

## 452 A Probability Distribution of Communities and Bipartites

453 **Theorem A.1.** *Given a graph  $\mathcal{G}$  and an ordering  $\pi$ , assuming there is a deterministic function that*  
 454 *provides the corresponding high-level graphs in a hierarchical order as  $\{\mathcal{G}^L, \mathcal{G}^{L-1}, \dots, \mathcal{G}^0\}$ , then:*

$$\begin{aligned} p(\mathcal{G} = \mathcal{G}^L, \pi) &= p(\{\mathcal{G}^L, \mathcal{G}^{L-1}, \dots, \mathcal{G}^0\}, \pi) = p(\mathcal{G}^L, \pi \mid \{\mathcal{G}^{L-1}, \dots, \mathcal{G}^0\}) \dots p(\mathcal{G}^1, \pi \mid \mathcal{G}^0) p(\mathcal{G}^0) \\ &= \prod_{l=0}^L p(\mathcal{G}^l, \pi \mid \mathcal{G}^{l-1}) \times p(\mathcal{G}^0) \end{aligned} \quad (9)$$

455 *Proof.* The factorization is derived by applying the chain rule of probability and last equality holds  
 456 as the graphs at the coarser levels are produced by a partitioning function acting on the finer level  
 457 graphs. Overall, this hierarchical generative model exhibits a Markovian structure.  $\square$

### 458 A.1 Proof of Theorem 3.1

459 **Lemma A.2.** *Given the sum of counting variables in the groups, the groups are independent and*  
 460 *each of them has multinomial distribution:*

$$\begin{aligned} p(\mathbf{w} = [\mathbf{u}_1, \dots, \mathbf{u}_M] \mid \{\mathbf{v}_1, \dots, \mathbf{v}_M\}) &= \prod_{m=1}^M \text{Mu}(\mathbf{v}_m, \boldsymbol{\lambda}_m) \\ \text{where: } \boldsymbol{\lambda}_m &= \frac{\boldsymbol{\theta}_m}{\mathbf{1}^T \boldsymbol{\theta}_m} \end{aligned}$$

461 *Here, probability vector (parameter)  $\boldsymbol{\lambda}_m$  is the normalized multinomial probabilities of the counting*  
 462 *variables in the  $m$ -th group.*

*Proof.*

$$\begin{aligned} p(\mathbf{w} \mid \{\mathbf{v}_1, \dots, \mathbf{v}_M\}) &= \frac{p(\mathbf{w})}{p(\{\mathbf{v}_1, \dots, \mathbf{v}_M\})} I(\mathbf{v}_1 = \mathbf{1}^T \mathbf{u}_1, \dots, \mathbf{v}_M = \mathbf{1}^T \mathbf{u}_M) \\ &= \frac{\frac{w!}{\prod_{i=1}^E w_i!} \prod_{i=1}^E \boldsymbol{\theta}_i^{w_i}}{\frac{w!}{\prod_{i=1}^M v_i!} \prod_{i=1}^M \boldsymbol{\alpha}_i^{v_i}} I(\mathbf{v}_1 = \mathbf{1}^T \mathbf{u}_1, \dots, \mathbf{v}_M = \mathbf{1}^T \mathbf{u}_M) \\ &= \frac{\frac{w!}{\prod_{i=1}^E w_i!} \boldsymbol{\theta}_1^{w_1} \dots \boldsymbol{\theta}_E^{w_E}}{\frac{w!}{\prod_{i=1}^M v_i!} (\mathbf{1}^T \boldsymbol{\theta}_1)^{v_1} \dots (\mathbf{1}^T \boldsymbol{\theta}_M)^{v_M}} \\ &= \frac{v_1!}{\prod_{i=1}^{E_1} \mathbf{u}_{1,i}!} \prod_{i=1}^{E_1} \boldsymbol{\lambda}_{1,i}^{\mathbf{u}_{1,i}} \times \dots \times \frac{v_M!}{\prod_{i=1}^{E_M} \mathbf{u}_{M,i}!} \prod_{i=1}^{E_M} \boldsymbol{\lambda}_{M,i}^{\mathbf{u}_{M,i}} \\ &= \text{Mu}(\mathbf{v}_1, \boldsymbol{\lambda}_1) \times \dots \times \text{Mu}(\mathbf{v}_M, \boldsymbol{\lambda}_M) \end{aligned}$$

463  $\square$

464 In a hierarchical graph, the edges has non-negative integer valued weights while the sum of all the  
 465 edges in community  $\mathcal{C}_i^l$  and bipartite graph  $\mathcal{B}_{ij}^l$  are determined by their corresponding edges in the  
 466 parent graph, i.e.  $w_{ii}^{l-1}$  and  $w_{ij}^{l-1}$  respectively. Let the random vector  $\mathbf{w} := [w_e]_e \in \mathcal{E}(\mathcal{G}^l)$  denote the  
 467 set of weights of all edges of  $\mathcal{G}^l$  such that  $w_0 = \mathbf{1}^T \mathbf{w}$ , its joint probability can be described as a  
 468 multinomial distribution:

$$\mathbf{w} \sim \text{Mu}(\mathbf{w} \mid w_0, \boldsymbol{\theta}^l) = \frac{w_0!}{\prod_{e=1}^{|\mathcal{E}(\mathcal{G}^l)|} w[e]!} \prod_{e=1}^{|\mathcal{E}(\mathcal{G}^l)|} (\boldsymbol{\theta}^l[e])^{w[e]}, \quad (10)$$

where  $\{\theta^l[e] \in [0, 1], \text{ s.t. } \mathbf{1}^T \theta^l = 1\}$  are the parameters of the multinomial distribution<sup>4</sup> Therefore, based on lemma A.2 these components are conditionally independent and each of them has a multinomial distribution:

$$p(\mathcal{G}^l | \mathcal{G}^{l-1}) \sim \prod_{i \in \mathcal{V}(\mathcal{G}^{l-1})} \text{Mu}([w_e]_e \in \mathcal{C}_i^l | w_{ii}^{l-1}, \theta_{ii}^l) \times \prod_{(i,j) \in \mathcal{E}(\mathcal{G}^{l-1})} \text{Mu}([w_e]_e \in \mathcal{B}_{ij}^l | w_{ij}^{l-1}, \theta_{ij}^l)$$

where  $\{\theta_{ij}^l[e] \in [0, 1], \text{ s.t. } \mathbf{1}^T \theta_{ij}^l = 1 \mid \forall (i, j) \in \mathcal{E}(\mathcal{G}^{l-1})\}$  are the parameters of the model.

Therefore, the log-likelihood of  $\mathcal{G}^l$  can be decomposed as the log-likelihood of its sub-structures:

$$\log p_{\phi^l}(\mathcal{G}^l | \mathcal{G}^{l-1}) = \sum_{i \in \mathcal{V}_{\mathcal{G}^{l-1}}} \log p_{\phi^l}(\mathcal{C}_i^l | \mathcal{G}^{l-1}) + \sum_{(i,j) \in \mathcal{E}_{\mathcal{G}^{l-1}}} \log p_{\phi^l}(\mathcal{B}_{ij}^l | \mathcal{G}^{l-1}) \quad (11)$$

□

**Bipartite distribution:** Let's denote the set of weights of all candidate edges of the bipartite  $\mathcal{B}_{ij}^l$  by a random vector  $\mathbf{w} := [w_e]_{e \in \mathcal{E}(\mathcal{B}_{ij}^l)}$ , its probability can be described as

$$\mathbf{w} \sim \text{Mu}(\mathbf{w} | w_{ij}^{l-1}, \theta_{ij}^l) = \frac{w_{ij}^{l-1}!}{\prod_{e=1}^{|\mathcal{E}(\mathcal{B}_{ij}^l)|} \mathbf{w}[e]!} \prod_{e=1}^{|\mathcal{E}(\mathcal{B}_{ij}^l)|} (\theta_{ij}^l[e])^{\mathbf{w}[e]} \quad (12)$$

where  $\{\theta_{ij}^l[e] \mid \theta_{ij}^l[e] \geq 0, \sum \theta_{ij}^l[e] = 1\}$  are the parameter of the distribution, and the multinomial coefficient  $\frac{n!}{\prod \mathbf{w}[e]!}$  is the number of ways to distribute the total weight  $w_{ij}^{l-1} = \sum_{e=1}^{|\mathcal{E}(\mathcal{B}_{ij}^l)|} \mathbf{w}[e]$  into all candidate edges of  $\mathcal{B}_{ij}^l$ .

**Community distribution:** Similarly, the probability distribution of the set of candidate edges for each community can be modeled jointly by a multinomial distribution but as our objective is to model the generative probability of communities in each level as an autoregressive process we are interested to decomposed this probability distribution accordingly.

## A.2 Proof of Theorem 3.3

For a random counting vector  $\mathbf{w} \in \mathbb{Z}_+^E$  with multinomial distribution  $\text{Mu}(w, \theta)$ , let's split it into  $M$  disjoint groups  $\mathbf{w} = [\mathbf{u}_1, \dots, \mathbf{u}_M]$  where  $\mathbf{u}_m \in \mathbb{Z}_+^{E_m}$ ,  $\sum_{m=1}^M E_m = E$ , and also split the probability vector as  $\theta = [\theta_1, \dots, \theta_M]$ . Additionally, let's define sum of all weights in  $m$ -th group by a random variable  $v_m := \sum_{e=1}^{E_m} \mathbf{u}_{m,e}$ .

**Lemma A.3.** Sum of the weights in the groups,  $\mathbf{u}_m \in \mathbb{Z}_+^{E_m}$ ,  $\sum_{m=1}^M E_m = E$  has multinomial distribution:

$$p(\{v_1, \dots, v_M\}) = \text{Mu}(w, [\alpha_1, \dots, \alpha_M])$$

where:  $\alpha_m = \sum \theta_m[i]$ . (13)

In the other words, the multinomial distribution is preserved when its counting variables are combined Siegrist (2017).

**Theorem A.4.** Given the aforementioned grouping of counts variables, the multinomial distribution can be modeled as a chain of binomials and multinomials:

$$\text{Mu}(w, \theta = [\theta_1, \dots, \theta_M]) = \prod_{m=1}^M \text{Bi}(w - \sum_{i < m} v_i, \eta_{v_m}) \text{Mu}(v_m, \lambda_m), \quad (14)$$

$$\text{where: } \eta_{v_m} = \frac{\mathbf{1}^T \theta_m}{1 - \sum_{i < m} \mathbf{1}^T \theta_i}, \quad \lambda_m = \frac{\theta_m}{\mathbf{1}^T \theta_m}$$

<sup>4</sup>It is analogous to the random trial of putting  $n$  balls into  $k$  boxes, where the joint probability of the number of balls in all the boxes follows the multinomial distribution.

495 *Proof.* Since sum of the weights of the groups,  $\mathbf{v}_m$ , are functions of the weights in the group:

$$p(\mathbf{w}) = p(\mathbf{w}, \{\mathbf{v}_1, \dots, \mathbf{v}_M\}) = p(\mathbf{w} | \{\mathbf{v}_1, \dots, \mathbf{v}_M\}) p(\{\mathbf{v}_1, \dots, \mathbf{v}_M\})$$

496 According to lemma A.3 sum of the weights of the groups is a multinomial and by lemma 3.2 it can  
497 be decomposed to a sequence of binomials:

$$p(\{\mathbf{v}_1, \dots, \mathbf{v}_M\}) = \text{Mu}(w, [\alpha_1, \dots, \alpha_M]) = \prod_{m=1}^M \text{Bi}(w - \sum_{i < m} \mathbf{v}_i, \hat{\eta}_m),$$

$$\text{where: } \alpha_m = \mathbf{1}^T \boldsymbol{\theta}_m, \hat{\eta}_e = \frac{\alpha_e}{1 - \sum_{i < e} \alpha_m}$$

498 Also based on lemma A.2, given the sum of the wights of all groups, the groups are independent and  
499 has multinomial distribution:

$$p(\mathbf{w} | \{\mathbf{v}_1, \dots, \mathbf{v}_M\}) = \prod_{m=1}^M \text{Mu}(\mathbf{v}_m, \boldsymbol{\lambda}_m)$$

$$\text{where: } \boldsymbol{\lambda}_m = \frac{\boldsymbol{\theta}_m}{\mathbf{1}^T \boldsymbol{\theta}_m}$$

500

□

## 501 B Graph Neural Network (GNN) architectures

502 To overcome limitations in the sparse message passing mechanism, Graph Transformers (GTs)  
503 (Dwivedi & Bresson, 2020) have emerged as a recent solution. One key advantage of GTs is the  
504 ability for nodes to attend to all other nodes in a graph, known as global attention, which addresses  
505 issues such as over-smoothing, over-squashing, and expressiveness bounds (Rampásek et al., 2022).  
506 GraphGPS provide a recipe for creating a more expressive and scalable graph transformer by making  
507 a hybrid message-passing graph neural networks (MPNN)+Transformer architecture. Additionally,  
508 recent GNN models propose to address the limitation of standard MPNNs in detecting simple  
509 substructures by adding features that they cannot capture on their own, such as the number of cycles.  
510 A framework for selecting and categorizing different types of positional and structural encodings,  
511 including *local*, *global*, and *relative* is provided in (Rampásek et al., 2022). Positional encodings, such  
512 as eigenvectors of the adjacency or Laplacian matrices, aim to indicate the spatial position of a node  
513 within a graph, so nodes that are close to each other within a graph or subgraph should have similar  
514 positional encodings. On the other hand, structural encodings, such as degree of a node, number of  
515 k-cycles a node belong to or the diagonal of the  $m$ -steps random-walk matrix, aim to represent the  
516 structure of graphs or subgraphs, so nodes that share similar subgraphs or similar graphs should have  
517 similar structural encodings.

518 In order to encode the node features of the augmented graphs in our model, we customized GraphGPS  
519 in various ways. We incorporated distinct initial edge features to distinguish augmented (candidate)  
520 edges from real edges. Furthermore, for bipartite generation, we apply a mask on the attention scores  
521 of the transformers of the augmented graph  $\mathcal{G}^l$  to restrict attention only to connected communities.  
522 Specifically, the  $i$ -th row of the attention mask matrix is equal to 1 only for the index of the nodes  
523 that belong to the same community or the nodes of the neighboring communities that are linked by a  
524 bipartite, and 0 (i.e., no attention to those positions) otherwise.

525 The time and memory complexity of GraphGPS can be reduced to  $\mathcal{O}(n + m)$  per layer by using  
526 *linear Transformers* such as Performer (Choromanski et al., 2020) for global graph attention, while  
527 they can be as high quadratic in the number of nodes if the original Transformer architecture is  
528 employed. Since our datasets are composed of graphs smaller than 1000 nodes, we leverage the  
529 original Transformer architecture.

## 530 C Experimental details

531 **Datasets:** For the benchmark datasetst, graph sizes, denoted as  $D_{dataset} =$   
532  $(|\mathcal{V}|_{max}, |\mathcal{V}|_{avg}, |\mathcal{E}|_{max}, |\mathcal{E}|_{avg})$ , are:  $D_{protein} = (500, 258, 1575, 646)$ ,  $D_{Ego} =$   
533  $(399, 144, 1062, 332)$ ,  $D_{Point-Cloud} = (5.03k, 1.4k, 10.9k, 3k)$ ,

Before training the models, we applied Louvain algorithm to obtain hierarchical graph structures for all of datasets and then trimmed out the intermediate levels to achieve uniform depth of  $L = 2$ . In case of  $\mathcal{HG}$ s with varying heights, empty graphs can be added at the root levels of those  $\mathcal{HG}$ s with lower heights to avoid sampling them during training. An 80%-20% split was randomly created for training and testing and 20% of the training data was used for validation purposes.

**Model Architecture:** In our experiments, the GraphGPS models consisted of 8 layers, while each level of hierarchical model has its own GNN parameters. The input node features were augmented with positional and structural encodings, which included the first 8 eigenvectors corresponding to the smallest non-zero eigenvalues of the Laplacian matrices and the diagonal of the random-walk matrix up to 8 steps. We leverage the original Transformer architecture for all detests except Point Cloud dataset which use Performer. The hidden dimensions were set to 64 for the Protein, Ego, and Point Cloud datasets, and 128 for the Stochastic Block Model and Enzyme datasets. The number of mixtures was set to  $K=20$ .

In comparison, the GRAN models utilized 7 layers of GNNs with hidden dimensions of 128 for the Stochastic Block Model, Ego, and Enzyme datasets, 256 for the Point Cloud dataset, and 512 for the Protein dataset. Despite having smaller model sizes, HiGen achieved better performance than GRAN.

For training, the HiGen models used the Adam optimizer (Kingma & Ba (2014)) with a learning rate of  $5e-4$  and default settings for  $\beta_1$  (0.9),  $\beta_2$  (0.999), and  $\epsilon$  ( $1e-8$ ).

The experiments for the Enzyme and Stochastic Block Model datasets were conducted on a MacBook Air with an M2 processor and 16GB RAM, while the rest of the datasets were trained using an NVIDIA L4 Tensor Core GPU with 24GB RAM as an accelerator.

## D Additional Results

Table 2 presents the results of various metrics for HiGen models on all benchmark datasets. The structural statistics are evaluated using the Total Variation kernel as the Maximum Mean Discrepancy (MMD) metric.

In addition, the table includes the average of random-GNN-based metrics (Thompson et al. (2022)) over 10 random Graph Isomorphism Network (GIN) initializations. The reported metrics are MMD with RBF kernel (GNN MMD), the harmonic mean of improved precision+recall (GNN F1 PR) and harmonic mean of density+coverage (GNN F1 DC).

Table 2: Various graph generative performance metrics for HiGen models on all benchmark datasets.

Model	Deg. ↓	Clus. ↓	Orbit ↓	Spec. ↓	GNN MMD ↓	GNN F1 PR ↑	GNN F1 DC ↑
<i>Enzyme</i>							
HiGen-m	6.61e-03	2.65e-02	2.15e-03	8.75e-03	2.15e-02	9.70e-01	8.97e-01
HiGen	2.31e-03	2.08e-02	1.51e-03	9.56e-03	1.80e-02	9.78e-01	9.83e-01
<i>Protein</i>							
HiGen-m	0.0041	0.109	0.0472	0.0061	6.71e-02	9.79e-01	9.85e-01
HiGen	0.0012	0.0435	0.0234	0.0025	6.71e-02	9.79e-01	9.85e-01
<i>Stochastic block model</i>							
HiGen-m	0.0017	0.0503	0.0604	0.0068	1.54e-01	9.12e-01	0.83
HiGen	0.0019	0.0498	0.0352	0.0046	4.32e-02	9.86e-01	1.07
<i>Ego</i>							
HiGen-m	0.011	0.063	0.021	0.013	4.20e-02	0.87	0.68
HiGen	1.9e-3	0.049	0.029	0.004	5.20e-02	0.88	0.69

### D.1 Point Cloud

We also evaluated HiGen on the *Point Cloud* dataset, which consists of 41 simulated 3D point clouds of household objects. This dataset consists of large graphs of approximately 1.4k nodes on average with maximum of over 5k nodes. In this dataset, each point is mapped to a node in a graph, and edges are connecting the k-nearest neighbors based on Euclidean distance in 3D space (Neumann et al. (2013)).



569 However, due to the quadratic growth of the number of candidate edges in the augmented graph  $\hat{\mathcal{G}}^l$  –  
570 the graph composed of all the communities and the candidate edges of all bipartites used in section  
571 3.2 for bipartite generation – memory limitations can arise when dealing with large graphs in the  
572 point cloud dataset. To address this issue, we can sample sub-graphs and generate one (or a subset of)  
573 bipartites at a time to fit the available memory. In our experimental study, we generated bipartites  
574 sequentially, sorting them based on the index of their parent edges in the parent level. In this case,  
575 the augmented graph  $\hat{\mathcal{G}}^l$  used for obtaining the node features of  $\mathcal{B}_{ij}^l$  consists of all the communities  
576  $\{C_k^l \mid \forall k \leq \max(i, j)\}$  and all the bipartites  $\{\mathcal{B}_{mn}^l \mid \forall (m, n) \leq (i, j)\}$ , augmented with the candidate  
577 edges of  $\mathcal{B}_{ij}^l$ . This model is denoted by HiGen-s in table 3.

578 The GraphGPS models that was used for this experiment have employed Performer (Choromanski  
579 et al., 2020) which offers linear time and memory complexity. The results in Table 3 highlights the  
580 performance improvement of HiGen-s in both local and global properties of the generated graphs.

Table 3: Comparison of generation metrics on benchmark 3D point cloud. The baseline results are obtained from (Liao et al., 2019).

Model	3D Point Cloud			
	Deg. ↓	Clus. ↓	Orbit ↓	Spec. ↓
<b>Erdos-Renyi</b>	3.1e-01	1.22	1.27	4.26e-02
<b>GRAN</b>	<b>1.75e-02</b>	5.1e-01	2.1e-01	7.45e-03
<b>HiGen-s</b>	3.48e-02	<b>2.82e-01</b>	<b>3.45e-02</b>	<b>5.46e-03</b>

581 An alternative approach is to sub-sample a large graph such that each augmented sub-graph consists  
582 of a bipartite  $\mathcal{B}_{ij}^l$  and its corresponding pair of communities  $C_i^l, C_j^l$ . This approach allows for  
583 parallel generation of bipartite sub-graphs but does not consider the connectivity between neighboring  
584 bipartites.

## 585 D.2 Ablation studies

586 In this section, two ablation studies were conducted to evaluate the sensitivity of HiGen with different  
587 node orderings and graph partitioning functions.

588 **Node Ordering** In our experimental study, the nodes in the communities of all levels are ordered  
589 using breadth first search (BFS) node ordering while the BFS queue are sorted by the total weight  
590 of edges between a node in the queue and predecessor nodes plus its self-edge. To compare the  
591 sensitivity of the proposed generative model against GRAN, we trained the models with default node  
592 ordering and random node ordering. The performance results, presented in Table 4, confirm that  
593 the proposed model is significantly less sensitive to the node ordering whereas the performance of  
594 GRAN drops considerably with non-optimal orderings.

Table 4: Ablation study on node ordering. Baseline HiGen used the BFS ordering and baseline GRAN used DFS ordering.  $\pi_1, \pi_2$  and  $\pi_3$  are default, random and  $\pi_3$  node ordering, respectively. Total variation kernel is used as MMD metrics of structural statistics. Also, the average of random-GNN-based metrics over 10 random GIN initialization are reported for MMD with RBF kernel (GNN MMD), the harmonic mean of improved precision+recall (GNN F1 PR) and harmonic mean of density+coverage (GNN F1 DC).

Model	Enzyme						
	Deg. ↓	Clus. ↓	Orbit ↓	Spec. ↓	GNN MMD ↓	GNN F1 PR ↑	GNN F1 DC ↑
<b>GRAN</b>	8.45e-03	2.62e-02	3.46e-02	2.11e-02	6.63e-02	9.50e-01	8.32e-01
<b>GRAN (<math>\pi_1</math>)</b>	1.75e-02	2.89e-02	3.78e-02	2.03e-02	6.51e-02	8.24e-01	6.69e-01
<b>GRAN (<math>\pi_2</math>)</b>	3.90e-02	3.24e-02	3.81e-02	2.38e-02	1.26e-01	8.31e-01	6.72e-01
<b>HiGen</b>	2.31e-03	2.08e-02	1.51e-03	9.56e-03	1.80e-02	9.78e-01	9.83e-01
<b>HiGen (<math>\pi_1</math>)</b>	1.83e-03	2.21e-02	6.75e-04	7.08e-03	1.78e-02	9.84e-01	9.77e-01
<b>HiGen (<math>\pi_2</math>)</b>	3.31e-03	2.34e-02	2.06e-03	9.10e-03	2.04e-02	9.47e-01	8.81e-01
<b>HiGen (<math>\pi_3</math>)</b>	1.34e-03	2.13e-02	6.94e-04	6.56e-03	1.90e-02	9.61e-01	9.74e-01

595 **Different Graph Partitioning** In this experimental study, we evaluated the performance of HiGen  
596 using different graph partitioning functions. Firstly, to assess the sensitivity of the hierarchical gener-

597 ative model to random initialization in the Louvain algorithm, we conducted the HiGen experiment  
 598 three times with different random seeds on the Enzyme dataset. The average and standard deviation  
 599 of performance metrics are reported in Table 5 which demonstrate that HiGen consistently achieves  
 600 almost similar performance across different random initializations.

601 Additionally, we explored spectral clustering (SC), which is a relaxed formulation of  $k$ -min-cut  
 602 partitioning (Shi & Malik, 2000), as an alternative partitioning method. To determine the number  
 603 of clusters, we applied SC to partition the graphs over a range of  $0.7\sqrt{n} \leq k \leq 1.3\sqrt{n}$ , where  $n$   
 604 represents the number of nodes in the graph. We computed the modularity score of each partition and  
 605 selected the value of  $k$  that yielded the maximum score.

606 The results presented in Table 5 demonstrate the robustness of HiGen against different graph parti-  
 607 tioning functions.

Table 5: Multiple initialization of Louvain partitioning algorithm and also min-cut partitioning

Model	Deg. ↓	Clus. ↓	Orbit↓	Enzyme Spec. ↓	GNN MMD ↓	GNN F1 PR ↑	GNN F1 DC ↑
HiGen	$2.64e-03 \pm 4.7e-4$	$2.09e-02 \pm 4.0e-4$	$7.46e-04 \pm 4.4e-4$	$1.74e-02 \pm 1.5e-3$	$2.00e-02 \pm 3.1e-3$	$.98 \pm 4.6e-3$	$.96 \pm 1.0e-2$
HiGen (SC)	2.24e-03	2.10e-02	5.59e-04	8.30e-03	2.00e-02	.98	.94

### 608 D.3 Graph Samples

609 Generated hierarchical graphs sampled from HiGen models are presented in this section.

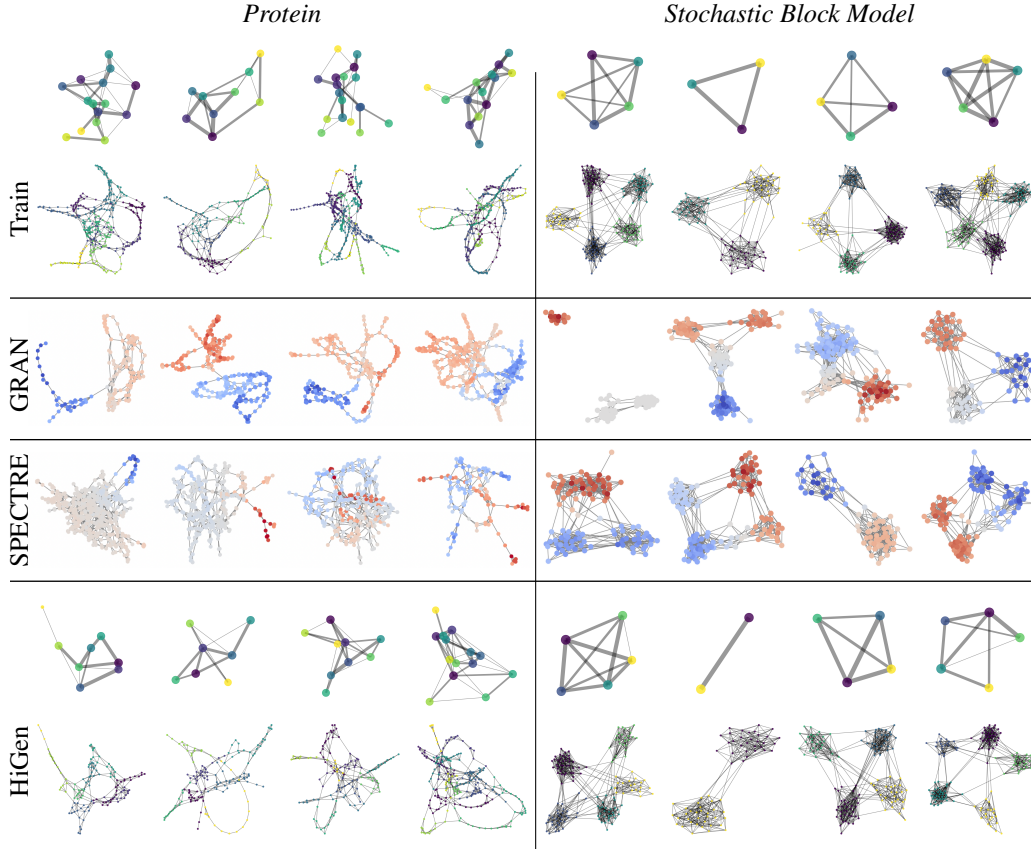


Figure 3: Samples from HiGen trained on *Protein* and *SBM*. Communities are distinguished with different colors and both levels are depicted. The samples for GRAN and SPECTRE are obtained from (Martinkus et al., 2022).

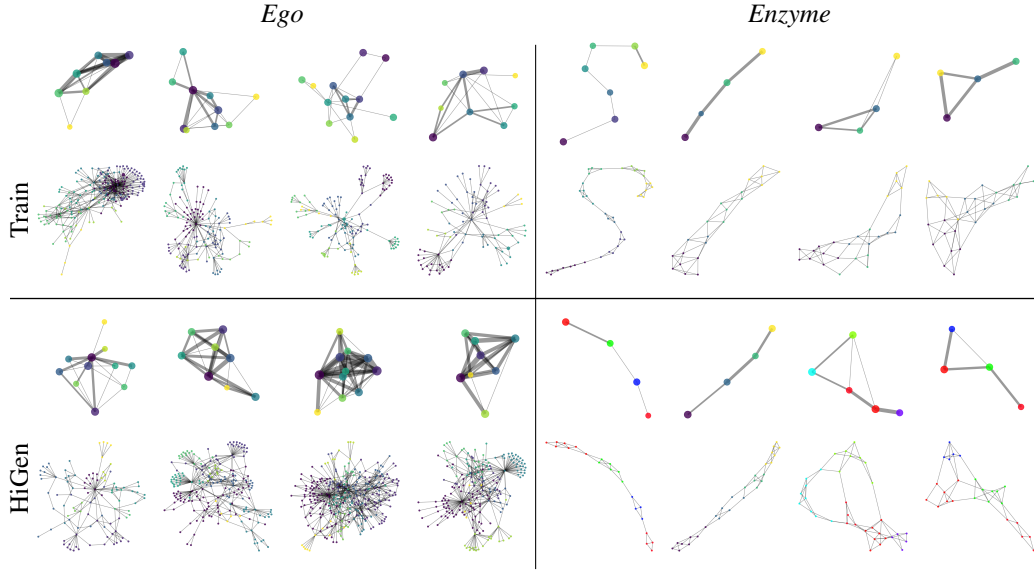


Figure 4: Samples from HiGen trained on *Protein* and *SBM*. Communities are distinguished with different colors and both levels are depicted.

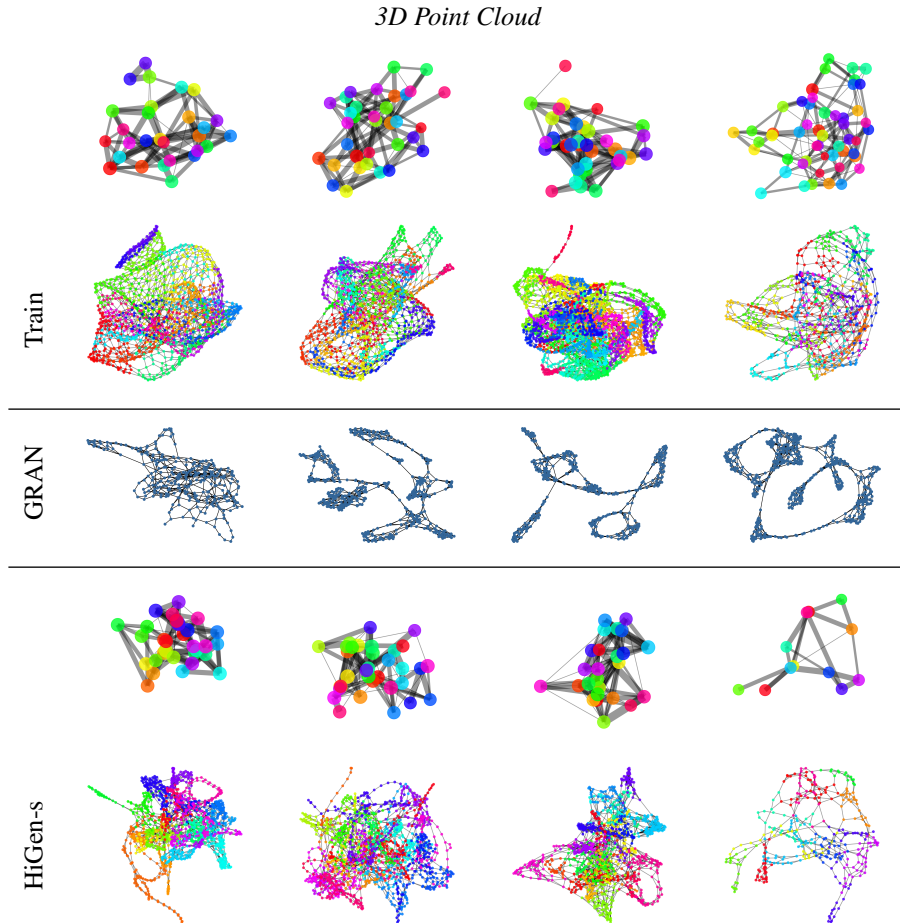


Figure 5: Samples from HiGen trained on *3D Point Cloud*. Communities are distinguished with different colors and both levels are depicted. The samples for GRAN are obtained from [Liao et al. 2019](#).