Dynamic Bayesian Networks as Formal Abstractions of Structured Stochastic Processes

Sadegh Soudjani[1], Alessandro Abate[2], and Rupak Majumdar[3]

1,3 Max Planck Institute for Software Systems (MPI-SWS), Germany
`{sadegh,rupak}@mpi-sws.org`
2 Department of Computer Science, University of Oxford, United Kingdom
`alessandro.abate@cs.ox.ac.uk`

Abstract

We study the problem of finite-horizon probabilistic invariance for discrete-time Markov processes over general (uncountable) state spaces. We compute discrete-time, finite-state Markov chains as formal abstractions of general Markov processes. Our abstraction differs from existing approaches in two ways. First, we exploit the structure of the underlying Markov process to compute the abstraction separately for each dimension. Second, we employ dynamic Bayesian networks (DBN) as compact representations of the abstraction. In contrast, existing approaches represent and store the (exponentially large) Markov chain explicitly, and run out of memory quickly.

We show how to construct a DBN abstraction of a Markov process satisfying an independence assumption on the driving process noise. We compute a guaranteed bound on the error in the abstraction w.r.t. the probabilistic invariance property; the dimension-dependent abstraction makes the error bounds more precise than existing approaches. Additionally, we show how factor graphs and the sum-product algorithm for DBNs can be used to solve the finite-horizon probabilistic invariance problem. Together, DBN-based representations and algorithms can be significantly more efficient than explicit representations of Markov chains for abstracting and model checking structured Markov processes.

1 Introduction

Markov processes over general uncountable state spaces appear in many areas of engineering such as power networks, transportation, biological systems, robotics, and manufacturing systems. The importance of this class of stochastic processes in applications has motivated a significant research effort into their foundations and their verification.

We study the problem of algorithmically verifying finite-horizon probabilistic invariance for Markov processes, which is the problem of computing the probability that a stochastic process remains within a given set for a given finite time horizon. For finite-state stochastic processes, there is a mature theory of model checking discrete-time Markov chains [5], and a number of probabilistic model checking tools [13, 17] that compute explicit solutions to the verification problem. On the other hand, stochastic processes taking values over uncountable state spaces in general lack explicit solutions and their numerical verification problems are undecidable even for simple dynamics [1]. A number of studies have therefore explored *abstraction* techniques that reduce the given stochastic process (over a general state space) to a finite-state process, while preserving properties in a quantitative sense [1, 7]. The abstracted model allows the application of standard model checking techniques over finite-state models. The work in [1] has further shown that an explicit error can be attached to the abstraction: this error is computed purely based on continuity properties of the concrete Markov process, and enables refine properties proved on the finite-state abstraction to properties of the original system. The overall approach has been extended to linear temporal specifications [23] and software tools have been developed to automate the abstraction procedure [9].

Conference title on which this volume is based on.
Editors: Billy Editor and Bill Editors; pp. 1–15
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

In previous works, the structure of the underlying Markov process (namely, the interdependence among its variables) has not been actively reflected in the abstraction algorithms, and the finite-state Markov chain has been always represented explicitly, which is quite expensive in terms of memory requirements. In many applications instead, the dynamics of the Markov process, which are characterized by a conditional kernel, often exhibit specific structural properties: more specifically, the dynamics of any state variable depends on only a small number of other state variables and the process noise driving each state variable is assumed to be independent. Examples of such structured systems are models of power grids and sensor-actuator networks as large-scale interconnected networks [22] and mass-spring-damper systems [3, 4].

We present an abstraction and model checking algorithm for discrete-time stochastic dynamical systems over general (uncountable) state spaces. Our abstraction constructs a finite-state Markov abstraction of the process, but differs from previous work in that it is based on a dimension-dependent partitioning of the state space. Additionally, we perform a precise dimension-dependent analysis of the error introduced by the abstraction, and our error bounds can be exponentially smaller than the general bounds obtained in [1]. Furthermore, we represent the abstraction as a dynamic Bayesian network (DBN) [14] instead of explicitly representing the probabilistic transition matrix. The Bayesian network representation provide polynomial sized representations (in the number of dimensions) for the Markov chain abstraction for which the explicit transition matrix can be exponential in the dimension. We show how factor graphs and the sum-product algorithm, developed for belief propagation in Bayesian networks, can be used to model check probabilistic invariance properties without constructing the transition matrix. Overall, our approach leads to significant reduction in computational and memory resources for model checking structured Markov processes and provides tighter error bounds.

The material is organised in six sections. Section 2 defines discrete-time Markov processes and the probabilistic invariance problem. Section 3 presents a new algorithm for abstracting a process to a DBN, together with the quantification of the abstraction error. We discuss efficient model checking of the constructed DBN in Section 4, and apply the overall abstraction algorithm to a case study in Section 5. Section 6 outlines some further directions of investigation. Proofs of statements are included in the Appendix.

2 Markov Processes and Probabilistic Invariance

2.1 Discrete-Time Markov Processes

Notation. We write $\mathbb{N}$ for the non-negative integers $\mathbb{N} = \{0, 1, 2, \ldots\}$ and $\mathbb{N}_n = \{1, 2, \ldots, n\}$. We use bold typeset for vectors and normal typeset for one-dimensional quantities.

We consider a discrete-time Markov process $\mathscr{M}_{\mathfrak{s}}$ defined over a general state space, and characterized by the tuple $(\mathcal{S}, \mathcal{B}, T_{\mathfrak{s}})$: $\mathcal{S}$ is the continuous state, which we assume to be endowed with a metric and to be separable[1]; $\mathcal{B}$ is the Borel σ-algebra associated to $\mathcal{S}$, which is the smallest σ-algebra containing all open subsets of $\mathcal{S}$; and $T_{\mathfrak{s}} : \mathcal{S} \times \mathcal{B} \to [0, 1]$ is a stochastic kernel, so that $T_{\mathfrak{s}}(\cdot, B)$ is a non-negative measurable function for any set $B \in \mathcal{B}$, and $T_{\mathfrak{s}}(\boldsymbol{s}, \cdot)$ is a probability measure on $(\mathcal{S}, \mathcal{B})$ for any $\boldsymbol{s} \in \mathcal{S}$. Trajectories (also called traces or paths) of $\mathscr{M}_{\mathfrak{s}}$ are sequences $(\boldsymbol{s}(0), \boldsymbol{s}(1), \boldsymbol{s}(2), \ldots)$ which belong to the set $\Omega = \mathcal{S}^{\mathbb{N}}$. The product σ-algebra on Ω is denoted by $\mathcal{F}$. Given the initial state $\boldsymbol{s}(0) = \boldsymbol{s}_0 \in \mathcal{S}$ of $\mathscr{M}_{\mathfrak{s}}$, the

[1] A metric space $\mathcal{S}$ is called separable if it has a countable dense subset.

stochastic Kernel $T_{\mathfrak{s}}$ induces a unique probability measure $\mathcal{P}$ on $(\Omega, \mathcal{F})$ which satisfies the Markov property: namely for any measurable set $B \in \mathcal{B}$ and any $t \in \mathbb{N}$

$$\mathcal{P}\left(\boldsymbol{s}(t+1) \in \ B|\boldsymbol{s}(0), \boldsymbol{s}(1), \ldots, \boldsymbol{s}(t)\right) = \mathcal{P}\left(\boldsymbol{s}(t+1) \in \ B|\boldsymbol{s}(t)\right) = T_{\mathfrak{s}}(\boldsymbol{s}(t), B).$$

We assume that the stochastic kernel $T_{\mathfrak{s}}$ admits a density function $t_{\mathfrak{s}} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$, such that $T_{\mathfrak{s}}(\boldsymbol{s}, B) = \int_B t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) d\bar{\boldsymbol{s}}$.

A familiar class of discrete-time Markov processes is that of stochastic dynamical systems. If $\{\boldsymbol{\zeta}(t),\ t \in \mathbb{N}\}$ is a sequence of independent and identically distributed (iid) random variables taking values in $\mathbb{R}^n$, and $\boldsymbol{f} : \mathcal{S} \times \mathbb{R}^n \to \mathcal{S}$ is a measurable map, then the recursive equation

$$\boldsymbol{s}(t+1) = \boldsymbol{f}(\boldsymbol{s}(t), \boldsymbol{\zeta}(t)), \quad \forall t \in \mathbb{N}, \quad \boldsymbol{s}(0) = \boldsymbol{s}_0 \in \mathcal{S}, \tag{1}$$

induces a Markov process that is characterized by the kernel

$$T_{\mathfrak{s}}(\boldsymbol{s}, B) = T_{\zeta}\left(\boldsymbol{\zeta} \in \mathbb{R}^n \ : \ \boldsymbol{f}(\boldsymbol{s}, \boldsymbol{\zeta}) \in B\right),$$

where T_{ζ} is the distribution of the r.v. $\boldsymbol{\zeta}(0)$ (in fact, of any $\boldsymbol{\zeta}(t)$ since these are iid random variables). In other words, the map $\boldsymbol{f}$ together with the distribution of the r.v. $\{\boldsymbol{\zeta}(t)\}$ uniquely define the stochastic kernel of the process. The converse is also true as shown in [12, Proposition 7.6]: any discrete-time Markov process $\mathscr{M}_{\mathfrak{s}}$ admits a dynamical representation as in (1), for an appropriate selection of function $\boldsymbol{f}$ and distribution of the r.v. $\{\boldsymbol{\zeta}(t)\}$.

Let us expand the dynamical equation (1) explicitly over its states $\boldsymbol{s} = [s_1, \ldots, s_n]^T$, map components $\boldsymbol{f} = [f_1, \ldots, f_n]^T$, and uncertainly terms $\boldsymbol{\zeta} = [\zeta_1, \ldots, \zeta_n]^T$, as follows:

$$\begin{aligned} s_1(t+1) &= f_1(s_1(t), s_2(t), \ldots, s_n(t), \zeta_1(t)), \\ s_2(t+1) &= f_2(s_1(t), s_2(t), \ldots, s_n(t), \zeta_2(t)), \\ &\vdots \\ s_n(t+1) &= f_n(s_1(t), s_2(t), \ldots, s_n(t), \zeta_n(t)). \end{aligned} \tag{2}$$

In this article we are interested in exploiting the knowledge of the structure of the dynamics in (2) for formal verification via abstractions [1, 7, 8]. We focus our attention to continuous (unbounded and uncountable) Euclidean spaces $\mathcal{S} = \mathbb{R}^n$, and further assume that for any $t \in \mathbb{N}$, $\zeta_k(t)$ are independent for all $k \in \mathbb{N}_n$. This latter assumption is widely used in the theory of dynamical systems, and allows for the following multiplicative structure on the conditional density function of the process:

$$t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) = t_1(\bar{s}_1|\boldsymbol{s}) t_2(\bar{s}_2|\boldsymbol{s}) \ldots t_n(\bar{s}_n|\boldsymbol{s}),$$

where the function $t_k : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}_{\geq 0}$ solely depends on the map f_k and the distribution of ζ_k. The reader is referred to Section 5 for the detailed computation of the functions t_k from the dynamical equations in (2).

▶ **Example 1.** Figure 1 shows a system of n masses connected by springs and dampers. For $i \in \mathbb{N}_n$, block i has mass m_i, the i^{th} spring has stiffness k_i, and the i^{th} damper has damping coefficient b_i. The first mass is connected to a fixed wall by the left-most spring/damper connection. All other masses are connected to the previous mass with a spring and a damper. A force ζ_i is applied to each mass, modelling the effect of a disturbance or of process noise. The dynamics of the overall system is comprised of the position and velocity of the blocks. It can be shown that the dynamics in discrete time take the form $\boldsymbol{s}(t+1) = \Phi \boldsymbol{s}(t) + \boldsymbol{\zeta}(t)$, where $\boldsymbol{s}(t) \in \mathbb{R}^{2n}$ with $s_{2i-1}(t), s_{2i}(t)$ indicating the velocity and position of mass i. The state transition matrix $\Phi = [\Phi_{ij}]_{i,j} \in \mathbb{R}^{2n \times 2n}$ is a band matrix with lower and upper bandwidth 3 and 2, respectively ($\Phi_{ij} = 0$ for $j < i - 3$ and for $j > i + 2$). ◀

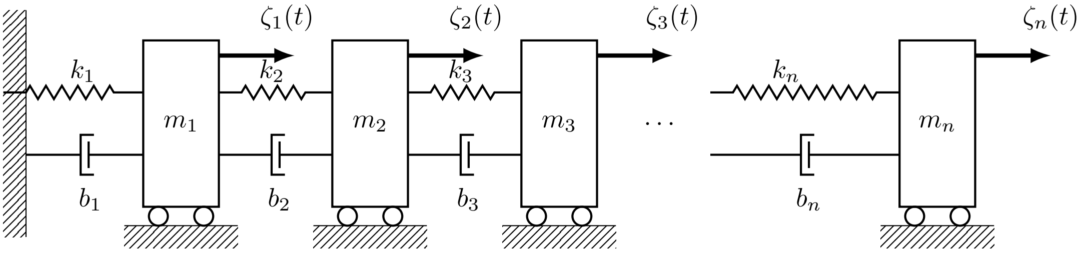

Figure 1 n-body mass-spring-damper system.

▶ **Example 2.** A second example of structured dynamical systems is a discrete-time large-scale interconnected system. Consider an interconnected system of $N_{\mathfrak{d}}$ heterogeneous linear time-invariant (LTI) subsystems described by the following stochastic difference equations:

$$\boldsymbol{s}_i(t+1) = \Phi_i \boldsymbol{s}_i(t) + \sum_{j \in N_i} G_{ij} \boldsymbol{s}_j(t) + B_i \boldsymbol{u}_i(t) + \boldsymbol{\zeta}_i(t),$$

where $i \in \mathbb{N}_{N_{\mathfrak{d}}}$ denotes the i^{th} subsystem and $\boldsymbol{s}_i \in \mathbb{R}^{n\times 1}, \boldsymbol{u}_i \in \mathbb{R}^{p\times 1}, \boldsymbol{\zeta}_i \in \mathbb{R}^{m\times 1}$ are the state, the input, and the process noise of subsystem i. The term $\sum_{j\in N_i} G_{ij}\boldsymbol{s}_j(t)$ represents the physical interconnection between the subsystems where N_i, $|N_i| \ll N_{\mathfrak{d}}$, is the set of subsystems to which system i is physically connected. The described interconnected system can be found in many application areas including smart power grids, traffic systems, and sensor-actuator networks [10]. ◀

2.2 Probabilistic Invariance

We focus on verifying probabilistic invariance, which plays a central role in verifying properties of a system expressed as PCTL formulae or as linear temporal specifications [5, 21, 23].

▶ **Definition 3** (Probabilistic Invariance). Consider a bounded Borel set $A \in \mathcal{B}$, representing a set of safe states. The finite-horizon *probabilistic invariance problem* asks to compute the probability that a trajectory of $\mathscr{M}_{\mathfrak{s}}$ associated with an initial condition $\boldsymbol{s}_0$ remains within the set A during the finite time horizon N:

$$p_N(\boldsymbol{s}_0, A) = \mathcal{P}\{\boldsymbol{s}(t) \in A \text{ for all } t = 0,1,2,\dots,N | \boldsymbol{s}(0) = \boldsymbol{s}_0\}.$$

This quantity allows us to extend the result to a general probability distribution $\pi : \mathcal{B} \to [0,1]$ for the initial state $\boldsymbol{s}(0)$ of the system as

$$\mathcal{P}\{\boldsymbol{s}(t) \in A \text{ for all } t = 0,1,2,\dots,N\} = \int_{\mathcal{S}} p_N(\boldsymbol{s}_0, A)\pi(d\boldsymbol{s}_0). \tag{3}$$

Solution of the probabilistic invariance problem can be characterized via the value functions $V_k : \mathcal{S} \to [0,1]$, $k = 0,1,2,\dots,N$, defined by the following Bellman backward recursion [1]:

$$V_k(\boldsymbol{s}) = \mathbf{1}_A(\boldsymbol{s}) \int_A V_{k+1}(\bar{\boldsymbol{s}}) t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) d\bar{\boldsymbol{s}} \quad \text{for} \quad k = 0,1,2,\dots,N. \tag{4}$$

This recursion is initialized with $V_N(\boldsymbol{s}) = \mathbf{1}_A(\boldsymbol{s})$, where $\mathbf{1}_A(\boldsymbol{s})$ is the indicator function which is 1 if $\boldsymbol{s} \in A$ and 0 otherwise, and results in the solution $p_N(\boldsymbol{s}_0, A) = V_0(\boldsymbol{s}_0)$.

Equation (4) characterizes the finite-horizon probabilistic invariance quantity as the solution of a dynamic programming problem. However, since its explicit solution is in general not available, the actual computation of the quantity $p_N(\boldsymbol{s}_0, A)$ requires N numerical integrations at each state in the set A. This is usually performed with techniques based on state-space discretization [6].

3 Formal Abstractions as Dynamic Bayesian Networks

3.1 Dynamic Bayesian Networks

A Bayesian network (BN) is a tuple $\mathfrak{B} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$. The pair $(\mathcal{V}, \mathcal{E})$ is a directed Acyclic Graph (DAG) representing the structure of the network. The nodes in $\mathcal{V}$ are (discrete or continuous) random variables and the arcs in $\mathcal{E}$ represent the dependence relationships among the random variables. The set $\mathcal{T}$ contains conditional probability distributions (CPD) in forms of tables or density functions for discrete and continuous random variables, respectively. In a BN, knowledge is represented in two ways: qualitatively, as dependences between variables by means of the DAG; and quantitatively, as conditional probability distributions attached to the dependence relationships. Each random variable $X_i \in \mathcal{V}$ is associated to a family of conditional probability distributions $\mathbb{P}(X_i|Pa(X_i))$, where $Pa(Y)$ represents the parent set of the variable $Y \in \mathcal{V}$: $Pa(Y) = \{X \in \mathcal{V}|(X, Y) \in \mathcal{E}\}$. A BN is called *two-layered* if the set of nodes $\mathcal{V}$ can be partitioned to two sets $\mathcal{V}_1, \mathcal{V}_2$ with the same cardinality such that only the nodes in the second layer $\mathcal{V}_2$ have an associated CPD.

A dynamic Bayesian network [14, 19] is a way to extend Bayesian networks to model probability distributions over collections of random variables $X(0), X(1), X(2), \ldots$ indexed by time t. A DBN[2] is defined to be a pair $(\mathfrak{B}_0, \mathfrak{B}_\rightarrow)$, where $\mathfrak{B}_0$ is a BN which defines the distribution of $X(0)$, and $\mathfrak{B}_\rightarrow$ is a two-layered BN that defines the transition probability distribution for $(X(t+1)|X(t))$.

3.2 DBNs as Representations of Markov Processes

We now show that any discrete-time Markov process $\mathscr{M}_\mathfrak{s}$ over $\mathbb{R}^n$ can be represented as a DBN $(\mathfrak{B}_0, \mathfrak{B}_\rightarrow)$ over n continuous random variables. The advantage of the reformulation is that it makes the dependencies between random variables explicit.

The BN $\mathfrak{B}_0$ is trivial for a given initial state of the Markov process $\boldsymbol{s}(0) = \boldsymbol{s}_0$. The DAG of $\mathfrak{B}_0$ has the set of nodes $\{X_1, X_2, \ldots, X_n\}$ without any arc. The Dirac delta distribution located in the initial state of the process is assigned to each node of $\mathfrak{B}_0$.[3] The DAG for the two-layered BN $\mathfrak{B}_\rightarrow = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ comprises a set of nodes $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2$, with $\mathcal{V}_1 = \{X_1, X_2, \ldots, X_n\}$ and $\mathcal{V}_2 = \{\bar{X}_1, \bar{X}_2, \ldots, \bar{X}_n\}$. Each arc in $\mathcal{E}$ connects a node in $\mathcal{V}_1$ to another node in $\mathcal{V}_2$; $(X_i, \bar{X}_j) \in \mathcal{E}$ if and only if $t_j(\bar{s}_j|\boldsymbol{s})$ is not a constant function of s_i. The set $\mathcal{T}$ assigns a CPD to each node $\bar{X}_j$ according to the density function $t_j(\bar{s}_j|\boldsymbol{s})$.

▶ **Example 4.** Consider the following stochastic linear dynamical system:

$$\boldsymbol{s}(t+1) = \Phi \boldsymbol{s}(t) + \boldsymbol{\zeta}(t) \quad t \in \mathbb{N}, \quad \boldsymbol{s}(0) = \boldsymbol{s}_0 = [s_{01}, s_{02}, \ldots, s_{0n}]^T, \tag{5}$$

where $\Phi = [a_{ij}]_{i,j}$ is the system matrix and $\boldsymbol{\zeta}(t) \sim \mathcal{N}(0, \Sigma)$ are independent Gaussian r.v. for any $t \in \mathbb{N}$. The covariance matrix Σ is assumed to be full rank. Consequently, a

[2] The DBNs considered in this paper are stationary (the structure of the network does not change with the time index t). They have no input variables and are fully observable: the output of the DBN model equals to its state.

[3] For a general initial probability distribution $\pi : \mathcal{B} \rightarrow [0, 1]$, a set of arcs must be added to reflect its possible product structure. This construction is not important at the current stage because of the backward recursion formulation of the probabilistic safety (please refer to (3) in Section 2.2).

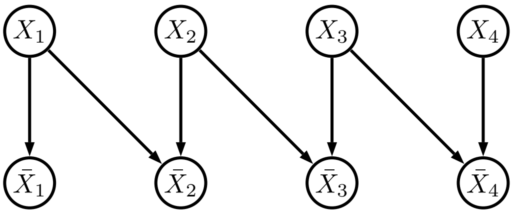

Figure 2 Two-layered BN $\mathfrak{B}_{\rightarrow}$ associated with the stochastic linear dynamical system in (5) for $n = 4$.

linear transformation can be employed to change the coordinates and obtain a stochastic linear system with a diagonal covariance matrix. Then without loss of generality we assume $\Sigma = diag([\sigma_1^2, \sigma_2^2, \dots, \sigma_n^2])$, which clearly satisfies the independence assumption on the process noise raised in Section 2.1. Model (5) for a lower bidiagonal matrix Φ can be expanded as follows:

$$\begin{aligned}
s_1(t+1) &= a_{11}s_1(t) + \zeta_1(t)\\
s_2(t+1) &= a_{21}s_1(t) + a_{22}s_2(t) + \zeta_2(t)\\
s_3(t+1) &= a_{32}s_2(t) + a_{33}s_3(t) + \zeta_3(t)\\
&\vdots\\
s_n(t+1) &= a_{n(n-1)}s_{n-1}(t) + a_{nn}s_n(t) + \zeta_n(t),
\end{aligned}$$

where $\zeta_i(\cdot)$, $i \in \mathbb{N}_n$ are independent Gaussian r.v. $\mathcal{N}(0, \sigma_i^2)$. The conditional density function of the system takes the following form:

$$t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) = t_1(\bar{s}_1|s_1)t_2(\bar{s}_2|s_1, s_2)t_3(\bar{s}_3|s_2, s_3)\dots t_n(\bar{s}_n|s_{n-1}, s_n).$$

The DAG of the two-layered BN $\mathfrak{B}_{\rightarrow}$ associated with this system is sketched in Figure 2 for $n = 4$. The BN $\mathfrak{B}_0$ has an empty graph on the set of nodes $\{X_1, \dots, X_n\}$ with the associated Dirac delta density functions located at s_{0i}, $\delta_d(s_i(0) - s_{0i})$. ◀

3.3 Finite Abstraction of Markov Processes as Discrete DBNs

Let $A \in \mathcal{B}$ be a bounded Borel set of safe states. We abstract the structured Markov process $\mathscr{M}_{\mathfrak{s}}$ interpreted in the previous section as a DBN with continuous variables to a DBN with discrete random variables. Our abstraction is relative to the set A. Algorithm 1 provides the steps of the abstraction procedure. It consists of discretizing each dimension into a finite number of bins.

In Algorithm 1, the projection operators $\Pi_i : \mathbb{R}^n \rightarrow \mathbb{R}$, $i \in \mathbb{N}_n$, are defined as $\Pi_i(\boldsymbol{s}) = s_i$ for any $\boldsymbol{s} = [s_1, \dots, s_n]^T \in \mathbb{R}^n$. These operators are used to project the safe set A over different dimensions, $D_i \doteq \Pi_i(A)$. In step 2 of the Algorithm, set D_i is partitioned as $\{D_{ij}\}_{j=1}^{n_i}$ (for any $i \in \mathbb{N}_n$, D_{ij}'s are arbitrary but non-empty, non-intersecting, and $D_i = \cup_{j=1}^{n_i} D_{ij}$). The corresponding representative points $z_{ij} \in D_{ij}$ are also chosen arbitrarily. Step 5 of the algorithm constructs the support of the random variables in $\mathfrak{B}_{\rightarrow}$, $\mathcal{V} = \{X_i, \bar{X}_i, i \in \mathbb{N}_n\}$, and step 6 computes the discrete CDPs $\mathbb{P}(\bar{X}_i|Pa(\bar{X}_i))$, reflecting the dependencies among the variables. For any $i \in \mathbb{N}_n$, $\Xi_i : Z_i \rightarrow 2^{D_i}$ represents a set-valued map that associates to any point $z_{ij} \in Z_i$ the corresponding partition set $D_{ij} \subset D_i$ (this is known as the "refinement map"). Furthermore, the abstraction map $\xi_i : D_i \rightarrow Z_i$ associates to any point

Algorithm 1 Abstraction of model $\mathscr{M}_\mathfrak{s}$ as a DBN with $\mathfrak{B}_\rightarrow = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ over discrete r.v.

```
Require: input model $\mathscr{M}_\mathfrak{s} = (\mathcal{S}, \mathcal{B}, T_\mathfrak{s})$, safe set $A$
1: Project safe set $A$ in each dimension $D_i \doteq \Pi_i(A)$, $i \in \mathbb{N}_n$
2: Select finite $n_i$-dimensional partition of $D_i$ as $D_i = \cup_{j=1}^{n_i} D_{ij}$, $i \in \mathbb{N}_n$
3: For each $D_{ij}$, select single representative point $z_{ij} \in D_{ij}, \{z_{ij}\} = \xi_i(D_{ij})$
4: Construct the DAG $(\mathcal{V}, \mathcal{E})$, with $\mathcal{V} = \{X_i, \bar{X}_i, i \in \mathbb{N}_n\}$ and $\mathcal{E}$ as per Section 3.2
5: Define $Z_i = \{z_{i1}, \ldots, z_{in_i}\}$, $i \in \mathbb{N}_n$, and take $\Omega_i = Z_i \cup \{\phi_i\}$ as the finite state space of
   two r.v. $X_i$ and $\bar{X}_i$, $\phi_i$ being dummy variables as per Section 3.3
6: Compute elements of the set $\mathcal{T}$, namely CPD $T_i$ related to the node $\bar{X}_i$, $i \in \mathbb{N}_i$, as
   $$T_i(\bar{X}_i = z | v(Pa(\bar{X}_i))) = \begin{cases} \int_{\Xi_i(z)} t_i(\bar{s} | v(Pa(\bar{X}_i))) d\bar{s}, & z \in Z_i,\ v(Pa(\bar{X}_i)) \cap \phi = \emptyset \\ 1 - \sum_{z \in Z_i} \int_{\Xi_i(z)} t_i(\bar{s} | v(Pa(\bar{X}_i))) d\bar{s}, & z = \phi_i,\ v(Pa(\bar{X}_i)) \cap \phi = \emptyset \\ 1, & z = \phi_i,\ v(Pa(\bar{X}_i)) \cap \phi \neq \emptyset \\ 0, & z \in Z_i,\ v(Pa(\bar{X}_i)) \cap \phi \neq \emptyset \end{cases}$$
Ensure: output DBN with $\mathfrak{B}_\rightarrow = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ over discrete r.v.
```

$s_i \in D_i$ the corresponding discrete state in Z_i. Additionally, notice that the absorbing states $\phi = \{\phi_1, \ldots, \phi_n\}$ are added to the definition of BN $\mathfrak{B}_\rightarrow$ so that the conditional probabilities $\mathbb{P}(\bar{X}_i | Pa(\bar{X}_i))$ marginalise to one. The function $v(\cdot)$ used in step 6 acts on (possibly a set of) random variables and provides their instantiation.

The construction of the DBN with discrete r.v. in Algorithm 1 is closely related to the Markov chain abstraction method in [1, 8]. The main difference lies in partitioning in each dimension separately instead of doing it for the whole state space. Absorbing states are also assigned to each dimension separately instead of having only one for the unsafe set. Moreover, Algorithm 1 stores the transition probabilities efficiently as a BN.

3.4 Probabilistic Invariance for the Abstract DBN

Given a DBN with discrete r.v. $\boldsymbol{z} = (X_1, X_2, \ldots, X_n)$ and a discrete set $Z_\mathfrak{a} \subset \prod_i \Omega_i$, the probabilistic invariance problem asks to evaluate the probability $p_N(\boldsymbol{z}_0, Z_\mathfrak{a})$ that a finite execution associated with the initial condition $\boldsymbol{z}(0) = \boldsymbol{z}_0$ remains within the set $Z_\mathfrak{a}$ during the finite time horizon $t = 0, 1, 2, \ldots, N$. Formally,

$$p_N(\boldsymbol{z}_0, Z_\mathfrak{a}) = \mathbb{P}(\boldsymbol{z}(t) \in Z_\mathfrak{a}, \text{ for all } t = 0, 1, 2, \ldots, N | \boldsymbol{z}(0) = \boldsymbol{z}_0).$$

This probability can be computed by a discrete analogue of the Bellman backward recursion (see [2] for details).

▶ **Theorem 5.** *Consider value functions $V_k^d : \prod_i \Omega_i \rightarrow [0, 1]$, $k = 0, 1, 2, \ldots, N$, computed by the backward recursion*

$$V_k^d(\boldsymbol{z}) = \mathbf{1}_{Z_\mathfrak{a}}(\boldsymbol{z}) \sum_{\bar{\boldsymbol{z}} \in \prod_i \Omega_i} V_{k+1}^d(\bar{\boldsymbol{z}}) \mathbb{P}(\bar{\boldsymbol{z}} | \boldsymbol{z}) \quad k = 0, 1, 2, \ldots, N, \tag{6}$$

and initialised with $V_N^d(\boldsymbol{z}) = \mathbf{1}_{Z_\mathfrak{a}}(\boldsymbol{z})$. Then the solution of the invariance problem is characterised as $p_N(\boldsymbol{z}_0, Z_\mathfrak{a}) = V_0^d(\boldsymbol{z}_0)$.

The discrete transition probabilities $\mathbb{P}(\bar{\boldsymbol{z}}|\boldsymbol{z})$ in Equation (6) are computed by taking the product of the CPD in $\mathcal{T}$. More specifically, for any $\boldsymbol{z}, \bar{\boldsymbol{z}} \in \prod_i \Omega_i$ of the form $\boldsymbol{z} = (z_1, z_2, \ldots, z_n), \bar{\boldsymbol{z}} = (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_n)$ we have

$$\mathbb{P}(\bar{\boldsymbol{z}}|\boldsymbol{z}) = \prod_i T_i(\bar{X}_i = \bar{z}_i | Pa(\bar{X}_i) = \boldsymbol{z}).$$

Our algorithm for probabilistic invariance computes $p_N(\boldsymbol{z}_0, Z_{\mathfrak{a}})$ to approximate $p_N(\boldsymbol{s}_0, A)$, for suitable choices of $\boldsymbol{z}_0$ and $Z_{\mathfrak{a}}$ depending on $\boldsymbol{s}_0$ and A. The natural choice for the initial state is $\boldsymbol{z}_0 = (z_1(0), \ldots, z_n(0))$ with $z_i(0) = \xi_i(\Pi_i(\boldsymbol{s}_0))$. For A, the n-fold Cartesian product of the collection of the partition sets $\{D_{ij}\}$, $i \in \mathbb{N}_n$ generates a cover of A as

$$\begin{aligned} A &\subset \bigcup \{D_{1j}\}_{j=1}^{n_1} \times \{D_{2j}\}_{j=1}^{n_2} \times \ldots \times \{D_{nj}\}_{j=1}^{n_n} \\ &= \bigcup_{\boldsymbol{j}} \{D_{\boldsymbol{j}} | \boldsymbol{j} = (j_0, j_1, \ldots, j_n), D_{\boldsymbol{j}} = D_{1j_1} \times D_{2j_2} \times \ldots \times D_{nj_n}\}. \end{aligned}$$

We define the safe set $Z_{\mathfrak{a}}$ of the DBN as

$$Z_{\mathfrak{a}} = \bigcup_{\boldsymbol{j}} \{(z_{1j_1}, z_{2j_2}, \ldots, z_{nj_n}), \text{ such that } A \cap D_{\boldsymbol{j}} \neq \emptyset \text{ for } \boldsymbol{j} = (j_1, j_2, \ldots, j_n)\}, \tag{7}$$

which is a discrete representation of the continuous set $\bar{A} \subset \mathbb{R}^n$

$$\bar{A} = \bigcup_{\boldsymbol{j}} \{D_{\boldsymbol{j}}, \text{ such that } \boldsymbol{j} = (j_1, j_2, \ldots, j_n), A \cap D_{\boldsymbol{j}} \neq \emptyset\}. \tag{8}$$

For instance $\bar{A}$ can be a finite union of hypercubes in $\mathbb{R}^n$ if the partition sets D_{ij} are intervals. It is clear that the set $\bar{A}$ is in general different form A.

There are thus two sources of error: first due to replacing A with $\bar{A}$, and second, due to the abstraction of the dynamics between the discrete outcome obtained by Theorem 5 and the continuous solution that results from (4). In the next section we provide a quantitative bound on the two sources of error.

3.5 Quantification of the Error due to Abstraction

Theorem 6 characterizes the error due to replacing the safe set A with $\bar{A}$.

▶ **Theorem 6.** *Solution of the probabilistic invariance problem with the time horizon N and two safe sets $A, \bar{A}$ satisfies the inequality*

$$|p_N(\boldsymbol{s}_0, A) - p_N(\boldsymbol{s}_0, \bar{A})| \leq MN\mathcal{L}(A\Delta\bar{A}), \quad \forall \boldsymbol{s}_0 \in A \cap \bar{A},$$

where $M = \sup\left\{t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) \middle| \boldsymbol{s}, \bar{\boldsymbol{s}} \in A\Delta\bar{A}\right\}$. $\mathcal{L}(B)$ denotes the Lebesgue measure of any set $B \in \mathcal{B}$ and $A\Delta\bar{A} \doteq (A\backslash\bar{A}) \cup (\bar{A}\backslash A)$ is the symmetric difference of the two sets $A, \bar{A}$.

The second contribution to the error is related to the discretization of Algorithm 1 which is quantified by posing regularity conditions on the dynamics of the process. The following Lipschitz continuity assumption restricts the generality of the density functions t_k characterizing the dynamics of model $\mathscr{M}_{\mathfrak{s}}$.

▶ Assumption 1. Assume the density functions $t_k(\bar{s}_i|\cdot)$ are Lipschitz continuous with the finite positive d_{ij}

$$|t_j(\bar{s}_j|\boldsymbol{s}) - t_j(\bar{s}_j|\boldsymbol{s'})| \leq d_{ij}\|s_i - s'_i\|,$$

with $\boldsymbol{s} = [s_1, \ldots, s_{i-1}, s_i, s_{i+1}, \ldots, s_n]$ and $\boldsymbol{s'} = [s_1, \ldots, s_{i-1}, s'_i, s_{i+1}, \ldots, s_n]$, for all $s_k, s'_k, \bar{s}_k \in D_k$, $k \in \mathbb{N}_n$, and for all $i, j \in \mathbb{N}_n$.

Note that $d_{ij} = 0$ if and only if $(X_i, \bar{X}_j) \notin \mathcal{E}$ in the DAG of the BN $\mathfrak{B}_{\rightarrow}$. Assumption 1 enables us to assign non-zero weights to the arcs of the graph and turn it into a weighted DAG. The non-zero weight $w_{ij} = d_{ij}\mathcal{L}(D_j)$ is assigned to the arc $(X_i, \bar{X}_j) \in \mathcal{E}$, for all $i, j \in \mathbb{N}_n$. We define the out-weight of the node X_i by $\mathcal{O}_i = \sum_{j=1}^{n} w_{ij}$ and the in-weight of the node $\bar{X}_j$ by $\mathcal{I}_j = \sum_{i=1}^{n} w_{ij}$.

▶ Remark. The above assumption implies Lipschitz continuity of the conditional density functions $t_j(\bar{s}_j|\boldsymbol{s})$. Using a chain of triangle inequalities results in

$$|t_j(\bar{s}_j|\boldsymbol{s}) - t_j(\bar{s}_j|\boldsymbol{s}')| \leq \mathcal{H}_j\|\boldsymbol{s} - \boldsymbol{s}'\|,$$

for all $\boldsymbol{s}, \boldsymbol{s}' \in \bar{A}, \bar{s}_j \in D_j$ where $\mathcal{H}_j = \sum_{i=1}^{n} d_{ij}$. The density function $t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s})$ is also Lipschitz continuous if the density functions $t_j(\bar{s}_j|\boldsymbol{s})$ are bounded, but the boundedness assumption is not necessary for our result to hold.

Assumption 1 enables us to establish Lipschitz continuity of the value functions W_k used in Bellman recursion (4) of the safety problem over the set $\bar{A}$. This continuity property is essential in proving an upper bound on the discretization error of Algorithm 1, which is presented in Corollary 8.

▶ **Lemma 7.** *Consider the value functions $W_k(\cdot)$, $k = 0, 1, 2, \ldots, N$, employed in Bellman recursion (4) of the safety problem over the set $\bar{A}$. Under Assumption 1, these value functions are Lipschitz continuous*

$$|W_k(\boldsymbol{s}) - W_k(\boldsymbol{s}')| \leq \kappa\|\boldsymbol{s} - \boldsymbol{s}'\|, \quad \forall \boldsymbol{s}, \boldsymbol{s}' \in \bar{A},$$

for all $k = 0, 1, 2, \ldots, N$ with the constant $\kappa = \sum_{j=1}^{n} \mathcal{I}_j$, where $\mathcal{I}_j$ is the in-weight of the node $\bar{X}_j$ in the DAG of the BN $\mathfrak{B}_{\rightarrow}$.

▶ **Corollary 8.** *The following inequality holds under Assumption 1:*

$$|p_N(\boldsymbol{s}_0, A) - p_N(\boldsymbol{z}_0, Z_{\mathfrak{a}})| \leq MN\mathcal{L}(A\Delta\bar{A}) + N\kappa\delta \quad \forall \boldsymbol{s}_0 \in A,$$

where $p_N(\boldsymbol{z}_0, Z_{\mathfrak{a}})$ is the invariance probability for the DBN obtained by Algorithm 1. The initial state of the DBN is $\boldsymbol{z}_0 = (z_1(0), \ldots, z_n(0))$ with $z_i(0) = \xi_i(\Pi_i(\boldsymbol{s}_0))$. The set $Z_{\mathfrak{a}}$ and the constant M are defined in (7) and Theorem 6, respectively. The diameter of the partition of Algorithm 1 is defined and used as $\delta = \sup\{\|\boldsymbol{s} - \boldsymbol{s}'\|, \forall \boldsymbol{s}, \boldsymbol{s}' \in D_{\boldsymbol{j}}, \forall \boldsymbol{j}\ \ D_{\boldsymbol{j}} \subset \bar{A}\}$.

The second error term in Corollary 8 is a linear function of the partition diameter δ, which depends on all partition sets along different dimensions. We are interested in proving a dimension-dependent error bound in order to parallelize the whole abstraction procedure along different dimensions. The next theorem gives this dimension-dependent error bound.

▶ **Theorem 9.** *The following inequality holds under Assumption 1:*

$$|p_N(\boldsymbol{s}_0, A) - p_N(\boldsymbol{z}_0, Z_{\mathfrak{a}})| \leq MN\mathcal{L}(A\Delta\bar{A}) + N\sum_{i=1}^{n} \mathcal{O}_i\delta_i \quad \forall \boldsymbol{s}_0 \in A, \tag{9}$$

with the constants defined in Corollary 8. $\mathcal{O}_j$ is the out-weight of the node X_i in the DAG of the BN $\mathfrak{B}_{\rightarrow}$. The quantity δ_i is the maximum diameter of the partition sets along the i^{th} dimension $\delta_i = \sup\{\|s_i - s_i'\|, \forall s_i, s_i' \in D_{ij}, \forall j \in \mathbb{N}_{n_i}\}$.

For a given error threshold ϵ, we can select the set $\bar{A}$ and consequently the diameters δ_i such that $MN\mathcal{L}(A\Delta\bar{A}) + N\sum_{i=1}^{n} \mathcal{O}_i\delta_i \leq \epsilon$. Therefore, generation of the abstract DBN, namely selection of the partition sets $\{D_{ij},\ j \in \mathbb{N}_i\}$ (according to the diameter δ_i) and computation of the CPD, can be implemented in parallel.

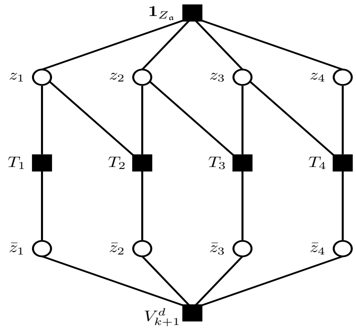

Figure 3 Factor graph of the linear stochastic system (5) for $n = 4$.

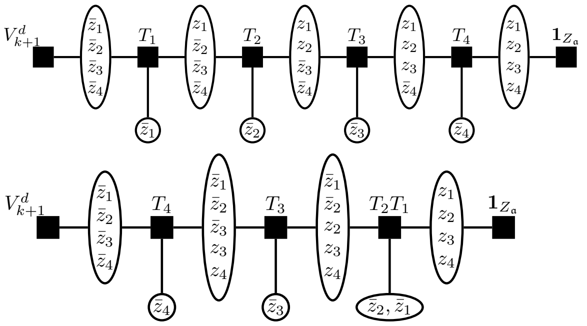
V^d_{k+1}
$\bar z_1, \bar z_2, \bar z_3, \bar z_4$
T_1
$z_1, \bar z_2, \bar z_3, \bar z_4$
T_2
$z_1, z_2, \bar z_3, \bar z_4$
T_3
$z_1, z_2, z_3, \bar z_4$
T_4
z_1, z_2, z_3, z_4
$\mathbf{1}_{Z_\mathfrak{a}}$
$\bar z_1$
$\bar z_2$
$\bar z_3$
$\bar z_4$
V^d_{k+1}
$\bar z_1, \bar z_2, \bar z_3, \bar z_4$
T_4
$\bar z_1, \bar z_2, \bar z_3, z_3, z_4$
T_3
$\bar z_1, \bar z_2, z_2, z_3, z_4$
T_2T_1
z_1, z_2, z_3, z_4
$\mathbf{1}_{Z_\mathfrak{a}}$
$\bar z_4$
$\bar z_3$
$\bar z_2, \bar z_1$

Figure 4 Spanning tree of the linear stochastic system in (5) for $n = 4$ and two orderings $(\bar z_1, \bar z_2, \bar z_3, \bar z_4)$ (top plot) and $(\bar z_4, \bar z_3, \bar z_2, \bar z_1)$ (bottom plot).

4 Efficient Model Checking of the Finite-State DBN

Existing numerical methods for model checking DBNs with discrete r.v. transform the DBN into an explicit matrix representation [11, 18, 20], which defeats the purpose of a compact representation. Instead, we show that the multiplicative structure of the transition probability matrix can be incorporated in the computation which makes the construction of $\mathbb{P}(\bar{\boldsymbol z}|\boldsymbol z)$ dispensable. For this purpose we employ *factor graphs* and the *sum-product algorithm* [16] originally developed for marginalising functions and applied to belief propagation in Bayesian networks. Suppose that a *global* function is given as a product of *local* functions, and that each local function depends on a subset of the variables of the global map. In its most general form, the sum-product algorithm acts on factor graphs in order to marginalise the global function, i.e., taking summation respect to a subset of variables, exploiting its product structure [16]. In our problem, we restrict the summation domain of the Bellman recursion (6) to $\prod_i Z_i$ because the value functions are simply equal to zero in the complement of this set. The summand in (6) has the multiplicative structure

$$g(\boldsymbol z, \bar{\boldsymbol z}) \doteq \mathbf{1}_{Z_\mathfrak{a}}(\boldsymbol z) V^d_{k+1}(\bar{\boldsymbol z}) \prod_i T_i(\bar X_i = \bar z_i | Pa(\bar X_i) = \boldsymbol z), \quad V^d_k(\boldsymbol z) = \sum_{\bar{\boldsymbol z} \in \prod_i Z_i} g(\boldsymbol z, \bar{\boldsymbol z}).$$

The function $g(\boldsymbol z, \bar{\boldsymbol z})$ depends on variables $\{z_i, \bar z_i, \, i \in \mathbb{N}_n\}$. The factor graph of $g(\boldsymbol z, \bar{\boldsymbol z})$ has $2n$ *variable nodes*, one for each variable and $(n+2)$ *function nodes* for local functions $\mathbf{1}_{Z_\mathfrak{a}}, V^d_{k+1}, T_i$. An arc connects a variable node to a function node if and only if the variable is an argument of the local function. The factor graph of Example 4 for $n = 4$ is presented in Figure 3 – factor graphs of general functions $g(\boldsymbol z, \bar{\boldsymbol z})$ are similar to that in Figure 3, the only part needing to be modified being the set of arcs connecting variable nodes $\{z_i, \, i \in \mathbb{N}_n\}$ and function nodes $\{T_i, \, i \in \mathbb{N}_n\}$. This part of the graph can be obtained from the DAG of $\mathfrak{B}_\rightarrow$ of the DBN.

The factor graph of a function $g(\boldsymbol z, \bar{\boldsymbol z})$ contains loops for $n \geq 2$ and must be transformed to a *spanning tree* using clustering and stretching transformations [16]. For this purpose the order of clustering function nodes $\{T_i, i \in \mathbb{N}_n\}$ and that of stretching variable nodes $\{z_i, \, i \in \mathbb{N}_n\}$ needs to be chosen. Figure 4 presents the spanning trees of the stochastic system in (5) for two such orderings. The variable nodes at the bottom of each spanning tree specify the order of the summation, whereas the function nodes considered from the left to the right indicate the order of multiplication of the local functions. The rest of the variable nodes show the arguments of the intermediate functions, which reflects the required

Algorithm 2 Greedy algorithm for obtaining the order of stretching variables and clustering functions in the factor graph

```
Require: Factor graph of the summand in Bellman recursion
Initialize the sets $\mathcal{U}_1 = \{z_i,\, i \in \mathbb{N}_n\}$, $\mathcal{U}_2 = \{\bar{z}_i,\, i \in \mathbb{N}_n\}$, $\mathcal{U}_3 = \{T_i,\, i \in \mathbb{N}_n\}$, $e = \kappa = \emptyset$
while $\mathcal{U}_1 \neq \emptyset$ do
    For any node $u \in \mathcal{U}_3$ compute $Pa(u)$ (resp. $Ch(u)$) as the elements of $\mathcal{U}_1$ (resp. $\mathcal{U}_2$) connected to $u$ by an arc in the factor graph
    Define the equivalence relation $R$ on $\mathcal{U}_3$ as $uR\bar{u}$ iff $Pa(u) = Pa(\bar{u})$
    Replace the set $\mathcal{U}_3$ with the set of equivalence classes induced by $R$.
    Combine all the variable nodes of $Ch(u)$ connected to one class
    Select $u \in \mathcal{U}_3$ with the minimum cardinality of $Pa(u)$ and put $e = (e, u), \kappa = (\kappa, Ch(u))$
    Update the sets $\mathcal{U}_1 = \mathcal{U}_1 \backslash Pa(u)$, $\mathcal{U}_2 = \mathcal{U}_2 \cup Pa(u) \backslash Ch(u)$, $\mathcal{U}_3 = \mathcal{U}_3 \backslash \{u\}$, and eliminate all the arcs connected to $u$
end while
Ensure: The order of variables $\kappa$ and functions $e$
```

memory for storing such functions. The computational complexity of the solution carried out on the spanning tree clearly depends on this ordering.

Algorithm 2 presents a greedy procedure that operates on the factor graph and provides an ordering of the variables and of the functions, in order to reduce the overall memory usage. This algorithm iteratively combines the function nodes and selects the next variable node, over which the summation is carried out. The output of this algorithm implemented on the factor graph of Example 4 are the orderings $\kappa = (\bar{z}_1, \bar{z}_2, \bar{z}_3, \bar{z}_4)$ and $e = (T_1, T_2, T_3, T_4)$, which is related to the spanning tree on top of Figure 4.

5 Comparison with the State-of-the-Art

In this section we compare our approach with the state-of-the-art abstraction procedure presented in [1] (referred to as AKLP in the following), which does not exploit the structure of the dynamics. The AKLP algorithm approximates the concrete model with a finite-state Markov chain by uniformly gridding the safe set. As in our work, the error bound of the AKLP procedure depends on the global Lipschitz constant of the density of the model, however it does not exploit its structure as proposed in this work. We compare the two procedures on (1) error bounds and (2) computational resources.

Consider the stochastic linear dynamical model in (5), where $\Phi = [a_{ij}]_{i,j}$ is an arbitrary matrix. The Lipschitz constants d_{ij} in Assumption 1 can be computed as $d_{ij} = |a_{ji}|/\sigma_j^2\sqrt{2\pi e}$, where e is Euler's constant. From Theorem 9, we get the following error bound:

$$e_s = MN\mathcal{L}(A\Delta\bar{A}) + \frac{N}{\sqrt{2\pi e}}\sum_{i,j=1}^{n}\frac{|a_{ji}|}{\sigma_j^2}\mathcal{L}(D_j)\delta_i.$$

On the other hand, the error bound for AKLP is

$$e_{\text{AKLP}} = MN\mathcal{L}(A\Delta\bar{A}) + \frac{Ne^{-1/2}}{(\sqrt{2\pi})^n\sigma_1\sigma_2\ldots\sigma_n}\|\Sigma^{-1/2}\Phi\|_2\delta\mathcal{L}(A).$$

In order to meaningfully compare the two error bounds, select set $A = [-\alpha, \alpha]^n$ and

$\sigma_i = \sigma, i \in \mathbb{N}_n$, and consider hypercubes as partition sets. The two error terms then become

$$e_s = \varsigma n\eta \left(\frac{\|\Phi\|_1}{n\sqrt{n}}\right), \quad e_{\mathrm{AKLP}} = \varsigma\eta^n\|\Phi\|_2, \quad \eta = \frac{2\alpha}{\sigma\sqrt{2\pi}}, \quad \varsigma = \frac{N\delta}{\sigma\sqrt{e}},$$

where $\|\Phi\|_1$ and $\|\Phi\|_2$ are the entry-wise one-norm and the induced two-norm of matrix Φ, respectively. The error e_{AKLP} depends exponentially on the dimension n as η^n, whereas we have reduced this term to a linear one ($n\eta$) in our proposed new approach resulting in error e_s. Note that $\eta \leq 1$ means that the standard deviation of the process noise is larger than the selected safe set: in this case the value functions (which characterize the probabilistic invariance problem) uniformly converge to zero with rate η^n; clearly the case of $\eta > 1$ is more interesting. On the other hand for any matrix Φ we have $\frac{\|\Phi\|_1}{n\sqrt{n}} \leq \|\Phi\|_2$. This second term indicates how sparsity is reflected in the error computation. Denote by r the degree of connectivity of the DAG of $\mathfrak{B}_{\rightarrow}$ for this linear system, which is the maximum number of non-zero elements in rows of matrix Φ. We apply Lemma 10 in the Appendix to matrix Φ to get the inequalities

$$\|\Phi\|_2 \leq \sqrt{nr}\max_{i,j}|a_{ij}|, \qquad \frac{\|\Phi\|_1}{n\sqrt{n}} \leq \frac{r}{\sqrt{n}}\max_{i,j}|a_{ij}|,$$

which shows that for a fixed dimension n, a sparse dynamics, compared to fully connected dynamics, results in better error bounds in the new approach.

In order to compare computational resources, consider the numerical values $N = 10$, $n = 4$, $\alpha = 1$, $\sigma = 0.2$, and the error threshold $\epsilon = 0.2$ for the lower bidiagonal matrix Φ with all the non-zero entries equal to one. To ensure the error is at most ϵ, the cardinality of the partition of each dimension for the uniform gridding and for the structured approach is 2.9×10^5 and 8.5×10^3, respectively. Then, AKLP requires storing 4.8×10^{43} entries (which is infeasible!), whereas the DBN approach requires 1.8×10^{12} entries ($\sim$ 8GB). The number of (addition and multiplication) operations required for computation of the safety probability are 1.2×10^{45} and 3.5×10^{21}, respectively. This shows a substantial reduction in memory usage and computational effort.

6 Conclusions and Future Directions

While we have focused on probabilistic invariance, our abstraction approach can be extended to more general properties expressed within the bounded-horizon fragment of PCTL [21] or to bounded-horizon linear temporal properties [23, 24], since the model checking problem for these logics reduce to computations of value functions similar to the Bellman recursion scheme. Our focus in this paper has been the foundations of DBN-based abstraction for general Markov processes: factored representations, error bounds, and algorithms. We are currently implementing these algorithms in the FAUST^2 tool [9], and scaling the algorithms using dimension-dependent adaptive gridding [8] as well as implementations of the sum-product algorithm on top of data structures such as algebraic decision diagrams (as in probabilistic model checkers [17]).

References

1 A. Abate, J.-P. Katoen, J. Lygeros, and M. Prandini. Approximate model checking of stochastic hybrid systems. *European Journal of Control*, 6:624–641, 2010.

2 A. Abate, M. Prandini, J. Lygeros, and S. Sastry. Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems. *Automatica*, 44(11):2724–2734, 2008.

3 A. Abate, S. Vincent, R. Dobbe, A. Silletti, N. Master, J. Axelrod, and C.J. Tomlin. A mechanical modeling framework for the study of epithelial morphogenesis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(6):1607–1620, Nov 2012.

4 J. Angeles. *Dynamic Response of Linear Mechanical Systems - Modeling, Analysis and Simulation*. Springer US, 2012.

5 C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

6 D.P. Bertsekas. Convergence of discretization procedures in dynamic programming. *IEEE Transactions on Automatic Control*, 20(3):415–419, 1975.

7 S. Esmaeil Zadeh Soudjani and A. Abate. Adaptive gridding for abstraction and verification of stochastic hybrid systems. In *QEST*, pages 59–69, 2011.

8 S. Esmaeil Zadeh Soudjani and A. Abate. Adaptive and sequential gridding procedures for the abstraction and verification of stochastic processes. *SIAM Journal on Applied Dynamical Systems*, 12(2):921–956, 2013.

9 S. Esmaeil Zadeh Soudjani, C. Gevaerts, and A. Abate. FAUST^2: Formal abstractions of uncountable-state stochastic processes. In *TACAS*, LNCS 9035, pages 272–286. Springer, 2015.

10 A. Gusrialdi and S. Hirche. Communication topology design for large-scale interconnected systems with time delay. In *American Control Conference*, pages 4508–4513, June 2011.

11 S.K. Jha, E.M. Clarke, C.J. Langmead, A. Legay, A. Platzer, and P. Zuliani. A Bayesian approach to model checking biological systems. In *Computational Methods in Systems Biology*, volume 5688 of *LNCS*, pages 218–234. Springer, 2009.

12 O. Kallenberg. *Foundations of Modern Probability*. Probability and its Applications. Springer Verlag, New York, 2002.

13 J.-P. Katoen, M. Khattri, and I. S. Zapreev. A Markov reward model checker. In *QEST*, pages 243–244. IEEE, 2005.

14 D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

15 L.Y. Kolotilina. Bounds for the singular values of a matrix involving its sparsity pattern. *Journal of Mathematical Sciences*, 137(3):4794–4800, 2006.

16 F.R. Kschischang, B.J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

17 M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *CAV*, volume 6806 of *LNCS*, pages 585–591, 2011.

18 C.J. Langmead. Generalized queries and Bayesian statistical model checking in dynamic Bayesian networks: Application to personalized medicine. In *Proc. 8th International Conference on Computational Systems Bioinformatics*, pages 201–212, 2009.

19 K.P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, Computer Science Division, 2002.

20 S.K. Palaniappan and P.S. Thiagarajan. Dynamic Bayesian networks: A factored model of probabilistic dynamics. In *ATVA*, LNCS, pages 17–25. Springer, 2012.

21 F. Ramponi, D. Chatterjee, S. Summers, and J. Lygeros. On the connections between PCTL and dynamic programming. In *HSCC*, pages 253–262, 2010.

22 V. Sundarapandian. Distributed control schemes for large-scale interconnected discrete-time linear systems. *Mathematical and Computer Modelling*, 41(2–3):313 – 319, 2005.

23 I. Tkachev and A. Abate. Formula-free finite abstractions for linear temporal verification of stochastic hybrid systems. In *HSCC*, pages 283–292, 2013.

24 I. Tkachev, A. Mereacre, J.-P. Katoen, and A. Abate. Quantitative automata-based controller synthesis for non-autonomous stochastic hybrid systems. In *HSCC*, pages 293–302, 2013.

A Proof of Statements

Proof of Theorem 6. The recursive equations for the probabilistic safety problem over sets $A, \bar{A}$ are as follows

$$V_N(\boldsymbol{s}) = \mathbf{1}_A(\boldsymbol{s}), \quad V_k(\boldsymbol{s}) = \int_A V_{k+1}(\bar{\boldsymbol{s}}) t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) d\bar{\boldsymbol{s}},$$

$$W_N(\boldsymbol{s}) = \mathbf{1}_{\bar{A}}(\boldsymbol{s}), \quad W_k(\boldsymbol{s}) = \int_{\bar{A}} W_{k+1}(\bar{\boldsymbol{s}}) t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) d\bar{\boldsymbol{s}},$$

for all $\boldsymbol{s} \in \mathbb{R}^n, k = 0, 1, 2, \dots, N-1$, where the solution of the safety problems are $p_N(\boldsymbol{s}_0, A) = V_0(\boldsymbol{s}_0)$ and $p_N(\boldsymbol{s}_0, \bar{A}) = W_0(\boldsymbol{s}_0)$. We prove inductively that the inequality $|V_k(\boldsymbol{s}) - W_k(\boldsymbol{s})| \leq M(N-k)\mathcal{L}(\bar{A}\Delta A)$ holds for all $\boldsymbol{s} \in A \cap \bar{A}$. This inequality is true for $k = N$. For any $k = 0, 1, 2, \dots, N-1$ and any state $\boldsymbol{s} \in A \cap \bar{A}$ we have

$$\begin{aligned}
|V_k(\boldsymbol{s}) - W_k(\boldsymbol{s})| \leq & \int_A |V_{k+1}(\bar{\boldsymbol{s}}) - W_{k+1}(\bar{\boldsymbol{s}})| t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) d\bar{\boldsymbol{s}} \\
& + \int_{A\backslash\bar{A}} V_{k+1}(\bar{\boldsymbol{s}}) t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) d\bar{\boldsymbol{s}} + \int_{\bar{A}\backslash A} W_{k+1}(\bar{\boldsymbol{s}}) t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) d\bar{\boldsymbol{s}} \\
& \leq M(N-k-1)\mathcal{L}(\bar{A}\Delta A) + M\mathcal{L}(\bar{A}\backslash A) + M\mathcal{L}(A\backslash\bar{A}) \\
& = M(N-k)\mathcal{L}(\bar{A}\Delta A).
\end{aligned}$$

The inequality for $k = 0$ proves upper bound $MN\mathcal{L}(\bar{A}\Delta A)$ on $|p_N(\boldsymbol{s}_0, A) - p_N(\boldsymbol{s}_0, \bar{A})|$. ◀

Proof of Lemma 7. The inequality holds for $k = N$. For $k = 0, 1, 2, \dots, N-1$ and any $\boldsymbol{s}, \boldsymbol{s}' \in \bar{A}$ we have

$$\begin{aligned}
|W_k(\boldsymbol{s}) - W_k(\boldsymbol{s}')| \leq & \int_{\bar{A}} W_{k+1}(\bar{\boldsymbol{s}}) |t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) - t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}')| d\bar{\boldsymbol{s}} \\
& \leq \int_{\bar{A}} |t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}) - t_{\mathfrak{s}}(\bar{\boldsymbol{s}}|\boldsymbol{s}')| d\bar{\boldsymbol{s}} = \int_{\bar{A}} \left| \prod_{i=1}^n t_i(\bar{s}_i|\boldsymbol{s}) - \prod_{i=1}^n t_i(\bar{s}_i|\boldsymbol{s}') \right| d\bar{\boldsymbol{s}} \\
& = \int_{\bar{A}} \left| \sum_{j=1}^n \left[\prod_{i=1}^{j-1} t_i(\bar{s}_i|\boldsymbol{s}') \prod_{i=j}^n t_i(\bar{s}_i|\boldsymbol{s}) - \prod_{i=1}^j t_i(\bar{s}_i|\boldsymbol{s}') \prod_{i=j+1}^n t_i(\bar{s}_i|\boldsymbol{s}) \right] \right| d\bar{\boldsymbol{s}} \\
& \leq \sum_{j=1}^n \int_{\bar{A}} \left[\prod_{i=1}^{j-1} t_i(\bar{s}_i|\boldsymbol{s}') \prod_{i=j+1}^n t_i(\bar{s}_i|\boldsymbol{s}) \, |t_j(\bar{s}_j|\boldsymbol{s}) - t_j(\bar{s}_j|\boldsymbol{s}')| \right] d\bar{\boldsymbol{s}} \\
& \leq \sum_{j=1}^n \int_{D_j} |t_j(\bar{s}_j|\boldsymbol{s}) - t_j(\bar{s}_j|\boldsymbol{s}')| \, d\bar{s}_j \\
& \leq \sum_{j=1}^n \mathcal{H}_j \|\boldsymbol{s} - \boldsymbol{s}'\| \mathcal{L}(D_j) = \|\boldsymbol{s} - \boldsymbol{s}'\| \sum_{j=1}^n \mathcal{H}_j \mathcal{L}(D_j) = \|\boldsymbol{s} - \boldsymbol{s}'\| \sum_{j=1}^n \mathcal{I}_j.
\end{aligned}$$

◀

▶ **Lemma 10.** *The entry-wise one-norm and two-norm of square matrices are equivalent:*

$$n\|\Phi\|_2 \leq \|\Phi\|_1 \leq n\sqrt{n}\|\Phi\|_2,$$

where n is the dimension of the matrix $\Phi = [a_{ij}]_{i,j} \in \mathbb{R}^{n\times n}$.

Proof of Lemma 10. Define $r_i(\Phi) = \sum_{j=1}^n |a_{ij}|$ and $c_j(\Phi) = \sum_{i=1}^n |a_{ij}|$. The Cauchy-Schwartz inequality implies that

$$c_j(\Phi) \leq \sqrt{n}\sqrt{\sum_{i=1}^n |a_{ij}|^2} = \sqrt{n}\|\Phi \boldsymbol{e}_1\|_2 \leq \sqrt{n}\|\Phi\|_2$$
$$\Rightarrow \|\Phi\|_1 = \sum_{j=1}^n c_j(\Phi) \leq \sum_{j=1}^n \sqrt{n}\|\Phi\|_2 = n\sqrt{n}\|\Phi\|_2,$$

where $\boldsymbol{e}_1 = [1,0,0,\ldots,0]^T$. On the other hand for any $\boldsymbol{s} = [s_1, s_2, \ldots, s_n]^T$ with $\|\boldsymbol{s}\|_2 = 1$,

$$\begin{aligned}\|\Phi \boldsymbol{s}\|_2 &= \left[\sum_{i=1}^n |a_{i1}s_1 + a_{i2}s_2 + \ldots + a_{in}s_n|^2\right]^{1/2} \leq \left[\sum_{i=1}^n \left(|a_{i1}|^2 + |a_{i2}|^2 + \ldots + |a_{in}|^2\right)\right]^{1/2}\\ &= \left[\sum_{i,j=1}^n |a_{ij}|^2\right]^{1/2} \leq \frac{n}{n^2}\sum_{i,j=1}^n |a_{ij}| = \frac{1}{n}\|\Phi\|_1.\end{aligned}$$

◀

As you see here the ratio $\|\Phi\|_1/\|\Phi\|_2$ is bounded from below by the dimension of the matrix and also from above by the $n\sqrt{n}$.

▶ **Lemma 11** ([15]). *The maximum singular value of a matrix can be bounded based on its sparsity pattern. In particular for any* Φ*,*

$$\|\Phi\|_2 \leq \max_{i,j:a_{ij}\neq 0} [r_i(\Phi)c_j(\Phi)]^{1/2}.$$