A Proofs of Message Flow Redundancy and Practical Implications

Corollary 1. *A message flow for output node u in the computational graph is a walk of the same length that ends at u in the original graph.*

Proof. Denote by p a message flow for output node u with length of k, where $p(k) = u$. GNNs based on WL-test iteratively update the node representation by aggregating the neighborhood information. Consequently, $p(l-1)$ is either $p(l)$ or one of $p(l)$'s neighbors, and so forth. Therefore, a message flow for output node u is a walk of the same length that ends at u in the original graph. □

Lemma 1. *The message flows that propagate the information of a trail are the walks of the trail from its beginning node to its ending node.*

Proof. Suppose tr is a trail of length r and p is a message flow of length k being generated from tr. Therefore, p consists of all edges of r, including backtracking edges and self-loop edges, and $p(k) = tr(r)$ and $p(0) = tr(0)$. Consequently, p is a walk of tr. Therefore, the message flows of a trail are walks of the trail from its beginning node to its ending node. □

Lemma 2. *Given a length-r trail, the number of length-k message flows derived from the trail is*

$$\sum_{s=0}^{\llcorner \frac{k-r}{2} \lrcorner} r^s (r+1+2s)^{(k-r-2s)},$$

where $1 \le r \le k$.

Proof. Without loss of generality, suppose a trail of length r and a message flow of length k derived from the trail, where $1 \le r \le k$. Lemma 1 states that the message flow is a walk of the trail, thus the message flow consists of all edges of r, including backtracking edges and self-loop edges. Since a backtracking edge results in an edge being visited twice, the maximum number of backtracking edges is $\llcorner \frac{k-r}{2} \lrcorner$. Let s be the number of backtracking edges, then the number of self-loop edges is $k-r-2s$. We can construct the message flow by adding backtracking edges and self-loop edges into the trail as illustrated in Figure 2. The backtracking edge of a self-loop edge is a self-loop edge. Thus, we begin by unorderly sampling s of the r backtracking edges with replacement. The number of combinations of unorderly sampling s edges from r edges is r^s. After the backtracking edges are placed, we need to unorderly sample $k-r-2s$ of the $r+1+2s$ self-loop edges with replacement. The number of combinations of unorderly sampling $k-r-2s$ edges from r edges is $(r+1+2s)^{(k-r-2s)}$. Consequently, the total number of possible message flows derived from the trail is computed as the number of combinations of backtracking edges and self-loops, as follows

$$\sum_{s=0}^{\llcorner \frac{k-r}{2} \lrcorner} r^s (r+1+2s)^{(k-r-2s)},$$

where $1 \le r \le k$. □

Redundancy Causes Message Over-squashing

Lemma 3. *Assume there is a graph with node set V, nodes $u, v \in V$, and the computational graph of a k-layer WL-test based GNN for u. If all length-k paths in the computational graph for u are activated with the same probability, we can measure the* ***relative influence of the input feature*** $\mathbf{x}_v$ ***on the node output*** $\mathbf{h}_u^{(k)}$*, on average, as*

$$\mathbb{E}\left(\frac{\partial \mathbf{h}_u^{(k)} / \partial \mathbf{x}_v}{\sum_{v' \in V} \partial \mathbf{h}_u^{(k)} / \partial \mathbf{x}_{v'}}\right) = \frac{\left[\prod_{l=k}^{1} \widetilde{A}\right]_{u,v}}{\sum_{v' \in V} \left[\prod_{l=k}^{1} \widetilde{A}\right]_{u,v'}}, \tag{9}$$

where $\widetilde{A} = A + I$ is the adjacency matrix of the graph with self-loop.

Proof. First, we rewrite the Equation 1 as follows.

$$\mathbf{h}_u^{(k)} = \psi^{(k)}\left(\mathbf{h}_u^{(k-1)}, \sum_{v\in V} A_{u,v}\phi^{(k)}\left(\mathbf{e}_{u,v}^{(k-1)}, \mathbf{h}_v^{(k-1)}\right)\right).$$

Without loss of generality, in order to simplify the notations, we assume that node attributes and hidden representations are scalar. We can compute the influence of $\mathbf{x}_v$ on $\mathbf{h}_u^{(k)}$ as

$$\begin{aligned}\frac{\partial \mathbf{h}_u^{(k)}}{\partial \mathbf{x}_v} &= \partial_1\psi^{(k)}\partial_{\mathbf{x}_v}\mathbf{h}_u^{(k-1)} + \partial_2\psi^{(k)}\sum_{j_k\in V} A_{u,j_k}\partial_2\phi^{(k)}\partial_{\mathbf{x}_v}\mathbf{h}_{j_k}^{(k-1)} \\ &= \sum_{j_k\in V}\widetilde{A}_{u,j_k}Z_{u,j_k}\partial_{\mathbf{x}_v}\mathbf{h}_{j_k}^{(k-1)},\end{aligned}$$

where

$$Z_{u,j_k} = \begin{cases}\partial_1\psi^{(k)}, & \text{if } j_k = u \\ \partial_2\psi^{(k)}\partial_2\phi^{(k)}, & \text{others}\end{cases}.$$

We recursively expands the layers and the equation can be rewritten as

$$\begin{aligned}\frac{\partial \mathbf{h}_u^{(k)}}{\partial \mathbf{x}_v} &= \sum_{j_k\in V}\widetilde{A}_{u,j_k}Z_{u,j_k}\partial_{\mathbf{x}_v}\mathbf{h}_{j_k}^{(k-1)} \\ &= \sum_{j_k\in V}\widetilde{A}_{u,j_k}Z_{u,j_k}\sum_{j_{k-1}\in V}\widetilde{A}_{j_k,j_{k-1}}Z_{j_k,j_{k-1}}\partial_{\mathbf{x}_v}\mathbf{h}_{j_{k-1}}^{(k-2)} \\ &= \sum_{j_k,\cdots,j_1\in V}\widetilde{A}_{u,j_k}\widetilde{A}_{j_k,j_{k-1}}\cdots\widetilde{A}_{j_2,j_1}Z_{u,j_k}Z_{j_k,j_{k-1}}\cdots Z_{j_2,j_1}\partial_{\mathbf{x}_v}\mathbf{h}_{j_1}^{(0)}.\end{aligned}$$

Since $\mathbf{h}_{j_1}^{(0)} = \mathbf{x}_{j_1}$, we have

$$\partial_{\mathbf{x}_v}\mathbf{h}_{j_1}^{(0)} = \begin{cases}1, \text{ if } j_1 = v \\ 0, \text{ others}\end{cases}.$$

Thus, the previous equation becomes

$$\frac{\partial \mathbf{h}_u^{(k)}}{\partial \mathbf{x}_v} = \sum_{j_k,\cdots,j_2\in V}\widetilde{A}_{u,j_k}\widetilde{A}_{j_k,j_{k-1}}\cdots\widetilde{A}_{j_2,v}Z_{u,j_k}Z_{j_k,j_{k-1}}\cdots Z_{j_2,v},$$

where $\{Z_{u,j_k}Z_{j_k,j_{k-1}}\cdots Z_{j_2,j_1} \mid \forall j_k,\ldots,j_2,j_1 \in V \text{ and } \widetilde{A}_{u,j_k}\widetilde{A}_{j_k,j_{k-1}}\cdots\widetilde{A}_{j_2,j_1} \neq 0\}$ is a collection of random variables dependent to $\mathbf{x}$. When all paths in the computational graph for the node u are activated with the same probability, these random variables have the same expectation. Thus, we conclude

$$\begin{aligned}\mathbb{E}\left(\frac{\partial\mathbf{h}_u^{(k)}/\partial\mathbf{x}_v}{\sum_{v'\in V}\partial\mathbf{h}_u^{(k)}/\partial\mathbf{x}_{v'}}\right) &= \frac{\sum_{j_k,\cdots,j_2\in V}\widetilde{A}_{u,j_k}\widetilde{A}_{j_k,j_{k-1}}\cdots\widetilde{A}_{j_2,v}}{\sum_{v'\in V}\sum_{j_k,\cdots,j_2\in V}\widetilde{A}_{u,j_k}\widetilde{A}_{j_k,j_{k-1}}\cdots\widetilde{A}_{j_2,v'}} \\ &= \frac{\left[\prod_{l=k}^{1}\widetilde{A}\right]_{u,v}}{\sum_{v'\in V}\left[\prod_{l=k}^{1}\widetilde{A}\right]_{u,v'}}.\end{aligned}$$

□

Lemma 4. *Assume there is a graph with node set V, the computational graph of a k-layer GNN based on WL-test for a node, and its message flow p. We compute* ***the relative influence of*** $\mathbf{x}_{p(0)}$ ***on*** $\mathbf{h}_{p(k)}^{(k)}$ ***along*** p*, on average, as*

$$\mathbb{E}\left(\frac{\partial\mathbf{h}_{p(k)}^{(k)}/\partial p}{\sum_{p'\in P}\partial\mathbf{h}_{p(k)}^{(k)}/\partial p'}\right) = \frac{\prod_{l=k}^{1}\widetilde{A}_{p(l),p(l-1)}}{\sum_{v'\in V}\left[\prod_{l=k}^{1}\widetilde{A}\right]_{p(k),v'}},$$

where $\widetilde{A} = A + I$ is the adjacency matrix of the graph with self-loop and P is the set of message flows that end at $tr(k)$.

Proof. By rewriting the Equation 1 as follows

$$\mathbf{h}_u^{(k)} = \psi^{(k)}\left(\mathbf{h}_u^{(k-1)}, \sum_{v\in V} A_{u,v}\phi^{(k)}\left(\mathbf{e}_{u,v}^{(k-1)}, \mathbf{h}_v^{(k-1)}\right)\right).$$

The *Jacobian* of $\mathbf{h}_{p(l)}^{(l)}$ *w.r.t* $\mathbf{h}_{p(l-1)}^{(l-1)}$ is computed as

$$\frac{\partial \mathbf{h}_{p(l)}^{(l)}}{\partial \mathbf{h}_{p(l-1)}^{(l-1)}} = \widetilde{A}_{p(l),p(l-1)} Z_{p(l),p(l-1)},$$

where

$$Z_{p(l),p(l-1)} = \left\{ \begin{array}{ll} \partial_1\psi^{(l)}, & \text{if } j_k = u \\ \partial_2\psi^{(l)}\partial_2\phi^{(l)}, & \text{others} \end{array} \right. .$$

The influence of $\mathbf{x}_{p(0)}$ on $\mathbf{h}_{p(k)}^{(k)}$ along p can be expanded recursively based on the above equation:

$$\frac{\partial \mathbf{h}_{p(k)}^{(k)}}{\partial p} = \prod_{l=k}^{1} \frac{\partial \mathbf{h}_{p(l)}^{(l)}}{\partial \mathbf{h}_{p(l-1)}^{(l-1)}} = \prod_{l=k}^{1} \widetilde{A}_{p(l),p(l-1)} Z_{p(l),p(l-1)},$$

where $\{\prod_{l=k}^{1} Z_{p(l),p(l-1)} \mid \forall p \in P$ and $\prod_{l=k}^{1} \widetilde{A}_{p(l),p(l-1)} \neq 0\}$ is a collection of random variables dependent to $\mathbf{x}$. When all paths in the computational graph for $p(k)$ are activated with the same probability, these random variables have the same expectation. Consequently, we compute **the relative influence of $\mathbf{x}_{p(0)}$ on $\mathbf{h}_{p(k)}^{(k)}$ along** p, on average, as

$$\mathbb{E}\left(\frac{\partial \mathbf{h}_{p(k)}^{(k)}/\partial p}{\sum_{p'\in P}\partial \mathbf{h}_{p(k)}^{(k)}/\partial p'}\right) = \mathbb{E}\left(\frac{\partial \mathbf{h}_{p(k)}^{(k)}/\partial p}{\sum_{v'\in V}\partial \mathbf{h}_{p(k)}^{(k)}/\partial \mathbf{h}_{v'}^{(0)}}\right) = \frac{\prod_{l=k}^{1} \widetilde{A}_{p(l),p(l-1)}}{\sum_{v'\in V}\left[\prod_{l=k}^{1} \widetilde{A}\right]_{p(k),v'}}.$$

□

Corollary 2. *Assume there is a graph with a trail* tr *and the computational graph of a* k*-layer WL-test based GNN for* tr*'s head node. We compute* ***the relative influence along*** tr*, on average, as*

$$\mathbb{E}\left(\frac{\partial \mathbf{h}_{tr(r)}^{(k)}/\partial tr}{\sum_{p'\in P}\partial \mathbf{h}_{p(k)}^{(k)}/\partial p'}\right) = \frac{\sum_{p'\in P^*}\prod_{r=k}^{1} \widetilde{A}_{p'(r),p'(r-1)}}{\sum_{v'\in V}\left[\prod_{r=k}^{1} \widetilde{A}\right]_{p(k),v'}},$$

where $\widetilde{A} = A + I$ *is the adjacency matrix of the graph with self-loop and* P^* *is the set of message flows derived from* tr*.*

Proof. The relative influence along tr equals to the summation of the relative influence along message flows derived from tr. □

B Proofs of RFGNN

B.1 Permutation equivalent

Corollary 3. *Redundancy-Free GNNs are permutation equivalent.*

Proof. Because any node permutation applied to a graph has no effect on the adjacency relationship between nodes, the resulting TPTs and TPF are independent of node permutation, therefore redundancy-Free GNNs are permutation equivalent. □

B.2 TPFH has maximum expressive power

Lemma 5. *Non-isomorphic graphs will generate non-isomorphic TPFs if the height of TPFs is not constrained.*

Proof. According to Definition 6, a TPF extracted from a graph would consists of all the epaths of the graph if the height of TPFs is not constrained (higher than the length of the longest path). Non-isomorphic graphs consists of different epaths. Therefore, non-isomorphic graphs will produce non-isomorphic TPFs if the height of TPFs is not restricted. ☐

B.3 TPFH is strictly more powerful than 1-WL

Inspired by those expressive power analysis of WL test based GNNs [16, 23, 18], we use TPFH to quantify the expressive power of RFGNN and compare with 1-WL.

The expressive power of 1-WL is dependent on WL subtrees extracted from graphs. According to [3], we restate the definition of the WL subtree as follows.

Definition 7. *Assume there is a graph G with a node u, the height k WL subtree of u, denoted as* $\texttt{WL subtree}^{k}_{(G,u)}$*, is a tree that is extracted from G, with the child nodes of each node in* $\texttt{WL subtree}^{k}_{(G,u)}$ *being the adjacent nodes of the node in G.*

Lemma 6. *For two graphs G_1 and G_2, and two nodes $u \in \mathcal{V}(G_1)$ and $v \in \mathcal{V}(G_2)$, if the height k WL subtree of u is non-isomorphic to that of v, the height k TPTs of them are non-isomorphic too.*

Proof. According to Definition 7, a height k WL subtree consists of walks of length k that have no self-loop and end at the root node. The definition of TPT (Definition 5) indicates that a height k TPT consists of epaths of length up to k that end at the root node. A walk can be converted into an epath without losing the structural information, by droping edges which are duplicate, backtracking or self-loops. Therefore, we can transform a height k WL subtree into its root's height k TPT without losing the structural information. If the height k WL subtrees of u and v are non-isomorphic, then these two WL subtrees hold different structural information. Therefore, the height k TPTs of node u and v hold different structural information, indicating that they are non-isomorphic. ☐

Lemma 7. *There exist two graphs G_1 and G_2 as well as two nodes, $u \in \mathcal{V}(G_1)$ and $v \in \mathcal{V}(G_2)$, where the height k WL subtree of u is isomorphic to that of v, but the height k TPTs of the two nodes are not.*

Proof. Assume that graph G_1 consists of two rings, each with r nodes, and graph G_2 consists of a ring with $2 * r$ nodes, with the same labels for all nodes in both graphs. Each node of G_1 has a height k WL subtree that is isomorphic to any height k WL subtree of G_2, where $k > r$. The height k TPT of any G_1 node, on the other hand, is not isomorphic to the height k WL subtree of any G_2 node. ☐

According to Lemma 6 and Lemma 7, two graphs that can be distinguished by 1-WL can also be distinguished by TPFH. However, TPFH is possible to distinguish some graph pairs that cannot be distingusihed by 1-WL. Therefore, we can conclude the expressive power of TPFH is better than 1-WL.

B.4 RFGNN is at most as powerful as TPFH.

We follow the proof of the expressive power of GIN [16] to demonstrate that RFGNN is at most as powerful as TPFH. We first prove that the neural networks are capable of modeling injective multiset functions for the subtree aggregation.

Lemma 10. *Assume $\mathcal{X}$ is countable. There exists a function $f : \mathcal{X} \to \mathbb{R}^n$ so that $h(X) = \sum_{x \in X} f(x)$ is unique for each multiset $X \subset \mathcal{X}$ of bounded size. Moreover, any multiset function g can be decomposed as $g(X) = \phi\left(\sum_{x \in X} f(x)\right)$ for some function ϕ.*

Proof. We first prove that there exists a mapping f so that $\sum_{x \in X} f(x)$ is unique for each multiset X of bounded size. Since $\mathcal{X}$ is countable, there exists a mapping $Z : \mathcal{X} \to \mathbb{N}$ from $x \in \mathcal{X}$ to natural numbers. As the cardinality of multisets X is bounded, there must exist a number $N \in \mathbb{N}$ so that $|X| < N$ for all X. Then an example of such f is $f(x) = N^{-Z(x)}$. This function f can be viewed as a more compressed form of an one-hot vector or N-digit presentation. Thus, $h(X) = \sum_{x \in X} f(x)$ is an injective function of multisets. $\phi\left(\sum_{x \in X} f(x)\right)$ is permutation invariant so it is a well-defined multiset function. For any multiset function g, we can construct such ϕ by letting $\phi\left(\sum_{x \in X} f(x)\right) = g(X)$. Note that such ϕ is well-defined because $h(X) = \sum_{x \in X} f(x)$ is injective. □

Then, we prove that the neural networks are capable of modeling injective functions over *(scalar, multiset)* pairs for the subtree update.

Corollary 4. *Assume $\mathcal{X}$ and $\mathcal{D}$ are both countable. There exists a function $f : \mathcal{X} \to \mathbb{R}^n$ and a function $b : \mathcal{D} \to \mathbb{R}^n$ so that $h(d, X) = b(d) + \sum_{x \in X} f(x)$ is unique for each pair (d, X), where $d \in \mathcal{D}$ is a rational number, and $X \subset \mathcal{X}$ is a multiset of bounded size. Moreover, any function g over such pairs can be decomposed as $g(d, X) = \varphi\left(b(d) + \sum_{x \in X} f(x)\right)$ for some function φ.*

Proof. We consider $f(x) = N^{-Z(x)}$, where N and $Z : \mathcal{X} \to \mathbb{N}$ are the same as defined in the proof of Lemma 10, and $b(d) = \varepsilon \cdot d$, where ε is an irrational number. Let $h(d, X) \equiv b(d) + \sum_{x \in X} f(x)$. Our goal is to demonstrate that for any $(d', X') \neq (d, X)$ with $d, d' \in \mathcal{D}$ and $X, X' \subset \mathcal{X}$, $h(d, X) \neq h(d', X')$ holds. If $d \neq d', X = X'$ or $d = d', X \neq X'$, then $b(d) + \sum_{x \in X} f(x) \neq z(d') + \sum_{x \in X'} f(x)$ holds, consequently, $h(d, X) \neq h(d', X')$ holds. If $d \neq d', X \neq X'$, we prove by contradiction that $h(d, X) \neq h(d', X')$ holds. We can similarly rewrite $h(d, X) = h(d', X')$ as

$$\varepsilon \cdot (d - d') = \sum_{x \in X'} f(x) - \sum_{x \in X} f(x). \tag{10}$$

Because ε is an irrational number and $d - d'$ is a non-zero rational number, L.H.S. of Equation 10 is irrational. On the other hand, R.H.S. of Equation 10, i.e., the sum of a finite number of rational numbers, is rational. Hence the equality in Equation 10 cannot hold, and thus we have reached a contradiction.

For any function g over the pairs (d, X), we can construct such φ for the desired decomposition by letting $\varphi\left(b(d) + \sum_{x \in X} f(x)\right) = g(d, X)$. Note that such φ is well-defined because $h(d, X) = b(d) + \sum_{x \in X} f(x)$ is injective. □

Without sacrificing generality, we assume a RFGNN that utilizes a height k TPF. RFGNN updates the representation of a node u at level $(l-1)$ as follows

$$\mathbf{h}_u^{(l-1)} = \psi^{(l)}\left(\mathbf{x}_u^{(l-1)}, \cup_{v \in \mathcal{C}^{(l-1)}(u)} \phi^{(l)}\left(\mathbf{e}_{u,v}^{(l)},\ \mathbf{h}_v^{(l)}\right)\right). \tag{7}$$

And RFGNN aggregates the root node representations of the TPTs to obtain the graph representation as follows

$$\mathbf{h}_G = \chi\left(\left\{\mathbf{h}_{root(\texttt{TPT})}^{(0)} \mid \texttt{TPT} \in \texttt{TPF}_G\right\}\right). \tag{8}$$

Corollary 4 suggests $\phi^{(k)}$ could be an injective function on the original node and edge features. If the input space of an injective function is countable, the output space of the injective function is countable too. Therefore, $\psi^{(k)}$ could be an injective function, according to Corollary 4. When $0 \leq l < k$, $\mathbf{h}_v^{(l)}$ is either transformed from an original node or a subtree feature updated by an injective function. Different nodes v could have different representations $\mathbf{h}_v^{(l)}$'s. Therefore $\phi^{(l)}$ and $\psi^{(l)}$ functions, $0 \leq l < k$, both could be injective functions. According to Corollary 4, any function g over *(scalar, multiset)* pairs can be decomposed as $g(d, X) = \varphi\left(b(d) + \sum_{x \in X} f(x)\right)$ for some function φ, therefore we can stack ψ functions to encode TPTs. According to Lemma 10, the χ function could be an injective function too.

In summary, RFGNN can be as powerful as TPFH if each layer of RFGNN are equipped with a sufficient number of layers and the parametric local aggregators they use can learn to be injective.

B.5 Advantages of RFGNN

Lemma 8. *Assume there is a height k RFGNN as in Equation 7, a graph G, and an epath ep within G of length $l \leq k$. We can measure the* ***influence of the node feature*** $\mathbf{x}_{ep[i]}^{(i)}$ ***on the tree representation*** $\mathbf{h}_{ep[0]}^{(0)}$*, where $0 \leq i \leq l$, as*

$$\frac{\partial \mathbf{h}_{ep[0]}^{(0)}}{\partial \mathbf{x}_{ep[i]}} = \left(\prod_{j=0}^{i-1} \left(\partial_2 \psi^{(j)} \partial_2 \phi^{(j)} \right) \right) \left(\partial_{\mathbf{x}_{ep[i]}} \psi^{(i)} + \partial_2 \psi^{(i)} \partial_{\mathbf{x}_{ep[i]}} \phi^{(i)} \right).$$

Similarly, we can measure the ***influence of the subtree representation*** $\mathrm{h}_{ep[i]}^{(i)}$ ***on the tree representation*** $\mathbf{h}_{ep[0]}^{(0)}$*, where $0 \leq i \leq l$, as*

$$\frac{\partial \mathbf{h}_{ep[0]}^{(0)}}{\partial \mathbf{h}_{ep[i]}^{(i)}} = \left(\prod_{j=0}^{i-1} \left(\partial_2 \psi^{(j)} \partial_2 \phi^{(j)} \right) \right) \partial_2 \psi^{(i)} \partial_{\mathbf{h}_{ep[i]}^{(i)}} \phi^{(i)}.$$

Proof. We rewrite Equation 7 as follows.

$$\mathbf{h}_u^{(k-1)} = \psi^{(k)} \left(\mathbf{x}_u^{(k-1)}, \sum_v A_{u,v}^{(k)} \phi^{(k)} \left(\mathbf{e}_{u,v}^{(k)}, \ \mathbf{h}_v^{(k)} \right) \right). \tag{11}$$

The computational graph of a height k RFGNN for a node could be regarded as a $(k+1)$-partite graph. Thus, we can use the matrix $A^{(k)}$ to indicate the adjacency between the $(k-1)$-partite and the k-partite of the computational graph (zero based index).

Given a path p of the computational graph with length no less than $(l+1)$, where $ep[i]$ is the i-th node (zero based index). We compute the influence of $\mathbf{x}_{ep[l]}^{(l)}$ on $\mathbf{h}_{ep[0]}^{(0)}$ through the path p as follows

$$\frac{\partial \mathbf{h}_{ep[0]}^{(0)}}{\partial \mathbf{x}_{ep[l]}^{(l)}} = \partial_2 \psi^{(0)} \partial_2 \phi^{(0)} \frac{\partial \mathbf{h}_{ep[1]}^{(1)}}{\partial \mathbf{x}_{ep[l]}^{(l)}}.$$

We iterate the computation above and see that the equation can be rewritten as

$$\frac{\partial \mathbf{h}_{ep[0]}^{(0)}}{\partial \mathbf{x}_{ep[l]}^{(l)}} = \left(\prod_{j=0}^{l-1} \left(\partial_2 \psi^{(j)} \partial_2 \phi^{(j)} \right) \right) \partial_{\mathbf{x}_{ep[l]}^{(l)}} \psi^{(l)}.$$

Similarly, we have

$$\frac{\partial \mathbf{h}_{ep[0]}^{(0)}}{\partial \mathbf{h}_{ep[l]}^{(l)}} = \left(\prod_{j=0}^{l-1} \left(\partial_2 \psi^{(j)} \partial_2 \phi^{(j)} \right) \right) \partial_2 \psi^{(l)} \partial_{\mathbf{h}_{ep[l]}^{(l)}} \phi^{(l)}.$$

□

Lemma 9. *Given a graph $G = (V, E)$, a node $u \in V$, and an edge $e \in E$ within the k-hop neighborhood of u. Let n be the number of rings in the k-hop neighborhood of u. We have that the number of repeations of e within the height k TPT of u is no more than $1 + n$.*

Proof. If there is no ring in u's k-hop neighborhood, then there is only a single path (of length up to k) from u to e. For each additional ring in the k-hop neighborhood of u, an epath from u and arriving at e is added, but the length of this epath may exceed k, and so it does not appear on the height k TPT of u. Consequently, the number of repeations of e within the height k TPT of u is up to $1 + n$. □

B.6 Graph Similarity Comparison

Intuitively, GNNs should be capable of encoding graphs with close graph edit distance (GED) [21] into similar representations, allowing us to assess graph similarity using a GNN. Since the output of any message passing GNN is dependent on the message flows derived from the original graph, **an ideal GNN should generate two message flow sets with a high degree of overlap for two graphs with a close graph edit distance (GED)**. We demonstrate that WL-test based GNNs may not fulfill this principle, whereas our RFGNN does.

We restate the definition of graph edit distance (GED) form the study [21].

Definition 8 (Graph Edit Distance, GED). *Generally, given a set of graph edit operations, the graph edit distance between two graphs G_1 and G_2, written as $GED(G_1, G_2)$ can be defined as*

$$GED(G_1, G_2) = \min_{(\epsilon_1,\dots,\epsilon_k)\in\mathcal{P}({G_1}_u,{G_2}_v)} \sum_{i=1}^{k} \tau(\epsilon_i),$$

where $\mathcal{P}(G_1, G_2)$ denotes the set of edit paths transforming G_1 into (a graph isomorphic to) G_2 and $\tau(\epsilon) \geq 0$ is the cost of each graph edit operation ϵ. The set of elementary graph edit operators typically includes [21]:

- *node insertion to introduce a single new labeled node to a graph.*
- *node deletion to remove a single (often disconnected) node from a graph.*
- *node substitution to change the label (or color) of a given node.*
- *edge insertion to introduce a new colored edge between a pair of nodes.*
- *edge deletion to remove a single edge between a pair of nodes.*
- *edge substitution to change the label (or color) of a given edge.*

For ease of discuss, we begin the analysis from one single node.

Definition 9 (Complete Rooted Subgraph and Rooted Subgraph). *Given a graph G with a node u. The radius-k complete rooted subgraph of u, denoted as ${G_u^k}^*$, is the k-hop neighborhood subgraph of u. Besides, a radius-k **rooted subgraph** of u, denoted as G_u^k, is a subgraph of ${G_u^k}^*$.*

Given a graph G with a node u, the computational graph for u is extracted from the radius-k complete rooted subgraph of u since GNNs (including both WL-test based GNNs and RFGNN) capture the radius-k complete rooted subgraph of u. Accordingly, we propose a new concept called rooted graph edit distance (RGED) to measure the similarity between two rooted graphs.

Definition 10 (Rooted Graph Edit Distance, RGED). *The rooted graph edit distance between two rooted graphs ${G_1}_u$ and ${G_2}_v$, written as $RGED({G_1}_u, {G_2}_v)$ can be defined as the graph edit distance between G_1 and G_2 in which u becomes v after the graph edit.*

We demonstrate that WL-test based GNNs are incapable of creating two message flow sets with a closer GED for certain pairs of GED-near graphs. Suppose there is a rooted graph ${G_1}_u^k$ and a rooted graph ${G_2}_u^k$ acquired by putting a node and an edge connecting the node into G_1. According to the definition of RGED, we have $RGED({G_1}_u^k, {G_2}_u^k)$ always equal to the cost of the addition of the node and edge. However, for WL-test based GNNs, the insertion of u_1 and an edge connecting it to G_1 will incur a different number of message flows for u depending on the paths connecting u and the added node. Especially when there are rings, the number of derived message flows can be large. Therefore, the message flows of two similar graphs created by WL-test based GNNs may vastly differ.

If there is no ring in both ${G_1}_u^k$ and ${G_2}_u^k$, the only difference between the two computational graphs created by RFGNN is the added node and edge, because of the non-redundancy of the computational graphs. On the other hand, if there are rings in ${G_1}_u^k$ and ${G_2}_u^k$, according to Lemma 9, the difference between the two computational graphs created by RFGNN is limited. Therefore, RFGNN would always generate two message flow sets with a significant degree of overlap for two graphs with a near graph edit distance (GED). Consequently, **RFGNN is more likely to be capable of comparing rooted graph similarity than WL-test based GNNs.** And hence, RFGNN is likely to be more capable of comparing graph similarity than WL-test based GNNs.

C Space and Time Complexity

Given a graph $G = (V, E)$ and a node $u \in V$. We measure the space complexity of GNNs by the number of nodes and edges whose information are required to store during message passing process. In addition, we investigate the time complexity of GNNs based on the number of nodes (aggregation and update operations) and edges (message passing operations) in the computational graphs to encode all nodes belonging G.

The number of nodes/edges in a TPF grows as its original graph's density increases. And a complete graph produces a TPF with more nodes than other graphs with the same number of nodes. If G is a complete graph and $\texttt{TPT}^{k}_{(G,u)}$ cannot express rings, any node v at l-layer of $\texttt{TPT}^{k}_{(G,u)}$ would have $|V| - 1 - l$ child nodes, where $0 \leq l < k$, because every node has $|V| - 1$ neighbors and a child node cannot be the same as any of its ancestor nodes. Then, we can conclude that $\texttt{TPT}^{k}_{(G,u)}$ has $1 + \sum\limits_{l=1}^{k} \prod\limits_{i=1}^{l} (|V| - i)$ nodes and $\sum\limits_{l=1}^{k} \prod\limits_{i=1}^{l} (|V| - i)$ edges. However, $\texttt{TPT}^{k}_{(G,u)}$ can express rings, so any node v at l-layer of $\texttt{TPT}^{k}_{(G,u)}$ additionally has one child node (i.e., u) so as to express a ring, where $2 \leq l \leq k - 1$. Therefore, $\texttt{TPT}^{k}_{(G,u)}$ has $1 + \sum\limits_{l=1}^{k} \prod\limits_{i=1}^{l} (|V| - i) + \sum\limits_{l=2}^{k-1} \prod\limits_{i=2}^{l} (|V| - i)$ nodes and $\sum\limits_{l=1}^{k} \prod\limits_{i=1}^{l} (|V| - i) + \sum\limits_{l=2}^{k-1} \prod\limits_{i=2}^{l} (|V| - i)$ edges. Consequently, $\texttt{TPF}^{k}_{G}$ has up to $|V| \left(1 + \sum\limits_{l=1}^{k} \prod\limits_{i=1}^{l} (|V| - i) + \sum\limits_{l=2}^{k-1} \prod\limits_{i=2}^{l} (|V| - i)\right)$ nodes and $|V| \left(\sum\limits_{l=1}^{k} \prod\limits_{i=1}^{l} (|V| - i) + \sum\limits_{l=2}^{k-1} \prod\limits_{i=2}^{l} (|V| - i)\right)$ edges.

From the analysis above, it can be determined that both the space and time complexities of a height k RFGNN are $\mathcal{O}\left(\frac{|V|!}{(|V|-k-1)!}\right)$.