Evolution and Future Prospects of Graph Generation Models

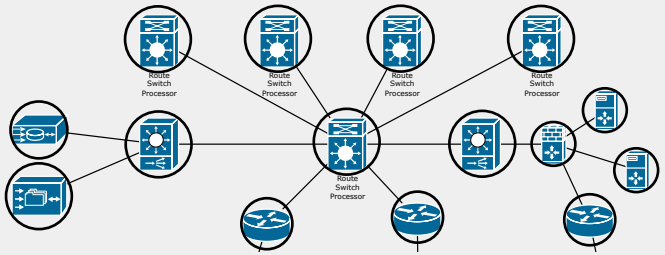
Graduate School of Engineering,
Nagaoka University of Technology

Associate Professor
Kohei Watabe

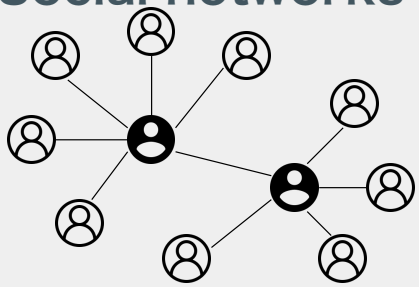
Presentation Overview

- A graph consisting of nodes and edges is an extremely versatile data structure.

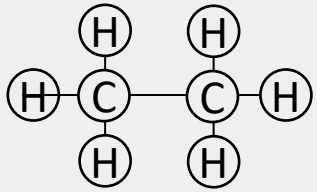
Communication networks



Social networks



Molecular structures



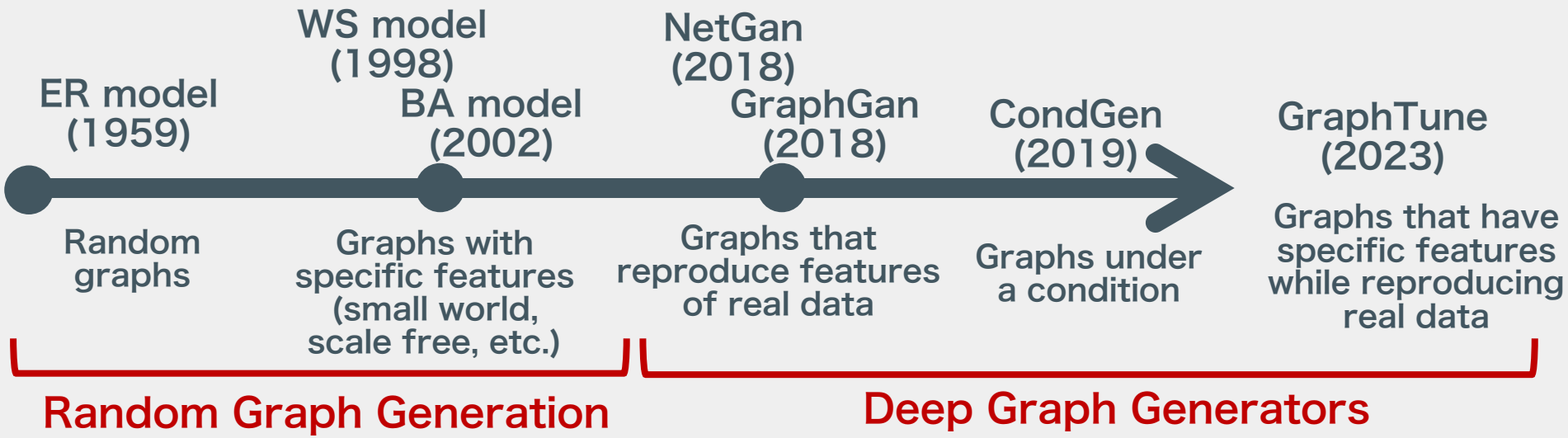
○ node (vertex)　　—— edge (link)

- The main topic of this talk is artificial graph generation techniques.
  - **Generating not present or future graph structures** from a small number of parameters or graph data.

- Applications of graph generation
  - Simulation of communication protocols
  - Information dissemination and community prediction in social networks
  - Code suggestion in programming
  - Development of drugs with novel molecular structures

History of Graph Generation Techniques

◆Graph generation techniques began with the ER model in 1959.



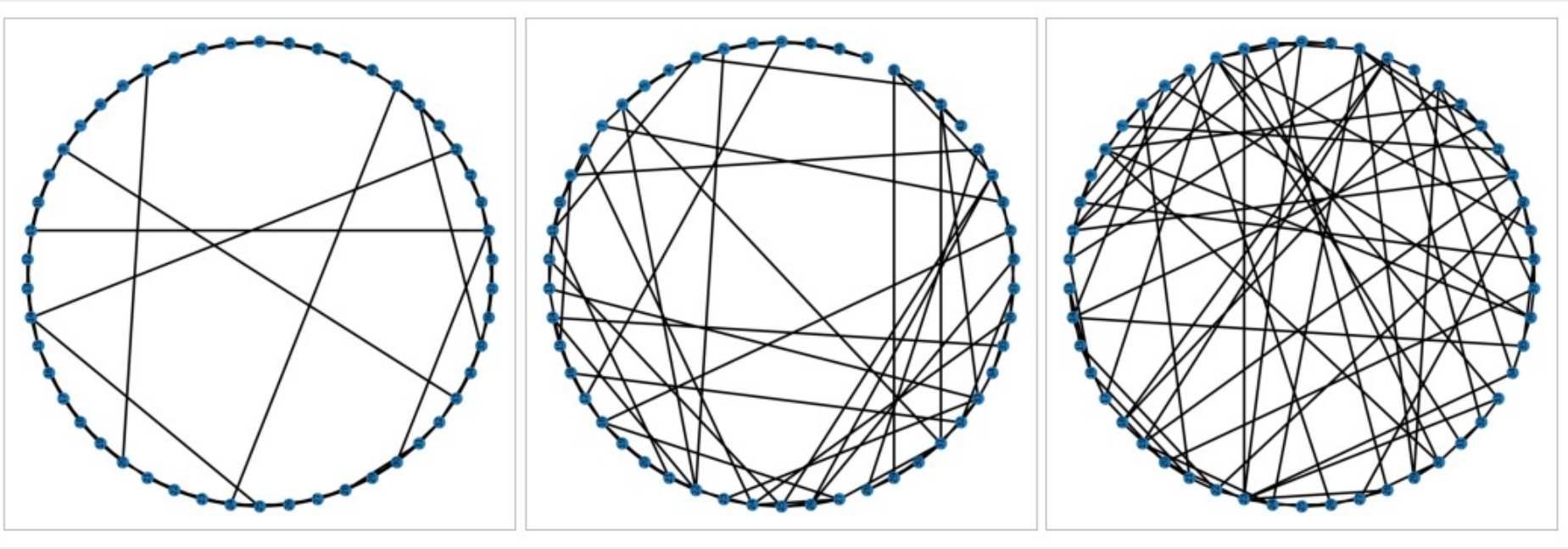
◆There are various features of graphs:

- ◆Average shortest path length, diameter
- ◆Clustering coefficient, modularity
- ◆Average degree, power-law exponent of degree distribution, edge density
- ◆Eigenvalues, degree centrality, betweenness centrality, PageRank

◆Common objective: **How to sample graphs with desired features from the huge space of graphs.**

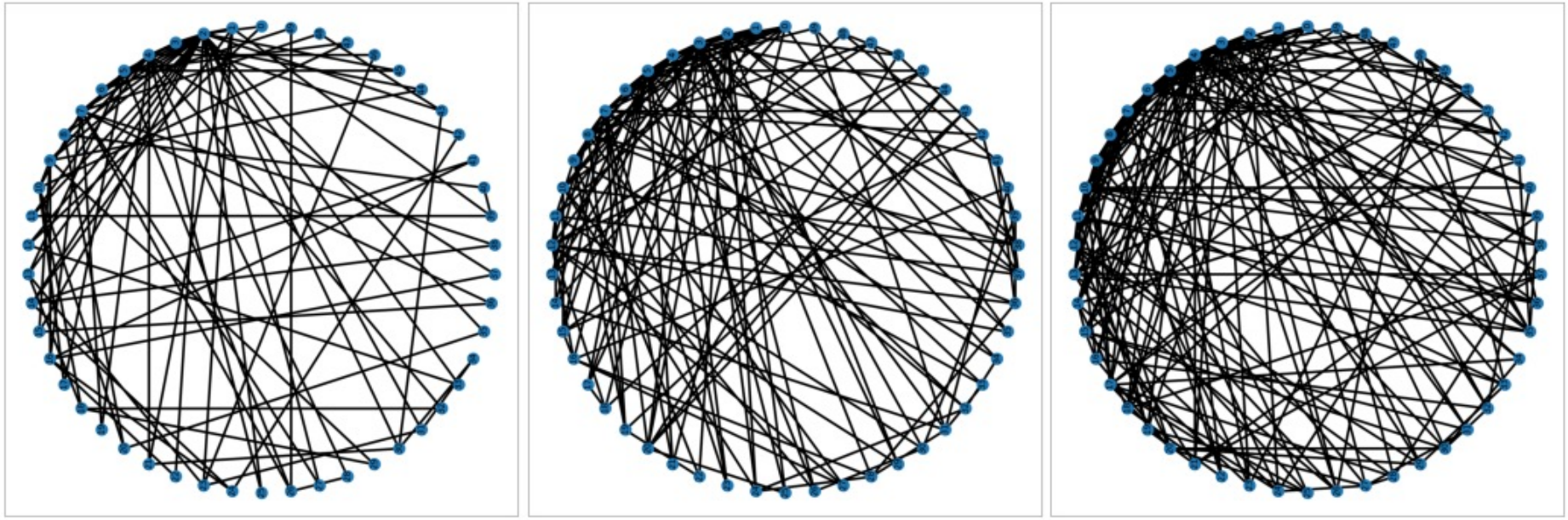
Random Generation Model – WS model

- Watts-Strogatz model is a graph generation model that **reproduce small-world properties**.
  - Small-world properties: For a number of nodes n, the average path length L increases at most logarithmically with n.
  - Input parameters: Number of nodes n, average degree $2K$, edge rewiring probability p
    1. Create n nodes.
    2. Construct a ring lattice with an average degree of $2K$.
    3. Rewire each edge with a probability of p.



Random Generative Model – BA model

- Barabási–Albert model is a graph generation model that **reproduce scale-free properties**.
  - Scale-free properties: The degree distribution $f(k)$ follows $f(k) \propto k^{-\gamma}$ $(2 \leq \gamma \leq 3)$.
  - Input parameters: Number of additional edges m , number of nodes n
    1. Create a complete graph consisting of m nodes.
    2. Add nodes with m m edges. However, the probability of an edge from the new node is proportional to the degree of the target node it connects to.
    3. Repeat Step 2 until the number of nodes reaches n.



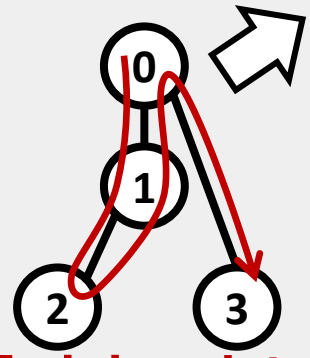
Deep Graph Generators – GraphGen

- Contrary to random graph generation, Deep Graph Generators **reproduce every features of real graphs**.
- Generated using Long Short Term Memory (LSTM) in deep learning, capable of sequence prediction.

1. Converting the training graphs into a sequence of edges.
2. Inputting sequence data and training the model to predict the subsequent edge.
3. Using the trained model recursively to generate the sequence.

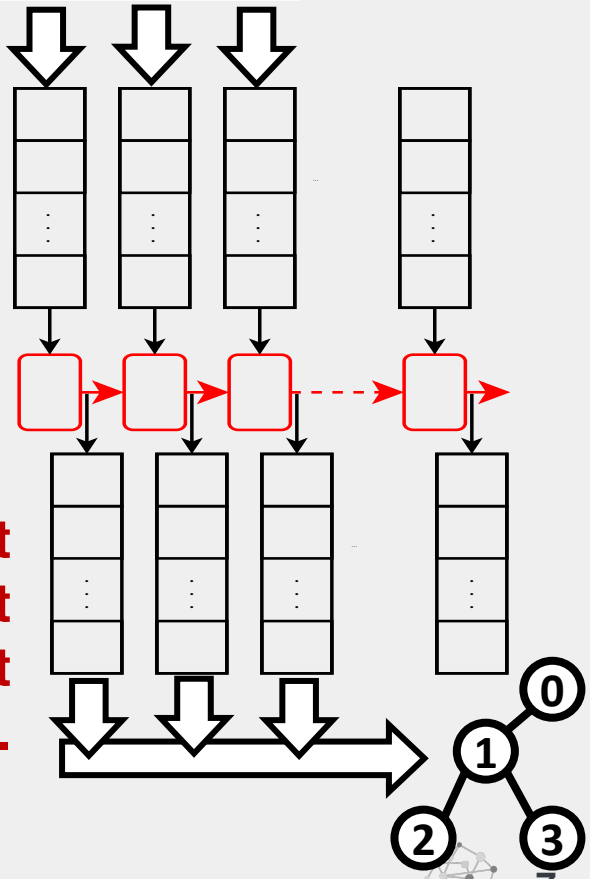
Converting edge sequence data using Depth-First Search (DFS).

Previous Node ID	0	1	0
Next Node ID	1	2	3



Training data

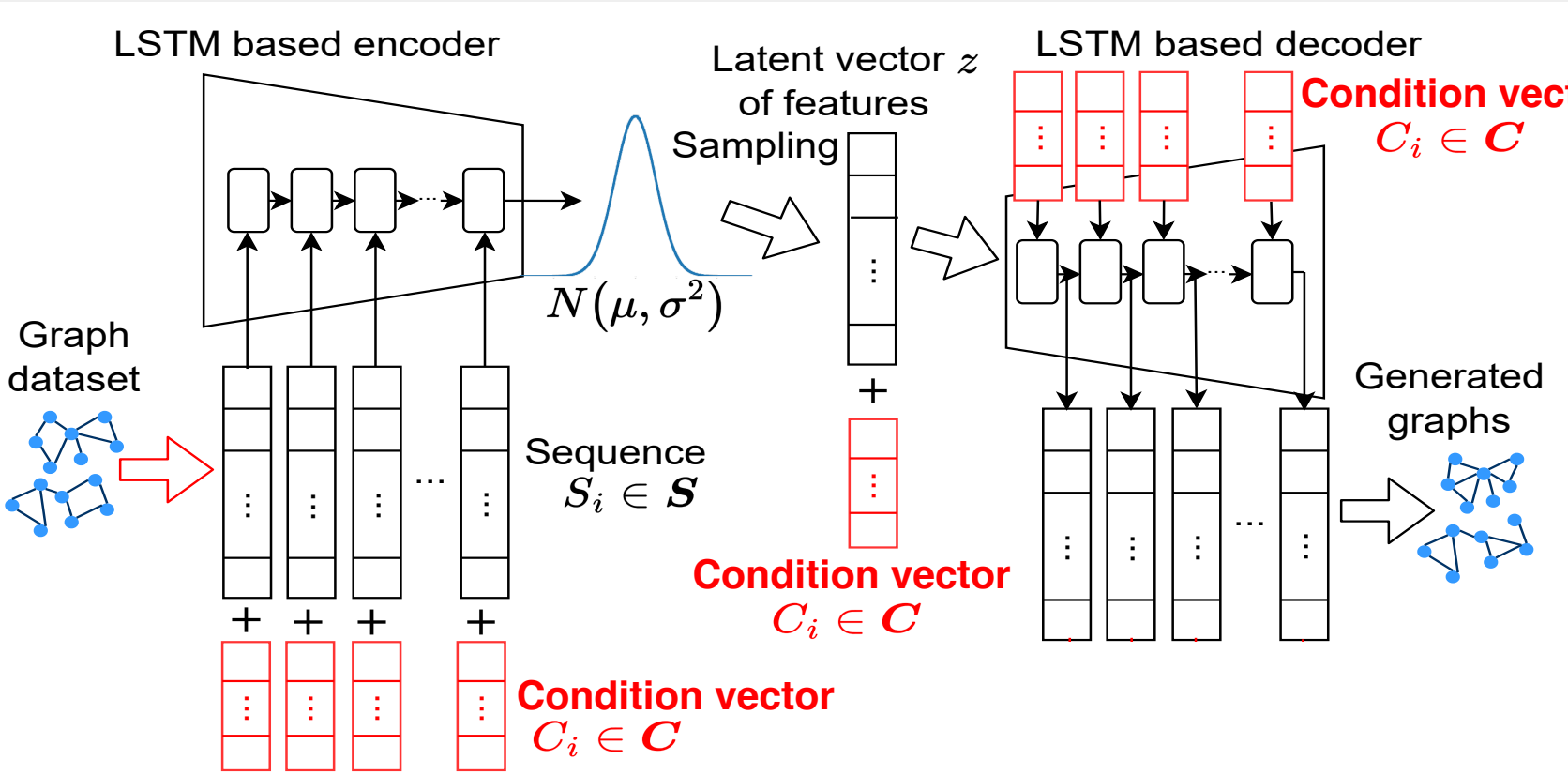
LSTM predict and output the subsequent edge.



Reconstructed data

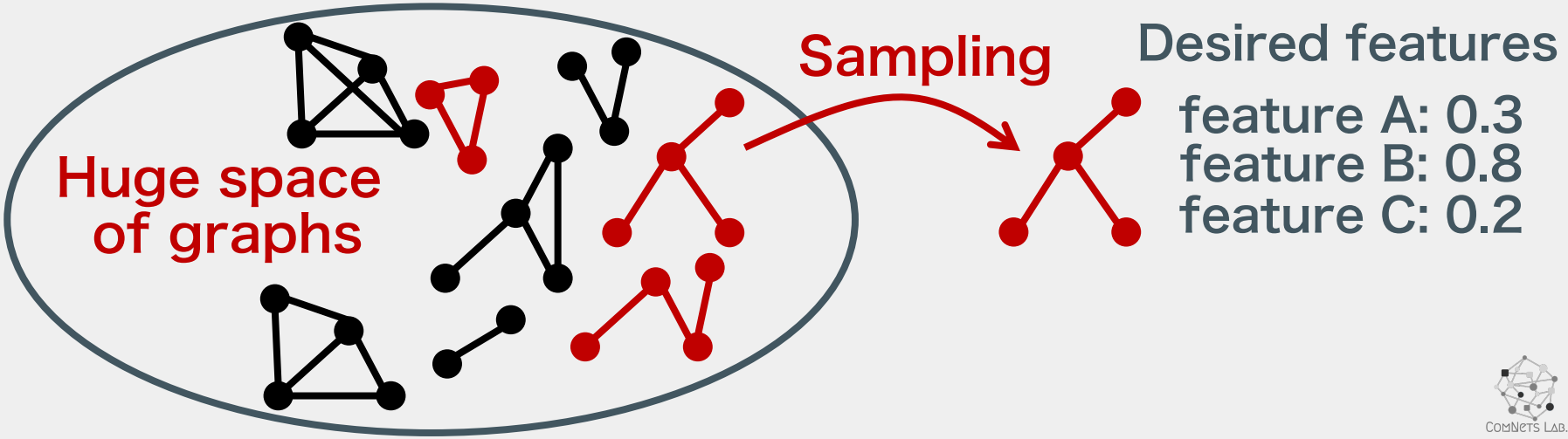
Deep Graph Generators – GraphTune

- A generative model capable of specifying arbitrary features by combining Variational AutoEncoder (VAE) and LSTM [1][2].
  - Learning to encode sequence data of graphs into vectors and then decode them back into their original sequences.
  - **By inputting a specified value of features as a condition vector, it can generate graphs with specified features.**



Objective of Graph Generation Techniques

- **Objective of Graph Generation Techniques: How to sample graphs with desired features from the huge space of graphs.**
  - WS Model: Aiming to reproduce small-world properties (average shortest path length).
  - BA Model: Aiming to reproduce scale-free properties (power-law exponent of degree distribution).
  - GraphGen: Aiming to reproduce the same features as training graph data.
  - GraphTune: Aiming to tune specific features while reproducing the other features of training graph data.



Experimental Settings for GraphTune

- GraphTune was trained using the following dataset:
  - Subgraphs of “who-follows-whom” graph of Twitter (Twitter dataset)

- Twitter dataset:
  - Attempted to tune only the **average shortest path length** by providing it as the condition vector while reproducing the other features of the dataset graphs.
  - Specified average shortest path lengths as 3.0, 4.0, and 5.0 in three patterns.

Graphs generated by GraphTune

- Distribution of generated graphs when trained on the Twitter dataset.
  - The overall distribution matches the training graph data (Real data).

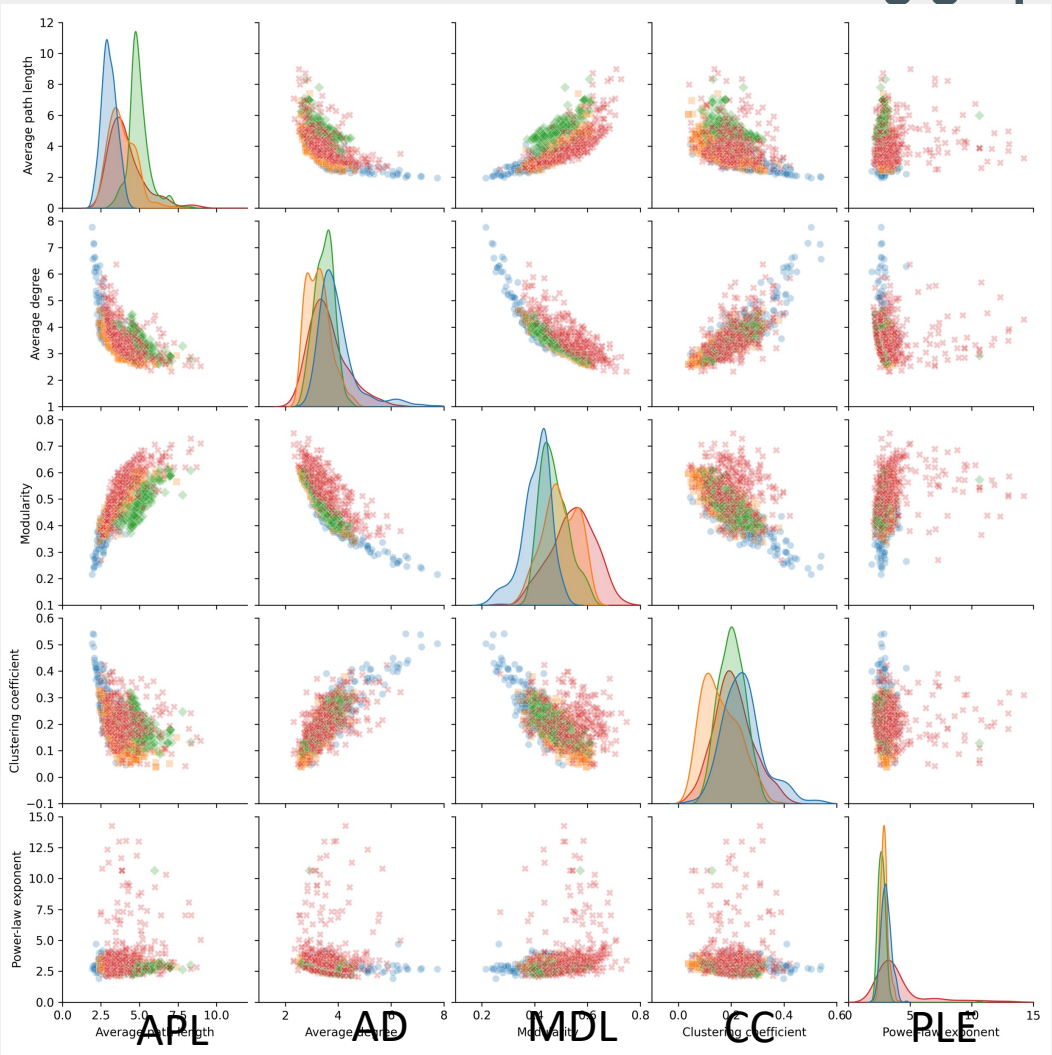
Average Path Length (APL)

Average degree(AD)

Modularity (MDL)

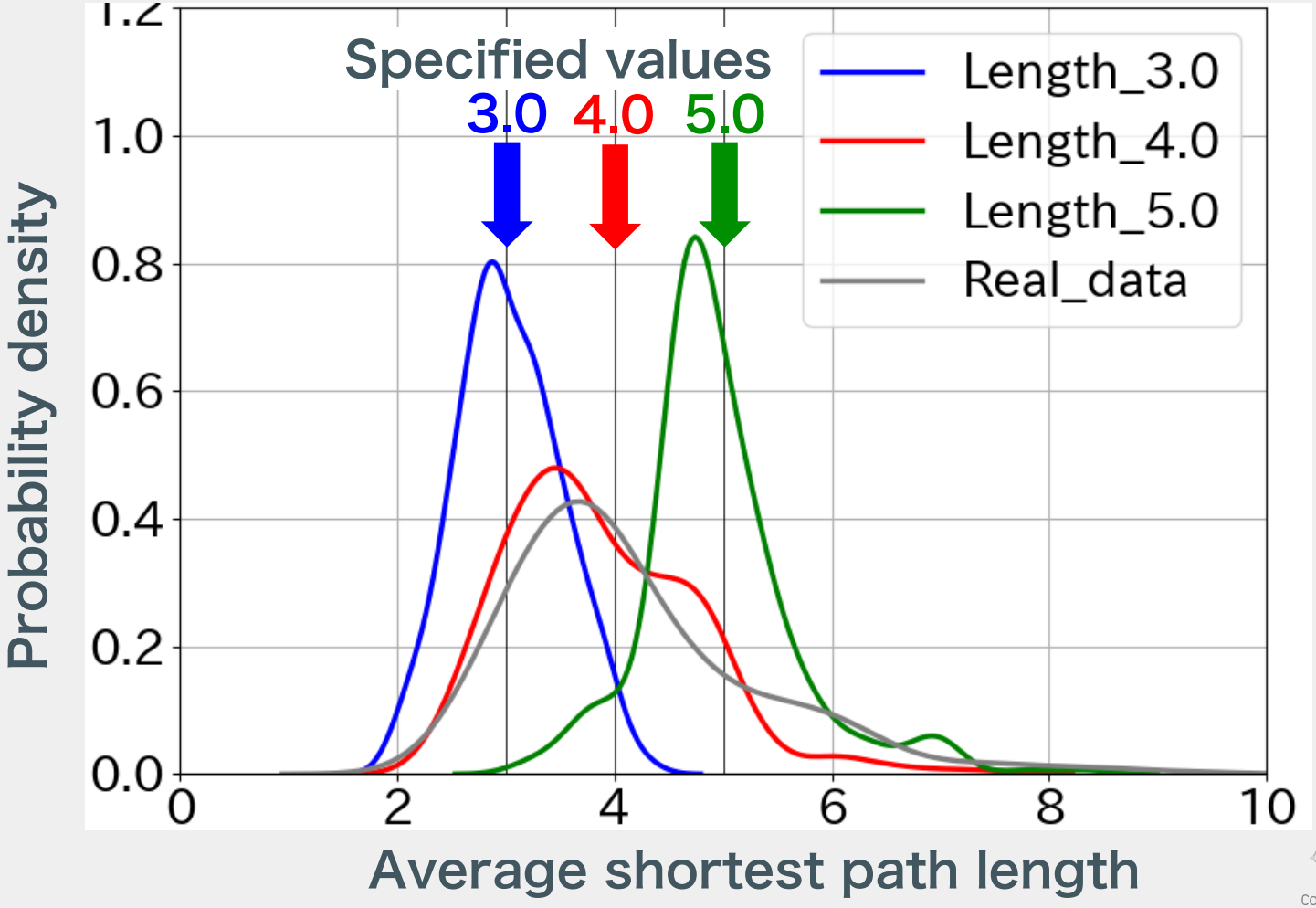
Clustering coefficient (CC)

Power-law exponent (PLE)



Graphs generated by GraphTune

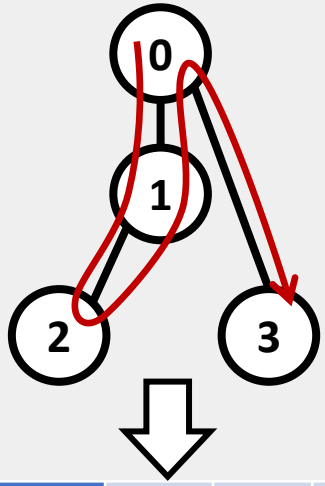
- The distribution of average shortest path length.
  - We can confirm that the distribution of generated graphs is concentrated around the specified value.



Experiments Regarding Sequence Conversion

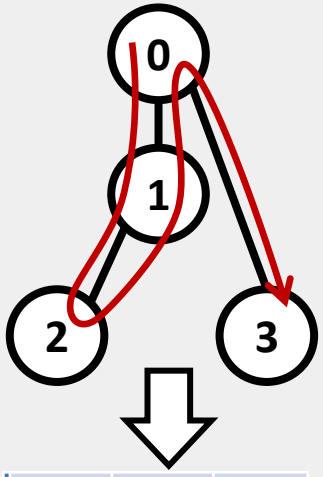
- Recently, we have been investigating for converting method to sequences from graphs.
- We investigated the impact of changing the deterministic DFS conversion to a random walk.

Deterministic DFS



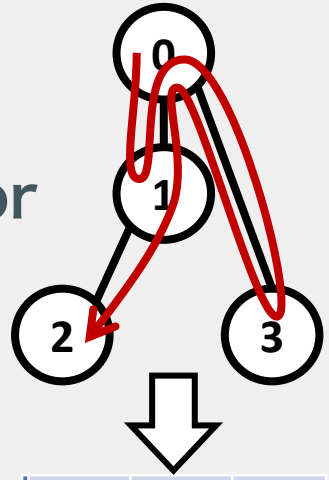
Previous Node ID	0	1	0
Next Node ID	1	2	3

2nd-order random walk



0	1	0
1	2	3

or



0	0	1
1	3	2

or ⋯

- 2nd-order random walk have 2 parameters p and q.
  - $p \gg q \Rightarrow$ DFS-like walk, $p \ll q \Rightarrow$ BFS-like walk

Graphs generated by GraphTune

- RMSE between the feature of generated graphs and the specified value are evaluated.
- Generally, the randomness of training data leads to training difficulty, but the accuracy improved with a random walk.
  - The impact of randomness is more pronounced when the training data size is small.

Deterministic DFS : 1.580 ± 0.343

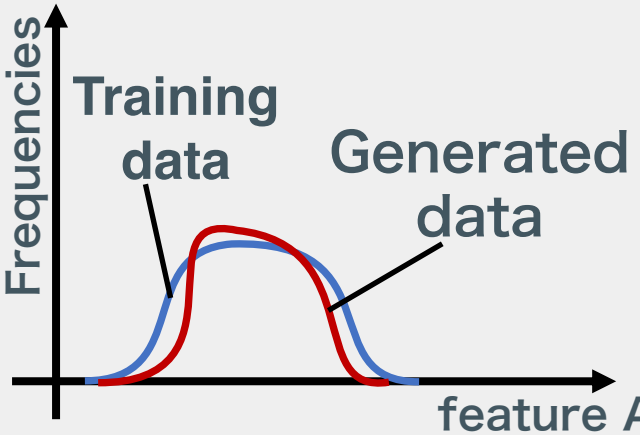
Better accuracy with randomness. ∨

parameter q \ parameter p	0.25	0.50	1.00	2.00	4.00
0.25	1.329 ± 0.436	1.339 ± 0.482	1.162 ± 0.318	1.153 ± 0.338	**0.831 ± 0.128**
0.50	1.461 ± 0.381	1.425 ± 0.488	1.257 ± 0.358	1.106 ± 0.247	0.961 ± 0.187
1.00	1.274 ± 0.191	1.403 ± 0.380	1.178 ± 0.256	1.084 ± 0.272	0.945 ± 0.314
2.00	1.300 ± 0.163	1.269 ± 0.268	1.318 ± 0.235	1.038 ± 0.214	1.027 ± 0.321
4.00	1.267 ± 0.172	1.266 ± 0.226	1.243 ± 0.196	1.219 ± 0.204	1.128 ± 0.284

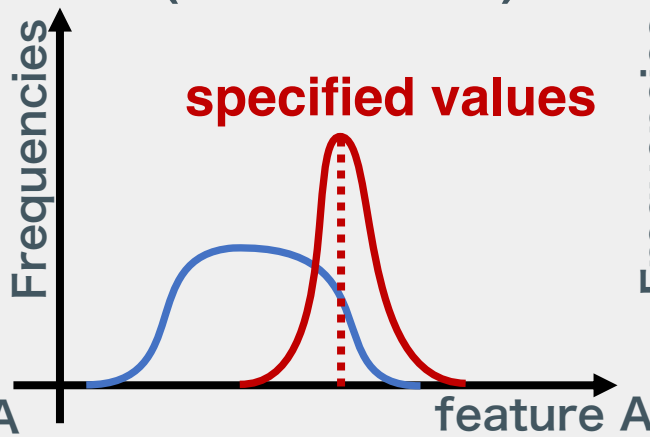
Future Prospects – Conditional Generation

- Graph generation techniques for reproducing training data have matured.
- The current focus is shifting towards the **development of conditional generation techniques.**
  - Improved precision in tunability
  - Diversification of tuning targets
  - Expansion of tunable regions

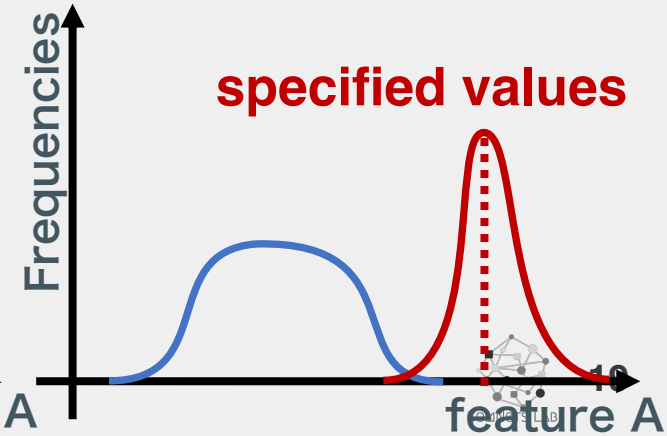
Generating graphs similar to training data (conventional)



Generating graphs that are similar to a subset of training data (current state)



Generating graphs in regions not present in the training data (future)



Summary

- Current Status
  - The graph generation problem involves sampling graphs with desired features from the huge space of graphs.
  - Generation techniques have become active in research, shifting from random generation to deep graph generators.
    - ER, WS, BA model ⇒ GraphGen, GraphTune

- Future Prospects
  - **Conditional generation techniques**, in particular, are promising as they **are still in the early stages of development**.
  - Once the technology for conditional graph generation matures, it may bring benefits to classical graph theory research as well.

References

[1] S. Nakazawa, Y. Sato, K. Nakagawa, S. Tsugawa, and K. Watabe, “A Tunable Model for Graph Generation Using LSTM and Conditional VAE,” Proc. of the 41st IEEE International Conference on Distributed Computing Systems (ICDCS 2021) Poster Track, 2021.

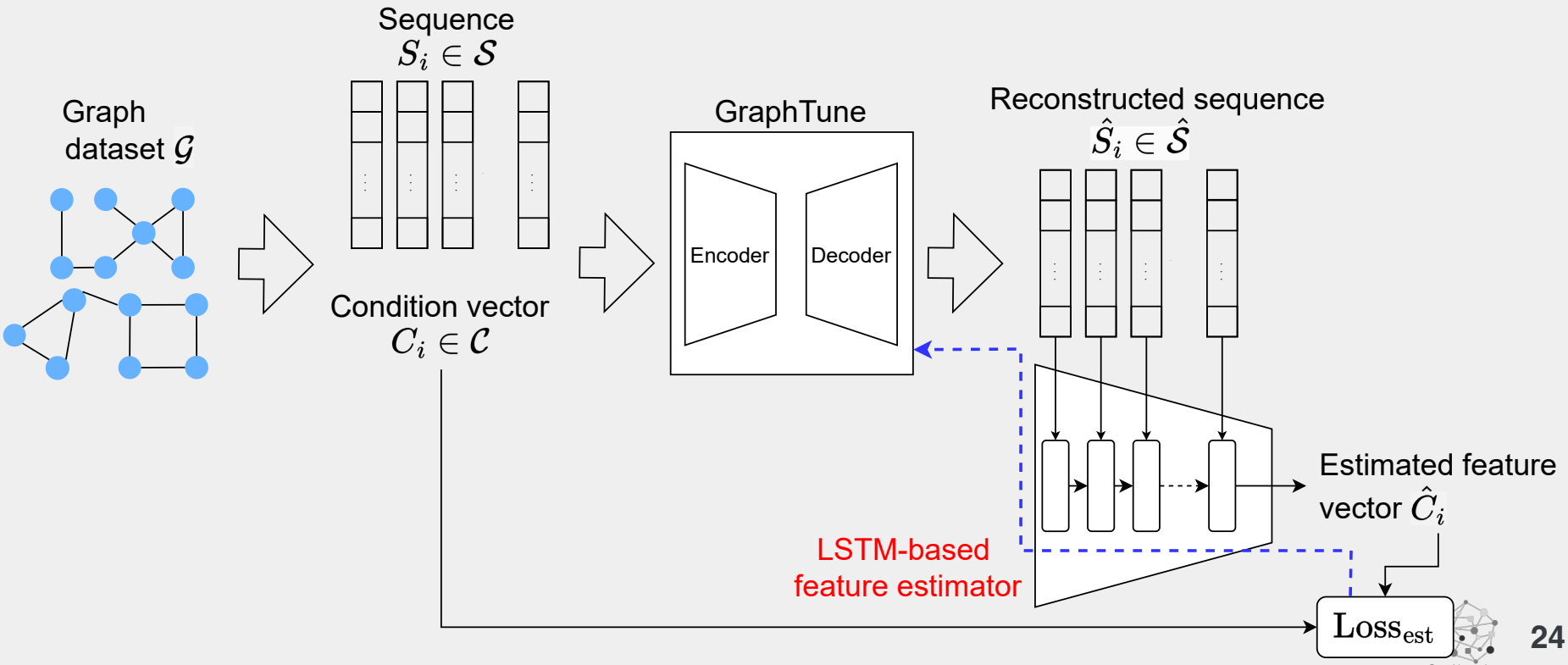
[2] K. Watabe, S. Nakazawa, Y. Sato, S. Tsugawa , and K. Nakagawa, “ GraphTune: A Learning-based Graph Generative Model with Tunable Structural Features,” IEEE Transactions on Network Science and Engineering, 2023.

[3] Takahiro Yokoyama, Yoshiki Sato, Sho Tsugawa, and Kohei Watabe, ``An Accurate Graph Generative Model with Tunable Features'', The 32nd International Conference on Computer Communications and Networks (ICCCN 2023) Poster Session, Honolulu, HI, USA , 2023.

Thank you for your kind attention.

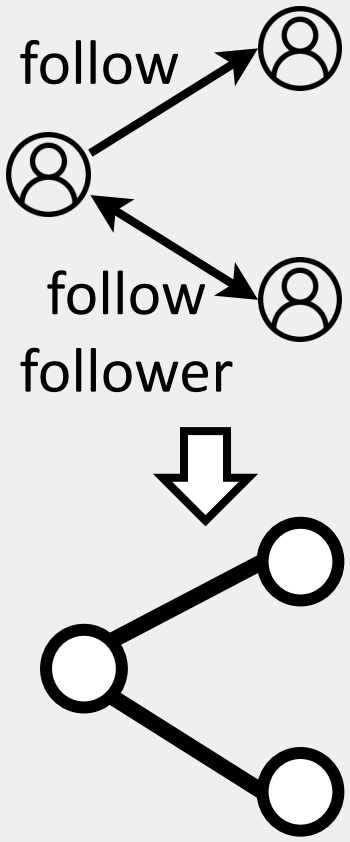
Deep Graph Generators – F-GraphTune

- An enhanced version of GraphTune, a generative model incorporating a feature estimator feedback mechanism for higher accuracy.
  - Feature estimator estimates a conditional vector representing the graph's characteristics from the reconstructed sequence.
  - Alternating training between GraphTune and Feature estimator.



ハイパーパラメータ

◆ 平均経路長など複数の特徴量を指定して精度を評価



◆提案モデル
- ◆最初の全結合層のサイズ：256
- ◆LSTMの隠れ層のサイズ：512
- ◆最後の全結合層のサイズ：1
- ◆GraphTune：提案論文で評価されたものを使用

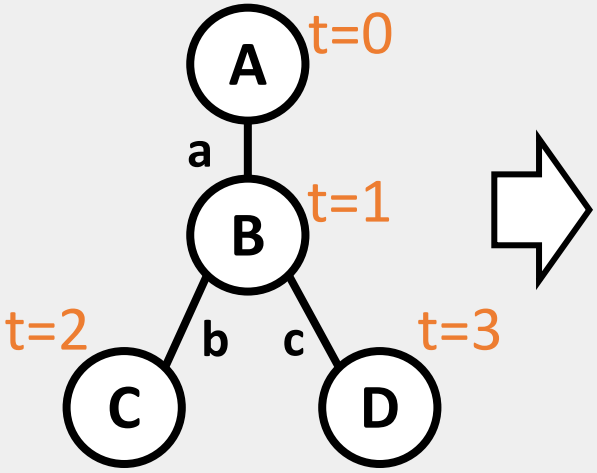
◆学習
- ◆最適化関数：Adam
- ◆学習率：0.001
- ◆Weight decay：0.0
- ◆勾配クリップの閾値：1.0
- ◆交互学習の反復回数：2
- ◆バッチサイズ：37
- ◆エポック数：10,000

Deep Graph Generators – GraphGen

◆グラフをシーケンスにする手法

1. グラフに深さ優先探索を行い探索順に各ノードにタイムスタンプを付与する
2. タイムスタンプ情報とノードラベル、ノード間のエッジラベルを基に5要素のベクトルに変換する
3. 各ノードを探索順毎にベクトルに変換することでグラフをシーケンスなデータに変換する

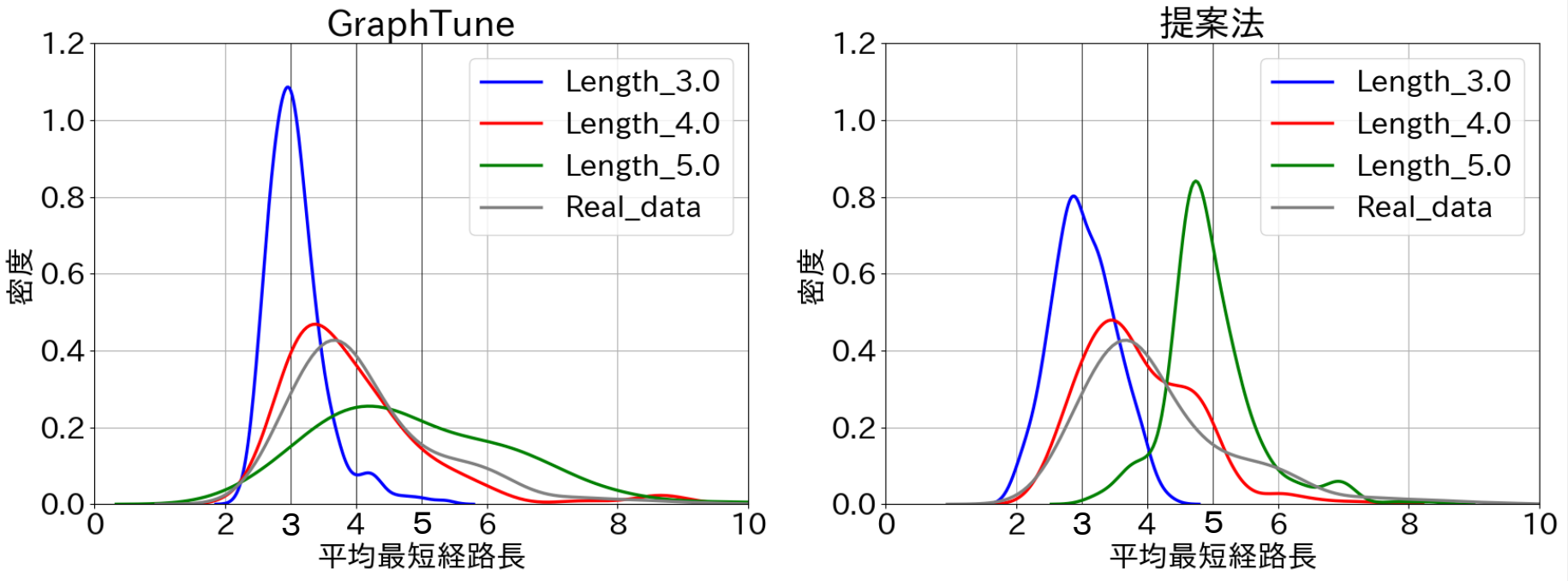
グラフを深さ優先探索



シーケンスデータへ

Time	Time stamp u	Time stamp v	Node label u	Node label v	Edge label
0	0	1	A	B	a
1	1	2	B	C	b
2	1	3	B	D	c

生成結果：分布

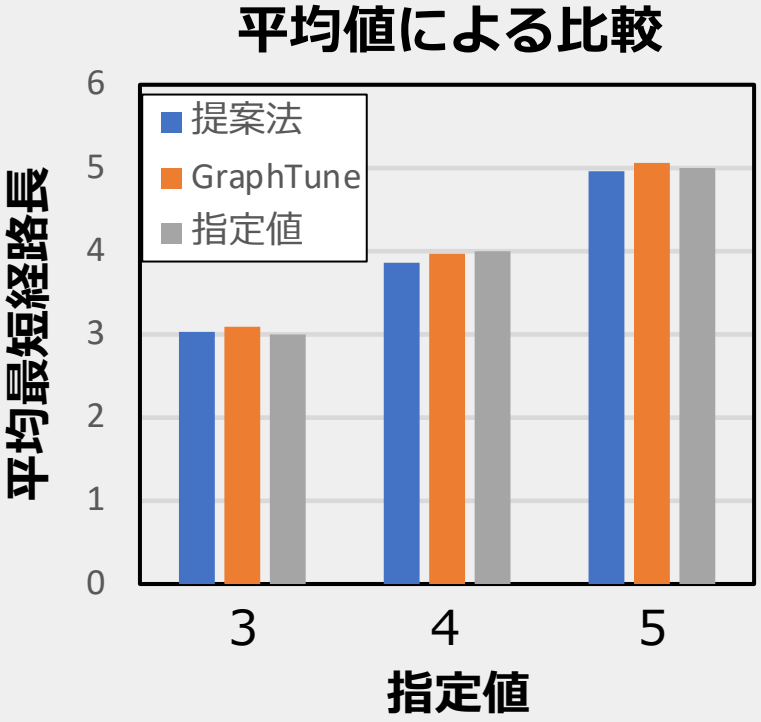


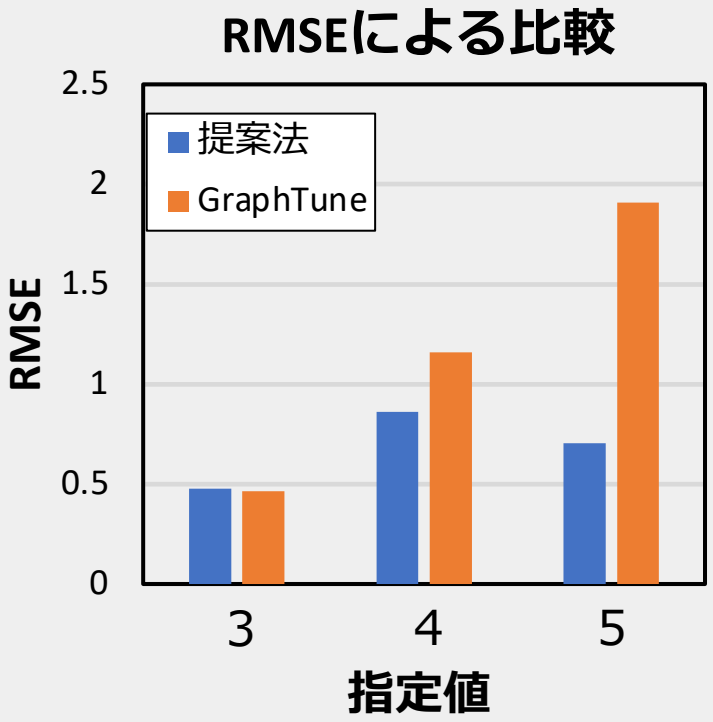
◆各手法による生成結果のカーネル密度推定プロットで分布を表示

◆平均最短経路長を5と指定した場合の分布は、GraphTuneよりも提案法の方がより指定値の近くに集中して分布している

生成結果：評価指標

◆評価指標による比較





◆平均値では2つの手法は指定値通りに平均値が推移

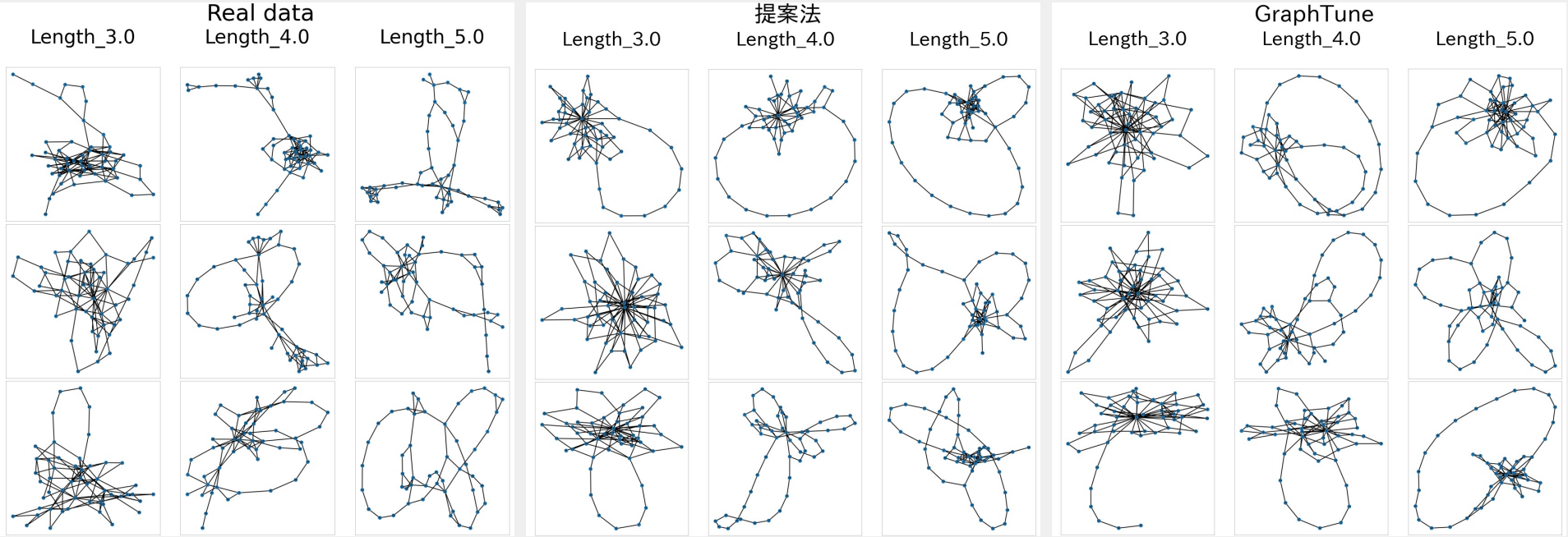
◆RMSEでは提案法がGraphTuneと同等かそれ以上の性能を示す

➢提案法は生成精度を高めることができたといえる

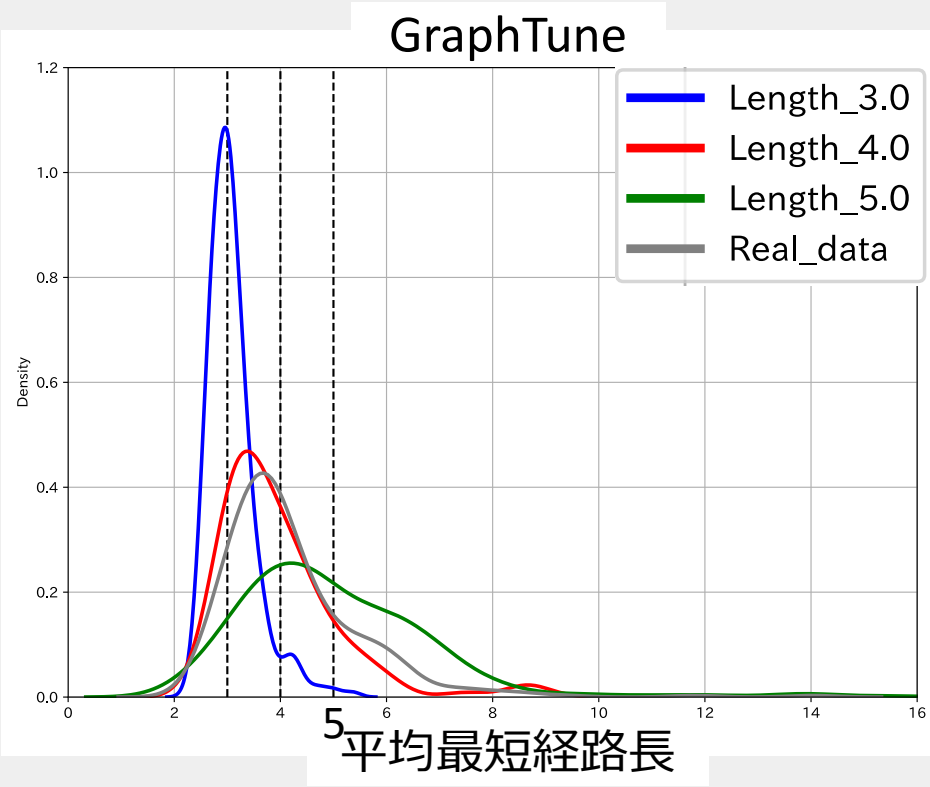
補足：生成されたグラフ

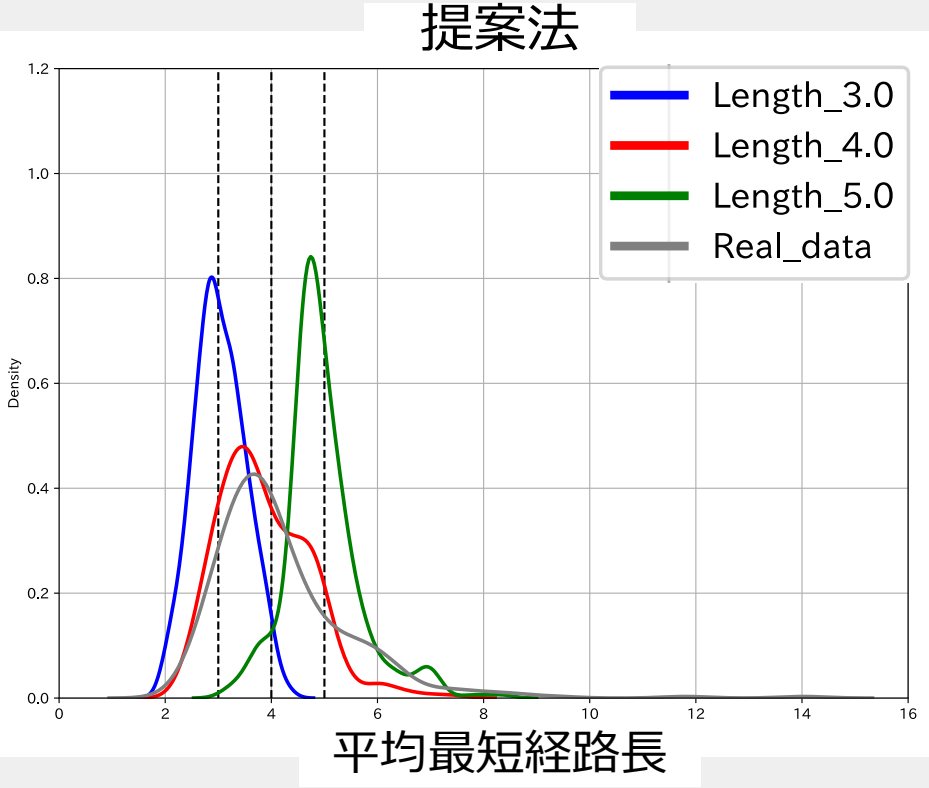
◆指定値ごとに生成されたグラフの一部を抜粋

◆Real dataはデータセットから指定値に最も近い3つのグラフを抜粋



補足：生成結果の分布の全体





◆各手法による生成結果のカーネル密度推定プロットで分布を表示

◆平均最短経路長を5と指定した場合の分布は、GraphTuneよりも提案法の方がより指定値の近くに集中して分布している

補足：生成結果の定量的評価

◆評価指標による比較

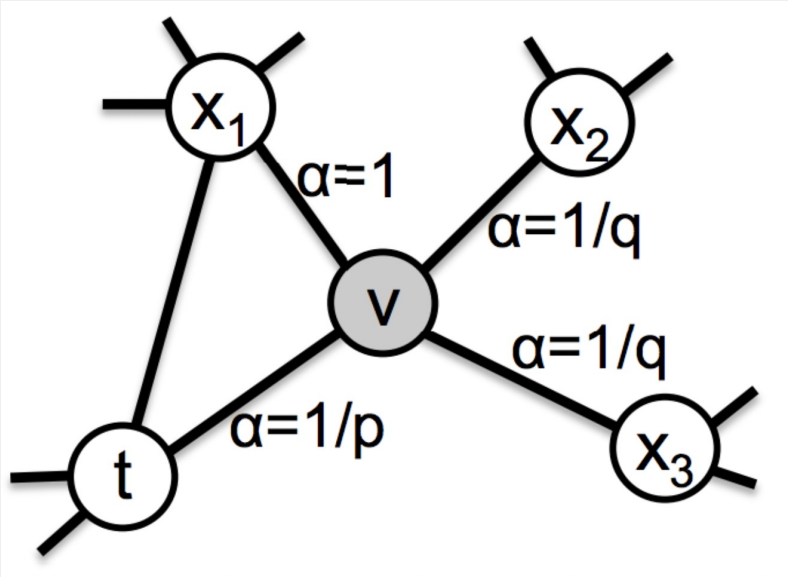
指標	Condition	提案法	GraphTune	Real data
平均値	3.0	**3.03**	3.09	4.23
	4.0	3.86	**3.97**	
	5.0	**4.96**	5.06	
RMSE	3.0	0.476	**0.463**	–
	4.0	**0.861**	1.16	
	5.0	**0.705**	1.91	

◆多くの場合で提案法はGraphTuneよりも優れたスコアである

◆GraphTuneが優れている場合でも、提案法との差は小さい

➢提案法は生成精度を高めることができたといえる

2nd order random walk



$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases}$$

◆遷移先のノードを決定する際、今いるノードの一つ前のノードからの距離に応じて選択確率にバイアスがかかるように設定する手法。

◆パラメータを　　p > q とするとDFS
　　q < p とするとBFS に近い遷移となる