Order Matters: Probabilistic Modeling of Node Sequence for Graph Generation

Xiaohui Chen [*1] Xu Han [*1] Jiajing Hu [1]
Francisco J. R. Ruiz [2] Liping Liu [1]

[1]Tufts University, US, [2]DeepMind, UK
[*]Equal Contribution

xiaohui.chen@tufts.edu

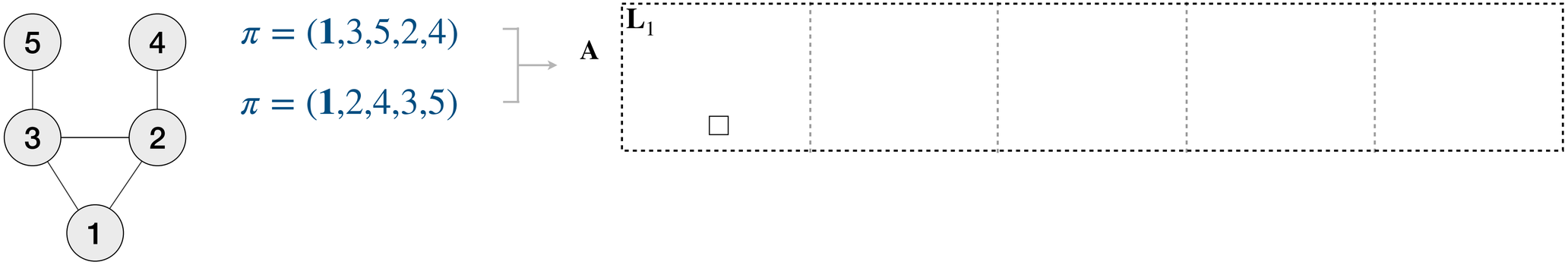
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]

   Consider **noder orderings** $(1,3,5,2,4)$ and $(1,2,4,3,5)$ and adjacency matrix $A = L + L^T$

$$P(A) = P(\otimes \mid L) \prod_{t=2}^{n} P(L_{t,:} \mid L_{1:(t-1)})$$



Graph G Node ordering π

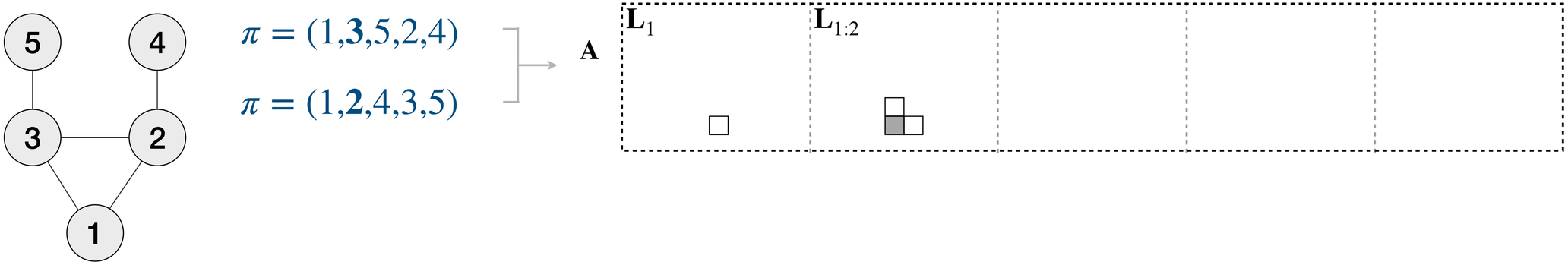
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]

   Consider **noder orderings** $(1,3,5,2,4)$ and $(1,2,4,3,5)$ and adjacency matrix $A = L + L^T$

$$P(A) = P(\otimes \mid L) \prod_{t=2}^{n} P(L_{t,:} \mid L_{1:(t-1)})$$



Graph G Node ordering π

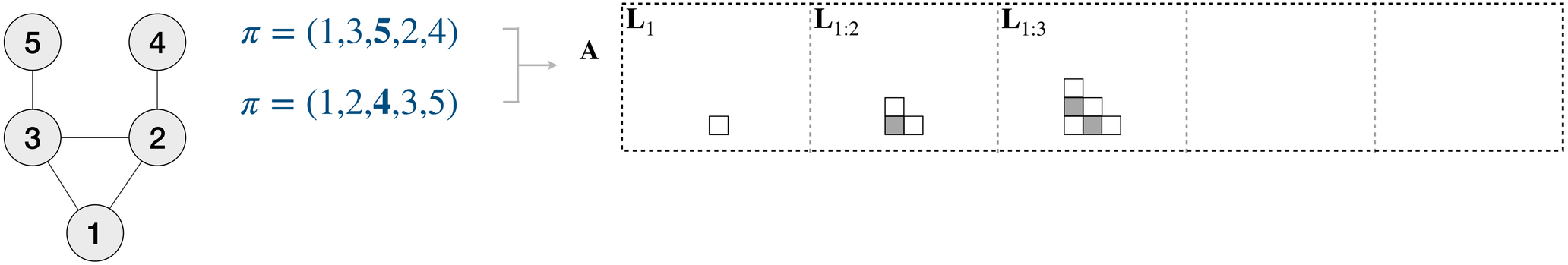
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]

   Consider **noder orderings** $(1,3,5,2,4)$ and $(1,2,4,3,5)$ and adjacency matrix $A = L + L^T$

$$P(A) = P(\otimes \mid L) \prod_{t=2}^{n} P(L_{t,:} \mid L_{1:(t-1)})$$



Graph G Node ordering π

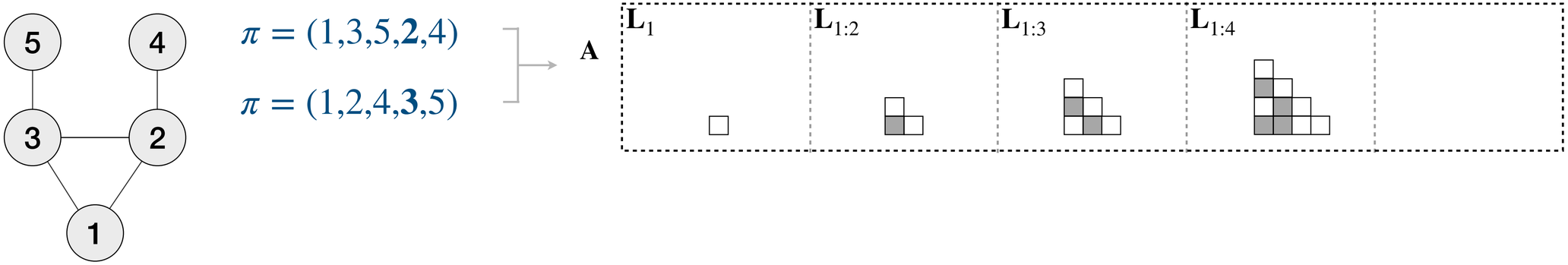
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]

   Consider **noder orderings** $(1,3,5,2,4)$ and $(1,2,4,3,5)$ and adjacency matrix $A = L + L^T$

$$P(A) = P(\otimes \mid L) \prod_{t=2}^{n} P(L_{t,:} \mid L_{1:(t-1)})$$



Graph G Node ordering π

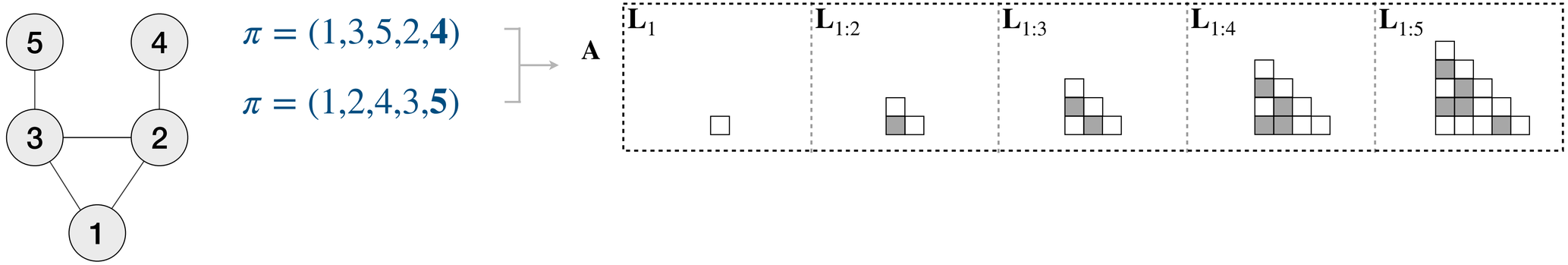
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]

   Consider **noder orderings** $(1,3,5,2,4)$ and $(1,2,4,3,5)$ and adjacency matrix $A = L + L^T$

$$P(A) = P(\otimes \mid L) \prod_{t=2}^{n} P(L_{t,:} \mid L_{1:(t-1)})$$



Graph G Node ordering π

xiaohui.chen@tufts.edu

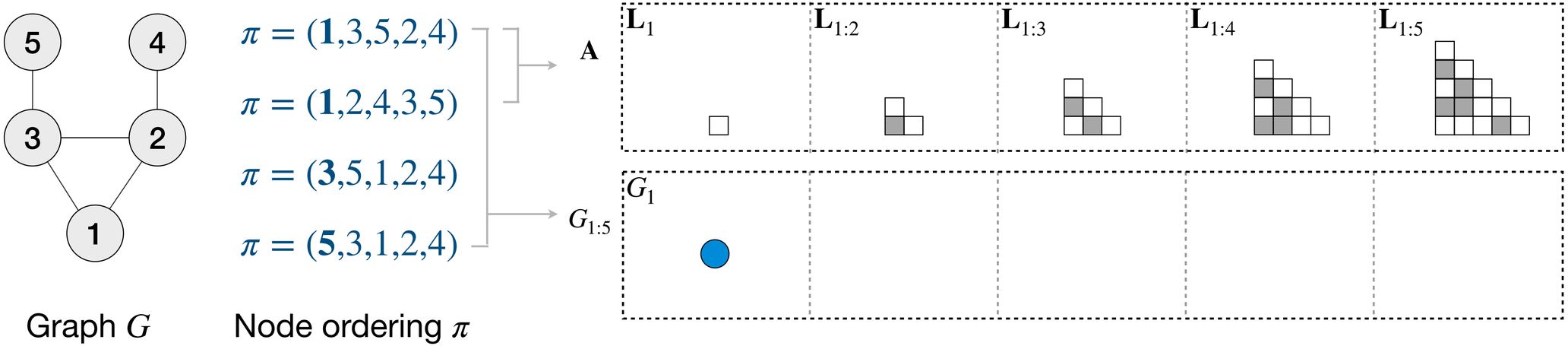
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]
2. Based on graph sequence [Li et al., 2018]

   Consider **node orderings** $(1,3,5,2,4)$, $(1,2,4,3,5)$, $(3,5,1,2,4)$, $(5,3,1,2,4)$ and graph sequence $G_{1:5}$

$$P(G_{1:n}) = P(\otimes \mid G_n) \prod_{t=2}^{n} P(G_t \mid G_{t-1})$$



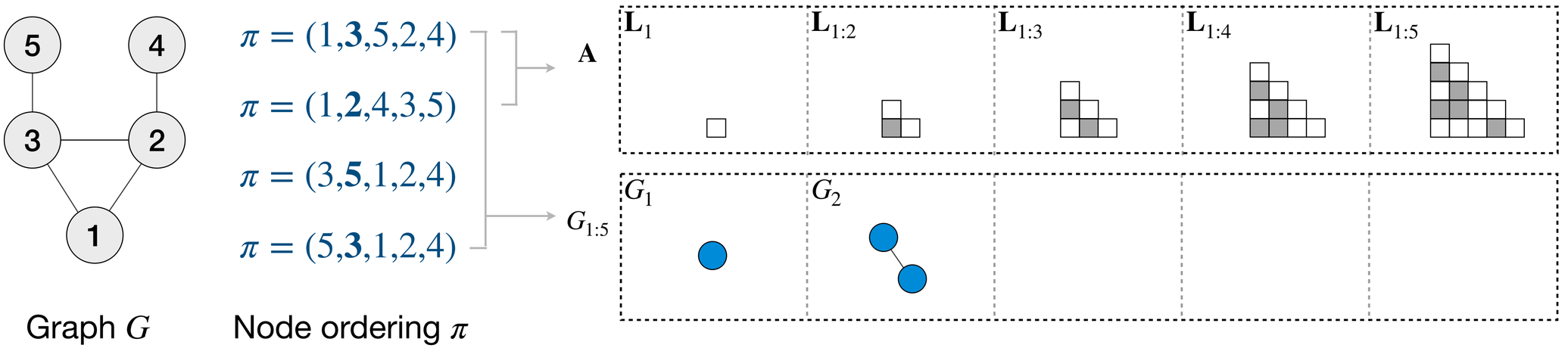
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]
2. Based on graph sequence [Li et al., 2018]

Consider **node orderings** $(1,3,5,2,4)$, $(1,2,4,3,5)$, $(3,5,1,2,4)$, $(5,3,1,2,4)$ and graph sequence $G_{1:5}$

$$P(G_{1:n}) = P(\otimes \mid G_n)\prod_{t=2}^{n} P(G_t \mid G_{t-1})$$



xiaohui.chen@tufts.edu

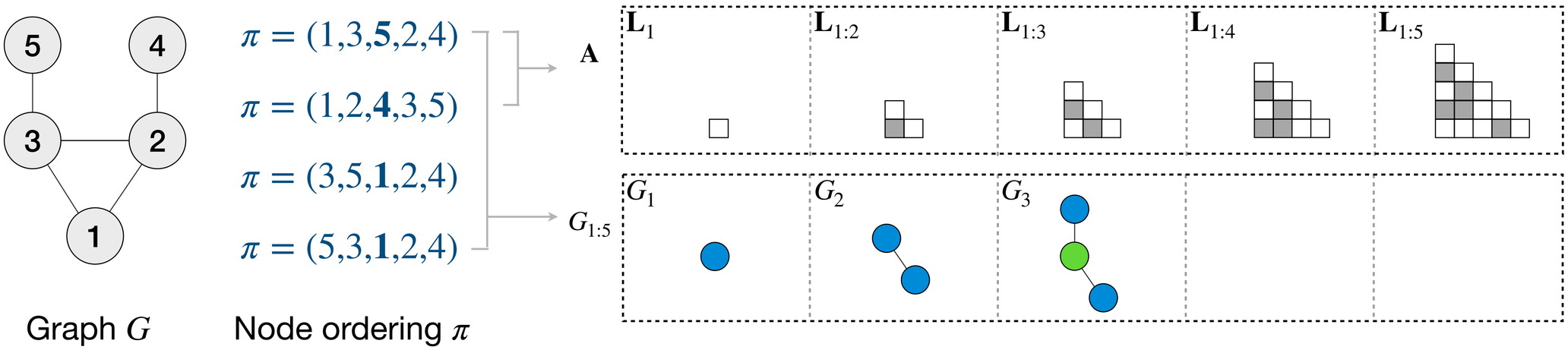
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]
2. Based on graph sequence [Li et al., 2018]

Consider **node orderings** $(1,3,5,2,4)$, $(1,2,4,3,5)$, $(3,5,1,2,4)$, $(5,3,1,2,4)$ and graph sequence $G_{1:5}$

$$P(G_{1:n}) = P(\otimes \mid G_n)\prod_{t=2}^{n} P(G_t \mid G_{t-1})$$



Graph G Node ordering π

xiaohui.chen@tufts.edu

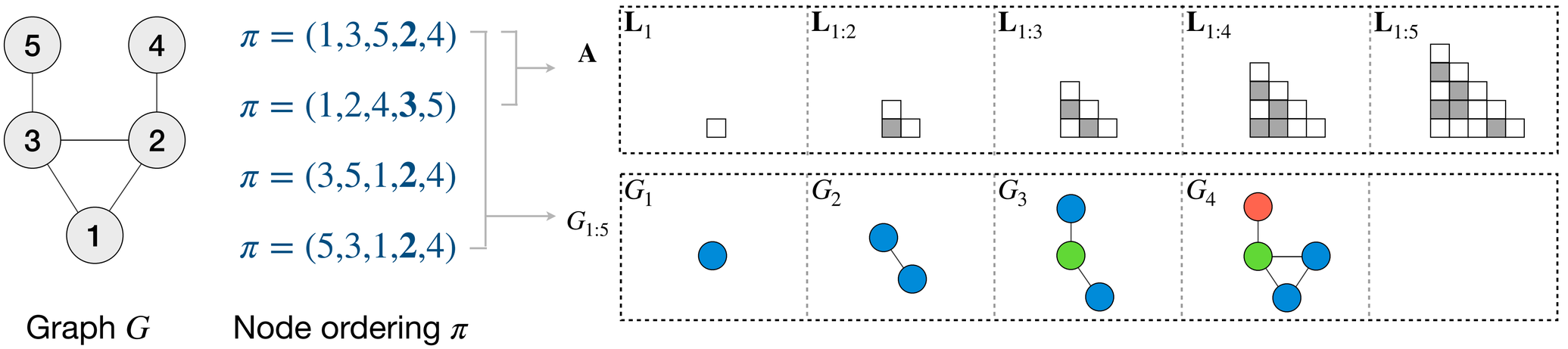
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]
2. Based on graph sequence [Li et al., 2018]

Consider **node orderings** $(1,3,5,2,4)$, $(1,2,4,3,5)$, $(3,5,1,2,4)$, $(5,3,1,2,4)$ and graph sequence $G_{1:5}$

$$P(G_{1:n}) = P(\otimes \mid G_n)\prod_{t=2}^{n} P(G_t \mid G_{t-1})$$



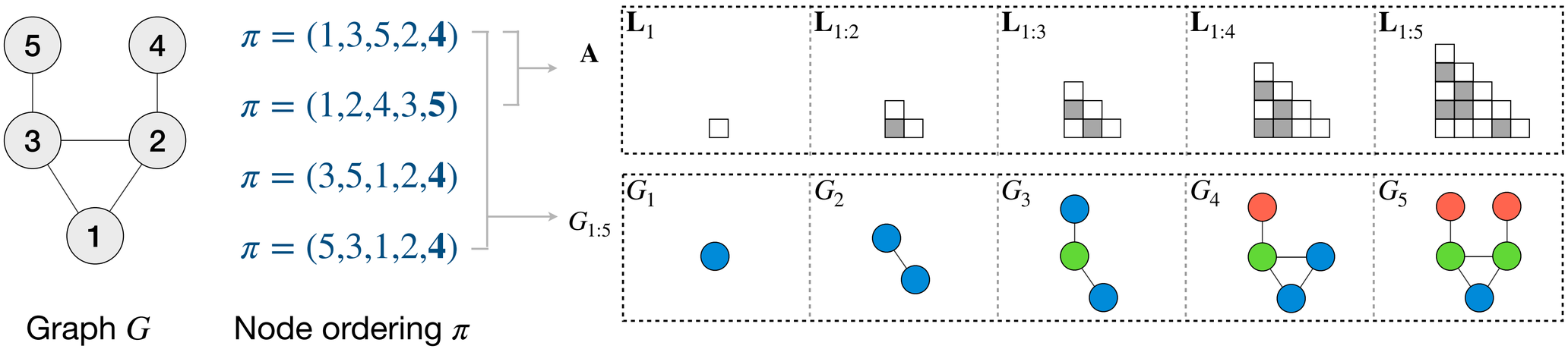
Problem of Autoregressive Graph Generation

Two types of autoregressive models:

1. Based on adjacency matrix [You et al., 2018; Liao et al., 2019; Shi et al., 2020; Goyal et al., 2020]
2. Based on graph sequence [Li et al., 2018]

Consider **node orderings** $(1,3,5,2,4)$, $(1,2,4,3,5)$, $(3,5,1,2,4)$, $(5,3,1,2,4)$ and graph sequence $G_{1:5}$

$$P(G_{1:n}) = P(\otimes \mid G_n)\prod_{t=2}^{n} P(G_t \mid G_{t-1})$$



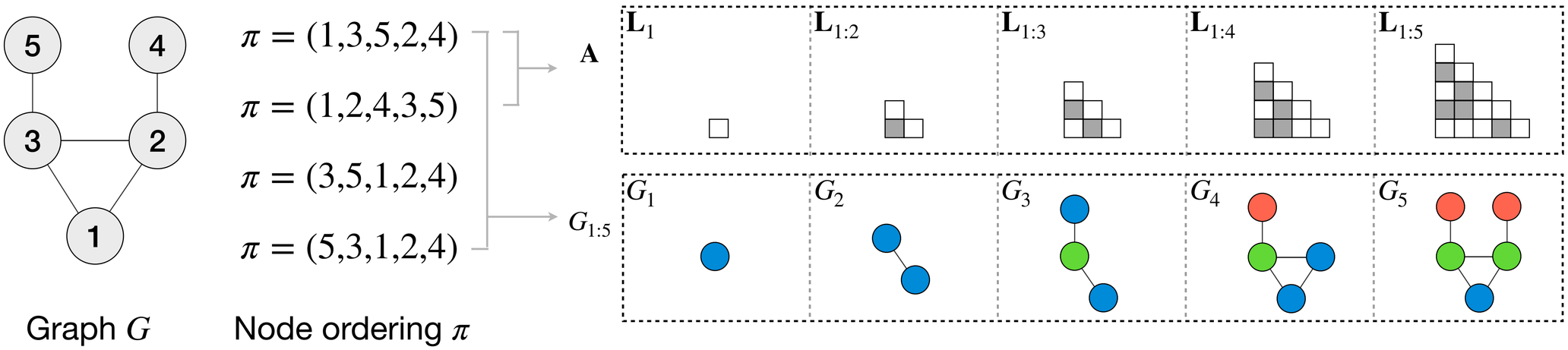
Graph G Node ordering π

xiaohui.chen@tufts.edu

Problem of Autoregressive Graph Generation

Observations:

1. A graph G does not naturally have a unique adjacency matrix A or sequence G
   - For that, we need to choose a node ordering π
   - A tuple (G, π) uniquely determines $A/G_{1:n}$, then we can fit the likelihood $p(A)/p(G_{1:n})$
2. There are multiple (G, π)-s leading to the same $A/G_{1:n}$ due to the **graph automorphism**
3. **An autoregressive generative model (which generates $A/G_{1:n}$) does not specify a distribution over π**



Graph G Node ordering π

Bridging Node Ordering π and $A/G_{1:n}$

- To fit a graph model via MLE, we need the marginal likelihood

$$P(G) = \sum_{\mathbf{A} \in \mathscr{A}(G)} P(\mathbf{A}) \qquad p(G) = \sum_{G_{1:n}:\ G_n = G} p(G_{1:n})$$

- The marginalization space of $A/G_{1:n}$ is very hard to characterize, while the space of π is very easy

$$P(G) = \sum_{(G,\pi)} P(G, \pi)$$

- Relation between $P(A)/P(G_{1:n})$ and $P(G, \pi)$

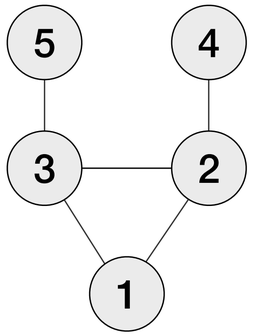
$$P(G, \pi) = \frac{1}{|\Pi[A]|} P(A); \quad |\Pi[A]| = \text{number of graph automorphism}$$

$$= \frac{1}{|\Pi[G_{1:n}]|} P(G_{1:n}); \quad |\Pi[G_{1:n}]| = \sum_{i=1}^{n} \text{orbit count of target node } i \text{ at } G_i$$

Bridging Node Ordering π and $A/G_{1:n}$: Example

$$P(G,\pi) = \frac{1}{|\Pi[A]|}P(A) = \frac{1}{2}P(A)$$

$$|\Pi[A]| = |\mathrm{Aut}(G)| = 2$$



Graph G

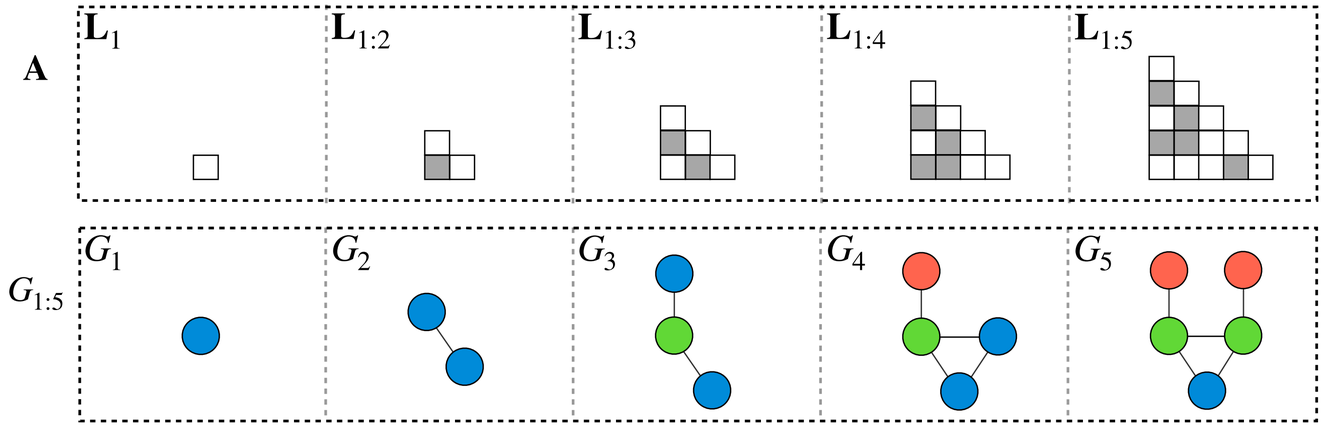
$\pi = (1,3,5,2,4)$

$\pi = (1,2,4,3,5)$

$\pi = (3,5,1,2,4)$

$\pi = (5,3,1,2,4)$

Node ordering π

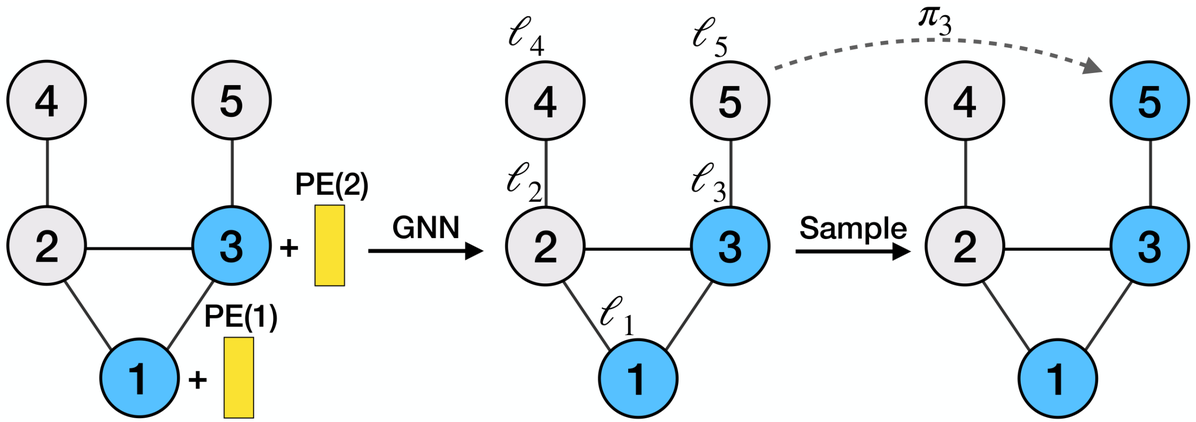


$$|\Pi[G_{1:n}]| = 1 \times 2 \times 2 \times 1 \times 2 = 8$$

$$P(G,\pi) = \frac{1}{|\Pi[G_{1:n}]|}P(G_{1:n}) = \frac{1}{8}P(G_{1:n})$$

Optimizing the Node ordering

- Introduce a variational distribution $q(\pi \mid G)$ to approximate $p(\pi \mid G)$
  - Parameterizing $q_\phi(\pi \mid G)$ using GNN
  - sample node recurrently to generate node ordering π



- Maximize ELBO w.r.t generative model θ and variational parameters ϕ

$$L(\theta, \phi, G) = \mathrm{E}_{q_\phi(\pi|G)}[\log p_\theta(G, \pi) - \log q_\phi(\pi \mid G)]$$

Optimizing the Node ordering

Algorithm 1 VI algorithm for training a graph model based on the adjacency matrix $\mathbf{A}$

Input: Dataset of graphs $\mathcal{G} = \{G_1, \ldots, G_n\}$, model p_θ, variational distribution q_ϕ, sample size S
Output: Learned parameters θ and ϕ
repeat
  for $G \in \mathcal{G}$ **do**
    Sample $\pi^{(1)}, \ldots, \pi^{(S)} \overset{\text{iid}}{\sim} q_\phi(\pi|G)$
    Obtain $\mathbf{A}^{(s)}$ from $(G, \pi^{(s)})$
    Set $p_\theta(G, \pi^{(s)}) = \frac{1}{|\Pi[\mathbf{A}^{(s)}]|} p_\theta(\mathbf{A}^{(s)})$
    Compute $\nabla_\phi \leftarrow \nabla_\phi L(\theta, \phi, G)$
    Compute $\nabla_\theta \leftarrow \nabla_\theta L(\theta, \phi, G)$
    Update ϕ, θ using the gradients ∇_ϕ, ∇_θ
  end for
until convergence of the parameters (θ, ϕ)

Experiment: Predictive Log-Likelihood

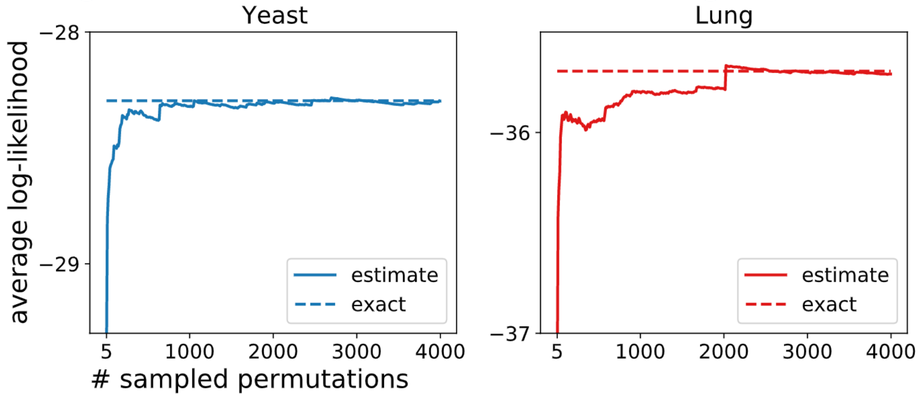
		Community-small log-like/ELBO	Citeseer-small log-like/ELBO	Enzymes log-like/ELBO	Lung log-like/ELBO	Yeast log-like/ELBO	Cora log-like/ELBO
DeepGMG	uniform	-206.2/-303.9	**-60.9/-67**	-281.9/-290.8	**-146.7/-225.7**	-115.1/128.9	-283.7/-295.2
	VI [ours]	**-124.8/-131.8**	**-59.6/-65.6**	**-145.8/-156.2**	**-146.1/-224.6**	**-105.4/-115.7**	**-227/-247.2**
GraphRNN	uniform	-154.6/-157.6	-101.9/-105.7	-340.3/-349.1	-232.4/ -242.2	-189.3/-200.1	-380.6/-401.8
	VI [ours]	**-53.7/-59.9**	**-89.6/-93.2**	**-274.9/-282.8**	**-155.9/-175.8**	**-109.1/-133.7**	**-345.3/-358.3**
GraphGEN	DFS	-263.74/NA	-73.0/NA	-574.2/NA	-140.1/NA	**-66.46/NA**	-199.5/NA
	VI [ours]	**-26.6/-35.0**	**-64.3/-71.1**	**-189.7/-213.8**	**-117.3/-125.5**	**-64.98/-72.39**	**-143.6/-152.3**

Approximate log-likelihood and ELBO of different generative models. For each model, we compare the default training algorithm with our method based on VI. The table shows

1) VI improves the model's predictive performance.

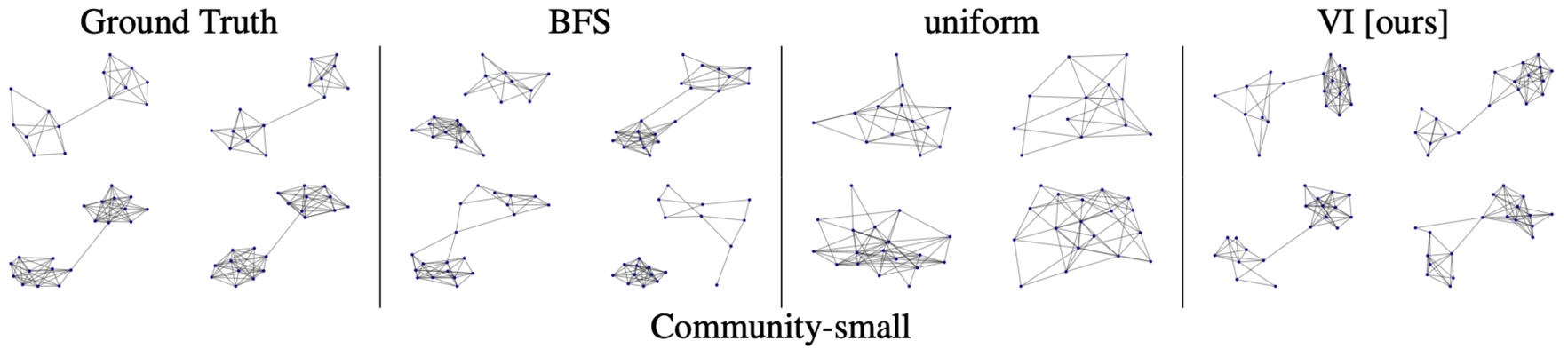
2) The variational bound is relatively tight.

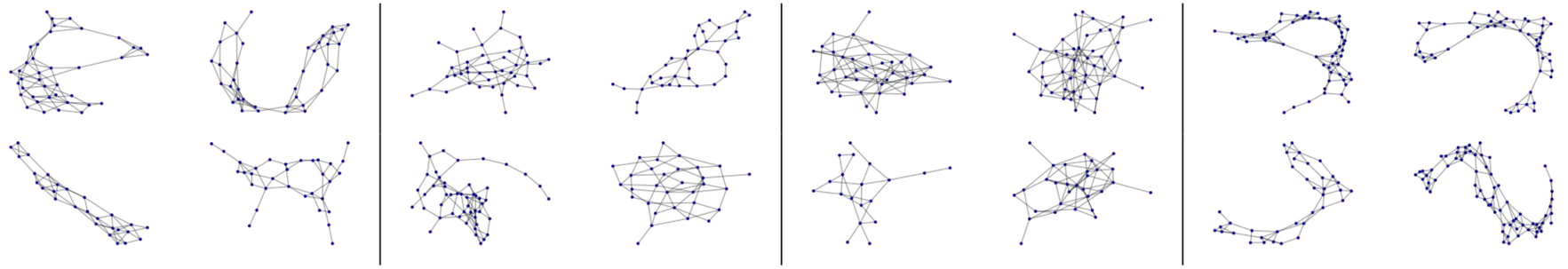
Tightness of the approximated log-likelihood

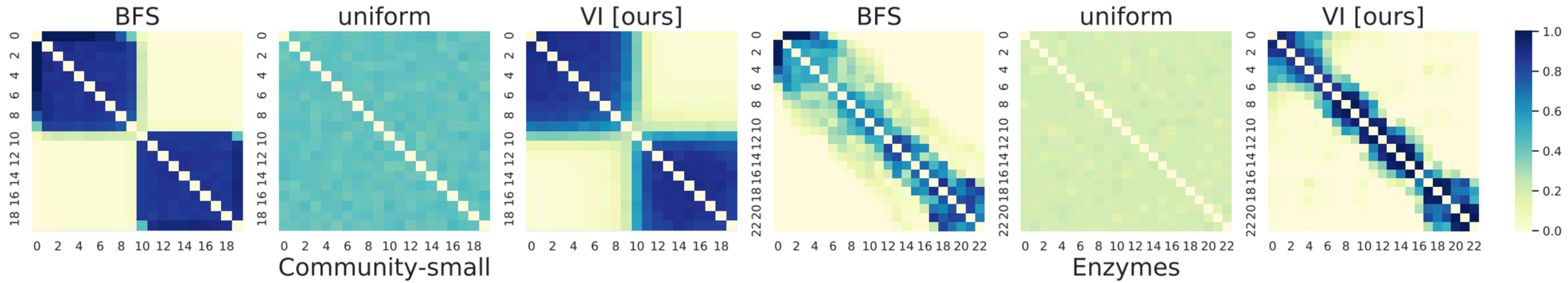


xiaohui.chen@tufts.edu

Experiment: Qualitative Analysis

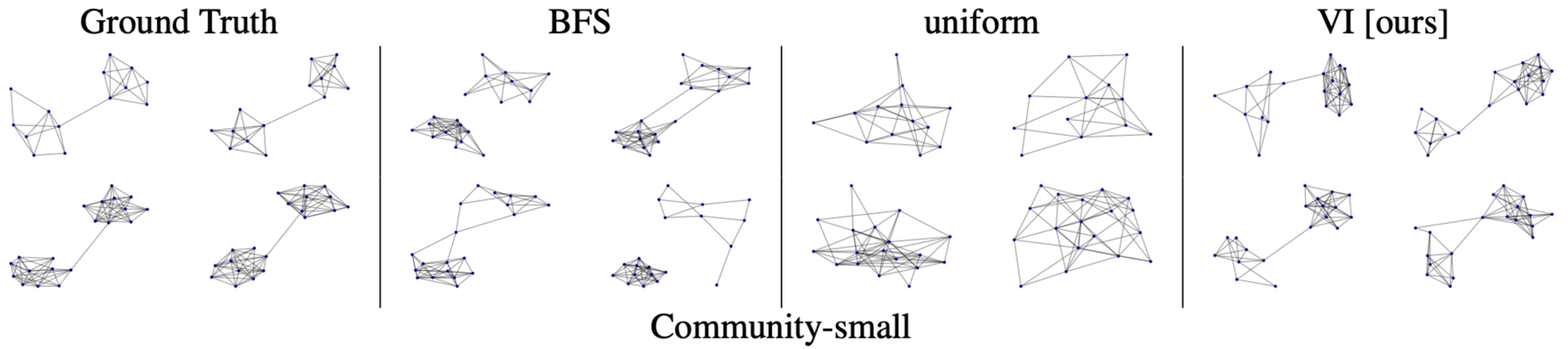


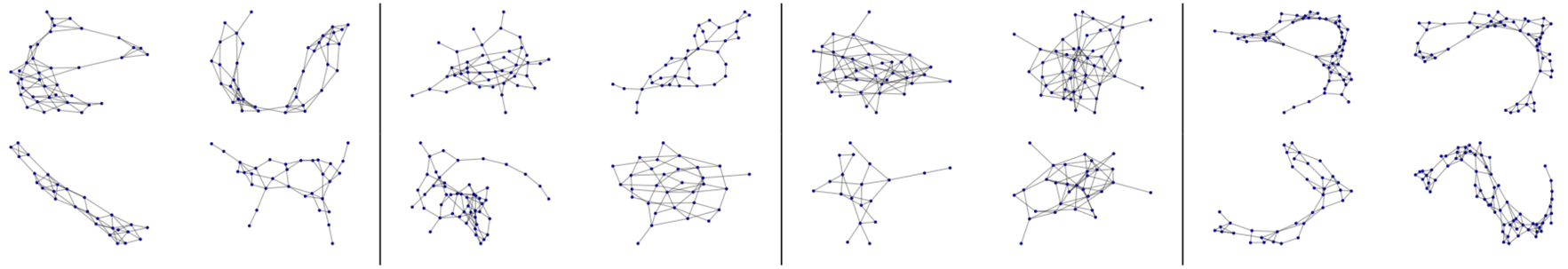


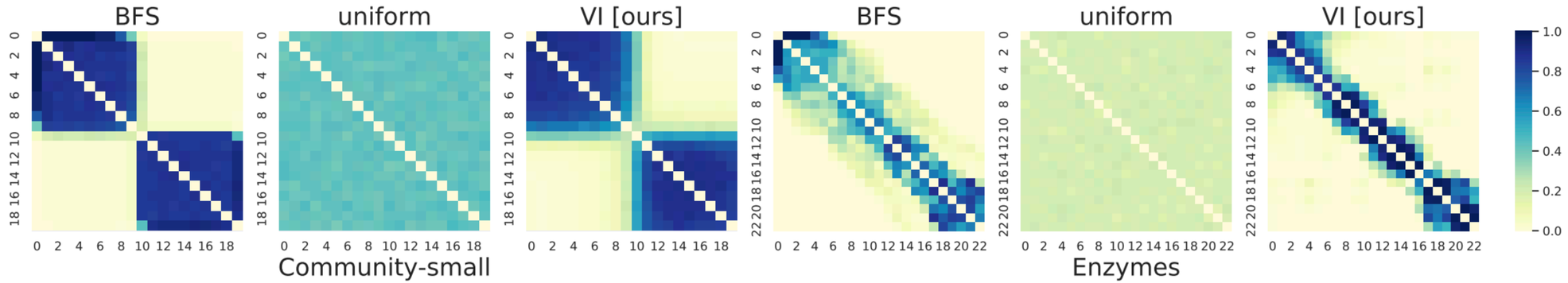


xiaohui.chen@tufts.edu

Experiment: Qualitative Analysis







xiaohui.chen@tufts.edu

Experiment: Quality of Generated Graphs

		Community-small			Citeseer-small			Enzymes		
		Deg.	Clus.	Orbit	Deg.	Clus.	Orbit	Deg.	Clus.	Orbit
DeepGMG	uniform	0.2	0.978	0.40	0.052	0.06	**0.005**	1.51	0.95	0.29
	VI [ours]	**0.178**	**0.921**	**0.338**	**0.028**	**0.014**	**0.005**	**1.01**	**0.48**	**0.27**
GraphRNN	BFS	0.034	0.11	0.009	0.016	**0.05**	0.004	0.03	0.085	0.043
	uniform	0.096	0.091	0.021	**0.009**	0.09	0.003	0.042	0.104	0.074
	VI [ours]	**0.018**	**0.01**	**0.008**	0.08	**0.05**	**0.002**	**0.015**	**0.067**	**0.02**
GraphGEN	DFS	0.695	0.931	0.178	0.047	**0.032**	0.017	0.716	0.456	0.078
	VI [ours]	**0.143**	**0.248**	**0.068**	**0.032**	0.078	**0.008**	**0.346**	**0.440**	**0.020**
		Lung			Yeast			Cora		
		Deg.	Clus.	Orbit	Deg.	Clus.	Orbit	Deg.	Clus.	Orbit
DeepGMG	uniform	0.206	**0.023**	0.224	0.547	0.242	0.470	**0.35**	0.27	0.11
	VI [ours]	**0.189**	**0.023**	**0.2**	**0.324**	**0.118**	**0.258**	0.36	**0.22**	**0.04**
GraphRNN	BFS	0.103	0.301	0.043	0.512	0.153	0.026	1.125	1.002	0.427
	uniform	1.213	**0.002**	0.081	0.746	0.351	0.070	0.188	0.206	0.200
	VI [ours]	**0.074**	0.060	**0.004**	**0.097**	**0.092**	**0.005**	**0.066**	**0.171**	**0.052**
GraphGEN	DFS	0.049	0.017	**0.000**	0.014	**0.003**	**0.000**	0.099	0.167	0.122
	VI [ours]	**0.022**	**0.008**	**0.000**	**0.012**	**0.003**	**0.000**	**0.056**	**0.103**	**0.069**

Summaries

Contributions

1. Analyzed autoregressive graph generative models
2. Provide an in-depth discussion of the automorphism issue that raises when calculating the marginal likelihood
3. Address the intractable marginalization over node orderings for fitting a graph generative model

Limitation

4. Computational speed

xiaohui.chen@tufts.edu

Thank You

xiaohui.chen@tufts.edu