Causal Discovery in Gene Regulatory Networks with GFlowNet: Towards Scalability in Large Systems

Trang Nguyen [*1,2,3], **Alexander Tong**[1,4], **Kanika Madan**[1,4], **Yoshua Bengio**[†1,4,5], and **Dianbo Liu**[†1,2]

[1]Mila – Quebec AI Institute
[2]National University of Singapore
[3]FPT Software AI Center
[4]Université de Montréal
[5]CIFAR AI Chair

Abstract

Understanding causal relationships within Gene Regulatory Networks (GRNs) is essential for unraveling the gene interactions in cellular processes. However, causal discovery in GRNs is a challenging problem for multiple reasons, including the existence of cyclic feedback loops and uncertainty that yields diverse possible causal structures. Previous works in this area either ignore cyclic dynamics (assume acyclic structure) or struggle with scalability. We introduce Swift-DynGFN as a novel framework that enhances causal structure learning in GRNs while addressing scalability concerns. Specifically, Swift-DynGFN exploits gene-wise independence to boost parallelization and to lower computational cost. Experiments on real single-cell RNA velocity and synthetic GRN datasets showcase the advancement in learning causal structure in GRNs and scalability in larger systems.

1 Introduction

Gene regulatory networks (GRNs) play a critical role in cell biology, orchestrating a highly intricate interplay of molecular interactions that ultimately govern the behavior of cells [20, 17]. Understanding causality entails deciphering the complex web of relationships between genes, shedding light on how the activity of one gene can intricately influence the expression or behavior of another [31, 1, 8, 22, 27]. However, uncovering causality within GRNs faces challenges, including the cyclic feedback loops [26, 3] and uncertainty that leads to multiple possible causal structures [3, 16, 14]. Prior efforts in this domain have encountered two restrictions. Firstly, most previous works formulate the problem as a Directed Acyclic Graph (DAG) [18, 23, 15, 2], thus overlooking the cyclic nature inherent in these networks. Secondly, scalability remains a concern when dealing with large GRNs [3, 15].

The *Bayesian dynamic structure learning* framework introduced in [3] and the incorporation with Generative Flow Networks (GFlowNets) [4, 5] sheds light on solving the first limitation. Particularly, to model causality within GRNs, Atanackovic et al. [3] simplifies structure learning in GRNs as a sparse identification problem within a dynamical system, utilizing RNA velocity data to estimate the rate of change of a gene's expression. In the context of dynamical systems, it is feasible to depict both the causal relations between variables and the system's changing behavior through time. Besides, GFlowNet plays a critical role in the work to model intricate distributions over cyclic structures. However, the scalability is still one restriction of their work on larger systems.

[*]Work done under an internship at Mila - Quebec AI Institute and National University of Singapore. Correspondence to trangphuongnnguyen@gmail.com.

[†]Equal contribution.

NeurIPS 2023 Generative AI and Biology Workshop.

In this study, we propose Swift-DynGFN to continue to navigate the intricacies of causal structures in GRNs and address scalability. Swift-DynGFN improves the architecture of GFlowNet and draws upon the *Bayesian dynamic structure learning* framework. Notably, we enhance causal structure learning by optimizing variable-wise influence, leveraging predictions from previous variables, and allowing the model to decide on the variable order to be processed. Furthermore, we improve the scalability by parallelizing the prediction process, reducing the sequential computation steps from n^2 to n (where n is the number of variables), significantly reducing the time and space requirements.

The main contributions of this work are summarized as follows:

- Introduce Swift-DynGFN that improves causal structure learning in GRNs by encouraging gene-wise causal influence and tackles the scalability challenge by enriching parallelization to reduce time and space costs.
- Experiment on real single-cell velocity showcases the capability to capture uncertainty and accurately infer causal relationships within GRNs.
- The experiment on synthetic GRN data proves the scalability potential of Swift-DynGFN.

2 Preliminaries

2.1 Dynamical systems in Single-cell RNA Velocity

Denote a finite dataset as $\mathcal{D}$, containing dynamic pairs (x, dx), where x represents a state in a time-invariant stochastic dynamical system and dx denotes its time derivative. In estimating the change rate in gene expression by leveraging *RNA velocity* [6], x and dx align to *gene expression levels* and *velocity* of changes in gene expression, respectively. We aim to learn posterior over explanatory graphs, G, that defines the sparsity graph pattern among variables in $\mathcal{D}$, represented as $Q(G|\mathcal{D})$.

2.2 Generative Flow Networks

Generative Flow Networks (GFlowNet) constitute a probabilistic framework designed to facilitate the generation of a diverse array of candidates over spaces of discrete objects [15, 4, 5]. Particularly, GFlowNet learns a probabilistic strategy for building structured objects, *e.g.* a graph, by generating a sequence of actions that gradually transform a partial object to a complete object by taking a sequence of actions where each action corresponds to adding an edge. Several training objectives have been used [15, 4, 25, 24], one of which is the *detailed balance* objective [5].

$$F(s)P_F(s'|s) = F(s')P_B(s|s') \tag{1}$$

Detailed Balance (DB) Training Objective Denoting a GFlowNet model parameterized by ψ that optimizes *forward policy* $P_F(s'|s, \psi)$ and *backward policy* $P_B(s'|s, \psi)$ corresponding to *Markovian flow* of a non-terminal state $F_\psi(s)$, the *DB constraint* [5] is presented as Equation 1 in transformation $s \to s'$. Subsequently, the *detail balance loss* is presented in Equation 2 that optimizes the *DB constraint*. With a terminal state s_n, a *Reward matching loss* [5, 25] is added, formulated as $\mathcal{L}_R(s_n) = (\log(R(s_n)) - \log(F_\psi(s_n)))^2$ with $R(s_n)$ implies to the reward obtained at state s_n.

$$\mathcal{L}_{DB}(s_{i-1}, s_i) = \left(\log \frac{F_\psi(s_{i-1})P_F(s_i|s_{i-1}, \psi)}{F_\psi(s_i)P_B(s_{i-1}|s_i, \psi)} \right)^2 \tag{2}$$

3 Proposed method: Swift-DynGFN

Intuitively, Swift-DynGFN enhances the variable-wise influence in predicting causal structures, raising parallelization for each variable and conducting sequential computing among different variables to alleviate time and space requirements.

Facilitating causal influence, we tackle two questions at each prediction step. First, "**What has been done?**". Specifically, as presented in Figure 1, the prediction of the current variable (red edges) is affected by predictions made on other nodes (gray edges) that ensure variables causally contribute to others. Second, "**What's next?**". Particularly, Swift-DynGFN selects the next node to be processed

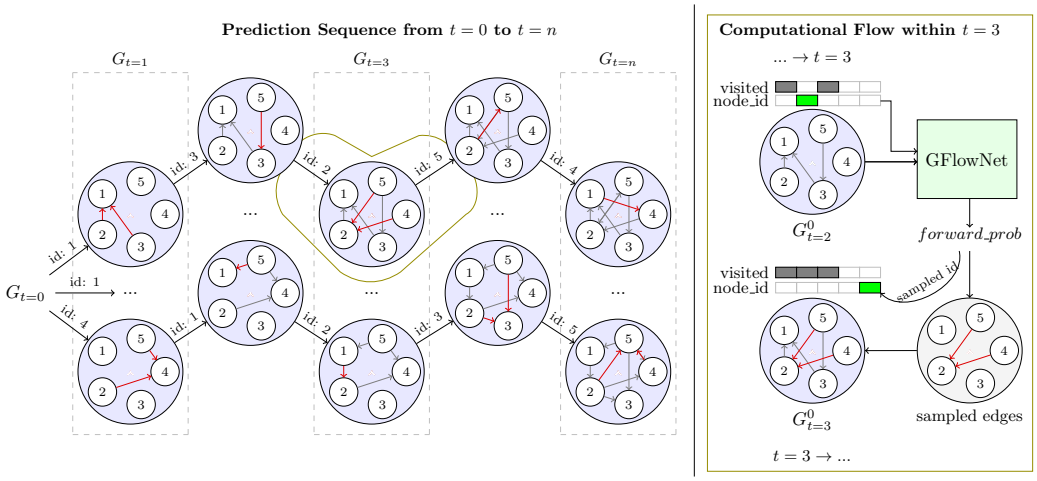

Figure 1: **Swift-DynGFN Intuition and Computation Flow**. Edges added from the previous turns are gray, and newly added edges are red. The model acknowledges *all* previously added edges and outputs (1) *all* incoming edges of the current node and (2) the node for the next turn (node_id). The visited binary mask marks nodes are done. A batch of graphs is predicted in parallel.

(light green slot in $node\_id$ vector in Figure 1) that harnesses the causal relationships in the precision of predictions for each other.

In terms of computational complexities, Swift-DynGFN design two strategies: (1) the prediction of incoming edges to the current node of interest is parallel, drastically reducing processing time and (2) we cut off the computation decomposition into each variable as observed in prior work [3] to avoid parameter outbreak when scaling up the number of nodes.

3.1 Variable-wise causal influence in Swift-DynGFN

Regarding paying attention to predictions made on previous variables, we formulate $Q(G|D)$ as in Equation 3. Particularly, $G \in \mathbb{R}^{B\times(n+1)\times(n\times n)}$ denotes states of B causal structures during $t \in [0 \dots n]$ steps, which have a shape of $n \times n$ per graph, where n is the number of variables. From the implementation point of view, the GFlowNet model takes incoming edges on visited nodes and the current node's index inputs.

$$Q(G|D) = \prod_{i\in 0\dots n} Q(G_{t=i}|G_{t=i-1}, D) \tag{3}$$

To decide the next variable to be processed, a variable's index that has not been visited yet is sampled as an action by the forward policy, along with the incoming edges of the current variable of interest. We employ the forward probability to sample both actions, with a binary mask to mask the set of visited nodes when selecting the subsequent variable. In the first turn ($t = 0$), only the action of the next variable index is taken into account, whereas incoming edges are not sampled since the node of interest has not been selected yet.

3.2 Parallelization in Swift-DynGFN

Different from prior approaches [3, 15, 4, 25, 24] that sample a single edge at a time, we sample all incoming edges of the current node from the same forward probability. Subsequently, the time complexity is dramatically optimized from n^2 to n regarding the times of sequential computations conducted. In addition, we avoid designing a GFlowNet in each node, as it introduces a potential threat to parameter explosion in a larger number of nodes. Instead, we operate a single GFlowNet model shared among variables, reducing the number of parameters required in larger systems.

Table 1: **Dynamic causal structure inference in GRN**. Reported scores are mean and std over five seeds. Swift-DynGFN outperforms all baselines and reduces computational complexities.

	Cellular System - RNA Velocity			
Methods	**Bayes-SHD↓**	**AUC↑**	**#Params**	**Duration (h)**
DynBCD	**2.79**±0.34	0.53±0.07	100	3.76
DynDiBS	6.82±0.78	0.46±0.03	50.0k	1.02
DynGFN	3.37±0.51	0.59±0.03	255.4k	1.16
Swift-DynGFN $_{best}$	**2.93**±0.21	**0.73**±0.04	87.4k	0.53
Swift-DynGFN $_{large}$	3.22±0.36	0.69±0.03	255.4k	0.99

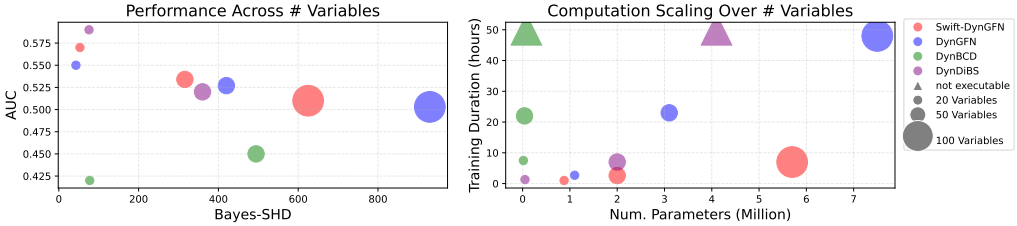

Figure 2: **Scalability comparison on synthetic data.** Swift-DynGFN shows strong scalability as it avoids complexities outbreak in large-scale systems while producing remarkable performance.

3.3 Optimization strategy in Swift-DynGFN

We utilize the *DB objective* as presented in Section 2.2 with the terminal state defined as G_n and reward as $R(G_d) = e^{-||dx_b - \widehat{dx_b}||_2^2 + \lambda_0 ||G_n||_0}$ where dx_b is the ground truth of the input batch x_b, and the $\lambda_0||G_n||_0$ term encourage sparsity of the GRNs.

4 Experiments

We conduct experiments to verify two hypotheses: $\mathcal{H}_1$ - Leveraging the causal relationship among genes enhances causal inference in GRN (Section 4.1) and $\mathcal{H}_2$- Swift-DynGFN reduces time and space complexities, contributing to improvements in large-scale systems (Section 4.2).

We compare our method to DynGFN, DynBCD, and DynDiBS, reported in DynGFN paper [3]. Particularly, besides GFlowNet, Atanackovic et al. [3] integrates BCD [13] and DiBS [23], which are designed for static systems, to *Bayesian dynamic structure learning*, denoted as DynBCD and DynDiBS. In addition, we employ Bayes-SHD and AUC metrics to evaluate the predicted structures over the true *graphs*.

4.1 Experiment on Single-Cell RNA-velocity Data

Dataset We investigate the cell cycle dataset of human Fibroblasts [30] that contains records of 5000 cells and more than 10,000 genes. Following Atanackovic et al. [3], we utilize a group of five genes, where Cdc25A activates Cdk1, which, in turn, inhibits Cdc25C, while the Mcm complex is correlated with Cdc25A but does not directly interact with Cdk1 in the cell cycle regulation. With this setting, the GRN system contains 81 admissible causal structures.

Supporting $\mathcal{H}_1$, Table 1 presents the performance and computation details of Swift-DynGFN alongside the reproduction of all baselines, experimented on RNA velocity on a single NVIDIA A100 and 4 CPUs. Overall, our proposed method delivers precise predictions while reducing computational complexities. Regarding prediction quality, Swift-DynGFN outperforms all baselines by a notable margin in both metrics, evident by the 0.14 increased AUC and 0.44 reduced Bayes-SHD compared to DynGFN. Regarding computational complexities, our best configuration, Swift-DynGFN $_{best}$, utilizes far fewer parameters and GPU hours for training than DynGFN. Even when configured to match DynGFN's parameter count, Swift-DynGFN $_{large}$, our method maintains superior performance and faster training time.

4.2 Experiment on Synthetic Data

Dataset We adopt the dataset generation using the indeterminacy model from Atanackovic et al. [3] (more detail is in Appendix C) and create a non-linear dynamical system $dx = \mathsf{sigmoid}(\mathbf{A}x)$. We design the number of variables to vary from 20, 50, and 100 with a fixed sparsity equal to 0.9 to be consistent to [3].

Examining $\mathcal{H}_2$, Figure 2 illustrates the Swift-DynGFN scalability, conducted on 2 NVIDIA A100-80GB GPUs and 8 CPUs. Swift-DynGFN consistently demonstrates robust scalability, requiring fewer computational resources in large-scale systems while delivering considerable performance. On the left, Swift-DynGFN outperforms or is on par with baselines as the number of nodes increases. On the right, Swift-DynGFN avoids computational outbreak, a phenomenon observed in baselines as the system scales up. For instance, with 100 nodes, DynGFN's training duration is five times longer than that of Swift-DynGFN, while DynBCD and DynDiBS struggle with CUDA memory requirements.

5 Conclusion

In this study, we proposed Swift-DynGFN that improves dynamical causal structure inference in large-scale systems. We leverage the GFlowNet model to strengthen the causal consideration while simultaneously parallelizing the computation to reduce time and space requirements. Our experiments on GRN and synthetic data illustrate Swift-DynGFN 's effectiveness in improving causal inference ability and scalability in dealing with large-scale systems compared to baselines.

Future Works: Potential directions to further improve this study include (1) extending the number of genes to estimate ability in large-scale GRN systems and (2) investigating the causal structure in diverse biological contexts, such as Metabolic Pathways and Immune Response.

Acknowledgement

We gratefully acknowledge the support received for this research. This research was enabled by computational resources provided by Mila. Each member involved in this research is funded by their primary institution.

References

[1] Syed Ahmed, Swarup Roy, and Jugal Kalita. Assessing the effectiveness of causality inference methods for gene regulatory networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17:56–, 01 2020. doi: 10.1109/TCBB.2018.2853728.

[2] Yashas Annadani, Jonas Rothfuss, Alexandre Lacoste, Nino Scherrer, Anirudh Goyal, Yoshua Bengio, and Stefan Bauer. Variational causal networks: Approximate bayesian inference over causal structures. *CoRR*, abs/2106.07635, 2021. URL `https://arxiv.org/abs/2106.07635`.

[3] Lazar Atanackovic, Alexander Tong, Jason Hartford, Leo J. Lee, Bo Wang, and Yoshua Bengio. Dyngfn: Towards bayesian inference of gene regulatory networks with gflownets, 2023.

[4] Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *arXiv preprint arXiv:2106.04399*, 2021.

[5] Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J. Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023. URL `http://jmlr.org/papers/v24/22-0364.html`.

[6] Volker Bergen, Marius Lange, Stefan Peidli, F. Wolf, and Fabian Theis. Generalizing rna velocity to transient cell states through dynamical modeling. *Nature Biotechnology*, 38:1–7, 12 2020. doi: 10.1038/s41587-020-0591-3.

[7] Ioan Gabriel Bucur, Tom Claassen, and Tom Heskes. Large-scale local causal inference of gene regulatory relationships. *International Journal of Approximate Reasoning*, 115:50–68, dec

2019. doi: 10.1016/j.ijar.2019.08.012. URL https://doi.org/10.1016%2Fj.ijar.2019.08.012.

[8] Guangyi Chen and Zhi-Ping Liu. Inferring causal gene regulatory network via greynet: From dynamic grey association to causation. *Frontiers in Bioengineering and Biotechnology*, 10, 2022. ISSN 2296-4185. doi: 10.3389/fbioe.2022.954610. URL https://www.frontiersin.org/articles/10.3389/fbioe.2022.954610.

[9] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *CoRR*, abs/1806.07366, 2018. URL http://arxiv.org/abs/1806.07366.

[10] Mathieu Chevalley, Yusuf Roohani, Arash Mehrjou, Jure Leskovec, and Patrick Schwab. Causalbench: A large-scale benchmark for network inference from single-cell perturbation data, 2023.

[11] Y. Chu, X. Wang, J. Ma, K. Jia, J. Zhou, and H. Yang. Inductive granger causal modeling for multivariate time series. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 972–977, Los Alamitos, CA, USA, nov 2020. IEEE Computer Society. doi: 10.1109/ICDM50108.2020.00111. URL https://doi.ieeecomputersociety.org/10.1109/ICDM50108.2020.00111.

[12] Tom Claassen and Tom Heskes. A logical characterization of constraint-based causal discovery. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, page 135–144, Arlington, Virginia, USA, 2011. AUAI Press. ISBN 9780974903972.

[13] Chris Cundy, Aditya Grover, and Stefano Ermon. BCD nets: Scalable variational approaches for bayesian causal discovery. *CoRR*, abs/2112.02761, 2021. URL https://arxiv.org/abs/2112.02761.

[14] Roozbeh Dehghannasiri, Byung-Jun Yoon, and Edward Dougherty. Efficient experimental design for uncertainty reduction in gene regulatory networks. 03 2015. doi: 10.13140/RG.2.1.3252.2085.

[15] Tristan Deleu, António Góis, Chris Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian structure learning with generative flow networks. In *Uncertainty in Artificial Intelligence*, pages 518–528. PMLR, 2022.

[16] Stojan Denic, B. Vasic, Charalambos Charalambous, and Ravishankar Palanivelu. Robust control of uncertain context-sensitive probabilistic boolean networks. *Systems Biology, IET*, 3: 279 – 295, 08 2009. doi: 10.1049/iet-syb.2008.0121.

[17] Frank Emmert-Streib, Matthias Dehmer, and Benjamin Haibe-Kains. Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. *Frontiers in Cell and Developmental Biology*, 2, 2014. ISSN 2296-634X. doi: 10.3389/fcell.2014.00038. URL https://www.frontiersin.org/articles/10.3389/fcell.2014.00038.

[18] Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, 10, 2019. ISSN 1664-8021. doi: 10.3389/fgene.2019.00524. URL https://www.frontiersin.org/articles/10.3389/fgene.2019.00524.

[19] Biwei Huang, Kun Zhang, Jiji Zhang, Joseph D. Ramsey, Ruben Sanchez-Romero, Clark Glymour, and Bernhard Schölkopf. Causal discovery from heterogeneous/nonstationary data. *CoRR*, abs/1903.01672, 2019. URL http://arxiv.org/abs/1903.01672.

[20] Guy Karlebach and Ron Shamir. Modelling and analysis of gene regulatory networks. *Nature reviews. Molecular cell biology*, 9:770–80, 10 2008. doi: 10.1038/nrm2503.

[21] Paola Lecca. Machine learning for causal inference in biological networks: Perspectives of this challenge. *Frontiers in Bioinformatics*, 1, 2021. ISSN 2673-7647. doi: 10.3389/fbinf.2021.746712. URL https://www.frontiersin.org/articles/10.3389/fbinf.2021.746712.

[22] Lin Li, Rui Xia, Wei Chen, Qi Zhao, Peng Tao, and Luonan Chen. Single-cell causal network inferred by cross-mapping entropy. *Briefings in Bioinformatics*, page bbad281, 08 2023. ISSN 1477-4054. doi: 10.1093/bib/bbad281. URL https://doi.org/10.1093/bib/bbad281.

[23] Lars Lorch, Jonas Rothfuss, Bernhard Schölkopf, and Andreas Krause. Dibs: Differentiable bayesian structure learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[24] Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. Learning gflownets from partial episodes for improved convergence and stability. *ArXiv*, abs/2209.12782, 2022. URL https://api.semanticscholar.org/CorpusID:252531657.

[25] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets, 2022.

[26] Alexander Mitrophanov and Eduardo Groisman. Positive feedback in cellular control systems. *BioEssays : news and reviews in molecular, cellular and developmental biology*, 30:542–55, 06 2008. doi: 10.1002/bies.20769.

[27] Kevin Murphy. Active learning of causal bayes net structure. 06 2001.

[28] Trang Nguyen, Amin Mansouri, Kanika Madan, Khuong Nguyen, Kartik Ahuja, Dianbo Liu, and Yoshua Bengio. Reusable slotwise mechanisms, 2023.

[29] Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Paul Beaumont, Konstantinos Georgatzis, and Bryon Aragam. Dynotears: Structure learning from time-series data, 2020.

[30] Andrea Riba, Attila Oravecz, Matej Durik, Sara Jiménez, Violaine Alunni, Marie Cerciat, Matthieu Jung, Céline Keime, William Keyes, and Nacho Molina. Cell cycle gene regulation dynamics revealed by rna velocity and deep-learning. *Nature Communications*, 13:2865, 05 2022. doi: 10.1038/s41467-022-30545-8.

[31] Swarup Roy, Dipankar Das, Dhrubajyoti Choudhury, Gunenja G. Gohain, Ramesh Sharma, and Dhruba K. Bhattacharyya. Causality inference techniques for in-silico gene regulatory network. In Rajendra Prasath and T. Kathirvalavakumar, editors, *Mining Intelligence and Knowledge Exploration*, pages 432–443, Cham, 2013. Springer International Publishing. ISBN 978-3-319-03844-5.

A Related Work

Bayesian Structure Learning Recent advancements in differentiable Bayesian methods for static structure learning, including DiBS [23], BCD-Nets [13], and DAG-GFlowNet [15], offer diverse graph parameterization approaches. While they excel in modeling uncertainty and structural distributions in smaller graphs, challenges arise when assuming natural dynamical systems adhere to DAGs, particularly when cyclic structures from feedback mechanisms complicate the search space.

Dynamic and Cyclic Structure Learning Dynamic and cyclic structure learning has seen limited development. Before DynGFN [3], the closest existing work in this area is CD-NOD [19], which has shown potential for extension to handle cyclic graphs besides the original purpose of harnessing non-stationary data to unveil causal relationships in scenarios with changing generative processes over time. In contrast, traditional approaches to capturing intricate and uncertainty, including NeuralODEs [9] that propose a sole explanatory structure, and DYNOTEARS [29] is a score-based approach for learning structure from time-series data.

Causal inference in large-scale biological systems Among attempts to inferencing causal relations in large-scale biological systems, various methodologies have been developed in causal inference within large-scale biological systems. These approaches can be broadly categorized into three main groups: constraint-based methods [11, 12], score-based methods [10, 21], and hybrid methods [7]. Constraint-based methods discern causal relationships by detecting statistical dependencies and independencies within the data. In contrast, score-based methods assign a score to each conceivable causal structure and select the structure with the most favorable score. On the other hand, hybrid methods amalgamate elements from constraint-based and score-based methodologies.

Recent years have witnessed a burgeoning interest in applying these causal inference techniques to extensive biological datasets [10]. For instance, researchers have harnessed these methods to unearth gene regulatory networks from gene expression data [30]. These networks offer valuable insights into the regulatory mechanisms steering gene expression and can pinpoint potential targets for therapeutic interventions. Despite these noteworthy advancements, several formidable challenges persist in this domain. Among the principal challenges is grappling with confounding variables that can introduce spurious causal connections. Additionally, addressing missing data, a common issue in biological datasets remains a substantial hurdle to overcome.

B Bayesian dynamic structure learning framework

$$p(G, \theta, \mathcal{D}) = p(\mathcal{D}|G, \theta)p(\theta|G)p(G) \tag{4}$$

We adopt the framework from Atanackovic et al. [3] that decomposes the generative model as in Equation 4. Subsequently, we employ the GFlowNet architecture to learn $P(G)$. Finally, we utilize the linear differential form $\frac{dx}{dt} = \mathbf{A}x$ to approximate the optimal θ to parameterize $P(\theta|G)$.

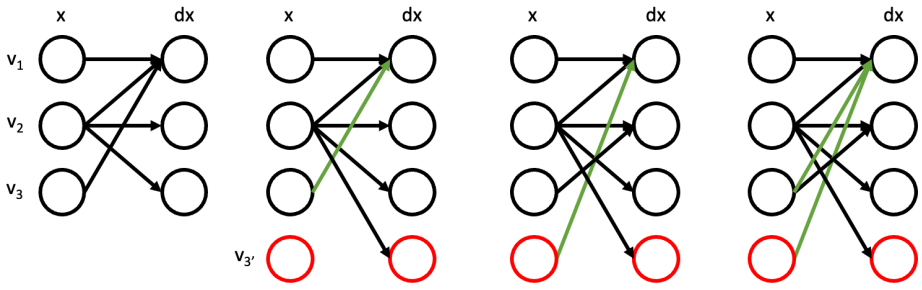

Figure 3: **The model of indeterminacy.** A new variable that mirrors the values of v_3 is added, introducing three potential explanations for the data (in green). The visualization is adopted from DynGFN paper [3].

C The Model of Indeterminacy

In this section, we summarize the model of indeterminacy, a strategy to generate the synthetic dataset investigated in this work, proposed by Atanackovic et al. [3] and visualized in Figure 3.

The purpose of the model of indeterminacy is to formulate a structure learning problem that contains many equivalent causal structures. Given the context of a dataset comprising pairs $(x, dx) \in \mathbb{R}^d \times \mathbb{R}^d$, which contains d variables, a new variable is introduced to have $d + 1$ variables in total. Specifically, the new variable replicates or is highly correlated with an existing variable v and inherits the same parents as v. This extension results in the emergence of various potential explanatory graphs. Finally, a sparsity penalty is applied to constrain the number of edges consistent in a valid graph and the number of possible cases.

D Extended of Potential Future Work

Decomposition The current approach relies heavily on a monolithic design, which may not always be conducive to achieving a high level of generalization. In future work, we plan to develop a strategy that balances decomposition and parameter management. For instance, we aim to incorporate reusable mechanisms proposed in RSM [28] to enhance the overall design.

Design variants As illustrated in Figure 2, although our proposed method exhibits superior scalability compared to other baselines, the performance of all models only marginally surpasses the random prediction threshold, as measured by the AUC metric. Therefore, further investigations and improvements in prediction quality are necessary. In addition to considering decomposition in the design, we will explore other avenues, such as enhancing sampling methods and the associated probability calculations, experimenting with alternative backbone models for GFlowNet beyond MLP, and exploring the potential benefits of fine-tuning.

Algorithm 1 Batch update training in Swift-DynGFN

Input: Data batch (x_b, dx_b)
B: batch of graphs computed parallelly
n: number of **variables**

GFlowNet Architecture
h_{id}, h_g: encoded sizes of node's index and the whole graph, respectively
h: hidden size
$h = h_{id} + h_g$
$\mathbf{MLP_{id}} : \mathbb{R}^n \to \mathbb{R}^{h_{id}}$
$\mathbf{MLP_g} : \mathbb{R}^{n \times n} \to \mathbb{R}^{h_g}$
$\mathbf{MLP_{FW}} : \mathbb{R}^h \to \mathbb{R}^{n+1}$ ▷ n dimensions for forward probability, and the last dimension for state flow

Model's computation flow
Inputs: $node\_id \in \mathbb{R}^{B \times n}$, $graphs \in \mathbb{R}^{B \times n \times n}$
$rep_{id} = \mathbf{MLP_{id}}(node\_id)$
$rep_g = \mathbf{MLP_g}(graphs)$
$rep = \text{cat}(rep_{id},\ rep_g,\ dim = -1)$
$pred = \mathbf{MLP_{FW}}(rep)$
$log\_forward = pred[:, :-1].\text{log\_softmax}()$
$log\_flow = pred[:, -1:]$
$log\_backward = \mathbf{0}_{B \times n}.\text{log\_softmax}()$ ▷ Uniform backward
Outputs: $log\_forward \in \mathbb{R}^{B \times n}$, $log\_backward \in \mathbb{R}^{B \times n}$, $flow \in \mathbb{R}^{B \times 1}$

Step 0. Initialization
$graphs \leftarrow \mathbf{0}_{B \times n \times n}$ ▷ Empty B graphs
$node\_id \leftarrow \mathbf{0}_{B \times n}$ ▷ Empty id of node of interest
$done\_mask \leftarrow \mathbf{0}_{B \times n}$ ▷ Empty visited mask
$ll\_diff \leftarrow \mathbf{0}_{n \times B}$

Training pipeline
for i in $0 \dots n$ **do** ▷ $n + 1$ times of execution
    Step 1. GFlowNet computation
    $log\_forward,\ log\_backward,\ log\_flow = \mathbf{model}(node\_id, graphs)$

    if i > 0 **then** ▷ Excluding the starting node
        Step 2. Sampling *all* incoming edges
        $actions \leftarrow \text{sample\_all\_incomming\_edges(log\_forward)}$
        $graphs[:, node\_id] \leftarrow actions$

    Step 3. Updating flows
    $ll\_diff[i] += log\_flow$
    $ll\_diff[i] += log\_forward.\text{gather}(actions)$
    if i > 0 **then**
        $ll\_diff[i-1] -= log\_flow$
        $ll\_diff[i-1] -= log\_backward.\text{gather}(actions')$
        $actions' \leftarrow actions$ ▷ Preparing for updating flows in the next turn
    if i == n **then** ▷ Reaching the last turn
        $log\_rewards \leftarrow -||dx_b - \widehat{dx_b}||_2^2 + \lambda_0 ||graphs||_0$
        $ll\_diff[node\_id] -= log\_rewards$
    else
        Step 4. Sampling the next node of interest
        $node\_id = (log\_forward - done\_mask \times \inf).\text{argmax}()$ ▷ Binary matrix
        $done\_mask += node\_id$

Optimization
$\mathcal{L}_{DB} = ll\_diff^2.\text{mean}()$