Constraint-Based Learning for Continuous-Time Bayesian Networks

Alessandro Bregoli A.BREGOLI1@CAMPUS.UNIMIB.IT
Università degli Studi di Milano-Bicocca, Milano, Italy

Marco Scutari SCUTARI@IDSIA.CH
Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Lugano, Switzerland

Fabio Stella FABIO.STELLA@UNIMIB.IT
Università degli Studi di Milano-Bicocca, Milano, Italy

Abstract

Dynamic Bayesian networks have been well explored in the literature as discrete-time models; however, their continuous-time extensions have seen comparatively little attention. In this paper, we propose the first constraint-based algorithm for learning the structure of continuous-time Bayesian networks. We discuss the different statistical tests and the underlying hypotheses used by our proposal to establish conditional independence. Finally, we validate its performance using synthetic data, and discuss its strengths and limitations. We find that score-based is more accurate in learning networks with binary variables, while our constraint-based approach is more accurate with variables assuming more than two values. However, more experiments are needed for confirmation.

Keywords: Continuous-time Bayesian networks, structure learning, constraint-based algorithm.

1. Introduction

Multivariate time-series data are becoming increasingly common in many domains such as healthcare, finance, telecommunications, social networks, e-commerce, and homeland security. Their size and dimensionality is set to continue to increase in the future, requiring automated algorithms to discover their probabilistic structure and to predict their trajectories over time.

In this paper we focus on the problem of learning the structure of continuous-time Bayesian networks (CTBNs; Nodelman et al., 2002) from data. This type of probabilistic graphical model has been successfully used to reconstruct transcriptional regulatory networks from time-course gene expression data (Acerbi et al., 2016), to model the presence of people at their computers (Nodelman and Horvitz, 2003), and to detect network intrusion (Xu and Shelton, 2008). The literature implements CTBN structure learning using score-based algorithms to maximize the Bayesian-Dirichlet equivalent (BDe) metric, while in this paper we design the first constraint-based algorithm.

The main contributions of this paper are:

- the design of the first constraint-based algorithm for the structure learning of CTBNs, which we call *Continuous-Time PC* (CTPC);
- the definition of different test statistics to assess conditional independence in CTBNs;
- an empirical performance comparison between score-based algorithms and our proposal.

The rest of the paper is organized as follows. Section 2 introduces CTBNs and the associated score-based structure learning algorithms. The proposed constraint-based algorithm and the associated conditional independence tests are presented in Section 3. We then compare score-based and constraint-based approaches in Section 4, and conclusions are summarized in Section 5.

2. Continuous-Time Bayesian Networks

CTBNs are a particular type of probabilistic graphical model that combine Bayesian networks (BNs; Koller and Friedman, 2009) and homogeneous Markov processes to model discrete-state continuous-time dynamical systems (Nodelman et al., 2002). Compared to their discrete-time counterpart, dynamic Bayesian networks (DBNs), they can efficiently model domains like those mentioned above in which variables evolve at different time granularities. The complexity of exact and approximate inference in CTBNs has been shown to be NP-hard (Sturlaugson and Sheppard, 2014).

2.1 Definitions and Notations

A CTBN models a stochastic process over a structured state space for a set of random variables $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$, where each $X_k \in \mathbf{X}$ takes value over a finite domain $Val(X_k)$. It encodes such a process in a compact form by factorizing its dynamics into local continuous-time Markov processes that depend on a limited set of states.

Definition 1 *(Nodelman et al., 2002). A CTBN $\mathcal{N}$ over $\mathbf{X}$ is characterized by two components:*

- *An initial distribution $\mathrm{P}^0(\mathbf{X})$, specified as a BN over $\mathbf{X}$.*
- *A continuous-time transition model specified as:*
  - *a directed (possibly cyclic) graph $\mathcal{G}$ whose nodes correspond to the $X_k \in \mathbf{X}$;*
  - *a conditional intensity matrix $\mathbf{Q}_{X_k \mid \mathbf{U}}$ for each X_k.*

The conditional intensity matrix (CIM) $\mathbf{Q}_{X_k \mid \mathbf{U}}$ consists of the set of intensity matrices

$$\mathbf{Q}_{X_k \mid \mathbf{u}} = \begin{bmatrix} -q_{x_1 \mid \mathbf{u}} & q_{x_1 x_2 \mid \mathbf{u}} & \cdots & q_{x_1 x_m \mid \mathbf{u}} \\ q_{x_2 x_1 \mid \mathbf{u}} & -q_{x_2 \mid \mathbf{u}} & \cdots & q_{x_2 x_m \mid \mathbf{u}} \\ \vdots & \vdots & \ddots & \vdots \\ q_{x_m x_1 \mid \mathbf{u}} & q_{x_m x_2 \mid \mathbf{u}} & \cdots & -q_{x_m \mid \mathbf{u}} \end{bmatrix}, \qquad m = |\, Val(X_k)|, \text{[1]}$$

one for each possible configuration $\mathbf{u}$ of the parents $\mathbf{U}$ of X_k in $\mathcal{G}$. The diagonal elements of $\mathbf{Q}_{X_k \mid \mathbf{u}}$ are such that $q_{x_i \mid \mathbf{u}} = \sum_{x_j \neq x_i} q_{x_i x_j \mid \mathbf{u}}$, where $q_{x_i \mid \mathbf{u}}$ is the parameter of the exponential distribution associated with state x_i of variable X_k. Therefore, $1/q_{x_i \mid \mathbf{u}}$ is the expected time that variable X_k stays in state x_i before transitioning to a different state x_j. The off-diagonal elements $q_{x_i x_j \mid \mathbf{u}}$ are proportional to the probability that X_k transitions from state x_i to state x_j when $\mathbf{U} = \mathbf{u}$. Note that, conditional on X_k, $\mathbf{Q}_{X_k \mid \mathbf{u}}$ can be equivalently summarized with two independent sets of parameters:

- $\mathbf{q}_{X_k \mid \mathbf{u}} = \left\{ q_{x_i \mid \mathbf{u}}, \forall x_i \in Val(X_k) \right\}$, the set of intensities of the exponential distributions of the *waits until the next transition*; and
- $\boldsymbol{\theta}_{X_k \mid \mathbf{u}} = \left\{ \theta_{x_i x_j \mid \mathbf{u}} = q_{x_i x_j \mid \mathbf{u}} / q_{x_i \mid \mathbf{u}}, \forall x_i, x_j \in Val(X_k), x_i \neq x_j \right\}$, the *probabilities of transitioning to specific states*.

1. For simplicity of notation, and without loss of generality, we omit the k subscript from m that implies that each X_k may have a different domain.

Therefore, a CTBN $\mathcal{N}$ over $\mathbf{X}$ can be equivalently described by a graph $\mathcal{G}$ together with the corresponding sets of parameters $\mathbf{q} = \left\{\mathbf{q}_{X_k \mid \mathbf{u}} : \forall X_k \in \mathbf{X}, x_i \in \mathit{Val}(X_k), \mathbf{u} \in \mathit{Val}(\mathbf{U})\right\}$ and $\boldsymbol{\Theta} = \left\{\boldsymbol{\theta}_{X_k \mid \mathbf{u}} : \forall X_k \in \mathbf{X}, x_i \in \mathit{Val}(X_k), \mathbf{u} \in \mathit{Val}(\mathbf{U})\right\}$.

It is important to note that we assume that only one variable in the CTBN can change state at any specific instant; and that its transition dynamics are specified by its parents via the CIM, while being independent of all other variables given its Markov Blanket.[2]

2.2 Structure Learning

Let $\mathcal{D} = \{\sigma_1, \ldots, \sigma_h\}$ be a sample consisting of h trajectories $\sigma_j = \{\langle t_1, X_{t_1}\rangle, \ldots, \langle T_j, X_{T_j}\rangle\}$, where T_j represents the length of trajectory σ_j, that is, the number of transitions. For each pair $\langle t_i, X_{t_i}\rangle$, we denote the time of the ith transition as t_i and the variable that leaves its current state at that time as X_{t_i}. Learning the structure of a CTBN from $\mathcal{D}$ can be cast as an optimization problem (Nodelman et al., 2003) in which we would like to find the graph $\mathcal{G}^*$ with the highest posterior log-probability given $\mathcal{D}$:

$$\ln \mathrm{P}(\mathcal{G} \mid \mathcal{D}) = \ln \mathrm{P}(\mathcal{G}) + \ln \mathrm{P}(\mathcal{D} \mid \mathcal{G}) \tag{1}$$

where $\mathrm{P}(\mathcal{G})$ is the *prior distribution over the space of graphs* spanning $\mathbf{X}$ and $\mathrm{P}(\mathcal{D} \mid \mathcal{G})$ is the *marginal likelihood* of the data given $\mathcal{G}$ averaged over all possible parameter sets.

The prior $\mathrm{P}(\mathcal{G})$ is usually assumed to satisfy the *structure modularity* property (Friedman and Koller, 2000), so that it decomposes as

$$\mathrm{P}(\mathcal{G}) = \prod_{X_k \in \mathbf{X}} \mathrm{P}(Pa(X_k) = \mathbf{U}). \tag{2}$$

For simplicity, the literature often assumes a uniform prior, that is, $\mathrm{P}(\mathcal{G}) \propto 1$.

The marginal likelihood $\mathrm{P}(\mathcal{D} \mid \mathcal{G})$ depends on the parameter prior $\mathrm{P}(\mathbf{q}, \boldsymbol{\Theta} \mid \mathcal{G})$, which is usually assumed to satisfy the *global parameter independence*, the *local parameter independence* and the *parameter modularity* properties (Heckerman et al., 1995) outlined below.

- *Global parameter independence*: the parameters $\mathbf{q}_{X_k \mid \mathbf{U}}$ and $\boldsymbol{\theta}_{X_k \mid \mathbf{U}}$ associated with each variable X_k in a graph $\mathcal{G}$ are independent:

$$\mathrm{P}(\mathbf{q}, \boldsymbol{\Theta} \mid \mathcal{G}) = \prod_{X_k \in \mathbf{X}} \mathrm{P}(\mathbf{q}_{X_k \mid \mathbf{U}}, \boldsymbol{\theta}_{X_k \mid \mathbf{U}} \mid \mathcal{G}). \tag{3}$$

- *Local parameter independence*: for each variable X_k, the parameters associated with each configuration $\mathbf{u}$ of parent set $\mathbf{U}$ are independent:

$$\mathrm{P}(\mathbf{q}_{X_k \mid \mathbf{U}}, \boldsymbol{\theta}_{X_k \mid \mathbf{U}} \mid \mathcal{G}) = \prod_{\mathbf{u} \in \mathit{Val}(\mathbf{U})} \prod_{x_i \in \mathit{Val}(X_k)} \mathrm{P}(\mathbf{q}_{x_i \mid \mathbf{u}}, \boldsymbol{\theta}_{x_i \mid \mathbf{u}} \mid \mathcal{G}). \tag{4}$$

- *Parameter modularity*: if variable X_k has the same parent set in two distinct graphs $\mathcal{G}$ and $\mathcal{G}'$, then the prior probability for the parameters associated with X_k should also the same:

$$\mathrm{P}(\mathbf{q}_{X_k \mid \mathbf{U}}, \boldsymbol{\theta}_{X_k \mid \mathbf{U}} \mid \mathcal{G}) = \mathrm{P}(\mathbf{q}_{X_k \mid \mathbf{U}}, \boldsymbol{\theta}_{X_k \mid \mathbf{U}} \mid \mathcal{G}'). \tag{5}$$

2. The definition of Markov blankets in CTBNs is the same as in BNs: a Markov blanket comprises the parents, the children and the spouses of the target node; and it graphically separates the target node from the rest of the network.

In the context of CTBNs, we assume that the priors over the waiting times and over the transition probabilities are independent as well:

$$\mathrm{P}(\mathbf{q}, \boldsymbol{\Theta} \,|\, \mathcal{G}) = \mathrm{P}(\mathbf{q} \,|\, \mathcal{G})\, \mathrm{P}(\boldsymbol{\Theta} \,|\, \mathcal{G}). \tag{6}$$

Nodelman et al. (2003) suggested conjugate priors for both $\mathbf{q}$ and $\boldsymbol{\Theta}$ in the form of

$$\mathrm{P}(\mathbf{q}_{x_i \,|\, \mathbf{u}}) \sim \mathit{Gamma}\left(\alpha_{x_i \,|\, \mathbf{u}}, \tau_{x_i \,|\, \mathbf{u}}\right), \tag{7}$$

$$\mathrm{P}(\boldsymbol{\theta}_{x_i \,|\, \mathbf{u}}) \sim \mathit{Dir}\left(\alpha_{x_i x_1 \,|\, \mathbf{u}}, \ldots, \alpha_{x_i x_m \,|\, \mathbf{u}}\right), \tag{8}$$

where $\alpha_{x_i \,|\, \mathbf{u}}, \tau_{x_i \,|\, \mathbf{u}}, \alpha_{x_i x_1 \,|\, \mathbf{u}}, \ldots, \alpha_{x_i x_m \,|\, \mathbf{u}}$ are the priors' hyperparameters. In particular, for any $X_k \,|\, \mathbf{U} = \mathbf{u}$, $\alpha_{x_i \,|\, \mathbf{u}}$ and $\alpha_{x_i x_j \,|\, \mathbf{u}}$ represent the pseudocounts for the number of transitions from state x_i to state x_j; and $\tau_{x_i \,|\, \mathbf{u}}$ represent the imaginary amount of time spent in each state x_i before any data is observed. Note that $\alpha_{x_i \,|\, \mathbf{u}}$ is inversely proportional to the number of joint states of the parents of X_i. After conditioning on the dataset $\mathcal{D}$, we obtain the following posterior distributions:

$$\mathrm{P}(\mathbf{q}_{x_i \,|\, \mathbf{u}} \,|\, \mathcal{D}) \sim \mathit{Gamma}\left(\alpha_{x_i \,|\, \mathbf{u}} + M_{x_i \,|\, \mathbf{u}}, \tau_{x_i \,|\, \mathbf{u}} + T_{x_i \,|\, \mathbf{u}}\right), \tag{9}$$

$$\mathrm{P}(\boldsymbol{\theta}_{x_i \,|\, \mathbf{u}} \,|\, \mathcal{D}) \sim \mathit{Dir}\left(\alpha_{x_i x_1 \,|\, \mathbf{u}} + M_{x_i x_1 \,|\, \mathbf{u}}, \ldots, \alpha_{x_i x_m \,|\, \mathbf{u}} + M_{x_i x_m \,|\, \mathbf{u}}\right), \tag{10}$$

where $T_{x_i \,|\, \mathbf{u}}$ and $M_{x_i x_j \,|\, \mathbf{u}}$ are the sufficient statistics of the CTBN. In particular, $T_{x_i \,|\, \mathbf{u}}$ is the amount of time spent by X_k in the state x_i and $M_{x_i x_j \,|\, \mathbf{u}}$ is the number of times that X_k transitions from the state x_i to the state x_j, given $\mathbf{U} = \mathbf{u}$.[3] The marginal likelihood $\mathrm{P}(\mathcal{D} \,|\, \mathcal{G})$ arising from these posteriors can be written as

$$\mathrm{P}(\mathcal{D} \,|\, \mathcal{G}) = \prod_{X_k \in \mathbf{X}} \mathrm{ML}(\mathbf{q}_{X_k \,|\, \mathbf{U}} : \mathcal{D})\, \mathrm{ML}(\boldsymbol{\theta}_{X_k \,|\, \mathbf{U}} : \mathcal{D}) \tag{11}$$

due to (3) and (6). $\mathrm{ML}(\mathbf{q}_{X_k \,|\, \mathbf{U}} : \mathcal{D})$ is the marginal likelihood of $\mathbf{q}_{X_k \,|\, \mathbf{U}}$,

$$\mathrm{ML}(\mathbf{q}_{X_k \,|\, \mathbf{U}} : \mathcal{D}) = \prod_{\mathbf{u} \in \mathit{Val}(\mathbf{U})} \prod_{x_i \in \mathit{Val}(X_k)} \frac{\Gamma\left(\alpha_{x_i \,|\, \mathbf{u}} + M_{x_i \,|\, \mathbf{u}} + 1\right) \left(\tau_{x_i \,|\, \mathbf{u}}\right)^{\alpha_{x_i \,|\, \mathbf{u}} + 1}}{\Gamma\left(\alpha_{x_i \,|\, \mathbf{u}} + 1\right) \left(\tau_{x_i \,|\, \mathbf{u}} + T_{x_i \,|\, \mathbf{u}}\right)^{\alpha_{x_i \,|\, \mathbf{u}} + M_{x_i \,|\, \mathbf{u}} + 1}}; \tag{12}$$

and $\mathrm{ML}(\boldsymbol{\theta}_{X_k \,|\, \mathbf{U}} : \mathcal{D})$ is the marginal likelihood of $\boldsymbol{\theta}_{X_k \,|\, \mathbf{U}}$,

$$\mathrm{ML}(\boldsymbol{\theta}_{X_k \,|\, \mathbf{U}} : \mathcal{D}) = \prod_{\mathbf{u} \in \mathit{Val}(\mathbf{U})} \prod_{x_i \in \mathit{Val}(X_k)} \frac{\Gamma\left(\alpha_{x_i \,|\, \mathbf{u}}\right)}{\Gamma\left(\alpha_{x_i \,|\, \mathbf{u}} + M_{x_i \,|\, \mathbf{u}}\right)} \prod_{x_j \in \mathit{Val}(X_k)} \frac{\Gamma\left(\alpha_{x_i x_j \,|\, \mathbf{u}} + M_{x_i x_j \,|\, \mathbf{u}}\right)}{\Gamma\left(\alpha_{x_i x_j \,|\, \mathbf{u}}\right)}. \tag{13}$$

The resulting $\mathrm{P}(\mathcal{D} \,|\, \mathcal{G})$ is the Bayesian-Dirichlet equivalent (BDe) metric for CTBNs (Nodelman, 2007) based on the priors (7) and (8), which satisfies assumptions (3), (4), and (5) by construction. The posterior in (1) can then be written in closed form as

$$\mathrm{P}(\mathcal{G} \,|\, \mathcal{D}) = \sum_{X_k \in \mathbf{X}} \log \mathrm{P}(\mathit{Pa}(X_k) = \mathbf{U}) + \log \mathrm{ML}(\mathbf{q}_{X_k \,|\, \mathbf{U}} : \mathcal{D}) + \log \mathrm{ML}(\boldsymbol{\theta}_{X_k \,|\, \mathbf{U}} : \mathcal{D}) \tag{14}$$

assuming (2) is satisfied.

Since $\mathcal{G}$ does not have acyclicity constraints in a CTBN, it is possible to maximize (14) by independently scoring the possible parent sets of each X_k. Furthermore, if we bound the maximum number of parents we can find the optimal $\mathrm{P}(\mathcal{G} \,|\, \mathcal{D})$ in polynomial time either by enumerating all possible parent sets or by using hill-climbing to add, delete or reverse arcs (Nodelman et al., 2003).

3. The number of times X_k leaves the state x_i when $\mathbf{U} = \mathbf{u}$ is $M_{x_i \,|\, \mathbf{u}} = \sum_{x_j \neq x_i} M_{x_i x_j \,|\, \mathbf{u}}$.

Algorithm 1 PC Algorithm

1. Form the complete undirected graph $\mathcal{G}$ on the vertex set $\mathbf{X}$.
2. For each pair of variables $X_i, X_j \in \mathbf{X}$, consider all the possible separation set from the smallest ($\mathbf{S}_{X_iX_j} = \varnothing$) to the largest ($\mathbf{S}_{X_iX_j} = \mathbf{X} \setminus \{X_i, X_j\}$). If there isn't any set $\mathbf{S}_{X_iX_j}$ such that $X_i \perp\!\!\!\perp X_j \,|\, \mathbf{S}_{X_iX_j}$ then the edge $X_i - X_j$ is removed from $\mathcal{G}$.
3. For each triple $X_i, X_j, X_k \in \mathcal{G}$ such that $X_i - X_j$, $X_j - X_k$, and X_i, X_j are not connected, orient the edges into $X_i \to X_j \leftarrow X_k$ if and only if $X_j \notin \mathbf{S}_{X_iX_j}$ for every $\mathbf{S}_{X_iX_j}$ that makes X_i and X_k independent.
4. The algorithm identifies the compelled directed arcs by iteratively applying the following two rules:
   (a) if X_i is adjacent to X_j and there is a strictly directed path from X_i to X_j then replace $X_i - X_j$ with $X_i \to X_j$ (to avoid introducing cycles);
   (b) if X_i and X_j are not adjacent but $X_i \to X_k$ and $X_k - X_j$, then replace the latter with $X_k \to X_j$ (to avoid introducing new v-structures).
5. Return the resulting CPDAG $\mathcal{G}$.

3. A Constraint-Based Algorithm for Structure Learning

Learning the structure of a BN is a problem that is well explored in the literature. Several approaches have been proposed spanning score-based algorithms, constraint-based algorithms and hybrid algorithms; a recent review is available from Scutari et al. (2019). Score-based algorithms find the BN structure that maximizes a given score function, while constraint-based algorithms use statistical tests to learn conditional independence relationships (called *constraints*) from the data and infer the presence or absence of particular arcs. Hybrid algorithms combine aspects of both score-based and constraint-based algorithms.

On the other hand, the only structure learning algorithm proposed for CTBNs is the score-based algorithm from Nodelman et al. (2003) we described in the previous section; to the best of our knowledge no constraint-based algorithm exists in the literature. After a brief introduction to constraint-based algorithms for BNs, we propose such an algorithm for CTBNs.

3.1 Constraint-Based Algorithms for BNs

Constraint-based algorithms for BN structure learning originate from the *Inductive Causation* (IC) algorithm from Pearl and Verma (1991) for learning causal networks. IC starts by finding pairs of nodes connected by an undirected arc as those are not independent given any other subset of variables. The second step identifies the v-structures $X_i \to X_k \leftarrow X_j$ among all pairs X_i and X_j of non-adjacent nodes which share a common neighbour X_k. Finally, IC identifies compelled arcs and orients them to build the completed partially oriented DAG (CPDAG) which describes the *equivalence class* the BN falls into.

However, steps 1 and 2 of the IC algorithm are unfeasible for non-trivial problems due to the exponential number of conditional independence relationships to be tested. The *PC algorithm*, which is briefly illustrated in Algorithm 1, was the first proposal addressing this issue; its modern incarnation is described in Colombo and Maathuis (2014), and we will use it as the foundation for CTBN structure learning below. PC starts from a fully-connected undirected graph. Then, for each pair of variables X_i and X_j it proceeds by gradually increasing the cardinality of the set of conditioning nodes $\mathbf{S}_{X_iX_j}$ until X_i and X_j are found to be independent or $\mathbf{S}_{X_iX_j} = \mathbf{X} \setminus \{X_i, X_j\}$. The remaining steps are identical to those of IC.

Neither IC nor PC (or other constraint-based algorithms, for that matter) require a specific test statistic to test conditional independence, making them independent from the distributional assumptions we make on the data.

3.2 The CTPC Structure Learning Algorithm

CTBNs differ from BNs in three fundamental ways: BNs do not model time, while CTBNs do; BNs are based on DAGs, while CTBNs allow cycles; and BNs model the dependence of a node on its parents using a conditional probability distribution, while CTBNs model it using a CIM. These differences make structure learning a simpler problem for CTBNs than it is for BNs.

Firstly, learning arc directions is an issue in BNs but not in CTBNs, where arcs are required to follow the arrow of time. Unlike BNs, which can be grouped into equivalence classes that are probabilistically indistinguishable, each CTBN has a unique minimal graphical representation (Nodelman et al., 2003). For instance, let a CTBN $\mathcal{N}$ have graph $\mathcal{G} = \{X \to Y\}$: unless trivially $X = Y$, $\mathcal{G}$ cannot generate the same transition probabilities as any CTBN $\mathcal{N}'$ with graph $\mathcal{G}' = \{X \leftarrow Y\}$.

Secondly, in CTBNs we can learn each parent set $Pa(X_k)$ in isolation, thus making any structure learning algorithm embarrassingly parallel. Acyclicity imposes a global constraint on $\mathcal{G}$ that makes it impossible to do the same in BNs.

Thirdly, each variable X_k is modelled conditional on a given function of its parent set $Pa(X_k)$: a conditional probability table for (discrete) BNs, a CIM for CTBNs. However, a CIM $\mathbf{Q}_{X_k \,|\, \mathbf{U}}$ describes the temporal evolution of the state of variable X_k conditionally on the state of its parent set $\mathbf{U}$. Hence we can not test conditional independence by using classic test statistics like the mutual information or Pearson's χ^2 that assume observations are independent (Koller and Friedman, 2009). Instead we need to adapt our definition of conditional independence to CTBNs to design a constraint-based algorithm for structure learning.

Definition 2 *Conditional Independence in a CTBN*

Let $\mathcal{N}$ be a CTBN with graph $\mathcal{G}$ over a set of variables $\mathbf{X}$. We say that X_i is conditionally independent from X_j given $\mathbf{S}_{X_iX_j} \subseteq \mathbf{X} \setminus \{X_i, X_j\}$ if

$$\mathbf{Q}_{X_i \,|\, x, \mathbf{s}} = \mathbf{Q}_{X_i \,|\, \mathbf{s}} \qquad \forall\, x \in \mathit{Val}(X_j), \forall\, \mathbf{s} \in \mathit{Val}(\mathbf{S}_{X_iX_j}). \tag{15}$$

If $\mathbf{S}_{X_iX_j} = \varnothing$, then X_i is said to be marginally independent from X_j.

It is important to note that Definition 2 is not symmetric: it is perfectly possible for X_i to be conditionally or marginally independent from X_j, while X_j is not conditionally or marginally independent from X_i. This discrepancy is, however, not a practical or theoretical concern because arcs are already non-symmetric (they must follow the arrow of time) and therefore we only test whether X_i depends X_j if X_j precedes X_i or the other way round.

As for the test statistics, we can test for conditional independence using $\mathbf{q}_{X_k \mid \mathbf{u}}$ (the waiting times) and, if we do not reject the null hypothesis of conditional independence, we can perform a further test using $\boldsymbol{\theta}_{X_k \mid \mathbf{u}}$ (the transitions); $\mathbf{q}_{X_k \mid \mathbf{u}}$ and $\boldsymbol{\theta}_{X_k \mid \mathbf{u}}$ have been defined to be independent in Section 2.1 so they can be tested separately. Note that conditional independence can be established by testing only the waiting times $\mathbf{q}_{X_k \mid \mathbf{u}}$ if the CTBN contains only binary variables. However, testing for conditional independence involves both waiting times and transitions in the general case in which variables can take more than two values. Since we consider that rates are the most important characteristic to to assess in a stochastic process, we decide without loss of generality to test $\mathbf{q}_{X_k \mid \mathbf{u}}$ first, and then $\boldsymbol{\theta}_{X_k \mid \mathbf{u}}$.

For $\mathbf{q}_{X_k \mid \mathbf{u}}$, we define the *null time to transition hypothesis* as follows.

Definition 3 *Null Time To Transition Hypothesis*

Given X_i, X_j and the conditioning set $\mathbf{S}_{X_i X_j}$, the null time to transition hypothesis of X_j over X_i is

$$q_{x \mid y,\mathbf{s}} = q_{x \mid \mathbf{s}} \qquad \forall\, x \in Val(X_i), \forall\, y \in Val(X_j), \forall\, \mathbf{s} \in Val(\mathbf{S}_{X_i X_j}). \tag{16}$$

For $\boldsymbol{\theta}_{X_k \mid \mathbf{u}}$, we define the *null state to state transition hypothesis* as follows.

Definition 4 *Null State To State Transition Hypothesis*

Given X_i, X_j and the conditioning set $\mathbf{S}_{X_i X_j}$, the null state to state transition hypothesis of X_j over X_i is

$$\theta_{x\cdot \mid y,\mathbf{s}} = \theta_{x\cdot \mid \mathbf{s}} \qquad \forall\, x \in Val(X_i), \forall\, y \in Val(X_j), \forall\, \mathbf{s} \in Val(\mathbf{S}_{X_i X_j}) \tag{17}$$

where we let $\theta_{x\cdot \mid y,\mathbf{s}}$ be off diagonal elements of matrix $\mathbf{Q}_{X_i \mid y,\mathbf{s}}$ corresponding to assignment $X_i = x$. It is worthwhile to mention that equality $\theta_{x\cdot \mid y,\mathbf{s}} = \theta_{x\cdot \mid \mathbf{s}}$ has to be understood in terms of corresponding components of vectors $\theta_{x\cdot \mid y,\mathbf{s}}$ and $\theta_{x\cdot \mid \mathbf{s}}$.

Definition 3 characterizes conditional independence for the times to transition for variable X_i when adding (or not) X_j to its parents; Definition 4 characterizes conditional independence for the transitions of X_i when adding (or not) X_j to its parents.

To test the *null time to transition hypothesis*, we use the F test to compare two exponential distributions from Lee and Wang (2003). In the case of CTBNs, the test statistic and the degrees of freedom take form

$$F_{r_1,r_2} = \frac{q_{x \mid \mathbf{s}}}{q_{x \mid y,\mathbf{s}}}, \qquad \text{with} \qquad r_1 = \sum_{x' \in Val(X_i)} M_{xx' \mid y,\mathbf{s}}, \qquad r_2 = \sum_{x' \in Val(X_i)} M_{xx' \mid \mathbf{s}} \tag{18}$$

To test the *null state to state transition hypothesis*, we investigated the use of the *two-sample chi-square* and *Kolmogorov-Smirnov* tests (Mitchell, 1971). For CTBNs the former takes form:

$$\chi^2 = \sum_{x' \in Val(X_i)} \frac{(K \cdot M_{xx' \mid y,\mathbf{s}} - L \cdot M_{xx' \mid \mathbf{s}})^2}{M_{xx' \mid y,\mathbf{s}} + M_{xx' \mid \mathbf{s}}}, \quad K = \sqrt{\frac{\sum_{x' \in Val(X_i)} M_{xx' \mid \mathbf{s}}}{\sum_{x' \in Val(X_i)} M_{xx' \mid y,\mathbf{s}}}}, L = \frac{1}{K}\ ; \tag{19}$$

and is asymptotically distributed as a $\chi^2_{|Val(X_i)|-1}$. The latter is defined as

$$D_{r_1,r_2} = \sup_{x' \in Val(X_i)} \left| \Theta_{xx' \,|\, \mathbf{s}} - \Theta_{xx' \,|\, y,\mathbf{s}} \right|, \qquad \Theta_{xx'} = \sum_{\substack{x'' \in Val(X_i) \\ x'' \leq x'}} \theta_{xx''}. \tag{20}$$

After characterising conditional independence, we can now introduce our constraint-based algorithm for structure learning in CTBNs. The algorithm, which we call *Continuous-Time PC* (CTPC), is shown in Algorithm 2. The first step is the same as the corresponding step of the PC algorithm in that it determines the same pattern of conditional independence tests. However, as discussed above, the hypotheses being tested are the *null time to transition hypothesis* and the *null state to state transition hypothesis*. The second step of CTPC differs from that in the PC algorithm. Since independence relationships are not symmetric in CTBNs, we can find the graph $\mathcal{G}$ of the CTBN without indentifying and then refining a CPDAG representing an equivalence class. Therefore, steps 3 and 4 of the PC algorithm are not needed in the case of CTBNs.

CTPC starts by initializing the complete directed graph $\mathcal{G}$ without loops (step 1). Note that while loops (that is, arcs like $X_i \to X_i$) are not included, cycles of length two (that is, $X_i \to X_j$ and $X_j \to X_i$) are, as well as cycles of length tree or more. Step 2 iterates over the X_i to identify their parents $\mathbf{U}$. This is achieved in step 2.2.1 by first testing for unconditional independence, then by testing for conditional independence gradually increasing the cardinality b of the considered separating sets. Each time Algorithm 2 concludes that X_i is independent from X_j given some separating set, we remove arc from node X_j to node X_i in step 2.2.2. At the same time, we also remove X_j from the current parent set $\mathbf{U}$. The iteration for X_i, X_j terminates either when X_j is found to be independent from X_i or there are no more larger separating sets to try because $b = |U|$; and the iteration over X_i terminates when there are no more X_j to test.

CTPC checks the *null time to transition hypothesis* (Definition 3) by applying the *test for two exponential means* (18). On the contrary, the *null state to state transition hypothesis* (Definition 4) can be tested using two different tests, i.e., the *two sample chi-square test* and the *two sample Kolmogorov-Smirnov test*. We call these two options CTPC$_{\chi^2}$ the and CTPC$_{\text{KS}}$, respectively.

Algorithm 2 Continuous-time PC Algorithm

1. Form the complete directed graph $\mathcal{G}$ on the vertex set $\mathbf{X}$.
2. For each variable $X_i \in \mathbf{X}$:
   - 2.1 Set $\mathbf{U} = \{X_j \in \mathbf{X} : X_j \to X_i\}$, the current parent set.
   - 2.2 For increasing values $b = 0, \ldots, n$, until $b = |\mathbf{U}|$:
     - 2.2.1 For each $X_j \in \mathbf{U}$, test $X_i \perp\!\!\!\perp X_j \,|\, \mathbf{S}_{X_i X_j}$ for all possible subsets of size b of $\mathbf{U} \setminus X_j$.
     - 2.2.2 As soon as $X_i \perp\!\!\!\perp X_j \,|\, \mathbf{S}_{X_i X_j}$ for some $\mathbf{S}_{X_i X_j}$, remove $X_j \to X_i$ from $\mathcal{G}$ and X_j from $\mathbf{U}$.
3. Return directed graph $\mathcal{G}$.

4. Numerical Experiments

We assess the performance of CTPC against that of the score-based algorithm from Nodelman et al. (2003) using synthetic data[4].

In particular, we generate random CTBNs as the combinations of directed graphs and the associated CIMs; and we generate random trajectories from each CTBN. We perform a simulation study using a factorial experimental design over different numbers of nodes $n = \{5, 10, 15, 20\}$, network densities $\{0.1, 0.2, 0.3\}$, number of states for the nodes $|Val(X_i)| = \{2, 3\}$ and different numbers of trajectories $h = \{100, 200, 300\}$. Each trajectory lasts on average for 100 units of time. Note that we only generate connected networks, hence absolute density is bounded below by $|X|$. We perform 10 replicates for each simulation configuration with $n < 10$; and 3 replicates for configurations with $n \geqslant 10$.

We measure the performance of the learning algorithms along two dimensions: structural accuracy and speed. In particular, structural accuracy is measured using the F1 score over the arcs, which is defined as follows,

$$F_1 = 2 \cdot \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \; ; \tag{21}$$

since there is no score equivalence in CTBNs, nor are networks constrained to be acyclic, comparing graphs is equivalent to evaluating a binary classification problem. As for speed, we evaluate the wall-clock time on a single core. While all algorithms we consider can be parallelized in some way, we prefer to avoid the confounding effect of varying degrees of parallelism overhead on speed.

4.1 Results

The results of our simulation study are summarized in Table 1 (for the score-based algorithm in Nodelman et al., 2003), Table 2 (for CTPC$_{\chi^2}$) and Table 3 (for CTPC$_{\text{KS}}$).

In the case of binary variables, the score-based algorithm performs better than the proposed constraint-based algorithms for any combination of network density, number of trajectories and number of nodes. CTPC$_{\chi^2}$ and CTPC$_{\text{KS}}$ have comparable performance, which is expected since in this case the two algorithms are identical (because the tests are identical, that is, we only test waiting times). However, CTPC$_{\chi^2}$ and CTPC$_{\text{KS}}$ have better F1 scores than the score-based algorithm for ternary variables.

CTPC$_{\chi^2}$ appears to perform marginally better than CTPC$_{\text{KS}}$ when $n < 20$, but the two algorithm are again comparable when $n = 20$ and $h = 300$. This suggests that CTPC$_{\chi^2}$ is more sample-efficient than CTPC$_{\text{KS}}$ with respect to the number of trajectories.

However, CTPC$_{\chi^2}$ and CTPC$_{\text{KS}}$ scale better than the score-based algorithm from Nodelman et al. (2003), which exhausts the 24GiB of memory allocated for the experiment and fails to complete in the allotted time as shown in the last line of Table 1.

4. Score-based learning has been performed by the CTBN-RLE package (Shelton et al., 2010), while CTPC has been implemented in Python and made available at the following GitHub https://github.com/AlessandroBregoli/ctbn_cba. CTBN-RLE uses Bayesian Score where a Gamma prior is used for parameters of the exponential distributions while a Dirichlet prior is used for transition probabilities. Hyperparameters are set to their default value, i.e., $\tau = 1$ for the Gamma prior, while $\alpha = 1$ for the Dirichlet prior.

Cardinality	Binary variables								
Network density	0.1			0.2			0.3		
# trajectories	100	200	300	100	200	300	100	200	300
5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
10	1.00	1.00	1.00	.990	.990	.990	1.00	1.00	1.00
15	1.00	1.00	1.00	1.00	1.00	1.00	.941	.993	1.00
20	.984	1.00	1.00	.987	1.00	1.00	.850	.922	.934
Cardinality	Ternary variables								
5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
10	1.00	1.00	1.00	.949	.949	1.00	.987	.934	.962
15	.971	.983	1.00	.800	.841	1.00	.541	.605	.765
20	—	—	—	—	—	—	—	—	—

Table 1: F_1-score for the score based algorithm.

Cardinality	Binary variables								
Network density	0.1			0.2			0.3		
# trajectories	100	200	300	100	200	300	100	200	300
5	.988	1.00	1.00	1.00	1.00	1.00	.992	1.0	1.00
10	1.00	.988	1.00	.970	.970	.970	.966	.973	.967
15	.980	.994	1.00	.949	.981	.993	.830	.903	.933
20	.968	.988	.992	.935	.989	.980	.787	.871	.883
Cardinality	Ternary variables								
5	.972	.921	.909	.973	.973	.973	.966	.953	.979
10	.938	.938	.950	.984	.992	.992	.981	.975	.970
15	.967	.962	.967	.966	.984	.984	.820	.871	.887
20	.944	.944	.939	.880	.904	.913	.583	.720	.761

Table 2: F_1-score for the CTPC$_{\chi^2}$ algorithm.

Cardinality	Binary variables								
Network density	0.1			0.2			0.3		
# trajectories	100	200	300	100	200	300	100	200	300
5	.988	1.00	1.00	1.00	1.00	1.00	.992	1.0	1.00
10	1.00	.988	1.00	.970	.970	.970	.966	.973	.967
15	.980	.994	1.00	.949	.981	.993	.830	.903	.933
20	.968	.988	.992	.935	.989	.980	.787	.871	.883
Cardinality	Ternary variables								
5	.667	.667	.667	.766	.771	.720	.871	.802	.785
10	.617	.623	.650	.811	.775	.780	.890	.886	.854
15	.762	.782	.775	.840	.863	.857	.775	.855	.875
20	.644	.624	.602	.820	.852	.859	.602	.704	.757

Table 3: F_1-score for the PC$_{KS}$CTBN algorithm.

5. Conclusions

In this paper we introduced the first constraint-based algorithm for structure learning in CTBNs, which we called CTPC, comprising both a suitable set of statistics for testing conditional independence and a heuristic algorithm based on PC.

Compared to the only score-based algorithm previously available in the literature (Nodelman et al., 2003), CTPC has better structural reconstruction accuracy when variables in the CTBN can assume more than two values. For binary variables, that score-based algorithm performs well, but its performance rapidly degrades when the number of states increases to three.

A major limitation of the proposed constraint-based algorithm is the computational cost which becomes problematic in domains with more than 20 variables. Even so, CTPC scales better than the score-based algorithm from Nodelman et al. (2003), which exhausts the 24GiB of memory allocated for the experiment and fails to complete in the allotted time.

Further experiments are needed to elucidate the behaviour of CTPC when the number of states of the variables increases. It would also be important to validate the performance of CTPC on real-world data. Unfortunately, we are not aware of any suitable real-world data set where ground truth is available, and thus we were unable to pursue this line of investigation. Furthermore, we are planning additional numerical experiments to evaluate the impact of the type-I-error threshold for the tests to better understand how to calibrate constraint-based algorithms.

References

E. Acerbi, E. Viganò, M. Poidinger, A. Mortellaro, T. Zelante, and F. Stella. Continuous Time Bayesian Networks Identify Prdm1 as a Negative Regulator of TH17 Cell Differentiation in Humans. *Scientific Reports*, 6:23128, 2016.

D. Colombo and M. H. Maathuis. Order-Independent Constraint-Based Causal Structure Learning. *Journal of Machine Learning Research*, 15:3921–3962, 2014.

N. Friedman and D. Koller. Being Bayesian about Bayesian Network Structure: A Bayesian Approach to Structure Discovery in Bayesian Networks. *Machine Learning*, 50:95–125, 2000.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20(3):197–243, 1995.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

E. T. Lee and J. Wang. *Statistical methods for survival data analysis*, volume 476. John Wiley & Sons, 2003.

B. Mitchell. A Comparison of Chi-Square and Kolmogorov-Smirnov Tests. *The Royal Geographical Society*, 3(4):237–241, 1971.

U. Nodelman. *Continuous Time Bayesian Networks*. PhD thesis, Stanford University, 2007.

U. Nodelman and E. Horvitz. Continuous Time Bayesian Networks for Inferring Users' Presence and Activities with Extensions for Modeling and Evaluation. Technical Report MSR-TR-2003-97, Microsoft Research, 2003.

U. Nodelman, C. R. Shelton, and D. Koller. Continuous Time Bayesian Networks. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, pages 378–387, 2002.

U. Nodelman, C. Shelton, and D. Koller. Learning Continuous Time Bayesian Networks. In *19th Conference on Uncertainty in AI (UAI)*, pages 451–458, 2003.

J. Pearl and T. Verma. A Theory of Inferred Causation. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, pages 441–452. Morgan Kaufmann Publishers Inc., 1991.

M. Scutari, C. E. Graafland, and J. M. Gutiérrez. Who Learns Better Bayesian Network Structures: Accuracy and Speed of Structure Learning Algorithms. *International Journal of Approximate Reasoning*, 115:235–253, 2019.

C. R. Shelton, Y. Fan, W. Lam, J. Lee, and J. Xu. Continuous time bayesian network reasoning and learning engine. *Journal of Machine Learning Research*, 11(Mar):1137–1140, 2010.

L. Sturlaugson and J. W. Sheppard. Inference Complexity in Continuous Time Bayesian Networks. In *The 30th Conference on Uncertainty in Artificial Intelligence (UAI 2014), Quebec City, Canada*, pages 772–779, 2014.

J. Xu and C. R. Shelton. Continuous Time Bayesian Networks for Host Level Network Intrusion Detection. In *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2008)*, pages 613–627, 2008.