Evaluating the Robustness of Causal Discovery Algorithms with Observations and Interventions in VNF Deployments

Tianzhu Zhang[1] **Davide Rinaldi**[1] **Fabio Pianese**[1] **Armen Aghasaryan**[1]

[1]AI Research Lab, ML & Systems Department, Nokia Bell Labs, Paris-Saclay, France

Abstract

Causal discovery (CD) incorporates a large collection of interdisciplinary research endeavors from statistics, computer science, and philosophy to uncover the true causal relationship from data and move beyond mere correlations to expose the underlying data generation mechanism. Despite the rich set of causal discovery algorithms, they also bear some common limitations, including demanding assumptions and lack of validation using real-world data, making their applicability in real systems questionable. This paper explores the practical challenges of performing causal discovery in real systems. We construct a controllable Network Function Virtualization (NFV) system that allows the deployment and perturbation of interconnected topologies of high-performance Virtual Network Functions (VNFs). Our contribution is a comparison of the ability of state-of-the-art CD algorithms to reconstruct the correct causal configuration from data in observational and interventional settings.

1 INTRODUCTION

In recent years, network softwarization techniques, especially NFV [Martins et al., 2014], are increasingly applied to deploy and provision high-speed, near real-time services, such as O-RAN [2018]. However, despite the advantages, such as enhanced service agility and reduced cost-of-ownership, NFV-powered network systems also bear some intrinsic drawbacks. In particular, the shared soft data planes are more susceptible to intermittent resource and operation contentions, making it challenging to pinpoint performance bottlenecks, especially in large-scale networks. Although machine learning models can make accurate predictions by characterizing correlations, they are quite limited in interpreting the underlying causal mechanisms and answering counterfactual queries, which are essential for managing production network systems. Causal discovery (CD) provides a promising alternative for network service and infrastructure providers to identify performance bottlenecks and fulfill Service-Level Agreements (SLAs). Compared to earlier attempts to apply CD for IT monitoring [Aït-Bachir et al., 2023], the VNF use case streamlines the collection of data about network events: instead of having to deal with possibly out-of-sync time series of heterogeneous indicators such as CPU load, RAM utilization, numbers of users, disk activity, etc., the reliance on individual throughput figures of simple network functions provides a more uniform and reliable gauge of system performance.

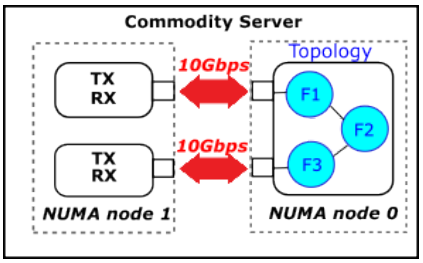


Figure 1: Testbed configuration

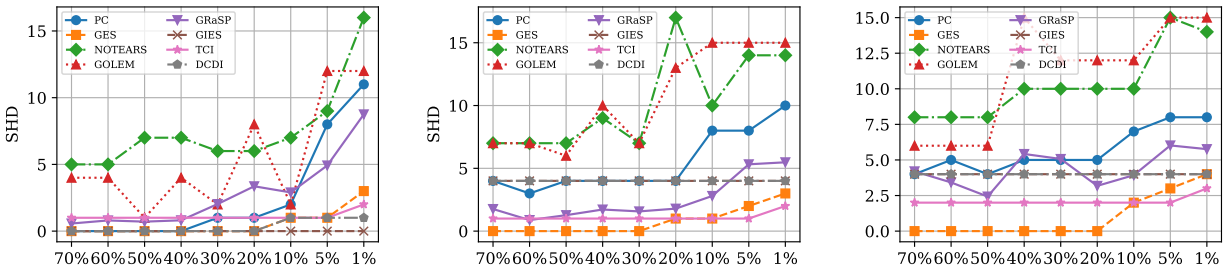
2 METHODOLOGY

We developed a testbed environment (Figure 1) to produce measurement data from high-speed network services with different topologies of VNF that are instrumented with a set of controllable sources of interference. The testbed allows for generating datasets with variable noise levels, suited for purely observational and intervention-based CD. The testbed and datasets are described in the supplementary material.

Our CD investigation relies on two key assumptions: we do not consider the time dependency between samples, and *causal sufficiency*, i.e., the measured variables include all common confounders. The latter hypothesis is reasonable for our testbed as spurious correlations among the CPU

Table 1: Causal Discovery Algorithms Covered in Our Study

Data	Class	Algorithm	Hyperparameters	Best SHD			Reference
				Linear	DAG-1	DAG-2	
Observational	Constraint-based	PC	CI test, significance α	0	3	4	[Spirtes et al., 2000]
	Constrained FCM	ICA-LiNGAM	-	14	14	16	[Shimizu et al., 2006]
		DirectLiNGAM	independence measure	13	12	13	[Shimizu et al., 2011]
	Score based	GES	max parents, score, reg	**0**	**0**	**0**	[Chickering, 2002]
		NOTEARS	loss, lr, reg	5	7	8	[Zheng et al., 2018]
		GOLEM	loss, lr, reg	2	6	6	[Ng et al., 2020]
	Permutation based	GRaSP	max parents, score	1	1	3	[Lam et al., 2022]
Interventional	Baseline	TCI	KL-div. threshold	1	1	2	Current paper
	Score based	GIES	score, reg	0	4	4	[Hauser and Bühlmann, 2012]
		DCDI	lr, reg, mlp	0	4	4	[Brouillard et al., 2020]
Abbreviations: lr = learning rate; reg = regularization parameter(s);							



(a) Linear Topology (b) DAG-1 Topology (c) DAG-2 Topology

Figure 2: SHD of the studied CD algorithms under different noise levels (on the abscissa)

cores are minimized by design. Moreover, data are aggregated on a 10s timescale: the time constants of the system being in the order of tens of ms, we deal with a steady-state system. Guided by these assumptions, we tested a selection of CD algorithms, reported in Table 1. The algorithm choice covers the main approaches: constraint-based, score-based, based on constrained functional causal models (FCM), and permutation-based. For purely observational CD algorithms implementation, we mostly relied on the causal-learn python library [Zheng et al., 2023]. We employed gCastle [Zhang et al., 2021] for the gradient-based approaches. For CD from interventional data, we considered the prototypical implementations by Brouillard et al. [2020]. We introduce here the Transitive Closure Induction (TCI) algorithm, which represents our baseline algorithm for intervention-based causal discovery. Further details, including our hyperparameter tuning procedures, can be found in the additional material.

3 RESULTS AND CONCLUSIONS

We conducted tests on three service topologies (see supplementary material). The linear topology was used for hyperparameter optimization, while the results were evaluated on DAG-1 and DAG-2. Such an evaluation strategy is plausible in a situation where a cloud provider conducts black-box testing on a live NFV system, where the network topology is typically unknown and only limited outside interference is allowed for experimentation. We assessed the accuracy of causal discovery using the *Structural Hamming Distance (SHD)* in Figure 2 and reported the best SHDs in Table 1. We also investigated the impact of controlled noise on causal assumptions. As the noise in the data generation process increased, the algorithms better captured dependencies among variables. This resulted in improved performance across all observational CD algorithms. Intervention-based algorithms remained unaffected by the noise level in our tests. Among purely observational algorithms, GES achieved the best results by accurately retrieving the ground-truth graph in both scenarios. On the other hand, for intervention-based CD algorithms, TCI demonstrated superior performance. It is worth noting that methods relying on more parameters, such as DCDI, exhibited sensitivity to the initial parameter choice and overfitted the linear topology. Since the system does not satisfy all the FCM assumptions required by the algorithm, LiNGAM failed to orient edges correctly.

Overall, our testbed investigation shows promising initial results. Classic algorithms, e.g., GES and PC, retrieve most edges correctly and generalize well across network topologies. However, further work is needed to extend this CD methodology to more complex network configurations.

References

Ali Aït-Bachir, Charles K. Assaad, Christophe de Bignicourt, Emilie Devijver, Simon Ferreira, Éric Gaussier, Hosein Mohanna, and Lei Zan. Case studies of causal discovery from it monitoring time series. *Proceedings of UAI 2023 Workshop on The History and Development of Search Methods for Causal Structure*, abs/2307.15678, 2023. URL `https://arxiv.org/abs/2307.15678`.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and Alexandre Drouin. Differentiable causal discovery from interventional data. *Advances in Neural Information Processing Systems*, 33: 21865–21877, 2020.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.

Paul Emmerich, Sebastian Gallenmüller, Daniel Raumer, Florian Wohlfart, and Georg Carle. MoonGen: A Scriptable High-Speed Packet Generator. In *Internet Measurement Conference 2015 (IMC'15)*, Tokyo, Japan, October 2015.

Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. *The Journal of Machine Learning Research*, 13(1):2409–2464, 2012.

Wai-Yin Lam, Bryan Andrews, and Joseph Ramsey. Greedy relaxations of the sparsest permutation algorithm. In *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 1052–1062, 2022.

Joao Martins, Mohamed Ahmed, Costin Raiciu, Vladimir Olteanu, Michio Honda, Roberto Bifulco, and Felipe Huici. {ClickOS} and the art of network function virtualization. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 459–473, 2014.

Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging {AI} applications. In *13th USENIX symposium on operating systems design and implementation (OSDI 18)*, pages 561–577, 2018.

Ignavier Ng, AmirEmad Ghassami, and Kun Zhang. On the role of sparsity and dag constraints for learning linear dags. *Advances in Neural Information Processing Systems*, 33:17943–17954, 2020.

O-RAN. "Open RAN Alliance", 2018. URL `https://www.o-ran.org/`.

Shohei Shimizu, Patrik O Hoyer, Aapo Hyvärinen, Antti Kerminen, and Michael Jordan. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10), 2006.

Shohei Shimizu, Takanori Inazumi, Yasuhiro Sogawa, Aapo Hyvarinen, Yoshinobu Kawahara, Takashi Washio, Patrik O Hoyer, Kenneth Bollen, and Patrik Hoyer. DirectLiNGAM: A direct method for learning a linear non-gaussian structural equation model. *Journal of Machine Learning Research-JMLR*, 12(Apr):1225–1248, 2011.

Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000.

Keli Zhang, Shengyu Zhu, Marcus Kalander, Ignavier Ng, Junjian Ye, Zhitang Chen, and Lujia Pan. gcastle: A python toolbox for causal discovery. *arXiv preprint arXiv:2111.15155*, 2021.

Wei Zhang, Guyue Liu, Wenhui Zhang, Neel Shah, Phillip Lopreiato, Gregoire Todeschi, KK Ramakrishnan, and Timothy Wood. OpenNetVM: A platform for high performance network service chains. In *Proceedings of the 2016 workshop on Hot topics in Middleboxes and Network Function Virtualization*, pages 26–31, 2016.

Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31, 2018.

Yujia Zheng, Biwei Huang, Wei Chen, Joseph Ramsey, Mingming Gong, Ruichu Cai, Shohei Shimizu, Peter Spirtes, and Kun Zhang. Causal-learn: Causal discovery in python. *arXiv preprint arXiv:2307.16405*, 2023.

Evaluating the Robustness of Causal Discovery Algorithms with Observations and Interventions in VNF Deployments (Supplementary Material)

Tianzhu Zhang[1] **Davide Rinaldi**[1] **Fabio Pianese**[1] **Armen Aghasaryan**[1]

[1]AI Research Lab, ML & Systems Department, Nokia Bell Labs, Paris-Saclay, France

A TESTBED, TOPOLOGIES, AND TRACE COLLECTION

Our testbed is deployed on a commodity server (Figure 1) with two discrete processors belonging to different Non-Uniform Memory Access (NUMA) [1] nodes. On one processor, we run MoonGen as the high-speed network traffic generator / monitor [Emmerich et al., 2015], whereas the network service topology under test is deployed on the other processor. We employ OpenNetVM [Zhang et al., 2016], a high-performance Network Function Virtualization platform that enables flexible and rapid deployment of common VNFs, such as firewalls and routers. Communication between the two NUMA nodes is realized via a pair of direct fiber-optic cables connecting the network interfaces. The testbed allows us to run experiments at the line rate of 10 Gbps, which is consistent with data center workloads.

For this paper, we work with three sample service topologies drawn from realistic use cases, depicted in Fig. 3. The leftmost "linear" topology implements a firewall and deep packet inspection (DPI) before routing the packet and scanning its payload on the way out to the bridge. The second "directed acyclic graph" (DAG-1) topology branches out after the firewall and splits the traffic between a DPI branch and a payload scan branch, both feeding into the bridge. The third topology (DAG-2) features a 3-way branch. Each virtual network function is executed on a different CPU core, ensuring mutual isolation of the computing resources. The functions communicate via shared memory, passing pointers to the packets stored in RAM as the processing follows the topology graph. At every network function, we introduce a controlled noise source as a parasite process that can consume a configurable share of the CPU core cycles. This device allows generating datasets with variable noise levels to emulate distribution shift and (soft) interventions. We utilize the observational algorithms implemented by two open-source libraries, namely Causal-learn [Zheng et al., 2023] and gCastle [Zhang et al., 2021].

Our testbed allows generating datasets with variable noise levels to emulate distribution shift and soft interventions, suited both for purely observational and intervention-based CD. We gather an observational dataset comprising 2000 measures of VNF variables for each topology configuration. Controlled noise is added to the input variable by changing its throughput according to a uniform distribution at regular 10-second intervals. Furthermore, we add a parasite process to each CPU core at the same intervals, based on an independent Bernoulli draw with a probability of 0.5. We compile different datasets corresponding to varying CPU shares (nine levels from 1% to 70%). This allows us to analyze the impact of noise on CD accuracy. Additionally, we compile a family of interventional datasets for each topology, obtained by intervening systematically on one variable at a time. We collect 200 samples for each intervened variable. Depending on the variable, interventions consist of limiting the available CPU share or the throughput. Also, in this scenario, we collect different datasets for each level of CPU noise, which are used for our main results.

B HYPERPARAMETERS AND TUNING

The performance of most CD algorithms depends heavily on the hyperparameter's tuning, which can be a delicate process due to the unsupervised nature of CD. In this study, we opted to tune the parameters for the linear topology and conducted a comparative evaluation of the remaining two topologies. We employed the library Optuna by Akiba et al. [2019], an automatic

[1]Each discrete processor has an independent memory subsystem connected via the PCIe bus.

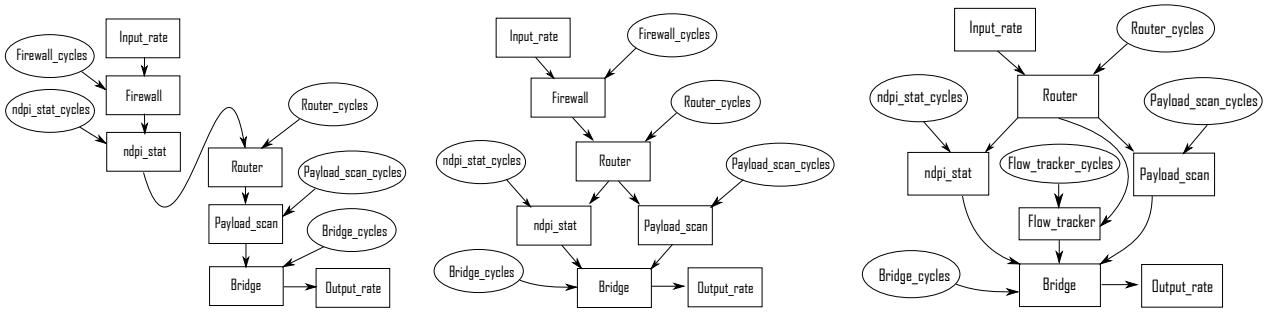
Table 2: Optimized Set of Hyperparameters

Algorithm	Hyperparameters
PC	α : 0.01, independence_test: fisherz, stable: True, (uc_rule: 0, uc_priority: 2, mvpc: False)
DirectLiNGAM	measure: pwling
GES	maxP: $\frac{\#variables}{2}$, score_func: local_score_BIC
NOTEARS	$\lambda 1$: 0.433, loss_type: l2, h_tol: 0.0099348, rho_max: 5.27026, max_iter: 1000, w_threshold: 0.038
GOLEM	λ_1 : 0.0977705, λ_2 : 0.14846, equal_variances: False, non_equal_variances: False, lr: 0.008, graph_thres: 0.95
GRaSP	maxP: $\frac{\#variables}{2}$, score_func: local_score_BIC
GIES	lambda-gies: 5
TCI	threshold: 1
DCDI	model: DCDI-G, num-layers: 2, hid-dim: 16, intervention: True optimizer: rmsprop, lr: 1e-3, reg-coeff: 0.8, h-threshold: 1e-8

hyperparameter optimization framework designed to find optimal hyperparameters efficiently. We used Ray [Moritz et al., 2018], a distributed computing framework that can scale ML workloads to accelerate tuning. We further adopted the Asynchronous Successive Halving Algorithm (ASHA), which features early stopping in large hyperparameter search spaces. We evaluate the causal discovery accuracy using the *Structural Hamming Distance (SHD)*, a classic metric to quantify the difference between two causal DAGs. Similar to the Hamming distance measuring the different bits of two equal-length strings in information theory, SHD accounts for the total missing, extra, and incorrect edges. Table 1 lists the hyperparameters tuned for each algorithm.

C TRANSITIVE CLOSURE INDUCTION

We introduce the Transitive Closure Induction (TCI) technique as a straightforward baseline benchmark for topology discovery. It aims to identify subsets of variables that exhibit variation together for each intervention. This variation is measured using a threshold on Kullback-Leibler divergence, assuming the Gaussian distribution of variables. The subsets are organized in a lattice structure, partially ordered by set inclusion. The output graph is obtained by computing the transitive reduction of this lattice. While the justification of TCI within the causal framework may not be fully established, it remains a valuable tool for retrieving the network VNF topology whenever it is transitively reduced. This hypothesis is met in our sample networks, and TCI demonstrated superior performance compared to many state-of-the-art algorithms.



(a) Linear (b) DAG-1 (c) DAG-2

Figure 3: The three VNF topologies used in our experiments