., Lim, D., Srinivasan, B., Cai, C., Balamurugan, G., Bronstein, M. M., and Maron, H. Equivariant subgraph aggregation networks. In *The Tenth International Conference on Learning Representations*, 2022.

Biggs, N., Lloyd, E. K., and Wilson, R. J. *Graph Theory, 1736-1936*. Oxford University Press, 1986.

Bodnar, C., Frasca, F., Otter, N., Wang, Y., Lio, P., Montufar, G. F., and Bronstein, M. Weisfeiler and Lehman go cellular: CW networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 2625–2640. Curran Associates, Inc., 2021a.

Bodnar, C., Frasca, F., Wang, Y., Otter, N., Montufar, G. F., Lio, P., and Bronstein, M. Weisfeiler and Lehman go topological: Message passing simplicial networks. In *International Conference on Machine Learning*, pp. 1026–1037. PMLR, 2021b.

Bouritsas, G., Frasca, F., Zafeiriou, S., and Bronstein, M. M. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):657–668, 2022.

Cai, J., Fürer, M., and Immerman, N. An optimal lower bound on the number of variables for graph identification. *Comb.*, 12(4):389–410, 1992.

Chen, Z., Chen, L., Villar, S., and Bruna, J. Can graph neural networks count substructures? In *Advances in Neural Information Processing Systems*, volume 33, pp. 10383–10395. Curran Associates, Inc., 2020.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive universal generalized pagerank graph neural network. In *9th International Conference on Learning Representations*, 2021.

Ding, Y., Orvieto, A., He, B., and Hofmann, T. Recurrent distance-encoding neural networks for graph representation learning. *CoRR*, abs/2312.01538, 2023.

D'Inverno, G. A., Bianchini, M., Sampoli, M. L., and Scarselli, F. On the approximation capability of GNNs in node classification/regression tasks. *arXiv preprint arXiv:2106.08992*, 2023.

Dong, Z., Zhang, M., Payne, P., Province, M. A., Cruchaga, C., Zhao, T., Li, F., and Chen, Y. Rethinking the power of graph canonization in graph representation learning with stability. In *The Twelfth International Conference on Learning Representations*, 2023.

Dwivedi, V. P., Joshi, C. K., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. Benchmarking graph neural networks. *J. Mach. Learn. Res.*, 24:43:1–43:48, 2023.

Feng, J., Chen, Y., Li, F., Sarkar, A., and Zhang, M. How powerful are k-hop message passing graph neural networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 4776–4790. Curran Associates, Inc., 2022.

Feng, J., Kong, L., Liu, H., Tao, D., Li, F., Zhang, M., and Chen, Y. Towards arbitrarily expressive GNNs in $O(n^2)$ space by rethinking Folklore Weisfeiler-Lehman. *CoRR*, abs/2306.03266, 2023.

Fey, M. and Lenssen, J. E. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Fortin, S. The graph isomorphism problem. *Technical report, Department of Computing. Science, The University of Alberta, Edmonton, Alberta, Canada,*, 1996.

Frasca, F., Bevilacqua, B., Bronstein, M., and Maron, H. Understanding and extending subgraph GNNs by rethinking their symmetries. In *Advances in Neural Information Processing Systems*, volume 35, pp. 31376–31390. Curran Associates, Inc., 2022.

Geerts, F. Walk message passing neural networks and second-order graph neural networks. *CoRR*, abs/2006.09499, 2020.

Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 4(2):268, 2018.

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

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., 2020.

Huang, N. T. and Villar, S. A short tutorial on the Weisfeiler-Lehman test and its variants. In *International Conference on Acoustics, Speech and Signal Processing*, pp. 8533–8537. IEEE, 2021.

Huang, Y., Peng, X., Ma, J., and Zhang, M. Boosting the cycle counting power of graph neural networks with I^2-GNNs. In *The Eleventh International Conference on Learning Representations*, 2022.

Immerman, N. and Lander, E. *Describing graphs: A first-order approach to graph canonization*. Springer, 1990.

Kiefer, S. and Neuen, D. The power of the Weisfeiler–Leman algorithm to decompose graphs. *SIAM Journal on Discrete Mathematics*, 36(1):252–298, 2022.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations*, 2017.

Kong, L., Chen, Y., and Zhang, M. Geodesic graph neural network for efficient graph representation learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 5896–5909. Curran Associates, Inc., 2022.

Kriege, N. M. Weisfeiler and Leman go walking: Random walk kernels revisited. In *Advances in Neural Information Processing Systems*, volume 35. Curran Associates, Inc., 2022.

Li, P., Wang, Y., Wang, H., and Leskovec, J. Distance encoding: Design provably more powerful neural networks for graph representation learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 4465–4478. Curran Associates, Inc., 2020.

Lim, D., Robinson, J. D., Zhao, L., Smidt, T., Sra, S., Maron, H., and Jegelka, S. Sign and basis invariant networks for spectral graph representation learning. In *International Conference on Learning Representations*, 2022.

Maron, H., Ben-Hamu, H., Serviansky, H., and Lipman, Y. Provably powerful graph networks. In *Advances in Neural Information Processing Systems*, volume 32, pp. 2153–2164. Curran Associates, Inc., 2019a.

Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. Invariant and equivariant graph networks. In *7th International Conference on Learning Representations*, 2019b.

Michel, G., Nikolentzos, G., Lutzeyer, J. F., and Vazirgiannis, M. Path neural networks: Expressive and accurate graph neural networks. In *International Conference on Machine Learning*, volume 202, pp. 24737–24755. PMLR, 2023.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. Weisfeiler and Leman go neural: Higher-order graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, pp. 4602–4609. AAAI Press, 2019.

Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., and Neumann, M. TUDataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, 2020a.

Morris, C., Rattan, G., and Mutzel, P. Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21824–21840. Curran Associates, Inc., 2020b.

Morris, C., Rattan, G., Kiefer, S., and Ravanbakhsh, S. Speqnets: Sparsity-aware permutation-equivariant graph networks. In *International Conference on Machine Learning*, pp. 16017–16042. PMLR, 2022.

Murphy, R. L., Srinivasan, B., Rao, V. A., and Ribeiro, B. Relational pooling for graph representations. In *International Conference on Machine Learning*, volume 97, pp. 4663–4673. PMLR, 2019.

Neuen, D. Homomorphism-distinguishing closedness for graphs of bounded tree-width. In *41st International Symposium on Theoretical Aspects of Computer Science*, volume 289, pp. 53:1–53:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

Nikolentzos, G., Dasoulas, G., and Vazirgiannis, M. k-hop graph neural networks. *Neural Networks*, 130:195–205, 2020.

Papp, P. A. and Wattenhofer, R. A theoretical comparison of graph neural network extensions. In *International Conference on Machine Learning*, pp. 17323–17345. PMLR, 2022.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Qian, C., Rattan, G., Geerts, F., Niepert, M., and Morris, C. Ordered subgraph aggregation networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 21030–21045. Curran Associates, Inc., 2022.

Robertson, N. and Seymour, P. D. Graph minors. II. Algorithmic aspects of tree-width. *Journal of algorithms*, 7 (3):309–322, 1986.

Robinson, R. W. and Wormald, N. C. Almost all regular graphs are Hamiltonian. *Random Structures & Algorithms*, 5(2):363–374, 1994.

Sterling, T. and Irwin, J. J. Zinc 15–ligand discovery for everyone. *Journal of chemical information and modeling*, 55(11):2324–2337, 2015.

Sun, Y., Deng, H., Yang, Y., Wang, C., Xu, J., Huang, R., Cao, L., Wang, Y., and Chen, L. Beyond homophily: Structure-aware path aggregation graph neural network. In *International Joint Conference on Artificial Intelligence*, pp. 2233–2240, 2022.

Thiede, E., Zhou, W., and Kondor, R. Autobahn: Automorphism-based graph neural nets. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29922–29934. Curran Associates, Inc., 2021.

Truong, Q. and Chin, P. Generalizing topological graph neural networks with paths. *CoRR*, abs/2308.06838, 2023.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *6th International Conference on Learning Representations*, 2018.

Wale, N. and Karypis, G. Acyclic subgraph based descriptor spaces for chemical compound retrieval and classification. In *IEEE International Conference of Data Mining*, volume 23, 04 2006.

Wallach, I., Dzamba, M., and Heifets, A. Atomnet: A deep convolutional neural network for bioactivity prediction in structure-based drug discovery. *CoRR*, abs/1510.02855, 2015.

Wang, G., Ying, R., Huang, J., and Leskovec, J. Multi-hop attention graph neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, August 2021*, pp. 3089–3096, 2021.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. Moleculenet: A benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *7th International Conference on Learning Representations*, 2019.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., and Liu, T.-Y. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems*, volume 34, pp. 28877–28888. Curran Associates, Inc., 2021.

You, J., Ying, R., and Leskovec, J. Position-aware graph neural networks. In *International Conference on Machine Learning*, pp. 7134–7143. PMLR, 2019.

You, J., Gomes-Selman, J. M., Ying, R., and Leskovec, J. Identity-aware graph neural networks. In *AAAI conference on artificial intelligence*, volume 35, pp. 10737–10745, 2021.

Zemlyachenko, V. N., Korneenko, N. M., and Tyshkevich, R. I. Graph isomorphism problem. *Journal of Soviet Mathematics*, 29:1426–1481, 1985.

Zhang, B., Feng, G., Du, Y., He, D., and Wang, L. A complete expressiveness hierarchy for subgraph GNNs via subgraph Weisfeiler-Lehman tests. In *International Conference on Machine Learning*, pp. 41019–41077. PMLR, 2023.

Zhang, B., Gai, J., Du, Y., Ye, Q., He, D., and Wang, L. Beyond Weisfeiler-Lehman: A quantitative framework for GNN expressiveness. In *The Twelfth International Conference on Learning Representations*, 2024.

Zhang, M. and Li, P. Nested graph neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 15734–15747. Curran Associates, Inc., 2021.

A. Related work

In this section, we provide a more comprehensive discussion on related work.

Walks vs. Paths. Although walks and paths in a graph are rather similar concepts, they have different implications for GNN expressivity. By definition, walks are sequences of adjacent nodes, and therefore paths are particular walks without node repetitions. However, due to the local nature of the message-passing procedure, which iteratively updates the feature of a node based on its neighborhood, GNNs at layer ℓ encode all the walks up to length ℓ for each node (see Kriege (2022)). Conversely, GNNs do not encode the identity of a node and therefore cannot identify repetitions of the same node in a walk. We thus argue that aggregating paths, in contrast to walks, does contribute additional information that can be helpful to distinguish between non-isomorphic graphs. Other approaches utilize walks in a way that increases the expressive power of GNNs. For example, Geerts (2020) proposes the walk Message Passing Neural Network (walk-MPNN), generalizing the second-order non-linear invariant network by Maron et al. (2019b). This model computes embeddings for pairs of nodes by encoding the walks between the two nodes. This architecture is strictly more powerful than 1-WL but is bounded in expressive power by 3-WL, regardless of the length of the walks. Indeed, increasing the length may allow to distinguish graphs with less iterations but cannot enhance expressivity. Conversely, the power of PATH-WL increases with the length of the paths. Therefore, PAIN is more powerful than walk-MPNN for some graph families such as strongly regular graphs (see Theorem 4.3).

Path-based GNNs. As argued in the previous paragraph, using paths to update node embeddings in GNNs can increase their expressive power. Several architectures have been devised that encode path information in different ways. For instance, Abboud et al. (2022) propose shortest path networks, which replace the topological neighborhood with shortest paths. They are provably more expressive than 1-WL, but not more expressive than 3-WL. Kong et al. (2022) consider geodesic information between pairs of nodes and show that they distinguish almost all d-regular graphs for which 1-WL consistently fails. Closely related to using shortest paths directly is the incorporation of distance information. This can be done either explicitly (You et al., 2019; Li et al., 2020; Ding et al., 2023) or implicitly (Ying et al., 2021; Nikolentzos et al., 2020; Feng et al., 2022). Li et al. (2020); You et al. (2019); Ding et al. (2023) propose position aware GNNs, whereas Ying et al. (2021) introduce a transformer architecture with spatial encoding. These approaches are quite similar to the general idea of using positional encodings in the context of graph representation learning (Abboud et al., 2021; Lim et al., 2022; Dong et al., 2023). Recently, several GNNs have been proposed that consider *all* paths instead (Sun et al., 2022; Michel et al., 2023; Truong & Chin, 2023). Truong & Chin (2023) introduce a color refinement test based on path complexes, a topological generalization of paths, which is strictly more expressive than 1-WL and not bounded by 3-WL. Sun et al. (2022) propose a graph neural network which samples from the set of all paths. Their model additionally learns structure and distance information with the help of a recurrent cell. Michel et al. (2023) propose pathNN, which aggregates paths instead of the standard topological neighborhood. They achieve strong empirical performance by additionally encoding shortest path distances in an LSTM cell. Please refer to the discussion of Remark C.3 for a detailed comparison between PATH-WL and pathNN.

k-Hop GNNs. Standard message-passing GNNs aggregate messages from the direct neighbors of each node, which are called the *first-hop* neighbors. Recently, many architectures have generalized the message-passing scheme by aggregating information from all the nodes within the k-hop neighborhood simultaneously (Nikolentzos et al., 2020; Chien et al., 2021; Abu-El-Haija et al., 2019; Wang et al., 2021). The idea of aggregating distant nodes, beyond the first-order neighborhood, is similar to that of path-based GNNs. However, in the k-hop aggregation we gather nodes with the same shortest path distance from a reference node, which is at most k. In contrast, the path-based aggregation provides a richer and context-aware representation, as it includes information such as the order of nodes within the same k-hop. Feng et al. (2022) and Papp & Wattenhofer (2022) extended the k-hop GNNs by encoding, respectively, the subgraph induced by the nodes in the k-th hop (KP-GNN) and the subgraph induced by the whole k-hop neighborhood (N_k). All these modifications increase the expressive power of GNNs beyond 1-WL (Feng et al., 2022, Prop.1). The k-hop GNNs are limited by 3-WL, including the KP-GNN (Feng et al., 2022, Thm. 2). On the contrary, considering the complete k-hop increases the expressivity because N_k is incomparable to 3-WL (Papp & Wattenhofer, 2022, Thm. 6.3). However, d-PATH-WL and N_k are fundamentally different graph coloring procedures: d-PATH-WL aggregates only paths up to a certain length which are endowed with distance information, while N_k aggregates the subgraph induced by the entire k-hop neighborhood.

B. Additional preliminaries

In the following, we provide some additional preliminaries.

FWL vs. OWL. 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 (Huang & Villar, 2021).
- $(k+1)$-WL is more expressive than k-WL for any k 2 (Cai et al., 1992).

The existing relation between k-OWL and k-FWL is the following:

$$k\text{-OWL} \quad (k \quad 1)\text{-FWL},$$

for any k 3 (Huang & Villar, 2021).

Definition B.1. A *tree* is a connected acyclic graph. A *rooted tree* is a tree in which one node is chosen as the root.

Definition B.2 (Neuen (2024))**.** Let $G = (V_G, E_G,)$ and $H = (V_H, E_H)$ be two graphs. We say that H is a *homomorphic image* of G if there is a surjective homomorphism $\phi : V_G \ ! \ V_H$ such that:

$$E_H = f(\phi(u), \phi(v)) j (u, v) \ 2 \ E_G g$$

We denote with spasm(G) the set of homomorphic images of G.

The *tree-width* of a graph H, denoted by $tw(H)$, can be defined in terms of a tree decomposition.

Definition B.3 (Robertson & Seymour (1986))**.** A *tree decomposition* of a graph $G = (V, E)$ is a pair (X, T) where $T = (I, A)$ is a tree, and $X = f X_i j i \ 2 \ I g$ is a family of subsets of V, such that:

(i) $\bigcup_{i \, 2 \, I} X_i = V$,

(ii) every edge of G has both of its endpoints in some X_i, for $i \ 2 \ I$, and

(iii) for all $i, j, k \ 2 \ I$, if j lies on the path from i to k in T, then $X_i \setminus X_k \quad X_j$.

The tree-width of a tree decomposition is $\max_{i \, 2 \, I} j X_i j \quad 1$. The tree-width of G is the minimum tree-width taken over all possible tree decompositions of G.

Definition B.4 (Neuen (2024))**.** The *hereditary tree-width* of G, denoted by $hdtw(G)$, is the maximum tree-width among the tree-widths of the homomorphic images of G, i.e.,

$$hdtw(G) := \max_{H \, 2 \, \text{spasm}(G)} tw(H)$$

Definition B.5 (Balakrishnan (1997))**.** An *Eulerian cycle*, also called an Eulerian circuit or Euler tour, in an undirected graph is a cycle that uses each edge exactly once. If such a cycle exists, the graph is called *Eulerian*.

Definition B.6 (Degree-invariant transformation)**.** Let $G = (V, E, X)$ be a graph. Let $T : E \ ! \ V \quad V$ 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.

Definition B.7 (Incomparability)**.** 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 be an ordering between them. If it holds that $R_1 \quad R_2$ and $R_1 \quad R_2$, the ordering relation is partial, and R_1 and R_2 are said to be *incomparable*.

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

Definition B.8 (Unfolding Tree)**.** The *unfolding tree* UT_v^l in graph $G = (V, E, \mathsf{X})$ of node $v \in V$ up to depth $l \in \mathsf{N}_0$ is defined as

$$\mathsf{UT}_v^l = \begin{cases} Tree(\mathsf{x}_v, ;) & \text{if } l = 0 \\ Tree\ \mathsf{x}_v, \mathsf{UT}^{l-1}_{N(v)} & \text{if } l > 0\,, \end{cases}$$

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

D'Inverno et al. (2023) and Kriege (2022) 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.

C. Proofs

In this section, we provide proofs and some additional discussion for our theoretical contributions.

C.1. Path-WL: A Path-Based WL test

Proposition C.1. *Let G, G' be the graphs in Figure 1. For every $\ell \geq 5$, $G \not\simeq_{\text{0-PATH-WL}^{\ell}(1)} G'$, while $G \simeq_{\text{1-WL}} G'$.*

Proof. Let $G = (V, E, \mathsf{X})$ and $G' = (V', E', \mathsf{X}')$ be the graphs in Figure 1. The partitioning induced on nodes by 1-WL is represented in Figure 1, where each color corresponds to a $\simeq_{1\text{-WL}}$ equivalence class. We can see that there exists a bijection from the equivalence classes in G to the equivalence classes in G' and we can thus conclude $G \simeq_{\text{1-WL}} G'$. For the other direction, first, let us notice the symmetry of the two graphs, where we can group the nodes into three different equivalence classes: (i) nodes with degree three, (ii) neighbors of degree three nodes, and (iii) nodes which are not adjacent to nodes with degree three. We now consider the path multisets up to length five, denoted by $0\text{-}P^5$, for each of the three node equivalence classes. In Figure 2, we can see that $0\text{-}P^5$ for nodes $v \in G$, $v' \in G'$ which belong to equivalence class (i) differ, as $0\text{-}P^5_v$ contains four paths of length five, while $0\text{-}P^5_{v'}$ contains only two paths of length five. Given the injectivity of the HASH function, for two nodes to obtain different colors it is sufficient to have different path multisets, as this ensures different colors from the first iteration onwards. The same argument applies to the remaining nodes in (ii) and (iii), which differ with respect to their multiplicities of paths of length five: (ii) in G, these nodes have five paths of length five, whereas in G' they have three paths of length five and for (iii) the nodes in G have three paths of length five, whereas in G' they have two paths of length five. Due to the different multiplicities of paths of length five, we conclude that 0-PATH-WL is able to distinguish the two graphs. □

Proposition C.2. *For every $d' \geq d \geq 0$, $\ell' \geq \ell \geq 1$ it holds that*

$$d\text{-PATH-WL}^{\ell} \preceq d'\text{-PATH-WL}^{\ell'},$$

and

$$d\text{-PATH-WL}^{\ell} \preceq d\text{-PATH-WL}^{\ell'}.$$

Proof. Due to the transitivity of the order relation $\preceq$, it suffices to prove the statement for $d' = d + 1$ and $\ell' = \ell + 1$. To demonstrate the non-decreasing expressive power of d-PATH-WL$^{\ell}$ with respect to d, first we need to show that for every ℓ, if two nodes u and v are s.t. $u \simeq_{(d+1)\text{-PATH-WL}^{\ell}} v$ then $u \simeq_{d\text{-PATH-WL}^{\ell}} v$. Let $\mathsf{c}^d_{v_i} := \mathsf{HASH}\ d\text{-}P^{\ell}_{v_i}$ be the color of node v_i after termination of d-PATH-WL. Hence, $\mathsf{c}^{d+1}_u = \mathsf{c}^{d+1}_v$ by hypothesis. Due to the injectivity of the HASH function, this implies that the multiset of paths with distance encoding up to length $d+1$ are equal, that is $(d+1)\text{-}P^{\ell}_u = (d+1)\text{-}P^{\ell}_v$. What we aim to infer is that $d\text{-}P^{\ell}_u = d\text{-}P^{\ell}_v$. By definition, for every path $((\mathsf{c}_v, \eta^d_{vv}), (\mathsf{c}_{v_1}, \eta^d_{vv_1}), \ldots, (\mathsf{c}_{v_\cdot}, \eta^d_{vv_\cdot})) \in d\text{-}P^{\ell}_v$, $\eta^d_{vv_i} \neq ;$ if $\eta^d_{vv_i} \neq d < d+1$. Therefore, it holds for any path in $(d+1)\text{-}P^{\ell}_v$ that all tuples $(c_{v_i}, \eta^{d+1}_{vv_i})$ are the same in $d\text{-}P^{\ell}_v$ if the distance between the nodes v and v_i is less or equal than d, otherwise $\eta^d_{vv_i}$ is $;$. Hence, we conclude that $d\text{-}P^{\ell}_u = d\text{-}P^{\ell}_v$. To prove the monotonicity of d-PATH-WL$^{\ell}$ as the path length increases, it is enough to consider two facts. The first is the injectivity of the HASH function, and the second is that $d\text{-}P^{\ell}_v$ contains paths *up to length* ℓ. Hence, if we can distinguish two nodes with a certain length ℓ, these different paths are also included in $d\text{-}P^{\ell+1}_v$. □

C.2. Importance of iterations.

In the following, we want to provide some intuition on the importance of iterations for improving expressivity. For instance, consider the two graphs G and H in Figure 8a. The two highlighted nodes v and u are 0-PATH-WL^3-equivalent, as they have the same multiset of paths of length up to three (see Figure 15). Hence, the output of the first iteration of 0-PATH-WL will result in the same color for u and v, as shown in Figure 8a. However, if we perform another iteration, we can distinguish the two nodes. Notably, with iterations, we can also observe that we may reduce the required path length to distinguish two non-isomorphic graphs. For example, in Figure 8b, we can see that the two graphs, which are indistinguishable with one iteration of 0-PATH-WL^3, can be distinguished with two iterations of 0-PATH-WL^2, reducing the required runtime by a factor of three.

(a) The two nodes v and u are 0-PATH-WL$^{3,(1)}$-equivalent, indeed they have the same color after the first iteration. Performing another iteration with the already colored nodes allows us to distinguish v and u. That is, 0-PATH-WL$^{3,(2)}_v \not\leftrightarrow$ 0-PATH-WL$^{3,(2)}_u$

(b) The two graphs are colored from the first iteration of 0-PATH-WL^2. The nodes u and v are not distinguishable, but their path multisets in the second iteration contain different paths. The output of the second iteration will thus result in different colors. That is, 0-PATH-WL$^{2,(2)}_v \not\leftrightarrow$ 0-PATH-WL$^{2,(2)}_u$

Remark C.3. For every $\ell \geq 1$, it holds that 0-PATH-WL$^{\ell,(\ell)}$ $\equiv$ pathNN (Michel et al., 2023) with path length ℓ

Discussion of Remark C.3. We compared d-PATH-WL and the annotation scheme characterizing pathNN with respect to the same number of iterations ℓ. We claim that one iteration of d-PATH-WL contains all information of the corresponding layer of pathNN. For example, in the first layer pathNN computes paths of length one (= the neighborhood) whereas 0-PATH-WL computes paths of length up to ℓ. At the second layer, pathNN aggregates paths of length two where each node feature is updated given the paths of length one, computed at the first layer. 0-PATH-WL aggregates the same set of paths up to length ℓ, but each node is updated with the paths of length up to ℓ, computed in the first layer. After the second iteration of 0-PATH-WL, one node may receive information from nodes at distance 2ℓ from it. At each step, 0-PATH-WL has access to all information that pathNN contains, with the same asymptotic complexity. Hence, the expressivity of 0-PATH-WL cannot be bounded by pathNN. Moreover, in the pathNN framework, the length of the paths must coincide with the number of iterations while these two parameters are independent in our algorithm. We often observe that these parameters positively affect each other: increasing the length may decrease the number of iterations needed to distinguish two graphs, and vice versa (see Section C.2). □

C.3. The discriminative power of paths

C.3.1. RELATION TO THE k-WL HIERARCHY.

Proposition C.4. For every path length $\ell > 1$ and every $d \geq 0$, d-PATH-WL$^\ell$ is strictly more expressive than 1-WL.

Proof. It is sufficient to prove the theorem for $d = 0$ because the addition of distance information to $d > 0$ does not decrease the expressive power of the test. The proof consists of two parts: (a) we demonstrate that 0-PATH-WL is as expressive as

1-WL, and (ii) we prove that ℓ-PATH-WL is more expressive than 1-WL. For (i), we prove that for every length $\ell > 1$, for every node u, v and every iteration $i \in \mathbb{N}$,

$$u \equiv_{\text{1-WL}^{(i)}} v \Rightarrow u \equiv_{\ell\text{-PATH-WL}^{(i)}} v.$$

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

$$\text{HASH}\left(c_v^{(i)}, \{\!\{c_w^{(i)} \mid w \in \mathcal{N}(v)\}\!\}\right) = \text{HASH}\left(c_u^{(i)}, \{\!\{c_w^{(i)} \mid w \in \mathcal{N}(u)\}\!\}\right) \Rightarrow \text{HASH}(\ell\text{-P}_v^{(i)}) = \text{HASH}(\ell\text{-P}_u^{(i)})$$

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 $\ell\text{-P}_v^{(i)} = \ell\text{-P}_u^{(i)}$. α is true if $c_v^{(i)} = c_u^{(i)}$ or if $\{\!\{c_w^{(i)} \mid w \in \mathcal{N}(v)\}\!\} = \{\!\{c_w^{(i)} \mid w \in \mathcal{N}(u)\}\!\}$, or both. Given that every element of $\ell\text{-P}_v^{(i)}$ is a sequence whose first element is $c_v^{(i)}$, i.e.,

$$\ell\text{-P}_v^{(i)} := \{\!\{(c_v^{(i)})\}\!\} \cup \{\!\{(c_v^{(i)}, c_w^{(i)})_{w \in \mathcal{N}(v)}\}\!\} \cup \{\!\{(c_v^{(i)}, c_w^{(i)}, \ldots, c_y^{(i)})_{w \in \mathcal{N}(v) \wedge y = \pi_\ell(p_v)}\}\!\},$$

where $\pi_j(p_v)$ denotes the j-th node of path p_v, we can simply conclude that if $c_v^{(i)} = c_u^{(i)}$ then $\ell\text{-P}_v^{(i)} = \ell\text{-P}_u^{(i)}$. Suppose that $c_v^{(i)} = c_u^{(i)}$. Then, $\{\!\{c_w^{(i)} \mid w \in \mathcal{N}(v)\}\!\} = \{\!\{c_w^{(i)} \mid w \in \mathcal{N}(u)\}\!\}$ implies $\{\!\{(c_v^{(i)}, c_w^{(i)})_{w \in \mathcal{N}(v)}\}\!\} = \{\!\{(c_u^{(i)}, c_w^{(i)})_{w \in \mathcal{N}(u)}\}\!\}$. Due to the fact that $\ell\text{-P}^{(i)}$ is a multiset of sequences of heterogeneous length, the fact that paths of length one are different is enough to conclude that $\ell\text{-P}_v^{(i)} = \ell\text{-P}_u^{(i)}$.

For (ii), we prove that ℓ-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 ℓ-PATH-WL is able to distinguish (see Figure 9 for one such example). □

Figure 9: The coloring after one iteration of ℓ-PATH-WL is enough to distinguish the two non-isomorphic graphs that 1-WL cannot distinguish. Note that the partitioning is the same but the colors of the nodes are different.

In order to prove Theorem C.6 we need the following Lemma.

Lemma C.5. Let C_ℓ be a cycle of length $\ell = \lfloor \frac{k}{2} \rfloor k$. Then C_ℓ has hereditary tree-width $\text{hdtw}(C_\ell) \leq k - 1$.

Proof. Recall that a graph G has hereditary tree-width $k - 1$ if the maximal treewidth among the graphs in the set of homomorphic images of G is $k - 1$ (See Definition B.4). In order to prove the lemma, it suffices to provide a lower bound on $\text{hdtw}(G)$, i.e., that there exists a graph H with treewidth $k - 1$, which is a homomorphic image of C_ℓ. We claim that H is a k-clique. Note that the tree-width of a k-clique (cf. Definition B.3) corresponds to the node degree, that is $k - 1$. It remains to prove that a k-clique is a homomorphic image of the cycle C_ℓ, with $\ell = \lfloor \frac{k}{2} \rfloor k$, that is, $\ell = k\frac{k-1}{2}$ for odd k and $\ell = \frac{k^2}{2}$ for even k.

First, consider k to be odd (and therefore $k - 1$ is even). Then, from Euler's Theorem, which states that a connected graph has an Eulerian cycle if and only if every node has even degree (Biggs et al., 1986), we can conclude that a k-clique, with k odd, contains an Eulerian cycle. Based on that, we claim that we can map the Eulerian cycle of length ℓ to the k-clique itself, via a surjective homomorphism (cf. Definition B.2). This corresponds to proving that the k-clique is the homomorphic image of a cycle C_ℓ (in this case C_ℓ coincides with the Eulerian cycle). In particular, the number of edges of the Eulerian cycle[1] is equal to the number of edges of the clique, that is $\ell = k\frac{k-1}{2}$. To guarantee that is a homomorphism, the nodes in the cycle must be connected in a way that for every edge in the cycle, there exists a corresponding edge

[1] Note that in a cycle, the number of edges corresponds to the number of nodes.

$(\varphi(u), \varphi(v))$ in the clique. This is ensured by the definition of Eulerian cycle, which traverses the edges of the clique exactly once. See Figure 10 for an example of such a homomorphism for $k = 5$. The resulting function φ is surjective because it maps a cycle of $\frac{k(k-1)}{2}$ nodes to a clique of k nodes. Hence, we proved the claim for k odd.

Now consider the case of an even k. We aim to show that a k-clique is the homomorphic image of some cycle C_ℓ, with $\ell = \frac{k^2}{2}$. Let us now remove a node from the k-clique (as shown in Figure 11a for the case $k = 6$). Deleting a node results in a $(k-1)$-clique, which we have just shown to be the homomorphic image of its Eulerian cycle. The following procedure illustrates how to build the cycle which will be homomorphically mapped to k-clique. We begin with the cycle on $\frac{k-2}{2}(k-1)$ edges, the Eulerian cycle of the $(k-1)$-clique (See Figure 11b). We want to add the edges that correspond to the missing edges in the k-clique to the cycle (cf. the dotted edges in Figure 11a). First, consider the minimal walk in the cycle, which starts at one node and ends in the same node. This corresponds to "opening" the cycle, doubling one node, and leaving the edges $x$ed. See for example Figure 12, where we doubled the node a. To preserve the homomorphism, the new node will be mapped to $\varphi(a)$: As a second step, we need to add to the cycle the edge corresponding to the edges linking the deleted node x to all the other nodes. We want to do this using the minimal number of edges. Each node in the cycle has degree two but x has $k-1$ neighbors, which is odd. Therefore we will connect x to pairs of neighbors (until $k-2$) and we know that one neighbor will be repeated twice. In order to utilize the minimum number of edges, the neighbor that will be repeated twice is the node that has been doubled in the previous step (a).[2] This number of copies of x will be $\frac{k}{2}$ and therefore the number of added edges will be $2(\frac{k}{2})$, because with x we add two neighbors at a time. It remains to link the neighbors together, with $(\frac{k}{2}-1)$ edges. The total amount of edges in the cycle will be:

$$(k-1)\frac{k-2}{2} + k + \frac{k}{2} - 1 = \frac{k^2}{2}$$

□

Theorem C.6. Let $d \geq 1$ and $k \geq 3$. Then, d-PATH-WL and k-WL are incomparable. Equivalently, the following holds:

(1) for every $k \geq 3$ there exists a path length ℓ such that d-PATH-WL$^\ell \not\equiv_{vk}$-WL;

(2) for every $\ell \geq 1$, there exists a k such that k-WL $\not\equiv_{vd}$-PATH-WL$^\ell$.

Proof. To prove the first part of the statement, we make use of a recent result from Neuen (2024, Theorem 1.3). This result states that for every graph F such that $\mathrm{htdtw}(F) > k$, the k-WL algorithm fails to detect subgraph counts of the pattern F. In our case, let F be a cycle on $\lfloor\frac{k+2}{2}\rfloor(k+2)$ nodes. From Lemma C.5, we know that the hereditary tree-width of a cycle on $\lfloor\frac{k+2}{2}\rfloor(k+2)$ nodes is at least $k+1 > k$. Then, from Corollary 4.6, we assert that for every $\ell \geq 1$, d-PATH-WL$^\ell$ can count cycles of any length. In particular, the path length needed to count the pattern F is $\lfloor\frac{k+2}{2}\rfloor(k+2) - 1$. That is, to distinguish two k-WL-equivalent graphs after the first iteration of d-PATH-WL$^\ell$, we need path length $\ell \geq \frac{k^2}{2}$.

To prove the second statement, we show that for every path length ℓ, one can always construct two graphs that are d-PATH-WL-equivalent but distinguishable by k-WL for $k \geq 3$. Consider the following graph construction (inspired by the proof for Papp & Wattenhofer (2022, Theorem 6.3)). We set $n = 2\ell + 1$ and construct two graphs in the following way: $G_{\frac{n}{2}}$, which is a cycle of length $\frac{n}{2}$, and $C_{\frac{n}{2},2}$, which consists of two disconnected cycles of length $\frac{n}{2}$ each. Next, we set $d = \ell$, which aligns with the maximum expressive power of d-PATH-WL$^\ell$. If the two graphs are indistinguishable by d-PATH-WL with $d = \ell$, they are indistinguishable for every $d \geq \ell$ (cf. Prop. 3.4). With d-PATH-WL$^\ell$, each node aggregates all the paths of length up to ℓ with shortest path distance $\leq \ell$. Since each node in $G_{\frac{n}{2}}$ as well as $C_{\frac{n}{2},2}$ is the starting point for exactly two paths for every length up to ℓ, it will have the same color assignment and the two graphs cannot be distinguished by d-PATH-WL. On the other hand, k-WL can distinguish the two graphs for every $k \geq 3$ (Kiefer & Neuen, 2022, Theorem 6.1). □

C.4. How is the path length related to the order of WL?

In Theorem 4.3, we compare the d-PATH-WL hierarchy with the k-WL one. The proof is constructive and provides, for each k, a pair of non-isomorphic graphs for which k-WL fails, but which can be distinguished by d-PATH-WL$^\ell$ with length $\ell \geq \frac{k^2}{2}$. However, the required length is always dependent on the graph at hand and it is not necessarily increasing with

[2] Of course, we can obtain a valid surjective homomorphism by creating a double copy of a node different from a, but it will result in the addition of more edges.

Figure 10: The construction of the homomorphism f from the cycle C_{10} to the 5-clique. The arrows connect the nodes via the function φ, which is surjective. The fact that the nodes are connected such that f is a homomorphism is guaranteed by the presence of the Eulerian cycle in the clique. Indeed, to construct the cycle it is enough to follow the Eulerian cycle on the clique, starting from any edge.

(a) (b)

Figure 11: (a) Deleting one node from the 6-clique, the red node, results in a 5-clique. (b) A 5-clique is the homomorphic image of a 10-cycle.

Figure 12: First step of the procedure: double the red node and open the cycle. The new node is mapped to u_1.

We present two examples to support this claim: The first one can be seen in Figure 4. For $n = 2^3$, we can distinguish the two graphs with $d = 0$ and path length $\ell = n$ whereas 1-WL cannot distinguish them. With $d = 1$ the required length is $\ell = n - 1$ because we can count the number of n-cycles. The second one is the pair of strongly regular graphs in Figure 3. While 3-WL cannot distinguish them, $d = 1$ and paths of length $\ell = 4$ are sufficient for our approach. These examples show that there are 3-WL-equivalent graphs that require shorter path lengths than 1-WL-equivalent graphs. This is in line with Zhang et al. (2024), where it is reported that 3-WL is not able to count cycles of length greater than seven. In this case, the required length to distinguish two graphs with a different cycle count is $\ell = \frac{k+2}{2}(k+2) - 1 = 9$ but shorter, $\ell = 7$. In this brief analysis of the relations between ℓ and k, we are not considering two parameters that play an important role in decreasing the required length: the addition of shortest path information and the iterative procedure. We refer to Table 2 in the paper to appreciate the empirical effectiveness of this information and to Appendix C.2 to see an example of the importance of iterations.

C.5. Counting cycles

We start with the following Lemma.

Lemma C.7. Each connected 2-regular graph on n nodes is isomorphic to a cycle C_n.

Proof. We prove the statement above by induction on the number of nodes. 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 $n + 1$ nodes. Consider one node $v \in V'$, 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 three nodes. Let G be the graph obtained by removing node v from G' and connecting (v_1, v_2) with a new edge (see Figure 13). The resulting G is connected, 2-regular with n nodes. Hence, for the inductive hypothesis, it is isomorphic to a cycle on n nodes C_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'. □

G^0 G

Figure 13: Sketch for the proof of Lemma C.7.

Theorem C.8. Let $\text{sub}(C_\ell, G, v) \neq \text{sub}(C_\ell, H, u)$ for some graphs G, H, nodes u, v and cycle C_ℓ. Then, $u \not\equiv_{1\text{-PATH-WL}^{\ell-1}} v$.

Proof. The cycle C_ℓ is identified by a path of length $\ell - 1$ where the last node is a marked node. That is, a cycle corresponds to paths of the form $((c_{v_1}, 0), \ldots, (c_{v_\ell}, 1))$. Due to the different cycle counts for the nodes v and u, their multisets of paths $1\text{-P}_v^{\ell-1}$ and $1\text{-P}_u^{\ell-1}$, will contain a different number of such paths. That is, $1\text{-P}_v^{\ell-1} \neq 1\text{-P}_u^{\ell-1}$ and given the injectivity of the HASH function, 1-PATH-WL can distinguish the two nodes. □

Corollary C.9. For every $k \geq 2$, there exists a path length ℓ such that 1-PATH-WL^ℓ is not bounded by {SubgraphGNN, Local 2-GNN, Folklore k-GNNs}.

Proof. The proof is a direct consequence of the fact that 1-PATH-WL can count arbitrary cycles given a sufficient path length (cf. Corollary 4.6). In particular, with path length ℓ, 1-PATH-WL can count cycles on $\ell + 1$ nodes, while SubgraphGNN, Local 2-GNN, and Folklore 2-GNN are limited to count cycles on maximum 7 nodes (Zhang et al., 2023). Moreover, the Folklore k-GNNs are characterized in expressive power by k-FWL $\equiv (k+1)$-WL, for every $k \geq 1$ (cf. B). Hence, given that 1-PATH-WL^ℓ is not bounded by k-WL (cf. Theorem 4.3) for length $\ell \geq k^2$, this concludes the proof. □

C.5.1. One iteration is almost all you need

Theorem C.10. There exists ℓ such that 0-PATH-WL^ℓ can distinguish the following pairs of infinite graph families:

1. Hamiltonian graphs of different orders at node and graph level,
2. Hamiltonian graphs and non-homogeneously traceable graphs at graph level, and
3. almost all connected d-regular graphs and disconnected d-regular graphs with m connected components at graph level.

In order to prove Theorem C.10 we prove (1) – (3) separately.

(1) Hamiltonian graphs of different order.

Corollary C.11. Let H_n and H_m be two Hamiltonian graphs with n and m nodes, respectively, and $n > m$. For any $v \in H_n$ and any $u \in H_m$, it holds that

$$v \not\equiv_{0\text{-PATH-WL}^\ell} u \quad \forall \ell \geq m.$$

Proof. By definition, a Hamiltonian graph H is a graph that contains a cycle C including all the nodes of the graph. Therefore, the length of the Hamiltonian path in H_m is $m - 1$, which is the longest path in the graph, containing all possible nodes. The same argument holds for H_n, with $n > m$. In particular, we are able to distinguish two graphs H_n and H_m by comparing their path multisets, i.e., after the first iteration, $0\text{-PATH-WL}_v^\ell \neq 0\text{-PATH-WL}_u^\ell \quad \forall \ell \geq m$. □

Figure 14: 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.

(2) Hamiltonian graphs and non-homogenously traceable graphs. The following theorem states that with sufficiently large ℓ, 0-PATH-WL$^\ell$ can always distinguish between two classes of graphs: non-homogeneously traceable graphs and Hamiltonian graphs.

Theorem C.12. Let G and H be two graphs of the same order. If G is a non-homogeneously traceable graph and H a Hamiltonian graph. Then,

$$G \not\simeq_{\text{0-PATH-WL}^{n-1}} H.$$

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. We denote with γ^{n-1} a generic Hamiltonian path from n. Formally, there exists a node $v \in G$ such that $h^{n-1}_v \neq \text{0-P}^{n-1}_v(G)$. 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 $u \in H$, $h^{n-1}_u \in \text{0-P}^{n-1}_u(H)$. At graph level, the conclusion follows. □

(3) Connected regular graphs vs. disconncected regular graphs. 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 C.13 (Connectivity). Let G be a disconnected graph with n connected components which are all d-regular graphs, with $d \geq 2$. Let s be the size of the smallest component. Let G' be a connected d-regular graph, such that $|V_G| = |V_{G'}|$. 0-PATH-WL can then distinguish almost every couple of such graphs G, G', whereas 1-WL cannot. In particular,

$$G \simeq_{\text{1-WL}} G' \quad \text{but} \quad G \not\simeq_{\text{0-PATH-WL}^s} G'.$$

Proof. Theorem 1 of Robinson & Wormald (1994) states that for 3 almost every[1] d-regular graph is Hamiltonian and from Lemma C.7 we know that every connected 2-regular graph is Hamiltonian. Hence let us consider all the connected components of G and the graph G' as Hamiltonian graphs. Then, from Corollary C.11 we know that two Hamiltonian graphs of different order can be distinguished by PATH-WL$^\ell$. The required length ℓ in this case is the size of the smallest connected component of G, i.e. $\ell = s$. Indeed, for every node v in G', 0-P^s_v contains a path of length s while 0-P^s_u = 0-P^{s-1}_u for every u in the smallest component of G. We can extend this reasoning to the graph level and conclude that $G \not\simeq_{\text{0-PATH-WL}^s} G'$. □

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 14 for an example of the transformation process which preserves the degree of the nodes.

Theorem C.14. 1-WL-equivalence at iteration ℓ and 0-PATH-WL$^{\ell,(1)}$-eqivalence are incomparable for every $\ell \geq 3$.

[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 (1994) for details.

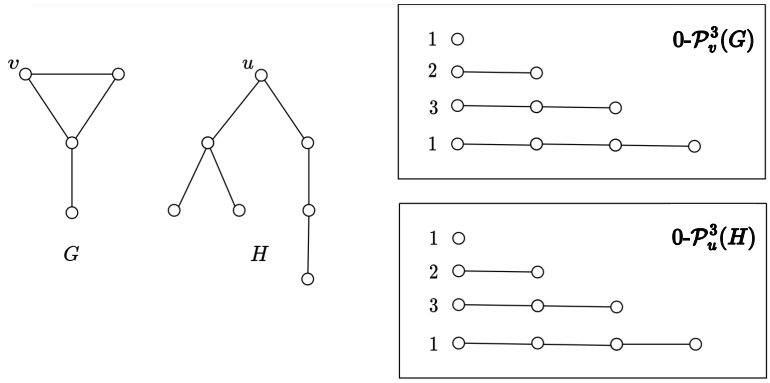

Figure 15: Graphs G and H and their respective path multisets for paths of length up to three for the node $v \in G$ and $u \in H$. These two graphs serve as a counterexample for paths being more discriminative than unfolding trees (and thus 1-WL).

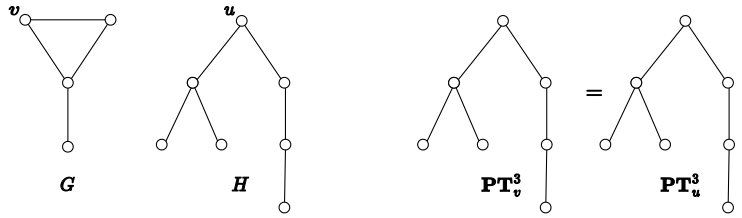

Figure 16: Counterexample showing that PT-equivalence is not more expressive than 1-WL. Indeed $v \not\equiv_{\text{1-WL}} u$ but the path-based unfolding trees for u and v are identical.

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

$$u \equiv_{\text{0-PATH-WL}^{k;(1)}} v \quad \text{and} \quad u \not\equiv_{\text{1-WL}} v \quad \text{at iteration } k.$$

and there exist nodes u^0, v^0 such that for some $t \in \mathbb{N}$

$$u^0 \not\equiv_{\text{0-PATH-WL}^{t;(1)}} v^0 \quad \text{and} \quad u^0 \equiv_{\text{1-WL}} v^0 \quad \text{at iteration } t.$$

In order to prove the theorem it suffices to show two examples: (i) one example, where 0-PATH-WL is able to distinguish between two nodes but 1-WL fails, and (ii) a second example where the converse holds. For (i), we can choose any graph class described before, or the famous instance shown in Figure 2. For (ii), we refer to Figure 15. The highlighted nodes in the figure have different unfolding trees at depth three (hence, $u \not\equiv_{\text{1-WL}} v$) but they are indistinguishable by the multiset of paths, at least for path length three.

□

Counterexample for Michel et al. (2023, Theorem 3.3) In Figure 16 we computed the path-based unfolding trees PT (cf. Michel et al. (2023, Definition 3.1) for the two nodes $v \in G$ and $u \in H$.

D. Additional Information on Experiments

In this section, we provide additional remarks on the experiments.

Distance Encoding. We do not use distance encoding for EXP. For SR we mark neighbors by adding a constant value of 1.0 to all the neighbors of the starting node within a path. For CSL and ZINC we use a different type of neighborhood encoding better suited for a learning task. For every distance encoding (even depth 0) we attach to each embedding in each path a learned vector that encodes some type of distance. In the case of $d = 1$, this vector encodes the position in the sequence. For the case of $d > 1$, this encodes the shortest path distance between the starting node and the current node in

the path. Finally, when d 1 we add an additional feature to the embedding of each node in the path that is one if it is a neighbor to the starting node in the path and zero otherwise.

Experimental Setup. For EXP, SR, ZINC and OGBG-MOLHIV we use a two-layer LSTM for f and summation for path-level pooling (AGG). We use summation as graph-level (READOUT) pooling for all of these datasets except OGBG-MOLHIV for which we use mean pooling. For ZINC we use a shared LSTM for f to reduce the number of parameters. The datasets ZINC and OGBG-MOLHIV also contain categorical node features which we project to the embedding dimension. We incorporate edge features by embedding them and then concatenating them to the vector embeddings in each path. For example in a path (v_1, v_2, v_3) we would attach to the embedding of v_2 the embedded feature of edge (v_1, v_2) and to v_3 the feature of (v_2, v_3). As the first node v_1 has no associated edge we simply concatenate an all-zeros vector to it. Finally, we reverse the paths on ZINC and OGBG-MOLHIV as this gives us a small boost in performance. We note that this has also been noticed by Michel et al. (2023). Due to numerical instabilities encountered with CSL, we vary some parameters for this dataset to combat these instabilities. For CSL we use a one-layer LSTM for f and the mean function for graph-level (READOUT) and path-level pooling (AGG). For CSL and ZINC, we add a two-layer multi-layer-perceptron (MLP) with ReLU activations and batch norm after each LSTM. For OGBG-MOLHIV this MLP has only a single layer to keep the final model as small as possible. We use PyTorch (Paszke et al., 2019) and PyTorch Geometric (Fey & Lenssen, 2019). We conduct our experiments on servers equipped with an RTX-3080 GPU and Intel Core i7-10700KF/i9-11900KF CPU.

Our code can be found at https://github.com/ocatias/ExpressivePathGNNs and https://github.com/tamaramagdr/synthetic-pain.

Training Time Comparison. In practice, it can be computationally infeasible to compute all paths. However, as current research on the expressivity of GNNs commonly focuses on molecules that have sparse structures this means that path-based GNNs such as our PAIN can be run efficiently on such datasets. We benchmark the run-time of PAIN on ZINC against GIN (Xu et al., 2019) and two subgraph GNNs: DS and DSS (Bevilacqua et al., 2022). For all four models, we select hyperparameters that yield the best validation set performance. For this, we train all four models on ZINC and report the mean and the standard deviation of the training time of each model. For DS and DSS we use a policy that extracts the 3-hop neighborhood (egonets) of every node. PAIN is identical to the model described in Section 5. All other models use an embedding dimension of 256 with a dropout rate of 0.5. GIN uses four message-passing layers, and both DS and DSS use five layers. All models are trained with early stopping. Table 6 shows that PAIN with path length three is only slightly slower than DS and DSS on ZINC (12k graphs) on consumer grade hardware. Finally, we would like to point out that it is not necessary to store every path up to length ℓ. For the sake of expressivity, it suffices to only store paths that are not part of larger paths. In preliminary experiments, this improved the runtime of PAIN by a speed-up factor of two.

Table 6: Average training time for different GNNs on ZINC. All models were trained with early stopping.

Model	Average Training Time (#)
GIN (Xu et al., 2019)	0.11 0.02 h
DS (Bevilacqua et al., 2022)	0.62 0.06 h
DSS (Bevilacqua et al., 2022)	0.69 0.06 h
PAIN (ours)	0.97 0.09 h