TOWARDS A COMPLETE LOGICAL FRAMEWORK FOR GNN EXPRESSIVENESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Designing expressive Graph neural networks (GNNs) is an important topic in graph machine learning fields. Traditionally, the Weisfeiler-Lehman (WL) test has been the primary measure for evaluating GNN expressiveness. However, high-order WL tests can be obscure, making it challenging to discern the specific graph patterns captured by them. Given the connection between WL tests and first-order logic, some have explored the logical expressiveness of Message Passing Neural Networks. This paper aims to establish a comprehensive and systematic relationship between GNNs and logic. We propose a framework for identifying the equivalent logical formulas for arbitrary GNN architectures, which not only explains existing models, but also provides inspiration for future research. As case studies, we analyze multiple classes of prominent GNNs within this framework, unifying different subareas of the field. Additionally, we conduct a detailed examination of homomorphism expressivity from a logical perspective and present a general method for determining the homomorphism expressivity of arbitrary GNN models, as well as addressing several open problems.

1 INTRODUCTION

Graph Neural Networks (GNNs) are the dominant approaches for learning graph-structured data and have achieved remarkable success over the past few years. Among them, Message Passing Neural Networks (MPNNs) (Kipf & Welling, 2016b) are prominent GNN models that learn node and graph representations by aggregating information from neighbors. However, a noticeable drawback of GNNs lies in their limited expressive power. Xu et al. (2018); Morris et al. (2019) discovered that the separation power of MPNNs is inherently restricted by 1-dimensional Weisfeiler-Lehman (1-WL) test. Subsequently, many studies have focused on enhancing expressiveness and designing more powerful GNN models using the k-WL framework as a metric.

While the k-WL hierarchy offers a systematic measure of GNN expressiveness that increases with k, it remains somewhat limited. First, it lacks interpretability. Despite 1-WL being a relatively straightforward procedure which aggregates neighborhood information, it is hard to understand what k-WL actually learns and how it surpasses $(k-1)$-WL. Second, WL tests are arguably too coarse to evaluate the expressive power of GNN models (Zhang et al., 2024; Morris et al., 2022; Puny et al., 2023): many works (Qian et al., 2022; Frasca et al., 2022) only provide loose upper bounds of expressiveness of their proposed models in terms of k-WL and most efficient GNNs are only proved to be more expressive than 1-WL by constructing specific example graphs (Zhang & Li, 2021; Bevilacqua et al., 2021; Zhang et al., 2023).

Apart from the WL hierarchy, some works systematically study GNN expressivity from various perspectives. For instance, Zhang et al. (2024) identified all substructures captured by several popular GNN models. Xu & Zou (2024) examined the approximate inference capabilities of popular GNN models. These works, although provide novel insights about GNN expressivity, still lack *extendability*: they do not provide a general method for analyzing the expressiveness of *arbitrary* GNN models using their theoretical framework. Thus, considerable effort is required when considering novel GNN variants.

To address these limitations, this paper studies GNN expressivity from a logical perspective. Previous research, such as Barceló et al. (2020), investigated the logical expressivity of MPNNs, while Huang et al. (2024) explored the logical expressivity of a specific class of GNN models for link prediction in knowledge graphs. However, these works study MPNNs and other GNN variants separately, leaving many popular models unexamined. Additionally, there still lacks a unified framework for assessing the logical expressivity of general GNN models.

Contributions. This paper presents a novel framework for assessing the logical expressivity of *arbitrary* GNN models, provided they can be represented through a series of combination and aggregation operations. We present a method for constructing the set of logical formulas captured by these GNN models. Using this framework, we describe the logical expressivity of popular GNN models in terms of graph-level, node-level, and link-level predictions. Furthermore, we demonstrate how several key topics in GNN expressivity, such as homomorphism expressivity, expressivity comparisons, and estimating WL upper bounds, can be effectively addressed by leveraging the logical expressivity results present in this study.

2 Background

Notations and definitions. We use $\{\}$ to denote sets and use $\{\{\}\}$ to denote multisets. The index set is denoted as $[n] = \{1, ..., n\}$. In this paper we consider finite, directed graphs with node and edge labels. Let $G = (\mathcal{V}_G, \mathcal{E}_G)$ be a graph where $\mathcal{V}_G$ denotes the set of nodes in G and $\mathcal{E}_G$ the set of edges. ℓ denotes the label function that maps nodes and edges to labels: $\ell(u)$ is the label of node u and $\ell(u, v)$ the label of (u, v) provided that edge (u, v) exists. $\mathcal{N}(u)$ denotes the set of neighbors of node u. We use symbols $\varphi, \psi, \phi, ...$ to refer to logic formulas.

In this paper we focus on logic formulas and GNN models that operate on nodes and more generally node tuples. For instance, logic formulas $\psi(x_1, x_2, x_3)$ and GNNs that learn representations for node pairs. We define the *order* of logic formulas and GNNs to be the size of node tuples considered, e.g. the order of ψ is 3 and the order of GNNs that compute representations for node pairs is 2. For notation brevity we use $\mathbf{u}, \mathbf{v}, \mathbf{x}, ... \in \mathcal{V}^k$ to refer to node tuples where $k \in \{0, 1, ...\}$ is the order of them, e.g. $\psi(x_1, x_2, x_3)$ is represented by $\psi(\mathbf{x})$ where $\mathbf{x}$ is a 3-order tuple $\mathbf{x} = (x_1, x_2, x_3)$. Specially, if $k = 1$, then $\mathbf{u} \in \mathcal{V}^1$ represents nodes in graphs; if $k = 0$, then $\mathbf{u} \in \mathcal{V}^0$ simple represents none.

Weisfeiler-Lehman tests and graph isomorphism. Two graphs $G = (\mathcal{V}_G, \mathcal{E}_G)$ and $H = (\mathcal{V}_H, \mathcal{E}_H)$ are isomorphic, denoted as $G \simeq H$, if $|\mathcal{V}_G| = |\mathcal{V}_H|$ and there exists a bijective permutation $\pi : \mathcal{V}_G \to \mathcal{V}_H$ satisfying: (1) $(u, v) \in \mathcal{E}_G \iff (\pi(u), \pi(v)) \in \mathcal{E}_H$ for $u, v \in \mathcal{V}_G$, (2) $\ell(u) = \ell(\pi(v))$ for $u \in \mathcal{V}_G$ and (3) $\ell(u, v) = \ell(\pi(u), \pi(v))$ for $(u, v) \in \mathcal{E}_G$. Such π is an isomorphism from G to H.

Weisfeiler-Lehman (WL) tests are a family of necessary tests for graph isomorphism. Apart from some corner cases (Cai et al., 1992), they are effective and computationally efficient tests for graph isomorphism. Its 1-dimensional variant iteratively aggregates the colors of nodes and their neighbors and then injectively hashes them into new colors. The algorithms decides two graphs non-isomorphic if the colors of two graphs are different.

Extending from classic WL tests, k-dimensional WL test (k-WL) refines colors for node tuples. At beginning, the color of a node tuple $\mathbf{u}$ is set to be injective w.r.t. the structure of $\mathbf{u}$, denoted as $\textsf{atp}(\mathbf{u})$. That is, for arbitrary two tuples $\mathbf{u} = (u_1, ..., u_k)$ and $\mathbf{v} = (v_1, ..., v_k)$, $\textsf{atp}(\mathbf{u}) = \textsf{atp}(\mathbf{v})$ *iff* there exists an isomorphism π for the subgraphs induced by nodes in $\mathbf{u}, \mathbf{v}$ and $\pi(u_i) = v_i$ for $i = 1, ..., k$. k-WL then recursively refines these colors until convergence. The details of WL tests are discussed in Appendix B.

First-order Logic. We briefly introduce first-order logic and its relation with graphs. Consider the following formula

$$\varphi(x) := \mathrm{Red}(x) \land \exists y \, (E(x, y) \land \mathrm{Blue}(y)) \, .$$

There are two variables $\textsf{var}(\varphi) = \{x, y\}$ in the formulation of φ, and φ has exactly one free variable $\textsf{free}(\varphi) = \{x\}$ which is not bounded by any quantifiers $\exists, \forall$. $\varphi(x)$ is $\textsf{true}$ *iff* x is Red and exists a Blue y such that $E(x, y)$ holds. It is straightforward to relate this formula with graphs: variables x, y are corresponded nodes in graphs and the predicates Blue, Red are corresponded to node labels while E is corresponded to edges. Therefore, $\varphi(x)$ determines whether a node x is Blue and has a Blue neighbor.

In this paper we focus on a fragment of the first-order logic which allows the utilization of counting quantifiers $\exists^{\geq N}$. The semantic of the quantifier $\exists^{\geq N}$ where $N \in \{1, 2, ...\}$ is to describe "there exists no less than N variables such that". For example, consider

$$\psi(x) := \exists^{\geq 2} y \left(E(x, y) \wedge \text{Blue}(y)\right).$$

$\psi(x)$ is **true** *iff* x has 2 or more Blue neighbors. Such a family of logic formulas is called First-order Logic with Counting quantifiers (FOC) and possesses following property.

Proposition 1. *(Cai et al., 1992) For any graphs G, H, k-WL assigns the same color to G and H* iff *all FOC formulas with no more than k variables classifies G and H the same.*

Graph neural networks. GNNs can be generally described as graph functions that are invariant under isomorphism. Most popular GNNs follow a *color refinement* paradigm Zhang et al. (2024) to achieve such invariance: they maintain a representation for each node (or more generally, node tuple) and iteratively updates these representations via combination and aggregation functions. Consider, for example, message passing neural networks (MPNNs) Morris et al. (2019), which maintains a representation $\chi^{(l)}(x)$ for node x at layer l. The representations are updated using the following formula:

$$\chi^{(l+1)}(x) = \text{COM}^{(l)}\left(\chi^{(l)}(x), \text{AGG}^{(l)}\left(\left\{\left\{\chi^{(l)}(y) \mid y \in \mathcal{N}(x)\right\}\right\}\right)\right),$$

where $\text{COM}^{(l)}(\cdot, \cdot)$ represents an arbitrary function combining two representations and $\text{AGG}^{(l)}(\{\{\cdot\}\})$ represents an arbitrary permutation-invariant function that aggregates a multi-set of representations.

There are also many other popular GNNs, which are listed in Appendix C. Generally, these models maintain a representation $\chi^{(l)}(\mathbf{u})$ for node tuple $\mathbf{u}$ at layer l. Let L be the total number of layers of a GNN model. Then the representation $\chi^{(L)}(\mathbf{u})$ for node tuple $\mathbf{u}$ at layer L serves as the output of the GNN. Since this paper studies the relationship between GNNs and logic formulas, we focus on GNNs with binary outputs (i.e., **true** and **false**).

3 Logical Expressivity of Graph Neural Networks

3.1 Equivalent Logic Sets

Given a GNN model M and a logic formula φ, let $\chi(\mathbf{u})$ be the output of M for node tuples $\mathbf{u} \in \mathcal{V}^k$. We say M captures φ if the results of φ are reproduced by M, Concretely, M captures φ if the orders of M and φ are equal, and $\varphi(\mathbf{u}) = \chi(\mathbf{u})$ holds for arbitrary graph G and $\mathbf{u} \in \mathcal{V}^k$. In this paper, we attempt to answer the question: *what logic formulas can GNN models capture?* This leads to the following definition of logical expressivity.

Definition 2. The equivalent logic set of a class of GNN models is a subset Φ of first order logic formulas captured by the GNN models, which satisfies:

- The order of $\varphi \in \Phi$ is equivalent to the order of M: suppose GNNs compute k-order representations, then $\varphi \in \Phi$ should also be k-order;
- For all $\varphi \in \Phi$, there exists a GNN model M such that for arbitrary graphs G and $\mathbf{u} \in \mathcal{V}_G^k$, $\varphi(\mathbf{u}) =$ **true** *iff* $\chi(\mathbf{u}) =$ **true** where χ is the output of M;
- Given any graphs G, H and $\mathbf{u} \in \mathcal{V}_G^k, \mathbf{v} \in \mathcal{V}_H^k$, the GNN models cannot distinguish $\mathbf{u}, \mathbf{v}$ *iff* all logic formulas $\varphi \in \Phi$ classify $\mathbf{u}, \mathbf{v}$ the same.

The equivalent logic sets therefore sufficiently describe the logical expressiveness of GNN models. Moreover, similar to the homomorphism expressivity (Zhang et al., 2024), the metric of equivalent logic sets is also *quantitative*, as we can identify distinct logic sets for different GNN models that precisely describe their expressiveness , making it finer than metrics based on graph isomorphism tests which only provide qualitative results. Moreover, the equivalent logic sets can also be used to compare the expressivity of different GNNs: a class of GNN models M_1 is more expressive than M_2 *iff* $\Phi_2 \subset \Phi_1$ where Φ_1, Φ_2 are the equivalent logic set of M_1 and M_2 respectively. Above all, the significance of logical expressivity lies in its interpretability: we can not only describe what patterns GNN models can capture, but also understand the expressivity gap of different models by studying the difference of the corresponding equivalent logic sets.

3.2 Describing Logical expressivity for GNNs

It is evident that the equivalent logic set of GNNs can be *infinite*. We utilize a recursive construction procedure to describe such sets, similar to previous works. Consider the set of graded model logic Φ (Barceló et al., 2020) which is the equivalent logic set of MPNNs for example . Φ is defined by specifying how its elements are recursively constructed: to begin with, let $\mathrm{Col}(x) \in \Phi$ where Col represents node colors. Each element of Φ is either Col, or one of the following:

$$\neg\varphi(x), \quad \varphi(x) \wedge \varphi'(x), \quad \exists^{\geq N}\left(E(x,y) \wedge \varphi(y)\right),$$

where $\varphi, \varphi' \in \Phi$ and N is a positive integer. Therefore, we define Φ by specifying how its elements are constructed, starting from the input node colors Col. For notation brevity we can abbreviate the definition into one line:

$$\varphi(x) := \exists^{\geq N}\left(\varphi'(x) \wedge E(x,y)\right) \mid \neg\varphi'(x) \mid \varphi'(x) \wedge \varphi''(x) \mid \mathrm{Col}(x), \tag{1}$$

with the convention that logic formulas φ together with its superscript variants φ', φ'' belong to the same logic set Φ. Eq. 1 discovered by Barceló et al. (2020) describes the equivalent logic set of MPNNs with undirected, homogeneous input graphs.

4 General Aggregate-Combine Networks

To formally discuss the logical expressivity of arbitrary GNN models, we first summarize GNN models including Message Passing Neural Networks (Xu et al., 2018), Higher-order GNNs (Morris et al., 2018), Subgraph GNNs (Bevilacqua et al., 2021), via a unified design paradigm namely General Aggregate-Combine Neural Networks (GACNNs). The basic idea is straightforward: we decompose the structure of different GNN layers into the same, principled *aggregation* and *combination* function series, which further enable us to study the expressive power of different GNN models via a unified framework.

Formally, let $\chi^{(l)}(\mathbf{u})$ be the representation of a k-order node tuple $\mathbf{u} \in \mathcal{V}^k$ computed by the l-th GACNN layer. The $(l+1)$-th GACNN layer takes $\chi^{(l)}(\mathbf{u})$ as input and evaluates $\chi^{(l+1)}(\mathbf{u})$ for $\mathbf{u} \in \mathcal{V}^k$. $\chi^{(l+1)}(\mathbf{u})$ is computed by $\chi^{(l)}(\mathbf{u})$ via a sequence of two operations: *combination* (denoted by COM) and *aggregation* (denoted by AGG) . $\mathrm{COM}(\cdot,\cdot)$ represents an arbitrary function combining two representations, and $\mathrm{AGG}(\{\!\{\cdot\}\!\})$ represents a permutation-invariant function that aggregates a multi-set of representations. The evaluation of a GACNN layer is decomposed into a series of intermediate variables $\{\chi_1^{(l)}, ..., \chi_K^{(l)}\}$. Denoting $\chi_0^{(l)} := \chi^{(l)}$ and $\chi_{K+1}^{(l)} = \chi^{(l+1)}$, we define either

$$\chi_i^{(l)}(\mathbf{u}) = \mathrm{COM}_i^{(l)}\left(\chi_j^{(l)}(\mathbf{u}), \chi_k^{(l)}(\mathbf{u})\right), \quad \text{or} \quad \chi_i^{(l)}(\mathbf{u}) = \mathrm{AGG}_i^{(l)}\left(\left\{\!\!\left\{\chi_j^{(l)}(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}(\mathbf{u})\right\}\!\!\right\}\right), \tag{2}$$

where $1 \leq j,k < i \leq K+1$, $\mathrm{COM}_i^{(l)}$ is a combination function, $\mathrm{AGG}_i^{(l)}$ is an aggregation function and $\mathcal{N}(\mathbf{u})$ is the generalized neighbor[1] of $\mathbf{u}$ defined by the GNN model. Specially, we denote by $\chi^{(0)} = \mathrm{INIT}(\mathbf{u})$ the initial representation of $\mathbf{u}$. The above definition generally expresses the aggregation and combination steps of GNN layers.

Example. To better introduce the idea of GACNNs we illustrate how MPNNs are described by the above GACNNs construction steps. Consider MPNNs Xu et al. (2018) whose layers are defined by

$$\chi^{(l+1)}(x) = \mathrm{COM}^{(l)}\left(\chi^{(l)}(x), \mathrm{AGG}^{(l)}\left(\left\{\!\!\left\{\chi^{(l)}(y) \mid y \in N(x)\right\}\!\!\right\}\right)\right),$$

[1] Certain GNNs generalize the concept of neighbor in graphs in different manners. For example, 2-FGNNs define the neighbor of a node pair (x,y) to be $\mathcal{N}_{\text{2-FGNN}}(x,y) = \{((x,z),(z,y)) \mid z \in \mathcal{V}\}$

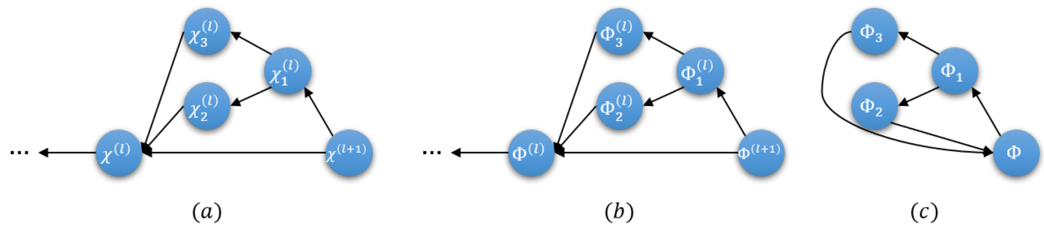


Figure 1: Left: the structure of the example GACNN. Middle: the structure of the equivalent logic sets, each corresponding to one of the nodes in the left GACNN. Right: the structure of the equivalent logic sets of the left GACNN, regardless of the number of layers l.

where x, y denotes nodes in graphs. We can simply decompose one layer of MPNN into

$$\chi^{(l+1)}(x) = \mathrm{COM}^{(l)}\left(\chi^{(l)}(x), \chi_1^{(l)}(x)\right),$$
$$\chi_1^{(l)}(x) = \mathrm{AGG}^{(l)}\left(\left\{\left\{\chi^{(l)}(y) \mid v \in N(x)\right\}\right\}\right).$$

In this manner we describe the MPNN layers using the principled GACNN framework. To better explain our framework, consider Local 2-GNN Zhang et al. (2024) whose layers are defined by

$$\begin{aligned}\chi^{(l+1)}(x,y) = \mathrm{COM}^{(l)}\Big(\chi^{(l)}(x,y),\ &\mathrm{AGG}_1^{(l)}\left(\left\{\left\{\chi^{(l)}(z,y) \mid z \in \mathcal{N}(x)\right\}\right\}\right), \\ &\mathrm{AGG}_2^{(l)}\left(\left\{\left\{\chi^{(l)}(x,z) \mid z \in \mathcal{N}(y)\right\}\right\}\right)\Big).\end{aligned} \tag{3}$$

Similarly, we can decompose one layer of Local 2-GNN into

$$\begin{aligned}&\chi^{(l+1)}(x,y) = \mathrm{COM}_1^{(l)}\left(\chi^{(l)}(x,y), \chi_1^{(l)}(x,y)\right), \qquad \chi_1^{(l)}(x,y) = \mathrm{COM}_2^{(l)}\left(\chi_2^{(l)}(x,y), \chi_3^{(l)}(x,y)\right), \\ &\chi_2^{(l)}(x,y) = \mathrm{AGG}_1^{(l)}\left(\left\{\left\{\chi^{(l)}(z,y) \mid z \in \mathcal{N}(x)\right\}\right\}\right), \chi_3^{(l)}(x,y) = \mathrm{AGG}_2^{(l)}\left(\left\{\left\{\chi^{(l)}(x,z) \mid z \in \mathcal{N}(y)\right\}\right\}\right),\end{aligned} \tag{4}$$

where $\mathrm{COM}_1^{(l)}, \mathrm{COM}_2^{(l)}$ are combination functions satisfying $\mathrm{COM}_1^{(l)}\left(\chi, \mathrm{COM}_2^{(l)}(\chi', \chi'')\right) = \mathrm{COM}^{(l)}(\chi, \chi', \chi'')$ for arbitrary representations χ, χ', χ''. Appendix E.6 illustrates how we build popular GNN variants using the GACNN framework.

Structure of GACNNs. The advantage of decomposing GNN layers into a series of the principled aggregation and combination procedures is that we can unify the study of complicated GNN models into the study of AGG and COM modules. Consider, for example, the Local 2-GNN model. The evaluation of this model can be illustrated as Figure 1 (a), where the node $\chi^{(l+1)}$ represents representations of all nodes at layer $l+1$ and is evaluated directly by $\chi_1^{(l)}$ and $\chi_2^{(l)}$, which are again evaluated by their children along the hierarchy until $\chi^{(l)}$. It is evident that by explicitly expanding the evaluation procedure of $\chi^{(l+1)}$, the whole and complicated GNN computation structure is broken down into small and simple pieces containing only two types of computation: let χ_p be a (parent) node, then if χ_p only has one child χ_l, χ_p is evaluated in the form of $\chi_p = \mathrm{AGG}_p(\{\{\chi_l\}\})$; otherwise χ_p has two children χ_l, χ_r and is evaluated by $\chi_p = \mathrm{COM}_p(\chi_l, \chi_r)$. In this manner we break down the computation procedure of GNNs into principled aggregation and combination procedure, each corresponding to a parent-children pair in the computation graph. We next investigate the logical expressivity of GACNNs by studying the *local property* of such parent-children pairs.

5 On the Equivalent Logic Fragment of Graph Neural Networks

In this section we discuss the separation power and function approximation property of general graph neural networks by providing the equivalent logic set of arbitrary GACNNs.

5.1 Equivalent Logic Sets for General Computation Procedure

For now let us relax the utilization of GACNN models and focus purely on the two types of computation units AGG and COM proposed in Section 4. Concretely, suppose a set $\{\chi_1, \chi_2, ..., \chi_K\}$ where χ_i maps a k-order node tuple $\mathbf{u}$ to its color $\chi_i(\mathbf{u})$ and each χ_i for $i \in [K]$ is defined by either $\chi_i(\mathbf{u}) = \mathrm{AGG}_i\left(\{\!\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\!\}\right)$, $\chi_i(\mathbf{u}) = \mathrm{COM}_i\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right)$, or $\chi_i(\mathbf{u}) = \mathrm{INIT}_i\left(\mathbf{u}\right)$ where $i > j, k$. We generalize the concept of equivalent logic sets to model the logical expressivity of $\{\chi_i\}_{i\in[K]}$.

Definition 3. The equivalent logic set of χ_i for $i \in [K]$ defined above is a subset Φ_i of first order logic formulas satisfying:

- The arity of $\varphi_i \in \Phi_i$ matches the output of χ_i;
- For all $\varphi_i \in \Phi_i$, there exists a series of functions $\{\mathrm{COM}_j\}_{j\in[i]}, \{\mathrm{AGG}_j\}_{j\in[i]}$ and $\{\mathrm{INIT}_j\}_{j\in[i]}$ such that for arbitrary graphs G and $\mathbf{u} \in \mathcal{V}_G^k$, $\varphi_i(\mathbf{u}) = \textbf{true}$ *iff* $\chi_i(\mathbf{u}) = \textbf{true}$;
- Given any graphs G, H and $\mathbf{u} \in \mathcal{V}_G^k, \mathbf{v} \in \mathcal{V}_H^k$, χ_i cannot distinguish $\mathbf{u}, \mathbf{v}$ *iff* all logic formulas $\varphi_i \in \Phi_i$ classify $\mathbf{u}, \mathbf{v}$ the same.

With the above definition, we are ready to introduce our main result. We next show that it is possible to find the equivalent logic sets of $\{\chi_i\}_{i\in[K]}$.

Theorem 4. *Given $\{\chi_1, \chi_2, ..., \chi_K\}$ defined above, there exists $\{\Phi_1, ..., \Phi_K\}$ where Φ_i is the equivalent logic set of χ_i for $i \in [K]$. Moreover, each $\varphi_i \in \Phi_i$ is given by:*

- $$\chi_i(\mathbf{u}) = \mathrm{AGG}_i\left(\{\!\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\!\}\right) \iff \varphi_i(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_j(\mathbf{v}) \wedge \mathbf{1}_{\mathbf{v}\in\mathcal{N}_i(\mathbf{u})}\right) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}),$$
- $$\chi_i(\mathbf{u}) = \mathrm{COM}_i\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right) \iff \varphi_i(\mathbf{u}) := \varphi_j(\mathbf{u}) \mid \varphi_k(\mathbf{u}) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}),$$
- $$\chi_i(\mathbf{u}) = \mathrm{INIT}_i\left(\mathbf{u}\right) \iff \varphi_i(\mathbf{u}) := \textbf{atp}(\mathbf{u}) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}),$$

where $\varphi_i', \varphi_i'' \in \Phi_i, \varphi_j \in \Phi_j, \varphi_k \in \Phi_k$.

Note that by denoting $\varphi_i(\mathbf{u}) := \textbf{atp}(\mathbf{u})$, we mean that $\varphi_i \in \Phi_i$ is capable of capturing all structures of the subgraph induced by $\mathbf{u}$. Concretely, for each possible color Col of **atp**, there exists a $\varphi^{\mathrm{Col}}(\mathbf{u})$ that is **true** if and only the color of $\mathbf{u}$ assigned by **atp** is Col. $\mathbf{1}$ is the indicator: $\mathbf{1}_{\text{condition}}$ is **true** *iff* the condition is satisfied.

Proof sketch. The theorem represents a major technical contribution, so we present a proof sketch below. Our proof is divided into two parts, presented in Appendix E.1. First, we show that each logic formula $\varphi_i \in \Phi_1$ can be captured by χ_i. Obviously, for $i = 1$, we only have $\varphi_1(\mathbf{u}) := \textbf{atp}(\mathbf{u}) \mid \neg\varphi_1'(\mathbf{u}) \mid \varphi_1'(\mathbf{u}) \wedge \varphi_1''(\mathbf{u})$, which can be directly described by $\chi_1(\mathbf{u}) = \mathrm{INIT}_1(\mathbf{u})$. By induction on i, we suppose all $\varphi_j \in \Phi_j$ for $j < i$ can be captured by χ_j. Then, given arbitrary φ_i, we provide a method to explicitly construct the corresponding $\{\mathrm{COM}_j\}_{j\in[i]}, \{\mathrm{AGG}_j\}_{j\in[i]}$ and $\{\mathrm{INIT}_j\}_{j\in[i]}$ functions so that φ_i is captured by χ_i.

In the next step, we prove that for any graphs G, H and $\mathbf{u} \in \mathcal{V}_G^k, \mathbf{v} \in \mathcal{V}_H^k$, χ_i cannot distinguish $\mathbf{u}, \mathbf{v}$ *iff* all $\varphi_i \in \Phi_i$ classify $\mathbf{u}, \mathbf{v}$ the same. By utilizing the fact that each $\varphi_i \in \Phi_i$ is captured by χ_i, the first direction is proved. It is therefore only necessary to show $\chi_i(\mathbf{u}) \neq \chi_i(\mathbf{v}) \Rightarrow$ there exists $\varphi_i \in \Phi_i$ satisfying $\varphi_i(\mathbf{u}) \neq \varphi_i(\mathbf{v})$. Again, we prove by induction on i. Suppose for all $j < i$ the statement holds. We then enumerate all possible cases where $\chi_i(\mathbf{u}) \neq \chi_i(\mathbf{v})$: for example if $\chi_i(\mathbf{u}) = \mathrm{COM}_i\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right)$, then there are three cases: (1) $\chi_j(\mathbf{u}) \neq \chi_j(\mathbf{v}), \chi_k(\mathbf{u}) = \chi_k(\mathbf{v})$; (2) $\chi_j(\mathbf{u}) = \chi_j(\mathbf{v}), \chi_k(\mathbf{u}) \neq \chi_k(\mathbf{v})$, and (3) $\chi_j(\mathbf{u}) \neq \chi_j(\mathbf{v}), \chi_k(\mathbf{u}) \neq \chi_k(\mathbf{v})$. For each case, we provide a method to construct $\varphi_i \in \Phi_i$ satisfying $\varphi_i(\mathbf{u}) \neq \varphi_i(\mathbf{v})$, thus concluding the proof.

The results in Theorem 4 specifies the construction of equivalent logic sets for arbitrary computation procedure built upon aggregation and combination functions, which is not only confined to graphs and GACNN models. As the central finding of this paper, it enables the study of complicated models built upon multiple heterogeneous layers and graphs containing node and edge labels. Consider

Figure 1 (b) for example: the equivalent logic set of each node in the computation graph in Figure 1 (a) only depends on its local neighbors. If the equivalent logic set $\Phi^{(l)}$ of $\chi^{(l)}$ is known, all logic sets in Figure 1 (b) are specified by Theorem 4. This indicates that once the input representations can be described by logic formulas, we are able to determine the logical expressivity of arbitrary functions over the input representations built upon aggregation and combination functions.

5.2 Main Results

In this section we formally describe our results for GACNNs. We assume a L-layer GACNN layers defined in the form of Eq. 2. For layer $l \in [L]$, let $\chi^{(l)}$ be the output representation at layer l and let $\left\{\chi_1^{(l)}, ..., \chi_K^{(l)}\right\}$ be the set of intermediate representations when computing $\chi^{(l+1)}$ from $\chi^{(l)}$. Similarly, we denote $\Phi^{(l)}$ to be the equivalent logic set of $\chi^{(l)}$ and $\Phi_i^{(l)}$ the equivalent logic set of $\chi_i^{(l)}$ for $i \in [K]$. Obviously $\Phi^{(l)}$ and $\Phi_i^{(l)}$ for $i \in [K]$ and $l \in [L]$ are directly defined by Theorem 4, which directly leads to the following result:

Corollary 5. *The equivalent logic set of L-layer GACNNs defined above is given by $\Phi^{(L)}$.*

Corollary 5 requires to specify the number of GACNN layers L. To derive a general result for all GACNNs regardless of the number of layers, we propose the following proposition.

Proposition 6. *Denote $\chi_0^{(l)} = \chi^{(l)}, \chi_{K+1}^{(l)} = \chi^{(l+1)}$ and $\Phi_{K+1} = \Phi_0 = \Phi$, Let $\Phi^{(l)}, \{\Phi_i^{(L)}\}_{i\in[K]}$ be the equivalent logic sets defined above. Then, $\Phi = \bigcup_{L=0}^{\infty} \Phi^{(L)}$ and $\Phi_i = \bigcup_{L=0}^{\infty} \Phi_i^{(L)}$ for $i \in [K]$ are defined by*

- $$\chi_i^{(l)}(\mathbf{u}) = \mathrm{AGG}_i^{(l)}\left(\left\{\left\{\chi_j^{(l)}(\mathbf{x}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\right\}\right\}\right) \text{ for } l \in [0, \infty)$$
$$\iff \varphi_i(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_j(\mathbf{v}) \wedge \mathbf{1}_{\mathbf{v}\in\mathcal{N}_i(\mathbf{u})}\right) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}) \mid \textsf{atp}(\mathbf{u}),$$
- $$\chi_i^{(l)}(\mathbf{u}) = \mathrm{COM}_i^{(l)}\left(\chi_j^{(l)}(\mathbf{u}), \chi_k^{(l)}(\mathbf{u})\right) \text{ for } l \in [0, \infty)$$
$$\iff \varphi_i(\mathbf{u}) := \varphi_j(\mathbf{u}) \mid \varphi_k(\mathbf{u}) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}) \mid \textsf{atp}(\mathbf{u}),$$

where $\varphi_i, \varphi_i', \varphi_i'' \in \Phi_i, \varphi_j \in \Phi_j, \varphi_k \in \Phi_k$ for $i \in \{0\} \cup [K+1]$.

The difference between Proposition 6 and Theorem 4 is that we now allow the construction of arbitrary number of GACNN layers. As long as GACNN layers share identical structure (which holds for most GNN models), we can fully describe their logical expressivity by Φ regardless of the number of layers. The structure of logic sets in Proposition 6 is illustrated in Figure 1 (c), where each set is irrelevant to the number of layers l and is constructed recursively. Given any class of GNN models, so long as we can break down a layer into a series of aggregation and combination operations, we can formally define its logical expressivity using Proposition 6.

About graph-level readout. Generally, GNNs compute graph representations by aggregating node tuple representations, i.e. $\chi_G = \mathrm{AGG}\left(\left\{\left\{\chi^{(L)}(\mathbf{u}) \mid \mathbf{u} \in \mathcal{V}^k\right\}\right\}\right)$ where L is the output layer. We determine the equivalent logic set of χ_G below.

Proposition 7. *The equivalent logic set Φ_G of the graph representation χ_G defined above is given by*

$$\varphi_G := \exists^{\geq N}\left(\varphi(\mathbf{u})\right) \mid \neg\omega_G' \mid \omega_G' \wedge \omega_G'',$$

where $\varphi_G, \varphi_G', \varphi_G'' \in \Phi_G$, and $\varphi \in \Phi$ is the equivalent logic set of $\chi^{(L)}$.

The results in this section provide a general method for determining the equivalent logic set of arbitrary GACNNs. In the remaining of this paper, we utilize these results to discuss several important topics implied by the logical expressiveness of GNNs.

6 Implications

With the complete description of logical expressivity for general GNN models in previous section, we now discuss how these results provide novel insights into understanding modern GNN frameworks. In this section, we highlight the significance of our theory by introducing important results induced by our theory.

6.1 Regarding Existing GNN models

First of all, we apply our results to formally describe the logical expressivity of popular GNNs including models for graph-level or node-level prediction MPNNs (Xu et al., 2018), Subgraph GNNs (Bevilacqua et al., 2021; Qian et al., 2022), Local GNNs (Zhang et al., 2024), Folklore-type GNNs (Zhang et al., 2024) and models for link prediction NBFNet (Zhu et al., 2021), SEAL (Zhang et al., 2020), etc. The details of these models are in Appendix C. For brevity we make the convention that the equivalent logic set of each class of GNNs is represented by Φ and denote by $\varphi, \varphi', ... \in \Phi$ the elements of Φ. To properly express Φ it is sometimes convenient to also define an auxiliary logic set which helps with the explanation of Φ. We denote $\psi, \psi', ... \in \Psi$ as such auxiliary logic sets. The result is summarized in Proposition 8. For notation brevity, since the terms in the form of $\varphi := \neg\varphi' \mid \varphi' \wedge \varphi'' \mid \textsf{atp}$ emerges in the definition of all logic formulas, they are omitted in the following description.

Proposition 8. *The equivalent logic sets of GNN models can be separately defined as:*

- ***MPNN**: $\varphi(x) := \exists^{\geq N} x\,(\varphi'(y) \wedge E(x, y))$, where E is the edge predicate.*
- ***Subgraph GNN (weak)**:* $\varphi(x) := \exists^{\geq N} y\,(\psi(x, y))\,, \psi(x, y) := \exists^{\geq N} z\,(\psi'(x, z) \wedge E(z, y))$.
- ***Subgraph GNN (strong)**:* $\varphi(x) := \exists^{\geq N} y\,(\varphi'(y) \wedge \psi(y, x))\,, \psi(x, y) := \exists^{\geq N} z\,(\psi(x, z) \wedge E(z, y)) \mid \varphi(y)$.
- ***NBFNet**: $\varphi(x, y) := \exists^{\geq N} z\,(\varphi'(x, z) \wedge E(z, y))$.*
- ***Local 2-GNN**: $\varphi(x, y) := \exists^{\geq N} z\,(\varphi'(x, z) \wedge E(z, y))) \mid \exists^{\geq N} z\,(E(x, z) \wedge \varphi'(z, y)))$*
- ***2-FGNN**: $\varphi(x, y) := \exists^{\geq N} z\,(\varphi'(x, z) \wedge \varphi''(z, y))$.*
- ***SEAL (MPNN)**:* $\varphi(x, y) := \exists^{\geq N} z\,(\psi(x, z, y))\,, \psi(x, z, y) := \exists^{\geq N} w\,(\psi(x, w, y) \wedge E(w, z)))$.
- ***2-GNN**: $\varphi(x, y) := \exists^{\geq N} z\,(\varphi'(x, z)) \mid \exists^{\geq N} z\,(\varphi'(z, y))$.*

Proof sketch. The proposition directly utilize the results in Proposition 6 to derive the equivalent logic sets of popular GNNs. We present a proof sketch for Local 2-GNNs and illustrate the pipeline for determining the expressivity of certain GNN models. First, we explicitly write down the GNN layers as Eq. 3. Then, we transform the GNN layers into GACNN layers by decomposing each layer into a sequence of AGG and COM functions, as in Eq. 4. By utilizing Proposition 6, we can directly obtain the equivalent logic set of Local 2-GNNs as below. (Again, we omit the terms in the form of $\varphi := \neg\varphi' \mid \varphi' \wedge \varphi'' \mid \textsf{atp}$ for notation brevity.)

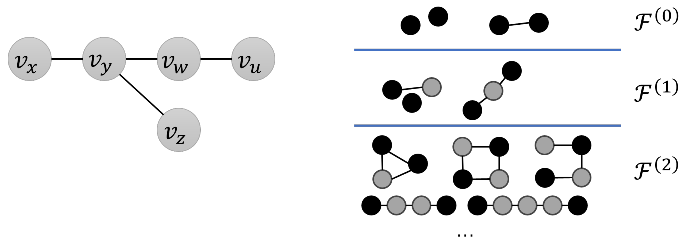
$$\begin{aligned} &\varphi(x, y) := \varphi'(x, y) \mid \varphi_1(x, y), && \varphi_1(x, y) := \varphi_2(x, y) \mid \varphi_3(x, y), \\ &\varphi_2(x, y) := \exists^{\geq N} z(\varphi(z, y) \wedge E(x, z)), && \varphi_3(x, y) := \exists^{\geq N} z(\varphi(x, z) \wedge E(z, y)), \end{aligned}$$

where $\varphi, \varphi' \in \Phi$ is the equivalent logic set of Local 2-GNNs, and $\varphi_1 \in \Phi_1, \varphi_2 \in \Phi_2, \varphi_3 \in \Phi_3$ are auxiliary logic sets. It is therefore only one step before the result in Proposition 8: by substituting the definition of φ_2, φ_3 into φ_1 and further substituting the definition of φ_1 into φ, we can write down the definition of φ into one line:

$$\varphi(x, y) := \varphi'(x, y) \mid \exists^{\geq N} z\,(\varphi'(x, z) \wedge E(z, y))) \mid \exists^{\geq N} z\,(E(x, z) \wedge \varphi'(z, y)))\,.$$

Removing the redundant term $\varphi(x, y) := \varphi'(x, y)$ directly yields the result in Proposition 8.

Proposition 8 gives a unified description of the logical expressivity of popular GNN models. The result of MPNNs follows Barceló et al. (2020). Following up, it is obvious that Subgraph GNNs (weak) surpasses MPNN by modeling more complex *relations* between nodes: rather than simply the edges E, they deploy the more general logic formulas ψ for modeling relations between nodes, which is obviously more powerful. Continuing, Subgraph GNN (strong) further strengthen ψ by not



(a) (b)

Figure 2: An illustration of the subgraph used for construct the homogeneous expressivity, (a) corresponds to the construction of $\varphi := \exists xyzwu\,((E(x,y) \wedge E(y,z) \wedge E(y,w) \wedge E(w,u))$. (b) corresponds to 2-FGNNs.

only allowing the single-source update pattern $\psi(x,y) := \exists^{\geq N} z\,(\psi'(x,z) \wedge E(z,y))$, but also aggregating information across different sources $\psi(x,y) := \varphi(y)$ (since $\varphi(y)$ aggregates $\psi(z,y)$ with different sources z). The rest of GNNs compute node-pair representations. Starting from NBFNet, it models the relation between two nodes $\varphi(x,y)$ by checking intermediate nodes z and its relation w.r.t. the two end nodes $\varphi'(x,z), E(z,y)$. This is useful for link prediction tasks, e.g. to predict whether two nodes are connected $\mathrm{Connect}(x,y) := \exists z\,(\mathrm{Connect}(x,y) \wedge E(y,z)) \mid E(x,y)$, which predicts the unknown $\mathrm{Connect}$ relation by utilizing the known edges E. This pattern is similar with the Bellman-Ford algorithm Baras & Theodorakopoulos (2022), which is a single-source shortest path algorithm. The logic formulas $\varphi(x,y)$ corresponding to NBFNet are also constructed with the single source node x. Local 2-GNN extends NBFNet by considering two sources x, y separately, which allows the construction of more complex logic formulas. 2-FGNNs further generalize by defining $\varphi(x,y)$ in a multi-source manner, analogous to Floyd shortest path algorithm. SEAL also defines $\varphi(x,y)$ in a multi-source manner, but it instead constructs $\psi(x,z,y)$ and uses the intermediate nodes z to perceive the relation between x, y simultaneously. 2-GNNs, although compute node-pair representations, are not suitable for link prediction since it fails to even express the simple logic rule $\mathrm{GrandParent}(x,y) := \exists z\,(\mathrm{Parent}(x,z) \wedge \mathrm{Parent}(z,y))$.

6.2 Structural Awareness of GNNs

There has been several works that study what graph structures different GNNs are aware of, such as cycles, cliques, etc. These concepts can be unified with logic formulas. For example, determining whether a node is in a 3-clique can be written as

$$\varphi_{\text{3-clique}}(x) := \exists y, z\,(E(x,y) \wedge E(y,z) \wedge E(z,x))\,.$$

Therefore, whether GNN models can capture 3-clique patterns depends on whether it captures $\varphi_{\text{3-clique}}$. However, determining 3-clique is a trivial task, and in practice it is often necessary to study whether GNNs can capture more complex structural patterns. We consider the concept of homogeneous expressivity proposed by Dell et al. (2018); Zhang et al. (2024).

Homomorphism expressivity. Homomorphism expressivity is a theory developed to precisely describe the structures of graphs being captured by GNNs. Concretely, let $G = (\mathcal{V}_G, \mathcal{E}_G), H = (\mathcal{V}_H, \mathcal{E}_H)$ be two graphs. A homomorphism from F to G is a mapping $\pi : \mathcal{V}_G \to \mathcal{V}_F$ that preserves labels (if any) and edges, i.e. $(\pi(u), \pi(v)) \in \mathcal{E}_H$ for all $(u,v) \in \mathcal{E}_G$ and $\ell(u) = \ell(\pi(u))$ for all nodes u, $\ell(u,v) = \ell(\pi(u), \pi(v))$ if there are node labels or edge labels respectively. $\mathsf{Hom}(F,G)$ is defined to be the number of homomorphisms from F to G. The crux is, to find all subgraphs F for GNNs such that, for all pairs of graphs G, H, GNNs distinguish $G, H \iff \mathsf{Hom}(F,G) \neq \mathsf{Hom}(F,H)$. Such a set of subgraphs is referred as the *homomorphism expressivity* of GNNs. Dell et al. (2018) gives the homomorphism expressivity for 1-WL (MPNNs), while Zhang et al. (2024) extends the results to several popular GNN models.

Similar as previous discussions, in this section we aim at providing a general method to determine the homomorphism expressivity of GACNNs, based on our findings about equivalent logic sets. Suppose we are given a class of GACNNs whose equivalent logic set is Φ. To simplify the discussion, we first assume no node / edge labels. We assume that the concept of neighbors in GACNNs is described by composition of edges: for example in MPNNs the neighbors of a node x is defined by $\mathbf{1}_{y\in\mathcal{N}(x)} := E(x, y)$ where E is the edge predicate; similarly in NBFNet $\mathbf{1}_{(x,z)\in\mathcal{N}_1(x,y)} := E(y, z)$. The homomorphism expressivity $\mathcal{F}$ can be constructed from the logic formulas in Φ via the following procedure.

1. Remove all formulas in Φ that contains negation $\neg$ or $\exists^{\geq n}$ where $n \geq 2$;
2. For each formula $\varphi \in \Phi$, add a graph F into $\mathcal{F}$ which is defined below:
   (a) There exists a bijective mapping τ from $\textsf{var}(\varphi)$ to $\mathcal{V}_F$, i.e. from the variables in φ (we avoid the reuse of variables)[2] to the nodes in F.
   (b) For any variables $x, y \in \textsf{var}(\varphi)$, $E(x, y)$ is a term in φ *iff* $E(\tau(x), \tau(y))$ is an edge of F.

A discussion about the reuse of variables and why we avoid this technique is in Appendix D. We now explain the procedure. Consider for example constructing a subgraph F for the logic formula $\varphi := \exists xyzwu\,((E(x, y) \wedge E(y, z) \wedge E(y, w) \wedge E(w, u))$. The construction of F is illustrated in Figure 2 (a), where F possesses a node corresponding to each variable x, y, z, w, u in φ and contains edges $E(x, y), E(y, z), E(y, w)$ and $E(w, u)$. We have the following result:

Theorem 9. *Given a class of GACNN models and suppose Φ be the equivalent logic set. Let $\mathcal{F}$ be the homomorphism expressivity constructed by Φ as discussed above. For all pairs of graphs G, H. the following statements are equivalent:*

1. $\textsf{Hom}(F, G) = \textsf{Hom}(F, H)$ *for all* $F \in \mathcal{F}$.
2. *All GACNNs do not distinguish G and H.*

Proof sketch. Theorem 9 represents another major technical contribution of the paper, so we present a proof sketch below. Given the homomorphism expressivity $\mathcal{F}$ and for any graph G, the intuition is that we can use logic formulas to count the number of homomorphisms from each $F \in \mathcal{F}$ to G. Consider the graph in Figure 2 (a) for example: the number of homomorphisms from it to G is N *iff* $\varphi := \exists^{=N}\mathbf{v}(E(v_x, v_y) \wedge E(v_y, v_w) \wedge E(v_y, v_z) \wedge E(v_w, v_u))$ evaluates $\textsf{true}$ in G, where $\mathbf{v} = (v_x, v_y, v_z, v_w, v_u)$. The next step of our proof is involved and shows that such φ can be expressed by logic formulas in the equivalent logic set and vice versa, which is presented in Appendix E.2, thus concluding the proof.

Theorem 9 validates the effectiveness of our construction procedure. Together, we provide a general method to identify the homomorphism expressivity for arbitrary GACNNs, which extends the known results in previous works (Dell et al., 2018; Zhang et al., 2024). Meanwhile, we have solved a conjecture in Zhang et al. (2024), i.e. when a GNN can be described by a GACNN, its homomorphism expressivity exists and is given by Theorem 9.

Example. We illustrate the strategy of recursively constructing homogeneous expressivity $\mathcal{F}$ by investigating 2-FGNNs, whose equivalent logic set Φ (removed negation and $\exists^{\geq N}$ for $N \geq 2$) is given by $\varphi(x, y) := \exists z\,(\varphi'(x, z) \wedge \varphi''(z, y)) \mid \varphi'(x, y) \wedge \varphi''(x, y) \mid \textsf{atp}(x, y)$. Let $\Phi^{(l)}$ be the equivalent logic set of l-layer 2-FGNNs. Let $\mathcal{F}^{(l)}$ be the homogeneous expressivity constructed at iteration l. For $\mathcal{F}^{(0)}$ at beginning, there are only two graphs in $\mathcal{F}^{(0)}$ corresponding to $\varphi^{(0)} \in \Phi^{(0)}$ where $\varphi^{(0)}(x, y) := \textsf{atp}(x, y)$, as illustrated in top of Figure 2 (b). At the next iteration, we consider the more complex $\Phi^{(1)}$, which is given by

$$\varphi^{(1)}(x, y) := \exists z\left(\varphi_1^{(0)}(x, z) \wedge \varphi_2^{(0)}(z, y)\right) \mid \varphi_1^{(0)}(x, y) \wedge \varphi_2^{(0)}(x, y) \mid \textsf{atp}(x, y).$$

We can simply construct $\mathcal{F}^{(1)}$ by reusing the known results about $\mathcal{F}^{(0)}$. Specially, to construct $\varphi^{(1)}(x, y) := \exists z\left(\varphi_1^{(0)}(x, z) \wedge \varphi_2^{(0)}(z, y)\right)$, we start from an empty graph F and add three nodes v_x, v_y, v_z corresponding to variables x, y, z in $\varphi^{(1)}$. Then, we replace (v_x, v_z) and (v_z, v_y) with the known subgraphs in $\mathcal{F}^{(0)}$, as illustrated in middle of Figure 2 (b). By continuing this procedure, the homomorphism expressivity $\mathcal{F}$ is constructed, as illustrated in bottom of Figure 2 (b).

[2]E.g. $\varphi(x) := \exists y(E(x, y) \wedge \exists x(E(x, y)))$. The variable x is reused. This is a technique often used in the context of logic to reduce the number of used symbols. In the construction of homogeneous expressivity we avoid this technique and write $\varphi(x)$ as $\varphi(x) := \exists y(E(x, y) \wedge \exists z(E(z, y)))$ so that all variables are explicitly expressed. As a result, there are 3 variables x, y, z in total.

6.3 Expressivity Comparison

It is also convenient to obtain the upper bounds with regard to WL tests thanks to the relation of logic and WL studied in Cai et al. (1992), as well as comparing the expressive power of different GNN models. For two classes of GNN models M_1, M_2, let Φ_1, Φ_2 be the equivalent logic sets of M_1 and M_2 respectively. Obviously, $\Phi_1 \subseteq \Phi_2$ indicates that M_2 is not weaker than M_1; Moreover, if $\Phi_1 \subset \Phi_2$, M_2 is strictly more expressive than M_1. Furthermore, to bound GNN models with k-WL, we have the following result:

Proposition 10. *Suppose the equivalent logic set of a class of GNN models is Φ. Then, the expressive power of the GNN models is bounded by k-WL,* iff *the number of variables of the logic formulas in Φ is at most k.*

Note that it is trival to check the number of variables in our setting: recall that in Proposition 6 the equivalent logic sets are defined by specifying the grammar of logic formulas. This implies that we can simply check the number of variables emerged in the grammar. For example, consider Subgraph GNN (weak). There are 2 free variables $\{x, y\}$ in $\varphi(x) := \exists^{\geq N} y\, (\varphi'(y) \wedge \psi(x,y)) \mid \neg\varphi'(x) \mid \varphi'(x) \wedge \varphi''(x) \mid \textbf{atp}(x)$ and 3 variables $\{x, y, z\}$ in $\psi(x,y) := \exists^{\geq N} z\, (\psi'(x,z) \wedge E(z,y)) \mid \neg\psi'(x) \mid \psi'(x) \wedge \psi''(x) \mid E(x,y)$. Therefore, the expressive power of Subgraph GNN (weak) is bounded by 3-WL. We summarize the section by introducing following results for popular GNN models.

Corollary 11. *The expressivity of GNN models satisfies: MPNNs $=$ 1-WL $<$ Subgraph GNNs (weak) $=$ NBFNet $<$ Subgraph GNNs (strong) $<$ Local 2-FGNN $<$ 2-FGNN $=$ 3-WL, 1-WL $<$ SEAL $<$ 4-WL.*

7 Limitation and Conclusion

Limitation. The results of this paper are applicable to GNNs that can be expressed by GACNNs. This includes most popular GNNs that learn graph-level, node-level or edge-level representations for both directed and undirected graphs with node and edge features. However, our framework are not applicable for GNNs which do not consist sole of aggregation and combination operations. For example, Graphormer-GD (Zhang et al., 2023) which injects distance information into node pairs and cannot be described by aggregation or combination layers. In the future, we plan to study GNNs that do not consist sole of aggregation and combination operations and investigate the necessary conditions for GNNs to possess equivalent logic sets.

Conclusion. In this paper we present a novel framework for systematically describe the logical expressivity of arbitrary GNN models built upon combination and aggregation operations. Utilizing the results, we analyze the logical expressivity of popular GNN models and provide new insight about many important topics in graph representation learning including expressivity comparison, structural awareness of GNNs, estimating WL expressivity, etc. Our framework serves as a toolbox to understand both existed and new GNN architectures: with new GNNs being designed, one can easily obtain the logical expressivity, study the substructures captured by them and bound these models with WL tests.

REFERENCES

Waïss Azizian and Marc Lelarge. Expressive power of invariant and equivariant graph neural networks. In *International Conference on Learning Representations*, 2020. URL https://api.semanticscholar.org/CorpusID:235358624.

John Baras and George Theodorakopoulos. *Path problems in networks*. Springer Nature, 2022.

Pablo Barceló, Egor V Kostylev, Mikael Monet, Jorge Pérez, Juan Reutter, and Juan-Pablo Silva. The logical expressiveness of graph neural networks. In *8th International Conference on Learning Representations (ICLR 2020)*, 2020.

Pablo Barcel'o, Floris Geerts, Juan L. Reutter, and Maksimilian Ryschkov. Graph neural networks with local graph parameters. In *Neural Information Processing Systems*, 2021. URL https://api.semanticscholar.org/CorpusID:235421961.

Leon Bergen, Timothy J. O'Donnell, and Dzmitry Bahdanau. Systematic generalization with edge transformers. In *NeurIPS*, 2021.

Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, G. Balamurugan, Michael M. Bronstein, and Haggai Maron. Equivariant subgraph aggregation networks. *ArXiv*, abs/2110.02910, 2021. URL https://api.semanticscholar.org/CorpusID:238407774.

Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yu Guang Wang, Pietro Lio', Guido Montúfar, and Michael M. Bronstein. Weisfeiler and lehman go cellular: Cw networks. In *Neural Information Processing Systems*, 2021a. URL https://api.semanticscholar.org/CorpusID:235606230.

Cristian Bodnar, Fabrizio Frasca, Yu Guang Wang, Nina Otter, Guido Montúfar, Pietro Lio', and Michael M. Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks. *ArXiv*, abs/2103.03212, 2021b. URL https://api.semanticscholar.org/CorpusID:232110693.

Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M. Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45:657–668, 2020. URL https://api.semanticscholar.org/CorpusID:219708613.

Jin-Yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12:389–410, 1989.

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

Benjamin Paul Chamberlain, Sergey Shirobokov, Emanuele Rossi, Fabrizio Frasca, Thomas Markovich, Nils Hammerla, Michael M Bronstein, and Max Hansmire. Graph neural networks for link prediction with subgraph sketching. In *ICLR*, 2023.

Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna. Can graph neural networks count substructures? *ArXiv*, abs/2002.04025, 2020. URL https://api.semanticscholar.org/CorpusID:211069434.

Leonardo Cotta, Christopher Morris, and Bruno Ribeiro. Reconstruction for powerful graph representations. *ArXiv*, abs/2110.00577, 2021. URL https://api.semanticscholar.org/CorpusID:238253248.

Holger Dell, Martin Grohe, and Gaurav Rattan. Lov\'asz meets weisfeiler and leman. *arXiv preprint arXiv:1802.08876*, 2018.

Fabrizio Frasca, Beatrice Bevilacqua, Michael M. Bronstein, and Haggai Maron. Understanding and extending subgraph gnns by rethinking their symmetries. *ArXiv*, abs/2206.11140, 2022. URL https://api.semanticscholar.org/CorpusID:249926758.

Floris Geerts and Juan L. Reutter. Expressiveness and approximation properties of graph neural networks. *ArXiv*, abs/2204.04661, 2022. URL `https://api.semanticscholar.org/CorpusID:248084856`.

Max Horn, Edward De Brouwer, Michael Moor, Yves Moreau, Bastian Alexander Rieck, and Karsten M. Borgwardt. Topological graph neural networks. *ArXiv*, abs/2102.07835, 2021. URL `https://api.semanticscholar.org/CorpusID:231934149`.

Xingyue Huang, Miguel Romero, Ismail Ceylan, and Pablo Barceló. A theory of link prediction via relational weisfeiler-leman on knowledge graphs. *Advances in Neural Information Processing Systems*, 36, 2024.

Nicolas Keriven and Gabriel Peyré. Universal invariant and equivariant graph neural networks. In *Neural Information Processing Systems*, 2019. URL `https://api.semanticscholar.org/CorpusID:152282292`.

Thomas Kipf and Max Welling. Variational graph auto-encoders. *ArXiv*, abs/1611.07308, 2016a.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016b.

S. Liu, Bernardo Cuenca Grau, Ian Horrocks, and Egor V. Kostylev. Indigo: Gnn-based inductive knowledge graph completion using pair-wise encoding. In *Neural Information Processing Systems*, 2021. URL `https://api.semanticscholar.org/CorpusID:245119728`.

Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. *ArXiv*, abs/1812.09902, 2018. URL `https://api.semanticscholar.org/CorpusID:56895597`.

Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *ArXiv*, abs/1905.11136, 2019a. URL `https://api.semanticscholar.org/CorpusID:166228757`.

Haggai Maron, Ethan Fetaya, Nimrod Segol, and Yaron Lipman. On the universality of invariant networks. *ArXiv*, abs/1901.09342, 2019b. URL `https://api.semanticscholar.org/CorpusID:59316743`.

Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI Conference on Artificial Intelligence*, 2018. URL `https://api.semanticscholar.org/CorpusID:52919090`.

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

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

Pál András Papp and Roger Wattenhofer. A theoretical comparison of graph neural network extensions. *ArXiv*, abs/2201.12884, 2022. URL `https://api.semanticscholar.org/CorpusID:246430444`.

Omri Puny, Derek Lim, Bobak Kiani, Haggai Maron, and Yaron Lipman. Equivariant polynomials for graph neural networks. In *International Conference on Machine Learning*, pp. 28191–28222. PMLR, 2023.

Chen Qian, Gaurav Rattan, Floris Geerts, Christopher Morris, and Mathias Niepert. Ordered subgraph aggregation networks. *ArXiv*, abs/2206.11168, 2022. URL `https://api.semanticscholar.org/CorpusID:249926461`.

Komal K. Teru, E. Denis, and William L. Hamilton. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, 2019. URL https://api.semanticscholar.org/CorpusID:211082667.

Erik H. Thiede, Wenda Zhou, and Risi Kondor. Autobahn: Automorphism-based graph neural nets. In *Neural Information Processing Systems*, 2021. URL https://api.semanticscholar.org/CorpusID:232092767.

Kristina Toutanova, Xi Victoria Lin, Wen tau Yih, Hoifung Poon, and Chris Quirk. Compositional learning of embeddings for relation paths in knowledge base and text. In *ACL*, 2016.

Boris Weisfeiler and Andrei Leman. The reduction of a graph to canonical form and the algebra which appears therein. *nti, Series*, 2(9):12–16, 1968.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Tuo Xu and Lei Zou. Rethinking and extending the probabilistic inference capacity of gnns. In *The Twelfth International Conference on Learning Representations*, 2024.

Jiaxuan You, Jonathan M. Gomes-Selman, Rex Ying, and Jure Leskovec. Identity-aware graph neural networks. In *AAAI Conference on Artificial Intelligence*, 2021. URL https://api.semanticscholar.org/CorpusID:231698873.

Bohang Zhang, Shengjie Luo, Liwei Wang, and Di He. Rethinking the expressive power of gnns via graph biconnectivity. *ArXiv*, abs/2301.09505, 2023. URL https://api.semanticscholar.org/CorpusID:256105774.

Bohang Zhang, Jingchu Gai, Yiheng Du, Qiwei Ye, Di He, and Liwei Wang. Beyond weisfeiler-lehman: A quantitative framework for gnn expressiveness. In *International Conference on Machine Learning*, 2024.

Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Neural Information Processing Systems*, 2018. URL https://api.semanticscholar.org/CorpusID:3573161.

Muhan Zhang and Yixin Chen. Inductive matrix completion based on graph neural networks. In *ICLR*, 2020.

Muhan Zhang and Pan Li. Nested graph neural networks. *ArXiv*, abs/2110.13197, 2021. URL https://api.semanticscholar.org/CorpusID:239885856.

Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin. Labeling trick: A theory of using graph neural networks for multi-node representation learning. In *Neural Information Processing Systems*, 2020. URL https://api.semanticscholar.org/CorpusID:239998439.

Lingxiao Zhao, Wei Jin, Leman Akoglu, and Neil Shah. From stars to subgraphs: Uplifting any gnn with local structure awareness. *ArXiv*, abs/2110.03753, 2021. URL https://api.semanticscholar.org/CorpusID:238531375.

Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. Neural bellman-ford networks: A general graph neural network framework for link prediction. In *Neural Information Processing Systems*, 2021. URL https://api.semanticscholar.org/CorpusID:235422273.

A Related Works

In this section we discuss related works that studies the expressive power of GNNs and several GNN models investigated in this paper.

Expressivity of GNNs. Studying the expressive power of GNNs has been a hot topic in graph machine learning community. Xu et al. (2018) investigate the expressive power of GNNs by relating MPNNs with 1-WL tests, making it possible to utilize many know results about WL tests for the analysis of GNNs. Barceló et al. (2020) study the logical expressiveness of MPNNs which is close to our work. Compared with them, we successfully design a method to describe the logical expressivity for arbitrary aggregation-combination networks and analyzed several important implications brought by our work, including homomorphism expressivity, comparison of expressive power, etc. Zhang et al. (2024) investigate several popular GNN models and study their expressive power in the perspective of homomorphisms. Compared with their work, we propose a general method to determine the homomorphism expressivity for arbitrary aggregation-combination networks while solving a conjecture in Zhang et al. (2024) in the meantime.

Higher order GNNs. Since the works of Xu et al. (2018); Morris et al. (2018) that relate GNNs with the 1-WL tests, it is straightforward to extend GNNs by imitating higher-order WL tests. Precisely, k-order WL tests assign colors for k-tuples of nodes and perform color aggregation between different tuples. Similarly, instead of learning representations for nodes, many works choose to apply the message passing paradigm in higher-order WL tests to GNNs and directly learn representations for node tuples (Morris et al., 2018; Maron et al., 2019a; 2018; 2019b; Keriven & Peyré, 2019; Azizian & Lelarge, 2020; Geerts & Reutter, 2022).

Subgraph GNNs. Since the higher order GNNs are often too expensive for larger graphs, many works try to find cheaper ways to design more expressive GNNs. A variety of works feed subgraphs to MPNNs. At each layer, a set of subgraphs is generated according to some predefined permutation-invariant policies, including node deletion (Cotta et al., 2021), edge deletion Bevilacqua et al. (2021), node marking (Papp & Wattenhofer, 2022), ego-networks (Zhao et al., 2021; Zhang & Li, 2021; You et al., 2021). We will focus on the unified ESAN framework proposed by Bevilacqua et al. (2021). Qian et al. (2022); Frasca et al. (2022) studied the expressive power of different branches of subgraph GNNs.

Substructure counting GNNs. There is another way to design GNNs that surpass 1-WL by constructing structural features for GNNs. Chen et al. (2020) showed that regular MPNNs cannot capture simple patterns such as cycles, cliques and paths. Bouritsas et al. (2020); Barcel'o et al. (2021) proposed to apply substructure counting as pre-processing, and add substructure information into node features. Bodnar et al. (2021b;a); Thiede et al. (2021); Horn et al. (2021) further designed novel WL variants and proposed fully-neural approaches that captures complex substructures.

GNNs for link prediction. Standard GNNs learn representations for each node. Early methods such as GAE Kipf & Welling (2016a) use GNN as an encoder and decode link representations as a function over node representation pairs. These methods are problematic in capturing complex graph structures, and might lead to poor performance. Later on, labeling trick was introduced by SEAL Zhang & Chen (2018) and adopted by GraIL Teru et al. (2019), IGMC Zhang & Chen (2020), INDIGO Liu et al. (2021), etc. These methods encode source and target nodes to mark them differently from the rest of the graph, and are proved to be more powerful than GAE. ID-GNN You et al. (2021) and NBFNet Zhu et al. (2021) both augments GNNs with the identity of the source nodes. Besides, All-path Toutanova et al. (2016) encodes relations as linear projections and proposes to efficiently aggregate all paths with dynamic programming. However, All-Path is restricted to bilinear models, has limited link prediction capability and is also not inductive. EdgeTransformer Bergen et al. (2021) utilizes attention mechanism to learn representations for nodes and links. While it also follows the 2-FWL message passing procedure, it operates directly on fully-connected graphs and have no proposals for simplifications as we do, thus it is not scalable to larger graphs. ELPH and BUDDY (Chamberlain et al., 2023) incorporate neighbor counting into node features to enhance the link prediction performance of MPNNs.

B WEISFEILER-LEHMAN TESTS

In this section we introduce the Weisfeiler-Lehman (WL) tests and their variants.

B.1 1-WL (COLOR REFINEMENT)

The classic 1-WL test (Weisfeiler & Leman, 1968) maintains a color for each node which is refined by aggregating the colors of their neighbors. It can be easily applied on node-featured graphs (Xu et al., 2018) as in Algorithm 1.

Algorithm 1: The 1-WL test (color refinement)

Input : $G = (\boldsymbol{A}, \boldsymbol{X})$
$l \leftarrow 0$;
$c_v^0 \leftarrow \mathrm{hash}(\boldsymbol{x}_v)$ for all $v \in \mathcal{V}_G$;
while *not converge* **do**
    $c_v^{l+1} \leftarrow \mathrm{hash}(c_v^l, \{\{c_u^l \mid u \in \mathcal{N}(v)\}\})$;
    $l \leftarrow l + 1$;
end
return $\{\{c_v^l \mid v \in \mathcal{V}_G\}\}$;

The iteration converges when the partitions of nodes no longer changes. The 1-WL test decides two graphs are non-isomorphic if the multisets of colors of the two graphs are different. The WL algorithm successfully distinguishes most pairs of graphs, apart from some special examples such as regular graphs. Similarly, given a subset of nodes C, 1-WL define its color as $\{\{c_v^l \mid v \in C\}\}$, and 1-WL distinguishes two set of nodes if the colors of them are differernt.

B.2 k-WL

The k-WL tests extend 1-WL to coloring k-tuples of nodes as in Algorithm 2, where we use $\boldsymbol{v}$ to denote a tuple of nodes, $G[\boldsymbol{v}]$ for ordered subgraphs. The neighbors $\mathcal{N}^k(\boldsymbol{v})$ are defined as follows: assume $\boldsymbol{v} = (v_1, ..., v_k)$, then $\mathcal{N}^k(\boldsymbol{v}) = (\mathcal{N}_1^k(\boldsymbol{v}), \mathcal{N}_1^k(\boldsymbol{v}), ..., \mathcal{N}_k^k(\boldsymbol{v}))$, where

$$\mathcal{N}_i^k(\boldsymbol{v}) = \{\{(v_1, ..., v_{i-1}, u, v_{i+1}, ..., v_k) \mid u \in \mathcal{V}\}\}.$$

Algorithm 2: The k-WL tests

Input : $G = (\boldsymbol{A}, \boldsymbol{X})$
$l \leftarrow 0$;
$c_{\boldsymbol{v}}^0 \leftarrow \mathrm{hash}(G[\boldsymbol{v}])$ for all $\boldsymbol{v} \in \mathcal{V}_G^k$;
while *not converge* **do**
    $c_{\boldsymbol{v}}^{l+1} \leftarrow \mathrm{hash}(c_{\boldsymbol{v}}^l, \{\{c_{\boldsymbol{u}}^l \mid \boldsymbol{u} \in \mathcal{N}^k(\boldsymbol{v})\}\})$;
    $l \leftarrow l + 1$;
end
return $\{\{c_{\boldsymbol{v}}^l \mid v \in \mathcal{V}_G \textit{ for all } v \in \boldsymbol{v}\}\}$;

B.3 k-FWL

The k-FWL (Cai et al., 1989) test is equally expressive with the $(k+1)$-WL test. It has the same initialization with $(k+1)$-WL. The neighbors $\mathcal{N}^k(\boldsymbol{v})$ are defined as follows: assume $\boldsymbol{v} = (v_1, ..., v_k)$, then $\mathcal{N}^k(\boldsymbol{v}) = \{\{\mathcal{N}_u^k(\boldsymbol{v}) \mid u \in \mathcal{V}\}\}$, where

$$\mathcal{N}_u^k(\boldsymbol{v}) = ((u, v_2, ..., v_k), (v_1, u, ..., v_k), ..., (v_1, ..., u, v_k)).$$

Algorithm 3: The k-FWL tests

Input : $G = (\boldsymbol{A}, \boldsymbol{X})$

1 $l \leftarrow 0$;
2 $c_{\boldsymbol{v}}^0 \leftarrow \mathrm{hash}(G[\boldsymbol{v}])$ for all $\boldsymbol{v} \in \mathcal{V}_G^k$;
3 **while** *not converge* **do**
4 $\quad c_{\boldsymbol{v}}^{l+1} \leftarrow \mathrm{hash}(c_{\boldsymbol{v}}^l, \{\{c_{\boldsymbol{u}}^l \mid \boldsymbol{u} \in \mathcal{N}^k(\boldsymbol{v})\}\})$;
5 $\quad l \leftarrow l + 1$;
6 **end**
7 **return** $\{\{c_{\boldsymbol{v}}^l \mid v \in \mathcal{V}_G \textit{ for all } v \in \boldsymbol{v}\}\}$;

B.4 Colors of k-WL / k-FWL

From the previous discussions k-WL and k-FWL both assign colors for k-tuples of nodes. The color of the graph G is defined by

$$c_G = \mathrm{Hash}(\{\{c_{\boldsymbol{v}} \mid \boldsymbol{v} \in \mathcal{V}^k\}\}).$$

Similarly, given any subset of nodes $\mathcal{S} \subseteq \mathcal{V}$, we also define its color as

$$c_{\mathcal{S}} = \mathrm{Hash}(\{\{c_{\boldsymbol{v}} \mid \boldsymbol{v} \in \mathcal{S}^k\}\}).$$

C About GNN models

In this section we briefly introduce several popular GNN models.

MPNN.

$$\chi^{(l+1)}(x) = \mathrm{COM}\left(\chi^{(l)}(x), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(y) \mid y \in \mathcal{N}(x)\right\}\right\}\right)\right).$$

Subgraph GNN (weak).

$$\chi^{(l+1)}(x) = \mathrm{AGG}\left(\left\{\left\{\chi^{(l+1)}(x, y) \mid y \in \mathcal{V}\right\}\right\}\right),$$
$$\chi^{(l+1)}(x, y) = \mathrm{COM}\left(\chi^{(l)}(x, y), \mathrm{AGG}\left(\chi^{(l)}(x, z) \mid z \in \mathcal{N}(y)\right)\right).$$

Subgraph GNN (strong).

$$\chi^{(l+1)}(x, y) = \mathrm{COM}\left(\chi^{(l)}(x, y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x, z) \mid z \in \mathcal{N}(y)\right\}\right\}\right), \chi^{(l)}(y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(z) \mid z \in \mathcal{N}(y)\right\}\right\}\right)\right),$$
$$\chi^{(l+1)}(x) = \mathrm{AGG}\left(\chi^{(l+1)}(y, x) \mid y \in \mathcal{V}\right).$$

NBFNet.

$$\chi^{(l+1)}(x, y) = \mathrm{COM}\left(\chi^{(l)}(x, y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x, z) \mid z \in \mathcal{N}(y)\right\}\right\}\right)\right).$$

Local 2-GNN.

$$\chi^{(l+1)}(x, y) = \mathrm{COM}\left(\chi^{(l)}(x, y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(z, y) \mid z \in \mathcal{N}(x)\right\}\right\}\right), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x, z) \mid z \in \mathcal{N}(y)\right\}\right\}\right)\right).$$

2-FGNN.

$$\chi^{(l+1)}(x, y) = \mathrm{COM}\left(\chi^{(l)}(x, y), \mathrm{AGG}\left(\left\{\left\{\mathrm{COM}\left(\chi^{(l)}(x, z), \chi^{(l)}(z, y)\right) \mid z \in \mathcal{V}\right\}\right\}\right)\right).$$

SEAL (MPNN).

$$\chi^{(l+1)}(x, z, y) = \mathrm{COM}\left(\chi^{(l)}(x, z, y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x, w, y) \mid w \in \mathcal{N}(z)\right\}\right\}\right)\right),$$
$$\chi^{(l+1)}(x, y) = \mathrm{AGG}\left(\left\{\left\{\chi^{(l+1)}(x, z, y) \mid z \in \mathcal{N}\right\}\right\}\right).$$

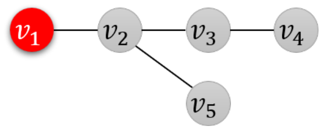


Figure 3: An example graph which has two groundings for $\varphi(x) := \mathrm{Red}(x) \wedge \exists y(E(y,x) \wedge \exists z(E(z,y)))$.

2-GNN

$$\chi^{(l+1)}(x,y) = \mathrm{COM}\left(\chi^{(l)}(x,y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(z,y) \mid z \in \mathcal{V}\right\}\right\}\right), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x,z) \mid z \in \mathcal{V}\right\}\right\}\right)\right).$$

D About variables in logic formulas

Consider the formula

$$\varphi(x) := \mathrm{Red}(x) \wedge \exists y(E(y,x) \wedge \exists z(E(z,y))).$$

The formula has a *free variable* x which is not bounded by the quantifier $\exists$ and two *quantified variables* y, z which are bounded by $\exists$. Therefore, the formula has 3 variables in total. Given a graph G, a *grounding* of $\varphi(x)$ in G is a mapping η from the variables in $\varphi(x)$ to the nodes in G. For example consider the graph in Figure 3. There are two groundings η_1, η_2 from $\varphi(x)$ to it, with $\eta_1(x) = v_1, \eta_1(y) = v_2, \eta_1(z) = v_3$ and $\eta_2(x) = v_1, \eta_2(y) = v_2, \eta_2(z) = v_5$.

To reduce the number of symbols used in logic formula, there is a trick which is to *reuse* the variable x and replace every occurrence of z in φ with x, leading to:

$$\varphi'(x) := \mathrm{Red}(x) \wedge \exists y(E(y,x) \wedge \exists x(E(x,y))).$$

To ground $\varphi'(x)$ on G, one still needs to substitute the variables in $\varphi'(x)$ with the nodes in G. This indicates that in Figure 3, we need to substitute the outer variable x in $\mathrm{Red}(x)$ with v_1 and the inner variable x in $\exists x(E(x,y))$ with v_3 or v_5. Therefore, when the variables are reused, the grounding is no longer a well-defined mapping from variables to nodes, and the essentially different variables x, z in $\varphi(x)$ are expressed by the same symbol x in $\varphi'(x)$. To avoid such clunky situations, we avoid the reuse of variables.

The properties of F constructed by φ. Recall that to construct the homomorphism expressivity, we construct a graph F for φ which is defined below:

1. There exists a bijective mapping τ from the variables in φ to the nodes in F.
2. For any variables x, y in φ, $E(x,y)$ is a term in φ *iff* $E(\tau(x), \tau(y))$ is an edge of F.

We define the concept of injective grounding:

Definition 12. An injective grounding from a logic formula φ to a graph G is a grounding from φ to G that maps different variables in φ to different nodes in G (without the reuse of variables).

It is now obvious that F is the minimum graph that contains an injective grounding from φ.

E Proof

E.1 Proof of Theorem 4

Theorem 4. *Given $\{\chi_1, \chi_2, ..., \chi_m\}$ defined above, there exists $\{\Phi_1, ..., \Phi_m\}$ where Φ_i is the equivalent logic set of χ_i for $i \in [k]$. Moreover, each $\varphi_i \in \Phi_i$ is given by:*

- $$\chi_i(\mathbf{u}) = \mathrm{AGG}\left(\{\{\chi_j(\mathbf{x}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right)$$
  $$\iff \varphi_i(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_j(\mathbf{v}) \wedge \mathbf{1}_{\mathbf{v}\in\mathcal{N}_i(\mathbf{u})}\right) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}),$$

- $$\chi_i(\mathbf{u}) = \mathrm{COM}\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right)$$
  $$\iff \varphi_i(\mathbf{u}) := \varphi_j(\mathbf{u}) \mid \varphi_k(\mathbf{u}) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}),$$
- $$\chi_i(\mathbf{u}) = \textsf{atp}(\mathbf{u}) \iff \varphi_i(\mathbf{u}) := \textsf{atp}(\mathbf{u}) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}),$$

where $\varphi_i', \varphi_i'' \in \Phi_i, \varphi_j \in \Phi_j, \varphi_k \in \Phi_k$.

Proof. Recall that the definition of equivalent logic set Φ of GNN models M is defined as:

1. The arity of $\varphi \in \Phi$ matches the output of M: suppose GNNs compute k-order representation, then $\varphi \in \Phi$ should be k-ary;
2. For all $\varphi \in \Phi$, there exists a GNN model M such that for arbitrary graphs G and $\mathbf{u} \in \mathcal{V}_G^k$, $\varphi(\mathbf{u}) = \textsf{true}$ *iff* $\chi(\mathbf{u}) = \textsf{true}$;
3. Given any graphs G, H and $\mathbf{u} \in \mathcal{V}_G^k, \mathbf{v} \in \mathcal{V}_H^k$, the GNN models cannot distinguish $\mathbf{u}, \mathbf{v}$ *iff* all logic formulas $\varphi \in \Phi$ classify $\mathbf{u}, \mathbf{v}$ the same.

Constraint 1 is naturally satisfied. Therefore, we need to prove Constraint 2 and 3. We prove by induction. At beginning we have $\chi_0(\mathbf{u}) = \textsf{atp}(\mathbf{u})$ and $\varphi_0(\mathbf{u}) = \textsf{atp}(\mathbf{u})$ thus the constraints are satisfied. Suppose we are to compute $\chi_i(\mathbf{u})$. By proof by induction the equivalent set Φ_j of χ_j for all $j < i$ are known. There are 3 situations:

1. $\chi_i(\mathbf{u}) = \textsf{atp}(\mathbf{u})$. Then $\varphi_i(\mathbf{u}) := \textsf{atp}(\mathbf{u}) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u})$ naturally satisfies.

2. $\chi_i(\mathbf{u}) = \mathrm{COM}\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right)$. Let Φ_j, Φ_k be the corresponding equivalent logic set of χ_j, χ_k respectively and satisfy the three constraints. Then the equivalent logic set Φ_i is then defined as

$$\varphi_i(\mathbf{u}) := \varphi_j(\mathbf{u}) \mid \varphi_k(\mathbf{u}) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}),$$

where $\varphi_i, \varphi' \in \Phi, \varphi_j \in \Phi_j, \varphi_k \in \Phi_k$. Let $\{\varphi_i^1, ..., \varphi_i^L\}$ be the series of sub-formulas of φ_i such that if φ_i^p is a sub-formula of φ_q then $p < q$. Again, we prove by induction on the sub-formula series of Φ_i.
1) At beginning for φ_i^1 we only have $\varphi_i^1(\mathbf{u}) := \varphi_j(\mathbf{u})$ or $\varphi_i(\mathbf{u}) := \varphi_k(\mathbf{u})$. In this case we let $\chi_i(\mathbf{u}) := \chi_j(\mathbf{u})$ or $\chi_i(\mathbf{u}) := \chi_k(\mathbf{u})$. Since Φ_j, Φ_k are the equivalent sets of χ_j, χ_k respectively, φ_i^1 is captured by χ_i.
2) Suppose at iteration l, all $\varphi_i^p(\mathbf{u})$ for $p < l$ can be captured by some $\chi_i^p(\mathbf{u}) = \mathrm{COM}^p(\chi_j(\mathbf{u}), \chi_k(\mathbf{u}))$. We show that by designing specific COM^l function there is also $\chi_i^l(\mathbf{u}) = \mathrm{COM}^l(\chi_j(\mathbf{u}), \chi_k(\mathbf{u}))$ that captures $\varphi_i^l(\mathbf{u})$. It is straightforward to prove: If $\varphi_i^l(\mathbf{u}) = \neg\varphi_i^q(\mathbf{u})$, then

$$\mathrm{COM}^l\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right) = 1 - \mathrm{COM}^p\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right).$$

If $\varphi_i^l(\mathbf{u}) = \varphi_i^p(\mathbf{u}) \wedge \varphi_i^q(\mathbf{u})$ then

$$\mathrm{COM}^l\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right) = \mathrm{COM}^p\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right) * \mathrm{COM}^q\left(\chi_j(\mathbf{u}), \chi_k(\mathbf{u})\right).$$

Otherwise we have $\varphi_i^1(\mathbf{u}) = \varphi_j(\mathbf{u})$ or $\varphi_i(\mathbf{u}) = \varphi_k(\mathbf{u})$. In this case φ_i^l can also be captured by χ_i as proven in 1).

Thus, we have shown that $\varphi_i(\mathbf{u}) \in \Phi_i$ is always captured by $\chi_i(\mathbf{u})$. Next, we show that Φ_i is maximal, i.e. all logic formulas that can be captured by χ_i belong to Φ_i. Since Φ_j, Φ_k are the equivalent sets of φ_j, φ_k respectively, we have:

$$\chi_j(\mathbf{u}) \neq \chi_j(\mathbf{v}) \Rightarrow \text{Exists } \varphi_j \in \Phi_j \text{ satisfying } \varphi_j(\mathbf{u}) \neq \varphi_j(\mathbf{v}),$$
$$\chi_k(\mathbf{u}) \neq \chi_k(\mathbf{v}) \Rightarrow \text{Exists } \varphi_k \in \Phi_k \text{ satisfying } \varphi_k(\mathbf{u}) \neq \varphi_k(\mathbf{v}).$$

Since for any $\mathbf{u}$ and $\mathbf{v}$, $\chi_i(\mathbf{u}) \neq \chi_i(\mathbf{v})$ indicates that either $\chi_j(\mathbf{u}) \neq \chi_j(\mathbf{v})$ or $\chi_k(\mathbf{u}) \neq \chi_k(\mathbf{v})$ (or both), by setting $\varphi_i(\mathbf{u}) := \varphi_j(\mathbf{u})$ or $\varphi_i(\mathbf{u}) := \varphi_k(\mathbf{u})$ respectively, we also have $\chi_i(\mathbf{u}) \neq \chi_j(\mathbf{u})$. Therefore, we have:

$$\chi_i(\mathbf{u}) \neq \chi_i(\mathbf{v}) \Rightarrow \text{Exists } \varphi_i \in \Phi_i \text{ satisfying } \varphi_i(\mathbf{u}) \neq \varphi_i(\mathbf{v}).$$

Thus the proof is completed.

3. $\chi_i(\mathbf{u}) = \mathrm{AGG}\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right)$. The proof is similar as above. Let Φ_j be the equivalent logic set of χ_j, which is defined as

$$\varphi_i(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_j(\mathbf{v}) \wedge \mathbf{1}_{\mathbf{v}\in\mathcal{N}_i(\mathbf{u})}\right) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}).$$

Let $\{\varphi_i^1, ..., \varphi_i^L\}$ be the series of sub-formulas of φ_i such that if φ_i^p is a sub-formula of φ_q then $p < q$. Again, we prove by induction on the sub-formula series of Φ_i.
1) At beginning for φ_i^1 we only have $\varphi_i^1(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_j(\mathbf{v}) \wedge \mathbf{1}_{\mathbf{v}\in\mathcal{N}_i(\mathbf{u})}\right)$. In this case we let

$$\chi_i(\mathbf{u}) = \mathrm{AGG}\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right) = \mathbf{1}\left\{\text{There are no less than } N \ \mathbf{v} \text{ such that } \varphi_j(\mathbf{v}) \text{ is } \textsf{true}\right\}.$$

Note that since all $\varphi_j \in \Phi_j$ can be captured by χ_j, the above AGG function can be realized.
2) Suppose at iteration l, all $\varphi_i^p(\mathbf{u})$ for $p < l$ can be captured by some $\chi_i^p(\mathbf{u}) = \mathrm{AGG}^p\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right)$. We show that by designing specific AGG^l function there is also $\chi_i^l(\mathbf{u}) = \mathrm{AGG}^l\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right)$ that captures $\varphi^l(\mathbf{u})$. It is also straightforward to prove: If $\varphi_i^l(\mathbf{u}) := \neg\varphi_i^q(\mathbf{u})$, then

$$\mathrm{AGG}^l\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right) = 1 - \mathrm{AGG}^q\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right).$$

If $\varphi_i^l(\mathbf{u}) := \varphi_i^p(\mathbf{u}) \wedge \varphi_i^q(\mathbf{u})$ then

$$\mathrm{AGG}^l\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right) = \mathrm{AGG}^p\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right) * \mathrm{AGG}^q\left(\{\{\chi_j(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\}\}\right).$$

Otherwise $\varphi_i^l(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_i^q(\mathbf{v}) \mid \mathbf{v} \in \mathcal{N}_i^l(\mathbf{u})\right)$ which is proved in 1).

Thus, we have shown that $\varphi_i(\mathbf{u}) \in \Phi_i$ is always captured by $\chi_i(\mathbf{u})$. Next, we show that Φ_i is maximal, i.e. all logic formulas that can be captured by χ_i belong to Φ_i. Since Φ_j is the equivalent set of φ_j, we have:

$$\chi_j(\mathbf{u}) \neq \chi_j(\mathbf{v}) \Rightarrow \text{Exists } \varphi_j \in \Phi_j \text{ satisfying } \varphi_j(\mathbf{u}) \neq \varphi_j(\mathbf{v}).$$

For any $\mathbf{u}$ and $\mathbf{v}$, $\chi_i(\mathbf{u}) \neq \chi_i(\mathbf{v})$ indicates that the multisets are different $\{\{\chi_j(\mathbf{w}) \mid \mathbf{w} \in \mathcal{N}_i(\mathbf{u})\}\} \neq \{\{\chi_j(\mathbf{w}) \mid \mathbf{w} \in \mathcal{N}_i(\mathbf{v})\}\}$. Then, there must exists a color X such that

$$\left|\{\mathbf{w} \mid \mathbf{w} \in \mathcal{N}_i(\mathbf{u}), \chi_j(\mathbf{w}) = X\}\right| \neq \left|\{\mathbf{w} \mid \mathbf{w} \in \mathcal{N}_i(\mathbf{v}), \chi_j(\mathbf{w}) = X\}\right|.$$

without loss of generality we assume $\left|\{\mathbf{w} \mid \mathbf{w} \in \mathcal{N}_i(\mathbf{u}), \chi_j(\mathbf{w}) = X\}\right| = N > \left|\{\mathbf{w} \mid \mathbf{w} \in \mathcal{N}_i(\mathbf{v}), \chi_j(\mathbf{w}) = X\}\right|$. Since χ_j is captured by Φ_j, there must exist $\varphi_j \in \Phi_j$ that expresses the color X: $\varphi_j(\mathbf{w}) = \textsf{true}$ *iff* $\chi_j(\mathbf{w}) = X$. By letting

$$\varphi_i(\mathbf{x}) := \exists^{\geq N}\mathbf{w}\left(\varphi_j(\mathbf{w}) \wedge \mathbf{1}_{\mathbf{w}\in\mathcal{N}_i(\mathbf{u})}\right),$$

it is evident that $\varphi_i(\mathbf{u}) = \textsf{true}$ while $\varphi_i(\mathbf{v}) = \textsf{false}$. Therefore, the proof completes. □

E.2 Proof of Theorem 9

Theorem 9. *Given a class of GACNN models and suppose Φ be the equivalent logic set. Let $\mathcal{F}$ be the homomorphism expressivity constructed by Φ as discussed above. For all pairs of graphs G, H. the following statements are equivalent:*

1. $\textsf{Hom}(F, G) = \textsf{Hom}(F, H)$ *for all* $F \in \mathcal{F}$.
2. *All GACNNs do not distinguish* G *and* H.

Proof. From Proposition 6 it is evident that statement 2 is equivalent to: All $\varphi \in \Phi$ do not distinguish G and H. We therefore instead prove:

$$\text{All } \varphi \in \Phi \text{ do not distinguish } G \text{ and } H \iff \textsf{Hom}(F, G) = \textsf{Hom}(F, H) \text{ for all } F \in \mathcal{F}.$$

We first prove the direction left to right. Given each $\varphi \in \Phi$ that does not contain negation $\neg$ or $\exists^{\geq N}$ where $N \geq 2$, recall that the corresponding F is constructed by:

1. Start from an empty graph F;
2. Construct the nodes of F: Add a node v_x for each variable x emerged in φ (we avoid the reuse of variables);

3. Construct the structure of F: Add an edge (v_x, v_y) for each edge term $E(x, y)$ in φ. Add F to $\mathcal{F}$.

Before we start, we first introduce some useful quantifiers $\exists^{=N}, \exists^{\leq N}$ which express "there exists exactly N" and "there exists no more than N" respectively. Note that the two quantifiers can be directly deduced by $\exists^{\geq N}$: $\exists^{\leq N} := \neg\exists^{\geq N+1}$ and $\exists^{=N} := \exists^{\geq N} \wedge \exists^{\leq N}$.

Note that all φ satisfying the constraint (i.e. without negation or $\exists^{\geq N}$ for $N > 1$) can be flattened into the form of:

$$\varphi := \exists \mathbf{x}_1 \exists \mathbf{x}_2 ... \exists \mathbf{x}_K E(x_{i_1}, x_{j_1}) \wedge ... \wedge E(x_{i_M}, x_{j_M}), \tag{5}$$

where $i_p, j_p \in [K]$ for all $p \in [M]$. We now prove that there is a logic formula $\psi \in \Phi$ that captures $\textsf{Hom}(F, G) = 1$, i.e. given arbitrary graph G, ψ is **true** *iff* $\textsf{Hom}(F, G) = 1$. Suppose F is constructed by φ as:

$$\varphi := \exists \mathbf{x}_1 \exists \mathbf{x}_2 ... \exists \mathbf{x}_K E(x_{i_1}, x_{j_1}) \wedge ... \wedge E(x_{i_M}, x_{j_M}).$$

Then by letting

$$\psi := \exists^{=1} \mathbf{x}_1 \exists^{=1} \mathbf{x}_2 ... \exists^{=1} \mathbf{x}_K E(x_{i_1}, x_{j_1}) \wedge ... \wedge E(x_{i_M}, x_{j_M}).$$

We now show that for arbitrary graph G, ψ is **true** *iff* $\textsf{Hom}(F, G) = 1$. By the construction of F each variable $\mathbf{x}_1, ..., \mathbf{x}_K$ is corresponded to a distinct node tuple in F and each term $E(x_{i_1}, x_{j_1}), ..., E(x_{i_m}, x_{j_m})$ is corresponded to an distinct edge in F. If ψ is **true** on G, then there exists a grounding $\mathbf{x}_1 \to \mathbf{u}_1, ..., \mathbf{x}_K \to \mathbf{u}_K$ such that all $E(x_{i_1}, x_{j_1}), ..., E(x_{i_m}, x_{j_m})$ are **true**. Then the mapping $\pi : F \to G$ that $\pi(v_{x_l}) = u_l$ for $l \in [K]$ where v_{x_l} is the node in F corresponding to the variable x_l in ψ. obviously π is a homomorphism from F to G, therefore $\textsf{Hom}(F, G) \geq 1$. Further more, suppose $\textsf{Hom}(F, G) > 1$, then there exists another $\pi' \neq \pi$ that is also a homomorphism from F to G, which indicates that the grounding for $E(x_{i_1}, x_{j_1}), ..., E(x_{i_M}, x_{j_M})$ to be **true** is not unique. In this case, ψ is not **true** because there exists not only one $\mathbf{x}_1, ..., \mathbf{x}_K$ such that $E(x_{i_1}, x_{j_1}), ..., E(x_{i_M}, x_{j_M})$ is **true**, violating the quantifiers $\exists^{=1}$ in ψ.

We next prove that there is also a logic formula $\psi \in \Phi$ that captures $\textsf{Hom}(F, \cdot) = N$ for arbitrary N: that is, for any graphs G, H, $\textsf{Hom}(F, G) \neq \textsf{Hom}(F, H) \Rightarrow$ there exists ψ such that ψ evaluates to different values on G and H. We prove by contradiction and assume all $\psi \in \Phi$ evaluates to the same value on G and H. We denote $\psi_{n_1 n_2 ... n_K} := \exists^{=n_1} \mathbf{x}_1 \exists^{=n_2} \mathbf{x}_2 ... \exists^{=n_K} \mathbf{x}_K \left(\bigwedge_{m \in [M]} E(x_{i_m}, x_{j_m}) \right)$. By assumption $\psi_{n_1 n_2 ... n_K}$ evaluates to the same value on G, H for any $n_1, n_2, ..., n_K$. We now construct a ψ such that ψ evaluates to **true** on G.

Let F be constructed by $varphi := \exists \mathbf{x}_1 \exists \mathbf{x}_2 ... \exists \mathbf{x}_K E(x_{i_1}, x_{j_1}) \wedge ... \wedge E(x_{i_M}, x_{j_M})$ such that $\textsf{Hom}(F, G) = N \neq \textsf{Hom}(F, H)$. Let $\mathbf{X}$ be the tuple of all variables in φ: $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_k)$. For all homomorphisms from F to G, let $\mathbb{V}$ be the set of images of $\mathbf{X}$ from F to G. We first pick all $\psi_{n_1 n_2 ... n_K}$ that evaluates to **true** on G. First, we have the following result:

Lemma 13. *If $n_k > N$ for some $k \in [K]$, then $\psi_{n_1 n_2 ... n_K} =$ **false** on G.*

Therefore, there are only finite $\psi_{n_1 n_2 ... n_K}$ that evaluates to **true** on G, and $n_k \leq N$ for all $k \in [K]$. For some $\psi_{n_1 n_2 ... n_K} =$ **true** on G, we refer to its *grounding* as a mapping from the variables $\mathbf{X}$ in ψ to the corresponding tuple of nodes $\mathbf{V}$ in G, and $\mathbf{V}$ is the *grounding result*. We then have the following result:

Lemma 14. *For $\psi_{n_1 n_2 ... n_K}$ and $\psi_{m_1 m_2 ... m_K}$, if $n_k \neq m_k$ for some $k \in [K]$, then then grounding results of $\psi_{n_1 n_2 ... n_K}$ and $\psi_{m_1 m_2 ... m_K}$ are different, i.e. there exists no $\mathbf{V}$ that is both a grounding result of $\psi_{n_1 n_2 ... n_K}$ and $\psi_{m_1 m_2 ... m_K}$.*

Let $\mathbb{S} = \{\psi_{n_1 n_2 ... n_K} \mid n_k \leq N \text{ for } k \in [K], \psi_{n_1 n_2 ... n_K} = \textbf{true} \text{ on } G\} = \{\psi_{n_1^l n_2^l ... n_K^l} \mid l \in [L]\}$ where $L = |\mathbb{S}|$. Then we have the following result:

$$\phi := \bigwedge_{l \in [L]} \psi_{n_1^l n_2^l ... n_K^l} \in \Phi \text{ evaluates } \textbf{true} \text{ on } G.$$

Moreover, according to Lemma 14 it is evident that the total number of different grounding results can be evaluated as

$$\begin{aligned}&\sum_{l\in[L]} \text{Number of grounding results of } \psi_{n_1^l n_2^l \ldots n_K^l}\\ =&\sum_{l\in[L]}\prod_{k\in[K]} n_k^l\end{aligned}$$

Since a grounding is also exactly a homomorphism from F to G, we have

$$\begin{aligned}&\sum_{l\in[L]} \text{Number of grounding results of } \psi_{n_1^l n_2^l \ldots n_K^l}\\ =&\textbf{Hom}(F, G) = N.\end{aligned}$$

By assumption, ϕ also evaluates to **true** on H, which indicates that $\psi_{n_1^l n_2^l \ldots n_K^l}$ evaluates to **true** for $l \in [L]$ on H. As a result,

$$\begin{aligned}&\textbf{Hom}(F, H)\\ \geq&\sum_{l\in[L]} \text{Number of grounding results of } \psi_{n_1^l n_2^l \ldots n_K^l}\\ =&N.\end{aligned}$$

This yields a contradiction where we assume $\textbf{Hom}(F, H) < \textbf{Hom}(F.G) = N$. Thus the proof completes.

We next prove the other direction, i.e. for any graphs G, H, there exists ψ such that ψ evaluates to different values on G and $H \Rightarrow \textbf{Hom}(F, G) \neq \textbf{Hom}(F, H)$. Without loss of generality we assume ψ evaluates to **true** on G and **false** on H. We first introduce the following lemma:

Lemma 15. *If there is $\psi \in \Phi$ such that ψ evaluates to different values on G and H, then there exists $\psi \in \Phi$ in the form of $\psi_{n_1 n_2 \ldots n_K} := \exists^{=n_1}\mathbf{x}_1\exists^{=n_2}\mathbf{x}_2\ldots\exists^{=n_K}\mathbf{x}_K \left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \bigwedge_{q\in[Q]} \neg E(x_{i_q}, x_{j_q})\right)$ that also evaluates to different values on G and H.*

Without loss of generality, we may now assume that there exists ψ in the form described by Lemma 15 that evaluates to **true** on G and **false** on H. We now prove that there must exists a $F \in \mathcal{F}$ such that $\textbf{Hom}(F, G) \neq \textbf{Hom}(F, H)$.

Given

$$\phi := \exists\mathbf{x}_1\exists\mathbf{x}_2\ldots\exists\mathbf{x}_K \left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \bigwedge_{q\in[Q]} \neg E(x_{i_q}, x_{j_q})\right),$$

let $\textbf{grd}(\phi, G)$ be the number of groundings from the variables in ϕ to G. Obviously **grd** is a extension to **hom** which allows negative edges $\neg E(x_{i_q}, x_{j_q}$ for $q \in [Q]$. We have the following result:

Lemma 16. *If $\textbf{Hom}(F, G) = \textbf{Hom}(F, H)$ for all $F \in \mathcal{F}$, then every $\phi \in \Phi$ of the above form satisfies:*

$$\textbf{grd}(\phi, G) = \textbf{grd}(\phi, H).$$

We define

$$\begin{aligned}\phi_{n_1 n_2 \ldots n_K} := &\exists\mathbf{x}_1^1\exists\mathbf{x}_1^2\ldots\exists\mathbf{x}_{x_1}^{n_1}\exists\mathbf{x}_2^{11}\mathbf{x}_2^{12}\ldots\exists\mathbf{x}_2^{1n_2}\exists\mathbf{x}_2^{21}\exists\mathbf{x}_2^{22}\ldots\exists\mathbf{x}_2^{n_1 n_2}\ldots\exists\mathbf{x}_K^{11\ldots1}\ldots\exists\mathbf{x}_K^{n_1 n_2 \ldots n_K}\\ &\left(\bigwedge_{p\in[P],} E(x_{i_p}^{l_1 l_2 \ldots l_{i_p}}, x_{j_p}^{l_1 l_2 \ldots l_{j_p}}) \bigwedge_{q\in[Q]} \neg E(x_{i_q}^{l_1 l_2 \ldots l_{i_q}}, x_{j_q}^{l_1 l_2 \ldots l_{j_q}})\right).\end{aligned}$$

Note that $\phi_{n_1 n_2 \ldots n_K} \in \Phi$ for arbitrary $n_1, n_2, \ldots, n_K$. We now prove by contradiction. Assume $\textbf{Hom}(F, G) = \textbf{Hom}(F, H)$ for arbitrary $F \in \mathcal{F}$. By Lemma 16 it is obvious that $\textbf{grd}(\phi_{n_1 n_2 \ldots n_K}, G) = \textbf{grd}(\phi_{n_1 n_2 \ldots n_K}, H)$ for all $n_1, n_2, \ldots, n_K$. First consider the case where

$\psi_{11\ldots1} = \textbf{true}$ on G. Obviously we have $\textbf{grd}(\phi_{11\ldots1}, G) = 1 = \textbf{grd}(\phi_{11\ldots1}, H)$, which indicates that there exists exactly one grounding from $\phi_{11\ldots1}$ to H and thus $\psi_{11\ldots1} = \textbf{true}$ on H, which yields a contradiction. We now assume $\psi_{N_1N_2\ldots N_K}$ evaluates to **true** on G. We prove that the evaluation of $\psi_{N_1N_2\ldots N_K}$ can be determined by the number of groundings from $\{\phi_{n_1n_2\ldots n_K} \mid n_k \in [N_k] \text{ for } k \in [K]\}$ to G, i.e.

$$(\textbf{grd}(\phi_{n_1n_2\ldots n_K}, G))_{n_k \in [N_k] \text{ for } k \in [K]}.$$

By proof by induction, we already know that $\psi_{11\ldots1}$ evaluates to **true** on G exactly when $\textbf{grd}(\phi_{11\ldots1}, G) = 1$ and thus $\psi_{11\ldots1}$ is captured by $(\textbf{grd}(\phi_{11\ldots1}, G))$. Now consider we are to prove $\psi_{N_1N_2\ldots N_K}$ is captured by $(\textbf{grd}(\phi_{n_1n_2\ldots n_K}, G))_{n_k \in [N_k] \text{ for } k \in [K]}$. Let us consider the groundings from $\phi_{N_1N_2\ldots N_K}$ to G. Obviously we can divide the groundings into two parts:

1. The non-injective groundings, i.e. groundings that maps different variables in $\phi_{N_1N_2\ldots N_K}$ to the same node in G.
2. The injective groundings, i,e, groundings that maps different variables in $\phi_{N_1N_2\ldots N_K}$ to different nodes in G.

Obviously the number of non-injective groundings can be computed by $\textbf{grd}(\phi_{n_1n_2\ldots n_K}, G)$ where for all $k \in [K]$ $n_k \leq N_k, k \in [K]$ and there exists $k \in [K]$ $n_k < N_k$. Thus, the number of injective homomorphisms can be evaluated. If the following constraints hold:

- The number of injective groundings from $\phi_{n_1n_2\ldots n_K}$ to G is larger than 0,
- The numbers of injective groundings from $\phi_{n_1+1n_2\ldots n_K}, \phi_{n_1n_2+1\ldots n_K} \ldots \phi_{n_1n_2\ldots n_K+1}$ to G are 0,

then obviously $\psi_{n_1\ldots n_K}$ evaluates to **true**. Therefore this yields a contradiction and the proof completes.

□

E.3 Proof of Corollary 5

Corollary 5. *The equivalent logic set of l-layer GACNNs defined above is given by $\Phi^{(l)}$.*

Proof. This is a direct result derived from Theorem 4 when we explicitly write down the computation procedure of a l-layer GACNNs. □

E.4 Proof of Proposition 6

Proposition 6. *The equivalent logic set of all GACNNs defined above is given by $\Phi = \bigcup_{l=0}^{\infty} \Phi^{(l)}$. Moreover, let $\Phi_i = \bigcup_{l=0}^{\infty} \Phi_i^{(l)}$ for $i \in [K]$, then Φ and $\{\Phi_i\}_{i\in[K]}$ exist and is defined by a similar procedure as Theorem 4. For the brevity of notation we denote $\chi^{(l)}$ as $\chi_{K+1}^{(l)}$, $\chi^{(l+1)}$ as $\chi_0^{(l)}$ and Φ as Φ_0 in the following description.*

- $$\chi_i^{(l)}(\mathbf{u}) = \text{AGG}\left(\left\{\left\{\chi_j^{(l)}(\mathbf{x}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\right\}\right\}\right)$$ $$\Longleftrightarrow \varphi_i(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_j(\mathbf{v}) \wedge \mathbf{1}_{\mathbf{v}\in\mathcal{N}_i(\mathbf{u})}\right) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}) \mid \textbf{atp}(\mathbf{u}),$$
- $$\chi_i^{(l)}(\mathbf{u}) = \text{COM}\left(\chi_j^{(l)}(\mathbf{u}), \chi_k^{(l)}(\mathbf{u})\right)$$ $$\Longleftrightarrow \varphi_i(\mathbf{u}) := \varphi_j(\mathbf{u}) \mid \varphi_k(\mathbf{u}) \mid \neg\varphi_i'(\mathbf{u}) \mid \varphi_i'(\mathbf{u}) \wedge \varphi_i''(\mathbf{u}) \mid \textbf{atp}(\mathbf{u}),$$

where $\varphi_i, \varphi_i', \varphi_i'' \in \Phi_i, \varphi_j \in \Phi_j, \varphi_k \in \Phi_k$ for $i \in \{0\} \cup [K]$.

Proof. Obviously, to consider all numbers of layers l simultaneously, the equivalent logic set is given by $\Phi = \bigcup_{l=0}^{\infty} \Phi^{(l)}$. Since for layer l the corresponding equivalent logic set Φ_l is given by

- $$\chi_i^{(l)}(\mathbf{u}) = \mathrm{AGG}\left(\left\{\left\{\chi_j^{(l)}(\mathbf{x}) \mid \mathbf{v} \in \mathcal{N}_i(\mathbf{u})\right\}\right\}\right)$$
$$\iff \varphi_i^{(l)}(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_j^{(l)}(\mathbf{v}) \wedge \mathbf{1}_{\mathbf{v}\in\mathcal{N}_i(\mathbf{u})}\right) \mid \neg\varphi^{(l)}{}'_i(\mathbf{u}) \mid \varphi^{(l)}{}'_i(\mathbf{u}) \wedge \varphi^{(l)}{}''_i(\mathbf{u}),$$
- $$\chi_i^{(l)}(\mathbf{u}) = \mathrm{COM}\left(\chi_j^{(l)}(\mathbf{u}), \chi_k^{(l)}(\mathbf{u})\right)$$
$$\iff \varphi_i^{(l)}(\mathbf{u}) := \varphi_j^{(l)}(\mathbf{u}) \mid \varphi_k^{(l)}(\mathbf{u}) \mid \neg\varphi^{(l)}{}'_i(\mathbf{u}) \mid \varphi^{(l)}{}'_i(\mathbf{u}) \wedge \varphi^{(l)}{}''_i(\mathbf{u}),$$

where $\varphi_i^{(l)}, \varphi_i^{(l)}{}', \varphi_i^{(l)}{}'' \in \Phi_i^{(l)}, \varphi_j^{(l)} \in \Phi_j^{(l)}, \varphi_k^{(l)} \in \Phi_k^{(l)}$ for $i \in \cup[K]$, and

$$\varphi^0(\mathbf{u}) := \textbf{atp}(\mathbf{u}).$$

It is obvious that the construction of Φ_i in Proposition 6 is a union of all $\Phi_i^{(l)}$ for $l \in [0, \infty)$: at beginning $\varphi_i := \textbf{atp}(\mathbf{u})$ thus at this moment $\Phi_i = \Phi_i^{(0)}$. Suppose at some iteration $\Phi_i = \Phi_i^{(l)}$. Then in next iteration we add

$$\varphi_i(\mathbf{u}) := \exists^{\geq N}\mathbf{v}\left(\varphi_j(\mathbf{v}) \wedge \mathbf{1}_{\mathbf{v}\in\mathcal{N}_i(\mathbf{u})}\right) \mid \varphi_j(\mathbf{u}) \mid \varphi_k(\mathbf{u}) \mid \neg\varphi'_i(\mathbf{u}) \mid \varphi'_i(\mathbf{u}) \wedge \varphi''_i(\mathbf{u}) \mid \textbf{atp}(\mathbf{u})$$

to Φ_i, and we still have $\Phi_i = \Phi_i^{(l+1)}$. Therefore $\Phi = \bigcup_{l=0}^{\infty} \Phi^{(l)}$ is given by Proposition 6. □

E.5 Proof of Proposition 7

Proposition 7. *The equivalent logic set Ψ of the graph representation χ_G defined above is given by*

$$\psi := \exists^{\geq N}\left(\varphi(\mathbf{u})\right) \mid \neg\psi' \mid \psi' \wedge \psi'',$$

where $\psi, \psi', \psi'' \in \Psi, \varphi \in \Phi$.

Proof. This is a direct result derived from Theorem 4. Since

$$\chi_G = \mathrm{AGG}\left(\{\{\chi(\mathbf{u}) \mid \mathbf{u} \in \mathcal{V}^k\}\}\right)$$

where $\mathcal{V}$ is the set of nodes in G and k is the order of $\mathbf{u}$, Ψ is specified by Theorem 4 as above. □

E.6 Proof of Proposition 8

Proposition 8. *The equivalent logic sets of GNN models can be separately defined as:*

- ***MPNN**: $\varphi(x) := \exists^{\geq N}x\left(\varphi'(y) \wedge E(x,y)\right)$, where E is the edge predicate.*
- ***Subgraph GNN (weak)**: $\varphi(x) := \exists^{\geq N}y\left(\psi(x,y)\right)$, and $\psi(x,y) := \exists^{\geq N}z\left(\psi'(x,z) \wedge E(z,y)\right)$.*
- ***Subgraph GNN (strong)**: $\varphi(x) := \exists^{\geq N}y\left(\varphi'(y) \wedge \psi(y,x)\right), \psi(x,y) := \exists^{\geq N}z\left(\psi(x,z) \wedge E(z,y)\right) \mid \varphi(y)$.*
- ***NBFNet**: $\varphi(x,y) := \exists^{\geq N}z\left(\varphi'(x,z) \wedge E(z,y)\right)$.*
- ***Local 2-GNN**: $\varphi(x,y) := \exists^{\geq N}z\left(\varphi'(x,z) \wedge E(z,y))\right) \mid \exists^{\geq N}z\left(E'(x,z) \wedge \varphi'(z,y))\right)$*
- ***2-FGNN**: $\varphi(x,y) := \exists^{\geq N}z\left(\varphi'(x,z) \wedge \varphi''(z,y)\right)$.*
- ***SEAL (MPNN)**: $\varphi(x,y) := \exists^{\geq N}z\left(\psi(x,z,y)\right), \psi(x,z,y) := \exists^{\geq N}w\left(\psi(x,w,y) \wedge E(w,z))\right)$.*
- ***2-GNN**: $\varphi(x,y) := \exists^{\geq N}z\left(\varphi'(x,z)\right) \mid \exists^{\geq N}z\left(\varphi'(z,y)\right)$.*

Proof. By utilizing the results from Proposition 6 and further simplify the resulted equivalent logic sets, we can easily obtain these results. Note that for brevity we omit the terms $\varphi := \neg\varphi' \mid \varphi' \wedge \varphi''$.

MPNN.

$$\chi^{(l+1)}(x) = \mathrm{COM}\left(\chi^{(l)}(x), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(y) \mid y \in \mathcal{N}(x)\right\}\right\}\right)\right)$$
$$\Rightarrow \varphi(x) := \varphi'(x) \mid \exists^{\geq N}x\left(\varphi'(y) \wedge E(x,y)\right)$$
$$\Rightarrow \varphi(x) := \exists^{\geq N}x\left(\varphi'(y) \wedge E(x,y)\right).$$

Subgraph GNN (weak). The layers are given by

$$\chi^{(l+1)}(x) = \mathrm{AGG}\left(\left\{\left\{\chi^{(l+1)}(x,y) \mid y \in \mathcal{V}\right\}\right\}\right),$$
$$\chi^{(l+1)}(x,y) = \mathrm{COM}\left(\chi^{(l)}(x,y), \mathrm{AGG}\left(\chi^{(l)}(x,z) \mid z \in \mathcal{N}(y)\right)\right).$$

Therefore the equivalent logic sets are given by

$$\varphi(x) := \exists^{\geq N} y\, (\psi(x,y)),$$
$$\psi(x,y) := \psi'(x,y) \mid \exists^{\geq N} z\, (\psi'(x,z) \wedge E(z,y)),$$

which can be directly simplified as

$$\varphi(x) := \exists^{\geq N} y\, (\psi(x,y)),$$
$$\psi(x,y) := \exists^{\geq N} z\, (\psi'(x,z) \wedge E(z,y)),$$

Subgraph GNN (strong). The layers are given by

$$\chi^{(l+1)}(x,y) = \mathrm{COM}\left(\chi^{(l)}(x,y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x,z) \mid z \in \mathcal{N}(y)\right\}\right\}\right), \chi^{(l)}(y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(z) \mid z \in \mathcal{N}(y)\right\}\right\}\right)\right),$$
$$\chi^{(l+1)}(x) = \mathrm{AGG}\left(\chi^{(l+1)}(y,x) \mid y \in \mathcal{V}\right).$$

Therefore the equivalent logic sets are given by

$$\psi(x,y) := \psi'(x,y) \mid \exists^{\geq N} z\, (\psi(x,z) \wedge E(z,y)) \mid \varphi(y) \mid \exists^{\geq N} z\, (\varphi(z) \wedge E(z,y)),$$
$$\varphi(x) := \exists^{\geq N} y\, (\psi(y,x)).$$

Substituting $\psi(x,y) := (\varphi(z) \wedge E(z,y))$ to the second line leads to

$$\varphi(x) := \exists^{\geq N} z\, (\varphi'(z) \wedge E(z,x)).$$

Therefore, the above Φ can also be described by

$$\varphi(x) := \exists^{\geq N} y\, (\varphi'(y) \wedge \psi(y,x)),$$
$$\psi(x,y) := \exists^{\geq N} z\, (\psi(x,z) \wedge E(z,y)) \mid \varphi(y).$$

NBFNet.

$$\chi^{(l+1)}(x,y) = \mathrm{COM}\left(\chi^{(l)}(x,y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x,z) \mid z \in \mathcal{N}(y)\right\}\right\}\right)\right)$$
$$\Rightarrow \varphi(x,y) := \varphi'(x,y) \mid \exists^{\geq N} z\, (\varphi'(x,z) \wedge E(z,y))$$
$$\Rightarrow \varphi(x,y) := \exists^{\geq N} z\, (\varphi'(x,z) \wedge E(z,y)).$$

Local 2-GNN.

$$\chi^{(l+1)}(x,y) = \mathrm{COM}\left(\chi^{(l)}(x,y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(z,y) \mid z \in \mathcal{N}(x)\right\}\right\}\right), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x,z) \mid z \in \mathcal{N}(y)\right\}\right\}\right)\right)$$
$$\Rightarrow \varphi(x,y) := \varphi'(x,y) \mid \exists^{\geq N} z\, (\varphi'(x,z) \wedge E(z,y))) \mid \exists^{\geq N} z\, (E'(x,z) \wedge \varphi'(z,y)))$$
$$\Rightarrow \varphi(x,y) := \exists^{\geq N} z\, (\varphi'(x,z) \wedge E(z,y))) \mid \exists^{\geq N} z\, (E'(x,z) \wedge \varphi'(z,y))).$$

2-FGNN.

$$\chi^{(l+1)}(x,y) = \mathrm{COM}\left(\chi^{(l)}(x,y), \mathrm{AGG}\left(\left\{\left\{\mathrm{COM}\left(\chi^{(l)}(x,z), \chi^{(l)}(z,y)\right) \mid z \in \mathcal{V}\right\}\right\}\right)\right)$$
$$\Rightarrow \varphi(x,y) := \varphi'(x,y) \mid \exists^{\geq N} z\, (\psi(x,y,z)), \psi(x,y,z) := \varphi(x,y) \mid \varphi(y,z)$$
$$\Rightarrow \varphi(x,y) := \exists^{\geq N} z\, (\varphi'(x,z) \wedge \varphi''(z,y)).$$

The last line holds because $\mathbf{1}(x,y) \in \Phi$ where $\mathbf{1}(x,y) \equiv$ **true** for all (x,y).

SEAL (MPNN). The layers are given by

$$\chi^{(l+1)}(x,z,y) = \mathrm{COM}\left(\chi^{(l)}(x,z,y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x,w,y) \mid w \in \mathcal{N}(z)\right\}\right\}\right)\right),$$
$$\chi^{(l+1)}(x,y) = \mathrm{AGG}\left(\left\{\left\{\chi^{(l+1)}(x,z,y) \mid z \in \mathcal{N}\right\}\right\}\right).$$

Therefore the equivalent logic sets are given by

$$\varphi(x,y) := \exists^{\geq N} z\left(\psi(x,z,y)\right),$$
$$\psi(x,z,y) := \exists^{\geq N} w\left(\psi(x,w,y) \wedge E(w,z)\right).$$

2-GNN

$$\chi^{(l+1)}(x,y) = \mathrm{COM}\left(\chi^{(l)}(x,y), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(z,y) \mid z \in \mathcal{V}\right\}\right\}\right), \mathrm{AGG}\left(\left\{\left\{\chi^{(l)}(x,z) \mid z \in \mathcal{V}\right\}\right\}\right)\right)$$
$$\Rightarrow \varphi(x,y) := \varphi'(x,y) \mid \exists^{\geq N} z\left(\varphi'(x,z)\right)) \mid \exists^{\geq N} z\left(\varphi'(z,y)\right))$$
$$\Rightarrow \varphi(x,y) := \exists^{\geq N} z\left(\varphi'(x,z)\right)) \mid \exists^{\geq N} z\left(\varphi'(z,y)\right)).$$

□

E.7 Proof of Proposition 10

Proposition 10. *Suppose the equivalent logic set of a class of GNN models is Φ. Then, the expressive power of the GNN models is bounded by k-WL,* iff *the number of variables of the logic formulas in Φ is at most k.*

Proof. Given any graphs G, H, Cai et al. (1992) states that the following statements are equivalent:

- k-WL distinguishes G, H;
- There is a FOC_k formula that distinguishes G, H.

Recall that FOC_k is a subset of first-order formula that allows quantifiers $\exists^{\geq N}$ but restricts the formulas to only possess k. Obviously, Φ in Proposition 10 is a subset of FOC_k, thus the expressive power of GNNs is bounded by k-WL. □

E.8 Proof of Corollary 11

Corollary 11. *The expressivity of GNN models satisfies: MPNNs = 1-WL < Subgraph GNNs (weak) = NBFNet < Subgraph GNNs (strong) < Local 2-FGNN < 2-FGNN = 3-WL, 1-WL < SEAL < 4-WL.*

Proof. By utilizing Proposition 6, Proposition 7 and Proposition 10, obviously Corollary 11 holds. □

E.9 Proof of Lemma 13

Lemma 13. *If $n_k > N$ for some $k \in [K]$, then $\psi_{n_1 n_2 \ldots n_K} = \textsf{false}$ on G.*

Proof. Obviously if $n_k > N$ for some $k \in [K]$, we have

$$\begin{aligned} &\textsf{Hom}(F,G) \\ \geq &\prod_{k \in [K]} n_k \\ > &N, \end{aligned}$$

which contradicts with the fact that $\textsf{Hom}(F,G) = N$. □

E.10 Proof of Lemma 14

Lemma 14. *For $\psi_{n_1n_2...n_K}$ and $\psi_{m_1m_2...m_K}$, if $n_k \neq m_k$ for some $k \in [K]$, then then grounding results of $\psi_{n_1n_2...n_K}$ and $\psi_{m_1m_2...m_K}$ are different, i.e. there exists no $\mathbf{V}$ that is both a grounding result of $\psi_{n_1n_2...n_K}$ and $\psi_{m_1m_2...m_K}$.*

Proof. Since $n_k \neq m_k$ for some $k \in [K]$, we assume that $n_l \neq m_l$ while $n_k = m_k$ for $k \in [l+1, K]$. We prove by contradiction. Suppose $\mathbf{V} = (\mathbf{v}_1, ..., \mathbf{v}_K)$ is both a grounding result of $\psi_{n_1n_2...n_K}$ and $\psi_{m_1m_2...m_K}$. This indicates that for $\psi_{n_1n_2...n_K}$, by fixing its variables to $\mathbf{x}_1 := \mathbf{v}_1, \mathbf{x}_2 := \mathbf{v}_2, ..., \mathbf{x}_{l-1} := \mathbf{v}_{l-1}$, there exists exactly n_l different groundings of $\mathbf{x}_l$ satisfying

$$\exists^{=n_{l+1}}\mathbf{x}_{l+1}\exists^{=n_{l+2}}\mathbf{x}_{l+2}...\exists^{=n_K}\mathbf{x}_K\left(\bigwedge_{m\in[M]} E(x_{i_m}, x_{j_m})\right)$$

in G. However, for $\psi_{m_1m_2...m_K}$ by fixing its variables to $\mathbf{x}_1 := \mathbf{v}_1, \mathbf{x}_2 := \mathbf{v}_2, ..., \mathbf{x}_{l-1} := \mathbf{v}_{l-1}$, there exists exactly m_l different groundings of $\mathbf{x}_l$ satisfying

$$\exists^{=m_{l+1}}\mathbf{x}_{l+1}\exists^{=m_{l+2}}\mathbf{x}_{l+2}...\exists^{=m_K}\mathbf{x}_K\left(\bigwedge_{m\in[M]} E(x_{i_m}, x_{j_m})\right)$$

in G. Since $m_k = n_k$ for $k \in [l+1, K]$ and $m_l \neq n_l$, this yields a contradiction. $\square$

E.11 Proof of Lemma 15

Lemma 15. *If there is $\psi \in \Phi$ such that ψ evaluates to different values on G and H, then there exists φ in the form of $\varphi := \exists^{=n_1}\mathbf{x}_1\exists^{=n_2}\mathbf{x}_2...\exists^{=n_K}\mathbf{x}_K\left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p})\bigwedge_{q\in[Q]} \neg E(x_{i_q}, x_{j_q})\right)$ that also evaluates to different values on G and H.*

Proof. We prove this by constructing φ of the form

$$\varphi := \exists^{=n_1}\mathbf{x}_1...\left(\bigwedge_{m\in[M]} E(x_{i_m}, x_{j_m})\right)$$

that explicitly captures the colors of χ. Concretely, similar as Theorem 4, suppose a series of functions $\{\chi_1, ..., \chi_L\}$ where χ_l is defined by

$$\chi_l(\mathbf{x}) = \mathrm{hash}\left(\chi_p(\mathbf{x}), \chi_q(\mathbf{x})\right),$$

$$\chi_l(\mathbf{x}) = \mathrm{hash}\left(\{\{\chi_p(\mathbf{y}) \mid y \in \mathcal{N}(\mathbf{x})\}\}\right),$$

or

$$\chi_l(\mathbf{x}) = \textsf{\textbf{atp}}(\mathbf{x}).$$

The difference between Theorem 4 and here is that we replace AGG and COM functions with injective hash function. Obviously the separation power of $(\chi_l)_{l\in[L]}$ here is no less than that in Theorem 4. The above procedure can be regarded as a general *color refinement* algorithm where the value of $\chi_l(\mathbf{x})$ is called the *color* of $\mathbf{x}$ computed by χ_l. We define the *signature* logic set Ψ_l of chi_l to be the set that satisfiesfor each color C, there exists $\psi_C \in \Phi_l$ such that

$$\psi_C(\mathbf{x}) = \textsf{\textbf{true}} \iff \chi(\mathbf{x}) = C.$$

We now provide a method to construct the signature logic set Ψ_l. We define:

1. $$\begin{aligned}\chi_l(\mathbf{x}) &= \mathrm{hash}\left(\chi_p(\mathbf{x}), \chi_q(\mathbf{x})\right)\\ \Rightarrow \psi_l(\mathbf{x}) &:= \psi_p(\mathbf{x}) \wedge \psi_q(\mathbf{x})\end{aligned}$$

2.
$$\chi_l(\mathbf{x}) = \text{hash}\left(\{\{\chi_p(\mathbf{y}) \mid y \in \mathcal{N}(\mathbf{x})\}\}\right)$$
$$\Rightarrow \psi_l(\mathbf{x}) := \exists^{=N}\mathbf{y}\left(\mathbf{1}_{\mathbf{y}\in\mathcal{N}(\mathbf{x})}\right) \wedge \exists^{=N_1}\mathbf{y}_1\left(\psi_p(\mathbf{y}) \wedge \mathbf{1}_{\mathbf{y}_1\in\mathcal{N}(\mathbf{x})}\right)$$
$$\wedge \exists^{=N_2}\mathbf{y}_2\left(\psi_p(\mathbf{y}) \wedge \mathbf{1}_{\mathbf{y}_2\in\mathcal{N}(\mathbf{x})}\right) \wedge ... \mid \exists^{=0}\mathbf{y}\mathbf{1}_{\mathbf{y}\in\mathcal{N}(\mathbf{x})},$$

where $N_2 \geq N_1 \geq 1$.

3.
$$\chi_l(\mathbf{x}) = \textsf{\textbf{atp}}(\mathbf{x})$$
$$\Rightarrow \psi_l(\mathbf{x}) := \mathbf{1}_{\textsf{\textbf{atp}}(\mathbf{x})=C}$$

where for each possible structure of a k-node graph (k is the order of $\mathbf{x}$; we consider node orders thus there are 2^k structures in total; suppose the nodes of the k-node graph are $v_1, ..., v_k$), there is a corresponding ψ_l that evaluates to **true** *iff* there is an isomorphism from the subgraph induced by $\mathbf{x}$ to the structure of the corresponding k-node graph that maps v_i to x_i for $i \in [k]$ where $\mathbf{x} = (x_1, ..., x_k)$.

We next prove that the above Ψ_l indeed is the signature logic set of χ_l. We denote $\psi_l^C(\mathbf{x})$ as the logic formula that evaluates to **true** *iff* $\chi_l(\mathbf{x}) = C$ where C is the color of $\mathbf{x}$ evaluated by χ_l. For situation 3 the statement obviously holds. For situation 1, suppose Ψ_p, Ψ_q are the signature logic sets of χ_p, χ_q respectively. For each color C_l of χ_l where $C_l = \text{hash}(C_p, C_q)$, we have

$$\psi_l^{C_l}(\mathbf{x}) := \psi_p^{C_p}(\mathbf{x}) \wedge \psi_q^{C_q}(\mathbf{x})$$

which is **true** *iff* $\chi_l(\mathbf{x}) = C_l$. Thus the statement still holds.

For situation 2, suppose Ψ_p is the signature logic set of χ_p. Each color C_l of χ_l is defined by

$$C_l = \text{hash}\left(\left\{\left\{C_p^1, C_p^2, ...\right\}\right\}\right) = \text{hash}\left(\left\{(C_p^1, N_1), (C_p^2, N_2), ...\right\}\right)$$

where $C_p^1, C_p^2, ...$ are colors produced by χ_p, and $N_1, N_2, ... \geq 1$ are the numbers of the colors $C_p^1, C_p^2, ...$ emerged in the multiset, We then have

$$\psi_l^{C_l}(\mathbf{x}) := \exists^{=N}\mathbf{y}\mathbf{1}_{\mathbf{y}\in\mathcal{N}(\mathbf{x})}\exists^{=N_1}\mathbf{y}_1\left(\psi_p^{C_p^1}(\mathbf{y}_1) \wedge \mathbf{1}_{\mathbf{y}_1\in\mathcal{N}(\mathbf{x})}\right)\wedge\exists^{=N_2}\mathbf{y}_2\left(\psi_p^{C_p^2}(\mathbf{y}_2) \wedge \mathbf{1}_{\mathbf{y}_2\in\mathcal{N}(\mathbf{x})}\right)\wedge....$$

Specially, if the multiset is empty, we have

$$\psi_l^{C_l} := \exists^{=0}\mathbf{y}\mathbf{1}_{\mathbf{y}\in\mathcal{N}(\mathbf{x})}.$$

Then, $\psi_l^{C_l}(\mathbf{x})$ is **true** *iff* $\chi_l(\mathbf{x}) = C_l$. $\psi_l^{C_l}$ is also in Ψ_l. Therefore, we have constructed the signature logic set of χ_l. Obviously, all $\psi_l \in \Psi_l$ can be written in the form of

$$\psi_l(\mathbf{x}) := \exists^{=N_1}\mathbf{x}_1 ... \exists^{=N_K}\mathbf{x}_K\left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \bigwedge_{q\in[Q]} (\neg E(x_{s_q}, x_{t_q}))\right).$$

For two graphs G, H, if there exists ψ that distinguishes them, then obviously the corresponding χ also distinguishes them. Without loss of generality, suppose the color of χ applied on G is C. Let ψ^C be the logic formula that evaluates **true** *iff* $\chi = C$. Then, we have

ψ^C evaluates to **true** on G and **false** on H.

Recall that since ψ^C can be written in the form of

$$\psi^C(\mathbf{x}) := \exists^{=N_1}\mathbf{x}_1 ... \exists^{=N_K}\mathbf{x}_K\left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \bigwedge_{q\in[Q]} (\neg E(x_{s_q}, x_{t_q}))\right),$$

the proof completes.

$\square$

E.12 Proof of Lemma 16

Lemma 16. *If* $\textbf{Hom}(F,G) = \textbf{Hom}(F,H)$ *for all* $F \in \mathcal{F}$*, then every* $\phi \in \Phi$ *of the above form satisfies:*

$$\textbf{grd}(\phi, G) = \textbf{grd}(\phi, H).$$

Proof. We prove by contradiction and assume $\textbf{Hom}(F,G) = \textbf{Hom}(F,H)$ for all $F \in \mathcal{F}$ but there exists $\phi \in \Phi$ given by

$$\phi := \exists \mathbf{x}_1 \exists \mathbf{x}_2 ... \exists \mathbf{x}_K \left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \bigwedge_{q\in[Q]} \neg E(x_{i_q}, x_{j_q}) \right)$$

that classifies G and H differently. Without loss of generality we assume ϕ evaluates to **true** in G. By the definition of **grd**, obviously if $\textbf{Hom}(F,G) = \textbf{Hom}(F,H)$ for all $F \in \mathcal{F}$, we have $\textbf{grd}(\psi, G) = \textbf{grd}(\psi, H)$ for all $\psi \in \Psi \subseteq \Phi$ where $\psi \in \Psi$ is of the form

$$\psi := \exists \mathbf{x}_1 \exists \mathbf{x}_2 ... \exists \mathbf{x}_K \left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \right),$$

i.e. ψ contains no negative edges $\neg E$. We next show that we can use $\textbf{grd}(\psi, \cdot)$ for $\psi \in \Psi$ to infer $\textbf{grd}(\phi, \cdot)$. We denote

$$\phi_q := \exists \mathbf{x}_1 ... \exists \mathbf{x}_K \left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \bigwedge_{r\in[q]} (\neg E(x_{s_r}, x_{t_r})) \right)$$

We now show that we can use the results $\textbf{grd}(\psi, \cdot)$ for $\psi \in \Psi$ to infer $\textbf{grd}(\phi^q, \cdot)$ for $q = 0, 1, ..., Q$. We denote $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_K)$. For $q = 0$, obviously $\phi^0 \in \Psi$ thus the statement naturally holds. Since the groundings of ϕ^0 consist of two parts:

- $\mathbf{X}$ that satisfy $\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge E(x_{s_1}, x_{t_1})$;
- $\mathbf{X}$ that satisfy $\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge \neg E(x_{s_1}, x_{t_1})$, corresponding to ϕ_1.

Obviously the two part do not intersect. Therefore, $\textbf{grd}(\phi_1, \cdot) = \textbf{grd}(\phi_0, \cdot) - \textbf{grd}(\varphi^{(1)}, \cdot)$ where

$$\varphi^{(1)} := \exists \mathbf{x}_1 ... \exists \mathbf{x}_K \left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge E(x_{s_1}, x_{t_1}) \right).$$

Similarly, to infer $\textbf{grd}(\phi_2, \cdot)$, the set of $\mathbf{X}$ that satisfy $\left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \right)$ consists of four non-intersect parts:

- $\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge E(x_{s_1}, x_{t_1}) \wedge E(x_{s_2}, x_{t_2})$,
- $\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge E(x_{s_1}, x_{t_1}) \wedge \neg E(x_{s_2}, x_{t_2})$,
- $\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge \neg E(x_{s_1}, x_{t_1}) \wedge E(x_{s_2}, x_{t_2})$,
- $\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge \neg E(x_{s_1}, x_{t_1}) \wedge \neg E(x_{s_2}, x_{t_2})$,

According to our assumption the first part is known. For the second part, since $\left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge E(x_{s_1}, x_{t_1}) \right)$ consists of two non-intersect parts:

- $\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge E(x_{s_1}, x_{t_1}) \wedge E(x_{s_2}, x_{t_2})$,
- $\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p}) \wedge E(x_{s_1}, x_{t_1}) \wedge \neg E(x_{s_2}, x_{t_2})$.

Thus the second part can also be inferred, and so does the third part. Therefore, the set of nodes that satisfy $\left(\bigwedge_{p\in[P]} E(x_{i_p}, x_{j_p})\right)$ can be inferred, and thus also $\textsf{\textbf{grd}}(\phi_2, G) = \textsf{\textbf{grd}}(\phi_2, H)$ Using the same strategy one can show that

$$\textsf{\textbf{grd}}(\phi_q, G) = \textsf{\textbf{grd}}(\phi_q, H)$$

for any q. Therefore, this yields the contradiction.

$\square$