Supplementary Material for "Adaptive Skeleton Construction for Accurate DAG Learning"

Xianjie Guo, Kui Yu*, Lin Liu, Peipei Li, and Jiuyong Li

1 Experiments on synthetic data

1.1 Comparison methods.

We compare the ADL algorithm with five representative local-to-global DAG learning algorithms, including GSBN[1] [1], MMHC[2] [2], SLL+C/G[3] [3] and GGSL [4], and seven well-established and state-of-the-art global DAG learning algorithms, including K2[4] [5], OBS [6], Improved-K2 [7], NOTEARS[5] [8], DAG-GNN[6] [9], GOLEM[7] [10] and DAG-NoCurl[8] [11].

1.2 Datasets.

We use an open-source toolkit [12] to first construct a DAG with 5,000 vertices, and then based on the causal mechanism shown in Eq. (1), we generate three datasets with 500, 1,000 and 5,000 samples, respectively.

$$Polynomial: y = (W_0 + \mathbf{X}W_1 + ... + \mathbf{X}^d W_d) + E, \quad (1)$$

where $\mathbf{X}$ denotes the vector of causes, E represents the noise variable accounting for all unobserved variables, and W_i $(i = 1, 2, .., d)$ is the weight coefficient vector. In our experiment, we add 40% Gaussian noise to generate datasets.

- *X. Guo, K. Yu, and P. Li are with the Key Laboratory of Knowledge Engineering With Big Data of Ministry of Education, Hefei University of Technology, Hefei 230601, China, also with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230601, China; e-mail: xianjieguo@mail.hfut.edu.cn, {yukui,peipeili}@hfut.edu.cn (*Corresponding author: Kui Yu).*
- *L. Liu and J. Li are with the UniSA STEM, University of South Australia, Adelaide 5095, Australia; e-mail: {Lin.Liu,Jiuyong.Li}@unisa.edu.au.*

*Manuscript received * *, *; revised * *, *.*

1. The source codes of GSBN are available at https://github.com/kuiy/CausalLearner.
2. The source codes of MMHC are available at http://mensxmachina.org/en/software/probabilistic-graphical-model-toolbox.
3. The source codes of SLL+C/G are available at https://www.cs.helsinki.fi/u/tzniinim/uai2012.
4. The source codes of K2 are available at https://github.com/bayesnet/bnt.
5. The implementation is publicly available at https://github.com/xunzheng/notears.
6. The code is available at https://github.com/fishmoon1234/DAG-GNN.
7. The code is available at https://github.com/ignavierng/golem.
8. The code is available at https://github.com/fishmoon1234/DAG-NoCurl.

1.3 Evaluation metrics.

We evaluate the performance of ADL and its rivals from three aspects: structure errors, structure correctness and efficiency. The *SHD* (Structural Hamming Distance) and *Ar_F1* metrics as shown below are used to measure structure error and structure correctness, respectively. The running time is utilized as the efficiency measure of the algorithms.

- *SHD* (Structural Hamming Distance) is the sum of the values of *Miss*, *Extra*, *Reverse* and *Undirected*, where *Miss* is the number of missing edges in the DAG learned by an algorithm against the true DAG, *Extra* is the number of extra edges in the learned DAG, *Reverse* is the number of edges with wrong directions according to the true DAG, and *Undirected* is the number of undirected edges in the learned DAG. In our experiments, we randomly orient the edges that cannot be oriented by a DAG learning algorithm, thus the value of *Undirected* is always 0 and Undirected metric is not shown in the experimental results. The smaller the value of SHD the better.
- Ar_F1=$\frac{2*Ar\_Precision*Ar\_Recall}{Ar\_Precision+Ar\_Recall}$. The $Ar\_Precision$ denotes the number of correctly predicted arrowheads in the output divided by the number of edges in the output of an algorithm, while the Ar_Recall denotes the number of correctly predicted arrowheads in the output divided by the number of true arrowheads in a test DAG. Compared to SHD, Ar_F1 not only considers erroneous edges, but also correct edges. A larger value of Ar_F1 is better.

1.4 Implementation details.

All experiments were conducted on a computer with Inter Core i9-10900 3.70-GHz CPU, NVIDIA GeForce RTX 3060 GPU and 64-GB memory. The significance level for conditional independence tests is set to 0.01. For continuous optimization based DAG learning methods (i.e., NOTEARS, DAG-GNN, GOLEM and DAG-NoCurl), we adopt 0.3 as the threshold to prune the DAGs obtained from those methods [10]. K2, OBS, Improved-K2, GSBN, MMHC and our algorithm are implemented in MATLAB, SLL+C/G and GGSL are implemented in C++, and NOTEARS, DAG-GNN, GOLEM and DAG-NoCurl are implemented in PYTHON.

1.5 Results of DAG learning on synthetic data

The experimental results on synthetic datasets are shown in Tables 1-3 below. In Tables 1-3, the symbol "-" denotes that an algorithm does not produce results on the corresponding dataset when the running time of the algorithm exceeded 12 hours or there is no enough memory space, and the best results are highlighted in bold face.

From Tables 1-3, we can see that only the MMHC algorithm and our algorithm can produce results on such a high-dimensional dataset, and our algorithm is always better than the MMHC algorithm on both effectiveness and efficiency. The experimental results on synthetic datasets further illustrate the scalability of our proposed algorithm.

2 Two Exemplars Learned by Our Method

In this section, we show two exemplar DAGs learned by our proposed algorithm with the marked missed, extra, reversed edges.

We first use one benchmark Bayesian network (BN), Child[9], to randomly generate two datasets, including 500 data samples and 1,000 data samples, respectively.

In the following, Fig. 1 shows the true DAG of Child network, Fig. 2 shows the DAG learned by our proposed method from the dataset with 500 samples, and Fig. 3 shows the DAG learned by our proposed method from the dataset with 1,000 samples.

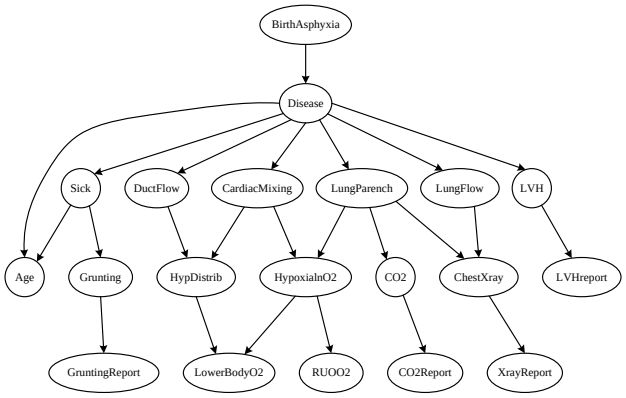

Fig. 1. The true Child network.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China (under grant 2021ZD0111801), in part by the Australian Research Council (under grants DP200101210, DP230101122).

References

[1] D. Margaritis and S. Thrun, "Bayesian network induction via local neighborhoods," *Advances in Neural Information Processing Systems*, vol. 12, 1999.

[2] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," *Machine Learning*, vol. 65, no. 1, pp. 31–78, 2006.

9. This benchmark BN is publicly available at http://www.bnlearn.com/bnrepository/.

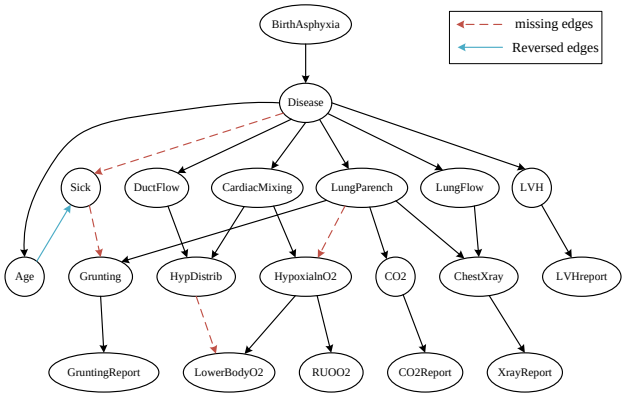

Fig. 2. The network learned by the ADL algorithm on Child with 500 samples.

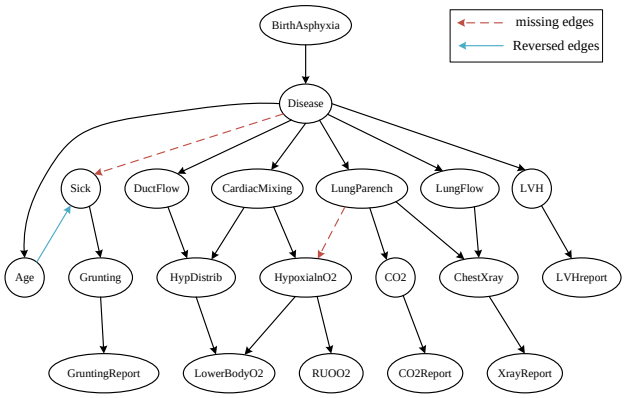

Fig. 3. The network learned by the ADL algorithm on Child with 1,000 samples.

[3] T. Niinimaki and P. Parviainen, "Local structure discovery in Bayesian networks," in *Proceedings of Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2012, pp. 634–643.

[4] T. Gao, K. Fadnis, and M. Campbell, "Local-to-global Bayesian network structure learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1193–1202.

[5] G. F. Cooper and E. Herskovits, "A Bayesian method for the induction of probabilistic networks from data," *Machine Learning*, vol. 9, no. 4, pp. 309–347, 1992.

[6] M. Teyssier and D. Koller, "Ordering-based search: a simple and effective algorithm for learning Bayesian networks," in *Proceedings of Conference on Uncertainty in Artificial Intelligence*, 2005, pp. 584–590.

[7] S. Behjati and H. Beigy, "Improved K2 algorithm for Bayesian network structure learning," *Engineering Applications of Artificial Intelligence*, vol. 91, p. 103617, 2020.

[8] X. Zheng, B. Aragam, P. K. Ravikumar, and E. P. Xing, "Dags with no tears: Continuous optimization for structure learning," *Advances in Neural Information Processing Systems*, vol. 31, pp. 9472–9483, 2018.

[9] Y. Yu, J. Chen, T. Gao, and M. Yu, "DAG-GNN: DAG structure learning with graph neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7154–7163.

[10] I. Ng, A. Ghassami, and K. Zhang, "On the role of sparsity and DAG constraints for learning linear DAGs," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 943–17 954, 2020.

[11] Y. Yu, T. Gao, N. Yin, and Q. Ji, "Dags with no curl: An efficient DAG structure learning approach," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 156–12 166.

[12] D. Kalainathan and O. Goudet, "Causal discovery toolbox: Uncover causal relationships in python," *arXiv preprint arXiv:1903.02278*, 2019.

TABLE 1
Experimental results on synthetic dataset with 500 samples. (↓ means that the lower, the better while ↑ denotes the higher, the better.)

Algorithm	SHD (↓)	Reverse (↓)	Miss (↓)	Extra (↓)	Ar_F1 (↑)	Ar_Precision (↑)	Ar_Recall (↑)	Time (↓)
K2	-	-	-	-	-	-	-	-
OBS	-	-	-	-	-	-	-	-
Improved-K2	-	-	-	-	-	-	-	-
GSBN	-	-	-	-	-	-	-	-
MMHC	4685	1427	207	3051	0.518	0.423	0.668	11724.85
SLL+C	-	-	-	-	-	-	-	-
SLL+G	-	-	-	-	-	-	-	-
GGSL	-	-	-	-	-	-	-	-
NOTEARS	-	-	-	-	-	-	-	-
DAG-GNN	-	-	-	-	-	-	-	-
GOLEM	-	-	-	-	-	-	-	-
DAG-NoCurl	-	-	-	-	-	-	-	-
ADL	**4168**	**1311**	**202**	**2655**	**0.554**	**0.462**	**0.693**	**6389.86**

TABLE 2
Experimental results on synthetic dataset with 1,000 samples. (↓ means that the lower, the better while ↑ denotes the higher, the better.)

Algorithm	SHD (↓)	Reverse (↓)	Miss (↓)	Extra (↓)	Ar_F1 (↑)	Ar_Precision (↑)	Ar_Recall (↑)	Time (↓)
K2	-	-	-	-	-	-	-	-
OBS	-	-	-	-	-	-	-	-
Improved-K2	-	-	-	-	-	-	-	-
GSBN	-	-	-	-	-	-	-	-
MMHC	3580	1143	61	2376	0.612	0.514	0.755	8277.6
SLL+C	-	-	-	-	-	-	-	-
SLL+G	-	-	-	-	-	-	-	-
GGSL	-	-	-	-	-	-	-	-
NOTEARS	-	-	-	-	-	-	-	-
DAG-GNN	-	-	-	-	-	-	-	-
GOLEM	-	-	-	-	-	-	-	-
DAG-NoCurl	-	-	-	-	-	-	-	-
ADL	**3214**	**1033**	**60**	**2121**	**0.643**	**0.548**	**0.778**	**3564.41**

TABLE 3
Experimental results on synthetic dataset with 5,000 samples. (↓ means that the lower, the better while ↑ denotes the higher, the better.)

Algorithm	SHD (↓)	Reverse (↓)	Miss (↓)	Extra (↓)	Ar_F1 (↑)	Ar_Precision (↑)	Ar_Recall (↑)	Time (↓)
K2	-	-	-	-	-	-	-	-
OBS	-	-	-	-	-	-	-	-
Improved-K2	-	-	-	-	-	-	-	-
GSBN	-	-	-	-	-	-	-	-
MMHC	2235	710	**8**	1517	0.741	0.654	0.854	4842.96
SLL+C	-	-	-	-	-	-	-	-
SLL+G	-	-	-	-	-	-	-	-
GGSL	-	-	-	-	-	-	-	-
NOTEARS	-	-	-	-	-	-	-	-
DAG-GNN	-	-	-	-	-	-	-	-
GOLEM	-	-	-	-	-	-	-	-
DAG-NoCurl	-	-	-	-	-	-	-	-
ADL	**2111**	**657**	**8**	**1446**	**0.755**	**0.669**	**0.865**	**3288.38**