Towards Building a Digital Twin of Complex System using Causal Modelling

Luka Jakovljevic[1,2], Dimitre Kostadinov[1], Armen Aghasaryan[1], and Themis Palpanas[2]

[1] Nokia Bell Labs France,
luka.jakovljevic@nokia.com
[2] University of Paris, France

Abstract. Complex systems, such as communication networks, generate thousands of new data points about the system state every minute. Even if faults are rare events, they can easily propagate, which makes it challenging to distinguish root causes of errors from effects among the thousands of highly correlated alerts appearing simultaneously in high volumes of data. In this context, the need for automated Root Cause Analysis (RCA) tools emerges, along with the creation of a causal model of the real system, which can be regarded as a digital twin. The advantage of such model is twofold: (i) it assists in reasoning on the system state, given partial system observations; and (ii) it allows generating labelled synthetic data, in order to benchmark causal discovery techniques or create previously unseen faulty scenarios (counterfactual reasoning). The problem addressed in this paper is the creation of a causal model which can mimic the behavior of the real system by encoding the appearance, propagation and persistence of faults through time. The model extends Structural Causal Models (SCMs) with the use of logical noisy-OR gates to incorporate the time dimension and represent propagation behaviors. Finally, the soundness of the approach is experimentally verified by generating synthetic alert logs and discovering both the structure and parameters of the underlying causal model.

Keywords: Digital Twin, Causality, Alerts, Complex Network Modelling

1 Introduction

Complex systems such as modern telecommunication networks, or distributed embedded systems, need to be continuously monitored to allow identification of failure situations. The monitoring systems generate huge volumes of alert logs and notifications, where it is extremely challenging to distinguish real fault symptoms from noisy alerts. Given the large number of highly correlated alerts that appear simultaneously in faulty situations, one still needs to model causal relations between alerts, in order to identify root causes of faults.

Experts in charge of monitoring and troubleshooting the system usually have knowledge about the system architecture, the possible types of faults and the

connections (dependencies) between components (e.g., topology). What is lacking is the model that can encode the system behavior and assist experts in understanding and reasoning on the system state. For example, it can be used to answer questions such as: how likely (and when) alert B will be observed, given that alert A is currently present?, how two simultaneous alerts B and C, which were never observed together, will propagate?, what is the most likely cause of the observed series of alarms?, etc.

The existing tools for modelling variables with causal relations [11, 14, 21, 23] are not suitable for these purposes, since they cannot adequately represent system behavior. On the other hand, the digital twin is emerging as a paradigm which can represent relations between system components and allow simulations or reasoning in unprecedented situations [7, 1]. A health monitoring model described in [12] uses Dynamic Bayesian Network (DBN) to model relations between variables. Similarly, system graph can be inferred from correlated patterns in the data [5]. To the best of our knowledge, all these tools and frameworks are only able to describe correlations or causal relations between variables, without the ability to encode probabilities of appearance, propagation and persistence of binary events such as faults, events and notifications.

In this paper, we propose a framework for building a digital twin that models faulty system behavior based on causal relations between observable alerts. An alert can be represented as a binary time series, where active state indicates a fault on the subsequent system component. The framework takes as input historical data i.e. past observed alerts, and builds a model of the system behavior in two steps. First, it extracts the structure of the causal relations between alerts (Directed Acyclic Graph - DAG), and then learns the parameters of the dependencies which drive the system behavior (e.g., frequency and duration of alert appearance, time lag needed to observe effect given presence of cause(s), etc). Second, it builds a causal model of the system by tackling the rarity of fault issues. Multi-causal dependencies follow the noisy-OR logic [18], which enables estimating the effect of multiple causes even if they have never been observed together. The noisy-OR model is commonly used to simplify the expression and computation of the parameters of the dependency between multiple independent causes and one common effect. It has largely been used in network diagnosis and fault localization problems [2, 8, 25, 13].

The resulting digital twin can answer the above mentioned questions and has several possible usages: generation of labelled synthetic data for algorithm optimisation and tuning, reasoning on the system state given partial observations, or simulation of unobserved fault scenarios.

The contribution of this paper is twofold. First it describes the framework for building a digital twin from observed historical data; second, it shows how Structural Causal Models (SCMs) [6, 19] can be used to encode the system behavior including alert appearance, propagation and persistence.

The rest of this paper is organized as follows: Prerequisites regarding noisy-OR and SCMs are given in Section 2. Section 3 describes our framework for building a causal model of a digital twin which models faulty system behaviour.

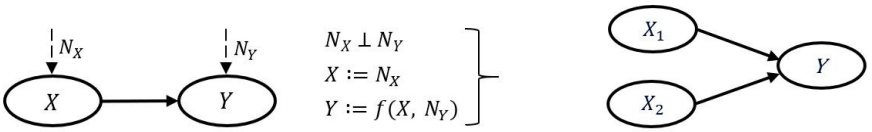
The experiments that simulate creation of a digital twin, by inferring both causal relations between variables and SCM parameters are discussed in Section 4, followed by the conclusion and future work.

2 Prerequisites

The approach presented in this paper is based on Structural Causal Models and uses noisy-OR gate to represent the impact of multiple causes on the same effect. This section briefly introduces the theory behind those two notions.

2.1 Structural Causal Models

A common way to represent causal relationships between variables is to use Structural Causal Models (SCM), referred also as Structural Equation Models (SEM) [3, 20, 10] in the case of linear relationships between the variables. Graphically, SCMs can be seen as Directed Acyclic Graphs (DAGs) [24] in which sets of Endogenous V and Exogenous U variables are connected by a set of functions F. This set of equations determine the values of the variables in V based on the values of the variables in U. They correspond to causal assumptions and can be seen as assignments, rather than mathematical equations. Intuitively, a DAG represents a flow of information, where the variables U are the inputs of the system, while the variables V are the nodes where that information is processed. Exogenous variables U correspond to unobserved influences in the model which can be treated as noise factors. In the simplest case with two variables, an SCM can be described as shown in Figure 1a, where variable Y (effect) is a child of a variable X (cause). N_X and N_Y are statistically independent noises i.e. $N_X \perp\!\!\!\perp N_Y$. Variable X depends only on a noise term N_X ($X := N_X$) while variable Y depends on the values of the parent variable X and its own noise N_Y ($Y := f(X,\ N_Y)$).



(a) Structural Causal Model (SCM) representing X →Y.

(b) Collider DAG (also called immorality or v-structure).

Fig. 1: Example of a SCM and a DAG.

2.2 Noisy-OR

The basic idea behind noisy-OR gate is to compactly represent the effect of multiple independent causes which are responsible for producing the same effect.

According to the leaky noisy-OR definition [9], equation allows the transfer of influence from multiple parent Boolean variables $x_1, \ldots, x_n$, where the child Boolean variable y can become active with probability:

$$P^{nor}(y = 1 \mid x_1, \ldots, x_n) = 1 - (1 - \lambda) \prod_{i=1}^{n} (1 - \nu_i)^{x_i} \tag{1}$$

For each $i = 1, ..., n$, the number ν_i (associated with x_i) that takes value between 0 and 1 is called **weight**. Number λ which also takes value between 0 and 1, is called **leak factor**. For the leak factor $\lambda = 0$, leaky noisy-OR function transforms to the original (standard) noisy-OR. In the following sections we will refer to the leaky noisy-OR as the noisy-OR, since in the majority of causality and Bayesian Network (BN) literature authors do the same.

As a comparison, causal probabilistic networks, also known as Bayesian Networks, would require 2^n probability parameters to express all states using Conditional Probability Tables (CPTs), where n is the number of parent variables. The main advantage of a noisy-OR gate is that it provides a practical compact representation of CPTs, by describing conditional probabilities using only $n + 1$ parameters (one parameter per parent variable plus one for the noise).

3 Causal Modelling for a Digital Twin

This section presents our framework for building a digital twin that mimics faulty behavior of a real system. First, we present steps needed to build a digital twin just by observing system alerts. Second, we define variables and rules needed to model fault appearance, propagation and persistence in time. Then, we describe how to build a causal model that encodes relations between system components. Lastly, we define steps needed to infer model structure and parameters from observational data.

3.1 Digital Twin Architecture

Many complex systems are composed of multiple interdependent components and use alerts to communicate on issues/faults in the system, where faults can propagate from one component to another. For example, fault on a base station can be caused by a fault on power equipment connected to that base station (Figure 2 - System Layers). Similarly, lack of computing resources on a server can cause faulty behavior on a base station. We separate three reasons for an alert occurrence:

- alert becomes active independently from other alerts i.e. indicates new fault in the component (case 1);
- alert becomes active due to the activity of some other alert i.e. indicates the propagation of a fault from another component (case 2);
- alert remains active after its occurrence at the previous time instant, i.e. the fault persists (case 3).

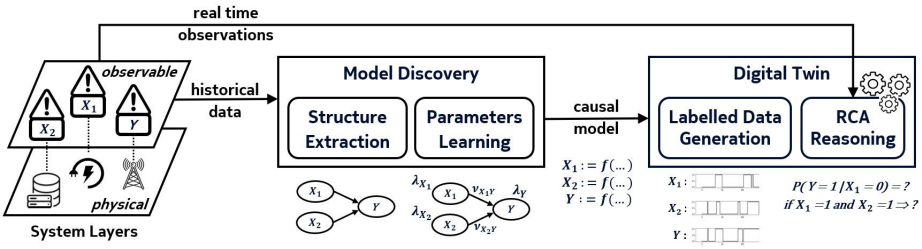


Fig. 2: Creation of a Digital twin : modelling behaviour of observable system states, enables mimicking fault appearance, propagation and persistence in time.

In order to model the system behavior and represent the three above mentioned cases, we introduce a framework for building a digital twin (Figure 2). It uses as input historical data coming from observable system layer to discover the system's model. This requires structure extraction and parameter learning. Structure extraction consists in learning causal relations between alerts. It can be done using existing causal discovery techniques [23, 22, 15, 17], which can in addition, discover causal dependency time lag (time series DAG). Then parameters can be learnt using techniques that infer conditional probability distribution of noisy-OR gates once the structure is known [16].

Once all parameters are inferred, causal model for a digital twin can be built, as will be described in more detail in the following subsections. The main advantage of having a digital twin for the real system lies in the ability to mimic system behavior. This allows (i) simulating previously unseen faulty scenarios and generating labelled data for algorithm optimization (ii) identifying root causes of faults by explaining observed alert logs using real time observations. Lastly, knowing the dynamics of fault propagation between components enables predictive maintenance tasks. Once an alert appears on a component, knowing when and which component will be affected next, can be used to predict the appearance of alerts. In the sequel, we will consider that active state of the alert corresponds to binary time series having value equal to one.

3.2 Defining Variables and Rules

In order to define the causal model used for modelling system behavior, this section starts by describing the rules used to express causal dependencies between variables, as well as the corresponding parameters. The model uses two types of endogenous SCM variables: gate variables and observed variables, where the latter correspond to system alerts. Given that the framework represents dependencies through time, all states of a given observed variable, which are involved in causal relations, are represented by a separate endogenous variable, indexed by the time instance they represent. For example, there can be two variables X_t and X_{t-1} representing the same observed variable X at two different moments, t and $t-1$ respectively. Gate variables are used to manage the probabilistic causal

dependencies between observed variables. Each gate variable has indexes for the cause variable and the effect variable that it links. When representing fault propagation, cause and effect variables have different names (and possibly different time indexes if the cause takes time to propagate). On the other hand, when representing fault persistence, the cause and the effect variables correspond to the same time series variable, just in different time states (the cause should have older time index that the effect).

Each endogenous variable has an exogenous variable associated with it, whose semantics differs, whether the endogenous variable is a gate or not. For observed (non-gate) variables, the corresponding exogenous variable controls the prior probability of turning the endogenous variable to 1 i.e. probability of a new fault, while for gate variables, it controls the propagation probability from the cause to the effect.

Once the graphical model of causal dependencies is defined, it is translated into equations based on the following rules:

- **Rule 1:** for each exogenous variable E_i, create an equation which assigns a value to it, using a random probabilistic generator with a given probability distribution. For example, a discrete random generator *disc* with parameter λ_i, denoted $disc(1-\lambda_i, \lambda_i)$ will assign to E_i a value 1 with probability λ_i and a value of 0 with probability $1-\lambda_i$:

$$E_i = disc(1-\lambda_i, \lambda_i) \tag{2}$$

- **Rule 2:** for each observed endogenous variable X_{t_i} which is not causally dependent from any other variable (i.e. is root variable), create an equation which assigns to it the value of its exogenous variable $E_{X_{t_i}}$:

$$X_t^i := E_{X_t^i} \tag{3}$$

- **Rule 3:** for each endogenous gate variable $Gate_{XY}$, create an equation which assigns a value to it using a logical AND operator over its exogenous variable $E_{Gate_{XY}}$ and the value of the input cause variable X:

$$Gate_{XY} := AND[X,\ E_{Gate_{XY}}] \tag{4}$$

- **Rule 4:** for each non root, non-gate, endogenous variable Y_t^i, create an equation which assigns a value to it using a logical OR operator over the values of all input gate variables $\{Gate_{PA_{Y_t^i} Y_t^i}\}$ and its exogenous variable $E_{Y_t^i}$:

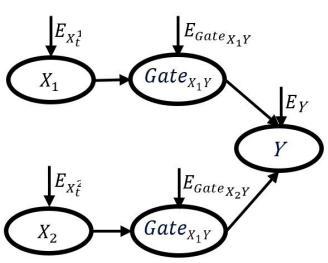
$$Y_t^i := OR[E_{Y_t^i},\ \{Gate_{PA_{Y_t^i} Y_t^i}\}] \tag{5}$$

  where PA corresponds to all parent nodes i.e. causes for variable Y_t^i.

3.3 Noisy-OR SCM Models

The digital twin takes as input a causal model, which allows modelling the three cases described in Subsection 3.1 in the following way. Probability of a new fault

(case 1) is represented by parameters of exogenous variables associated to non-gate endogenous variables, using Rule 1, which translates this probability into faults. These exogenous variables are then assigned to the non-gate endogenous variables using Rules 2 or 4, depending on whether the variable is a root or not, respectively. If variable is not a root node, the model includes propagation of a fault from parent components (case 2), incorporating also Rule 3. The SCM that models these two cases using all four rules is presented on Figure 3. This model corresponds to the noisy-OR definition (Equation 1) for collider structure on Figure 1b, where parents X_1 and X_2 independently cause child Y.



Exogenous variables:

$E_{X_1} = disc\,(1 - \lambda_{X_1}, \lambda_{X_1})$

$E_{X_2} = disc\,(1 - \lambda_{X_2}, \lambda_{X_2})$

$E_{Gate_{X_1Y}} = disc\,(1 - \nu_{X_1Y}, \nu_{X_1Y})$

$E_{Gate_{X_2Y}} = disc\,(1 - \nu_{X_2Y}, \nu_{X_2Y})$

$E_Y = disc\,(1 - \lambda_Y, \lambda_Y)$

Endogenous variables:

$X_1 := E_{X_1}$

$X_2 := E_{X_2}$

$Gate_{X_1Y} := AND[X_1, E_{Gate_{X_1Y}}]$

$Gate_{X_2Y} := AND\left[X_2, E_{Gate_{X_2Y}}\right]$

$Y := OR[E_Y, Gate_{X_1Y}, Gate_{X_2Y}]$

Fig. 3: Noisy-OR SCM.

Persistence of a fault on the same component (case 3) is realised by introducing (unobserved) endogenous gate variable between two time instances of the same variable ($Gate_{X_{t-1}X_t}$ i.e. $Gate_{Y_{t-1}Y_t}$ on Figure 4). Likewise, introducing gate variable between two time instances of different time series ($Gate_{X_{t-1}Y_t}$) creates a time lag for propagation of fault from another component. The SCM on Figure 4 combines previously described effects, therefore modelling all three cases from the Subsection 3.1. Alert Y_t can become active due to the activity of alert X_t, regarded as parent in causal graph. At the same time, both alerts have the probability of entering and maintaining their active state, independently from each other. Methodology and all equations can be easily generalized to multiple children and parents ($X_t^i;\ i = 1, ..., n$) with propagation intervals of arbitrary length $t - lag$.

3.4 Model Discovery from Observable Data

Building a digital twin requires inferring structure and parameters from historical data, which will serve as input for a causal model (Figure 2 - Model Discovery). Once the structure (DAG and *lag*-s) are properly identified using causal discovery techniques, the parameters of the SCM need to be estimated.

This consists in learning the parameters of the probability distributions used to assign values to n random variables (alerts - nodes in a DAG) which is similar to learning parameters in Bayesian Networks structure [16]. The order in which

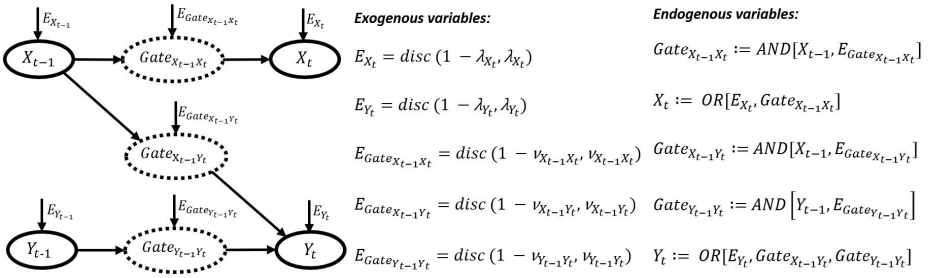


Fig. 4: Example of an SCM for modelling 2 binary time series (X_t causing Y_t) where propagation time lag between variables is 1 ($lag = 1$).

parameters are learnt is important. First, for each variable X_t^i $(i = 1, ..., n)$, the new fault appearance probability $\lambda_{X_t^i}$ is computed from all observations in which all parent variables of X_t^i are inactive, including its own past. Then, the alert persistence probability $\nu_{X_{t-lag}^i X_t^i}$ is inferred from samples where all parents of X_t^i are inactive, except the samples representing its own past (i.e. X_{t-lag}^i). Finally, alert propagation probabilities $\nu_{X_{t-lag}^i X_t^j}$ are estimated by considering the cases in which only one of the parents for X_t^j is active at a time $(j \neq i)$. In this last case, due to alert persistence, the sample representing the past of X_t^j may also be active, but as it's propagation probability has already been estimated, there is only one unknown parameter to be learnt.

4 Experiments

The goal of this section is to validate the framework's capability to infer the digital twin of a system, using just the observational data coming from the system alert logs. Synthetic datasets used in the tests are generated using the framework presented in Section 3. The experiment is performed in two phases: first, we test the capability of different causal discovery techniques to infer the causal relations between alerts (DAG) from observational data. Second, we test the ability to correctly learn noisy-OR SCM parameters, which corresponds to alert characteristics, once the structure is properly identified.

4.1 Setup

The data generation process of the proposed framework is implemented in Python. The implementation code, dataset samples and the notebooks for running the simulations are publicly available[3]. Each test is repeated 10 times, time series length is 1000 points and evaluation results are shown as average and standard deviation across the 10 runs. Tests are executed on a VM with CPU @ 2.2 GHz and 16GB of RAM.

[3]https://github.com/nokia/causal-digital-twin

Datasets The data generation process consists in 3 steps:

1) Defining the structure of the underlying model, represented as a DAG. Directed Acyclic Graphs with n nodes are randomly generated and the mean edge degree is fixed to $d = 3$ (taking into account in and out edges in directed graph). Graphs of different size n are generated with $n \in \{5, 10, 15, 20, 50\}$. DAGs are generated according to ER (Erdős–Rényi) model using probability of p to assign new edges, where p is computed as $p = \frac{d}{n-1}$.

2) Parametrizing the model is done by assigning $\lambda_{X_t^i}$ and $\nu_{X_{t-lag}^i X_t^i}$ probabilities to graph nodes and $\nu_{X_{t-lag}^i X_t^j}$ and lag are assigned to graph edges, where $i, j = 1, ..., n$. Given that some causal discovery techniques, such as PCMCI, do not apply on instantaneous propagation, lag equal to zero is not used in data generation process. DAG attributes are used to instantiate exogenous and endogenous variables of the SCM, where nodes in a DAG correspond to variables in SCM, while edges correspond to causal relation in SCM.

3) Generating a dataset with n variables, each representing a binary time series. Edges in DAG represent ground truth causal relations between variables.

Causal Discovery Techniques Four state-of-the-art multivariate causal discovery techniques are used for evaluation across the tests, as representatives of different methods for learning causal relations, as listed in Table 1.

Table 1: Causal Discovery techniques.

Algorithm	Short Description	Parameters used
PCMCI [23]	constraint-based	$\alpha = 5\%$, $\tau_{max} = 3$, CMIsymb
PCMCI+ [22]	constraint-based	$\alpha = 5\%$, $\tau_{max} = 3$, CMIsymb
TCDF [15]	convolutional neural networks	$K = 4$, $L = 0$
DYNOTEARS [17]	score-based	$\tau_W = 5\%$, $lag_{max} = 3$

Evaluation Measures The accuracy of discovered causal relations is measured based on the fact that the discovery of a directed edge in a DAG can be treated as a binary classification problem, since all techniques used in experiments output directed edges. In [4], the precision and recall (true positive rate) are defined as TP/(TP+FP) and TP/(TP+FN), respectively, where True Positives (TP) are defined as correctly identified links in the underlying DAG. Similarly for False Positives (FP) and False Negatives (FN). The $F1$ score is defined as $2 * Precision * Recall/(Precision + Recall)$. Tested causal discovery algorithms can additionally output specific time lag along with detected edge. If technique outputs wrong or multiple time lags for the same causal relation, it is not penalized i.e. only incorrectly detected edge (causal relation between variables) is penalized.

4.2 Results

Parameter range in synthetic datasets is chosen to mimic the behavior of alerts in real systems, where faults are rare events. In addition, faults propagate on interconnected components, while alerts turn off once the system recovers from faults. Probability for SCM parameters, used in experiments, are listed in Table 2, where uncertainty of 5% is introduced in order to have more probabilistic scenario.

Table 2: Parameter ranges for the tests in Figure 5.

Parameter	**Value range**	**Description**
$\lambda_{X_t^i}$	(0 - 0.05]	probability of a new fault on X_t^i
$\nu_{X_{t-lag}^i X_t^i}$	(0 - 0.05]	probability that fault persists on X_t^i
$\nu_{X_{t-lag}^i X_t^j}$	[0.95 - 1.0]	probability of a fault propagating from X_t^i to X_t^j
lag	[1,2,3]	causal dependency time lag

First, the ability of causal discovery techniques to detect underlying causal graph from observational datasets is tested. Figure 5 represents the F1, precision and recall scores of different algorithms for various graph sizes. PCMCI+ and PCMCI are able to correctly identify a DAG and maintain high F1 score as graph size grows. TCDF has high precision for small graph structures, although this technique suffers from lower accuracy as the number of variables increases. DYNOTEARS has stable precision across datasets of different sizes, yet suffers from low recall, since it only detects around 20% of edges from the ground truth set. Second, given correctly identified graph structure, our approach using Maximum Likelihood Estimation (MLE) is able to learn the SCM parameters asymptotically as the number of time samples increases (RMSE of ˜1% for variable length of 100k).

5 Conclusions

This work proposes a general-purpose framework for modelling faulty behaviors in complex systems. The model relies on noisy-OR logic components combined into a causal DAG structure by using Structural Causal Models (SCMs). Explicit representation of fault propagation rules characterizes the impact of multiple causes on the same node, while introduction of lagged variables in SCM incorporates the time dimension. Our approach enables modelling the appearance, propagation and persistence of faults, which is suitable for representing multivariate binary time series such as alerts, events, or notifications. Synthetic datasets are generated from this model, showcasing that we are able to recover both causal relations and parameters in the SCM only from observational data, which allows creation of digital twins for the real systems. Our framework can

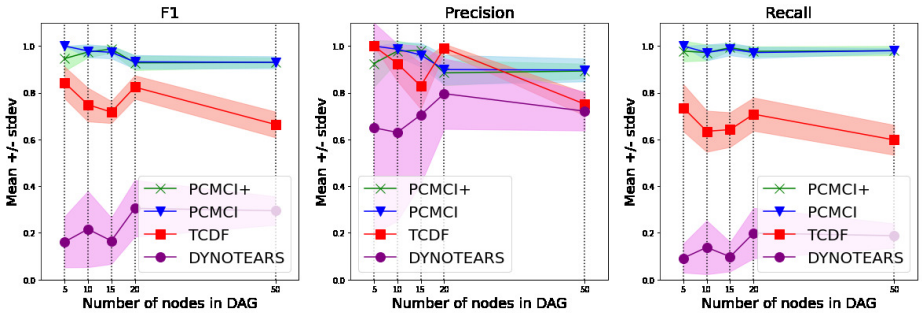


Fig. 5: Accuracy and execution time for different graph sizes ($d = 3$ and SCM parameters are listed in Table 2).

be broadened to allow modelling categorical and continuous time series by using generalizations of the noisy-OR gate. In our future work, we will study extensions to enable changes in causal graph structure and function parameters, which would allow reasoning on interventions, counterfactuals and distribution shifts. In particular, structure changes and distribution shifts can model occurrences of severe fault modes preceded by normal situations with recurrent minor alarms. Last but not the least, we are planning evaluation campaigns of our framework confronted to alarm data collected from deployed communication networks.

References

1. van der Aalst, W.M., Hinz, O., Weinhardt, C.: Resilient digital twins (2021)
2. Agnieszka, O., Marek, J.D., Hanna, W.: Learning bayesian network parameters from small data sets: application of noisy-or gates. International Journal of Approximate Reasoning 27(2), 165–182 (2001)
3. Aldrich, J.: Autonomy. Oxford Economic Papers 41(1), 15–34 (01 1989)
4. Andrews, B., Ramsey, J., Cooper, G.F.: Learning high-dimensional directed acyclic graphs with mixed data-types. PMLR 104:4-21 (2019)
5. Banerjee, A., Dalal, R., Mittal, S., Joshi, K.P.: Generating digital twin models using knowledge graphs for industrial production lines. UMBC Information Systems Department (2017)
6. Bollen, K.A.: Structural equation models with observed variables. Structural Equations with Latent Variables pp. 80–150 (1989)
7. Boschert, S., Heinrich, C., Rosen, R.: Next generation digital twin. In: Proc. tmce. vol. 2018, pp. 7–11. Las Palmas de Gran Canaria, Spain (2018)
8. Fallet-Fidry, G., Weber, P., Simon, C., Iung, B., Duval, C.: Evidential network-based extension of leaky noisy-or structure for supporting risks analyses. IFAC Proceedings Volumes 45(20), 672–677 (2012)
9. Fenton, N.E., Noguchi, T., Neil, M.: 'an extension to the noisy-or function to resolve the "explaining away" deficiency for practical bayesian network problems', ieee trans. Knowl. Data Eng 31, 2441–2445 (2019)
10. Hoover, K.D.: Causality in economics and econometrics. In: The New Palgrave Dictionary of Economics, : Palgrave Macmillan UK, pp. 1–13 (2017)

11. Lawrence, A.R., Kaiser, M., Sampaio, R., Sipos, M.: Data Generating Process to Evaluate Causal Discovery Techniques for Time Series Data. Causalens NIPS 2020 workshop (2020)
12. Li, C., Mahadevan, S., Ling, Y., Wang, L., Choze, S.: A dynamic bayesian network approach for digital twin. In: 19th AIAA Non-Deterministic Approaches Conference. p. 1566 (2017)
13. Liang, R., Liu, F., Liu, J.: A belief network reasoning framework for fault localization in communication networks. Sensors (12 2020)
14. Mirylenka, K., Cormode, G., Palpanas, T., Srivastava, D.: Conditional heavy hitters: detecting interesting correlations in data streams. VLDB J. 24(3) (2015)
15. Nauta, M., Bucur, D., Seifert, C.: Causal discovery with attention-based convolutional neural networks. Machine Learning and Knowledge Extraction 1(1), 312–340 (2019), https://www.mdpi.com/2504-4990/1/1/19
16. Oniśko, A., Druzdzel, M.J., Wasyluk, H.: Learning bayesian network parameters from small data sets: Application of noisy-or gates. International Journal of Approximate Reasoning 27(2), 165–182 (2001)
17. Pamfil, R., Sriwattanaworachai, N., Desai, S., Pilgerstorfer, P., Georgatzis, K., Beaumont, P., Aragam, B.: Dynotears: Structure learning from time-series data. In: International Conference on Artificial Intelligence and Statistics (2020)
18. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc (1988)
19. Pearl, J.: Causality. Cambridge university press (2009)
20. Pearl, J., Glymour, M., Jewell, N.P.: Causal inference in statistics: A primer. John Wiley & Sons (2016)
21. Ramsey, J., Malinsky, D., Bui, K.V.: algcomparison: Comparing the performance of graphical structure learning algorithms with tetrad. arXiv preprint arXiv:1607.08110 (2016)
22. Runge, J.: Discovering contemporaneous and lagged causal relations in autocorrelated nonlinear time series datasets. In: Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI). PMLR (2020)
23. Runge, J., Nowack, P., Kretschmer, M., Flaxman, S., Sejdinovic, D.: Pcmci - detecting causal associations in large nonlinear time series datasets. Sci. Adv 5, 11 (11 2019)
24. Thulasiraman, K., Swamy, M.N.S.: Graphs: Theory and Algorithms. Wiley, John & Sons (2011)
25. Zhou, K., Martin, A., Pan, Q.: The belief noisy-or model applied to network reliability analysis. ArXiv abs/1606.01116 (2016)