Why are DBNs sparse?

Shaunak Chatterjee
Computer Science Division
University of California
Berkeley, CA 94720
shaunakc@cs.berkeley.edu

Stuart Russell
Computer Science Division
University of California
Berkeley, CA 94720
russell@cs.berkeley.edu

Abstract

Real stochastic processes operating in continuous time can be modeled by sets of stochastic differential equations. On the other hand, several popular model families, including hidden Markov models and dynamic Bayesian networks (DBNs), use discrete time steps. This paper explores methods for converting DBNs with infinitesimal time steps into DBNs with finite time steps, to enable efficient simulation and filtering over long periods. An exact conversion—summing out all intervening time slices between two steps—results in a completely connected DBN, yet nearly all human-constructed DBNs are sparse. We show how this sparsity arises from well-founded approximations resulting from differences among the natural time scales of the variables in the DBN. We define an automated procedure for constructing an approximate DBN model for any desired time step and prove error bounds for the approximation. We illustrate the method by generating a series of approximations to a simple pH model for the human body, demonstrating speedups of several orders of magnitude compared to the original model.

1 Introduction

Since their introduction by Dean and Kanazawa (1989), dynamic Bayesian networks (DBNs) have proved to be a flexible and effective tool for representing and reasoning about stochastic systems that evolve over time. DBNs include as special cases hidden Markov models (HMMs) (Baum and Petrie, 1966), factorial HMMs (Ghahramani and Jordan, 1997), hierarchical HMMs (Fine *et al.*, 1998), discrete-time Kalman filters (Kalman, 1960), and several other families of discrete-time models.

As explained in detail in Section 2, a DBN represents the state of a system by the values of a set of variables in a *time slice*, with connections between slices representing the stochastic evolution of the system. Of particular importance is the fact that DBNs are often *sparse*—each variable in a given slice includes among its parents only a small subset of variables from the preceding slice. Thus, a DBN may require exponentially fewer parameters than an equivalent HMM.

Although there have been some attempts at DBN structure learning (Friedman *et al.*, 1998), for the most part DBNs are built by hand. As with ordinary (non-temporal) Bayesian networks, this is a somewhat opaque process fraught with errors; but for DBNs, there is the additional issue of *choosing the size of the time step* Δ that separates the time slices. As we will see, the choice of Δ has a dramatic effect on both the computational cost of the model and the proper topology of the DBN. Folk wisdom in the field—borrowed perhaps from standard practice in simulation of differential equations—suggests that Δ needs to be small enough so that the fastest-changing variable in the model has only a small probability of changing its state in time Δ. Unfortunately, in many systems this results in gross inefficiency. For example, the body's pH setpoint changes on a timescale of days or weeks, while breathing rate (which affects pH) changes on a timescale of seconds; hence, a system that models both is forced to perform inference over millions of time steps in order to track the pH setpoint over an extended period. This issue motivated the development of continuous-time Bayes nets (CTBNs) (Nodelman *et al.*, 2002), which avoid committing to any fixed time step. Another approach, appropriate for regular but widely separated observations and for certain restricted classes of models, is to convert a natural small-Δ model into an equivalent model whose Δ matches

Appearing in Proceedings of the 13^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, Chia Laguna Resort, Sardinia, Italy. Volume 9 of JMLR: W&CP 9.

the observation frequency (Aleks *et al.*, 2009).

The approach we take in this paper is to think about how one might convert a continuous-time model—a CTBN or a set of stochastic differential equations (SDEs)—into an equivalent, or approximately equivalent, discrete-time DBN for a given Δ. This provides some insight into why DBNs have the structures that they do, and also yields an automatic procedure for choosing time steps and DBN structures, such that simulation over long periods can be both efficient and provably (approximately) accurate.

Let us assume that the system can be modeled exactly by a set of n coupled SDEs that is sparse; we can think of this model as a sparse DBN with a very small time step δ. Now, if we increase the time step to, say, $n\delta$ by summing out $n-1$ intervening steps in the model, the resulting model will be completely connected (unless the original model has disjoint components). This presents a puzzle, since most human-designed DBNs are sparse even with very large Δ. Such models must implicitly be making approximations. In this paper, we will show how these approximations are a natural outcome when the variables have widely different timescales (rates of evolution).

In a deterministic dynamic model, the idea of using a wide separation of timescales to simplify the model goes back at least to work by Michaelis and Menten (1913); see Iwasaki and Simon (1994), Gómez-Uribe *et al.* (2008) for more recent surveys. The general analysis involves finding gaps in the eigenspectrum of the coefficient matrix of the system of differential equations. Here, we provide the simplest possible example: a system of two variables, s and f, where s (the "slow" variable) influences f (the "fast" variable) but not vice versa:

$$\frac{ds}{dt} = a_1 s \; ; \qquad \frac{df}{dt} = b_1 s + b_2 f \tag{1}$$

where we assume $|a_1| \ll |b_2|$ and both negative. Viewed as a (deterministic) DBN, this looks like Figure 1(a). The exact solution for some time t is

$$s = S_0 e^{a_1 t} \; ; f = \left(\frac{b_1 S_0}{a_1 - b_2}\right) e^{a_1 t} + \left(F_0 - \frac{b_1 S_0}{a_1 - b_2}\right) e^{b_2 t} \tag{2}$$

where S_0 and F_0 are initial values for s and f. This is represented by the DBN structure shown in Figure 1(b) for a large finite time step Δ. Although f is nominally a "fast" variable, the solution shows that, for $t \gg 1/|b_2|$, f follows a slowly changing equilibrium value that depends on s. Thus, we need model only the dynamics of s and can compute $f(t)$ directly from $s(t)$. This corresponds to the DBN structure in Figure 1(c). With this structure, we can use a large Δ because s changes very slowly.

Effective *model reduction* methods have been developed in the dynamical systems literature for deriving such simplified deterministic models in a semi-automated fashion. Moreover, any discrete-state DBN model can be converted into an equivalent deterministic dynamical system whose variables are the occupancy probabilities of individual states, and model-reduction techniques can be applied to this system. Unfortunately, this approach involves an *exponential blowup* in the model size; furthermore, even if it can be computed, the reduced version would not necessarily correspond to a meaningful sparse model in terms of the original variables.

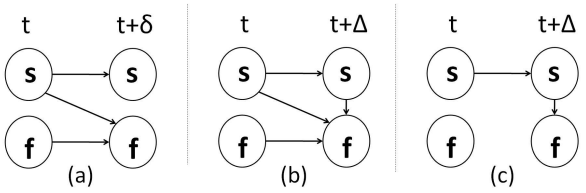

Figure 1: Two variable DBN: The slow variable s is independent of the fast variable f. (a) Exact model for small time-step δ. (b) Exact model for large time-step Δ. (c) Approximate model for large time-step Δ.

Instead, we work directly with the DBN, beginning with a very-small-time-step model, identifying time steps Δ that nicely separate the model's time scales, and deriving the corresponding reduced DBN for each such Δ. A salient feature of the algorithm is that it avoids building the intractably large full transition matrix. For large Δ, accurate simulation over very long time periods becomes possible; moreover, the per-time-step inference cost for the reduced models can be much less than for the original models, since the models become sparser as Δ becomes larger. The larger time-step combined with the simpler model result in speed-ups of several orders of magnitude compared to the original model.

Sections 2 presents DBNs and other relevant definitions. Section 3 introduces an example DBN that models the body's pH control system. Section 4 presents the approximation scheme, a proof of its correctness and an analysis of the associated error. Section 5 extends these ideas to obtain a general set of rules to construct an approximate DBN for a large time-step. Section 6 presents results on the accuracy and computational cost of the approximate DBNs of the pH control mechanism.

2 Definitions

A dynamic Bayesian network (DBN) (Dean and Kanazawa, 1989) is a discrete-time model of a stochastic dynamical system. The system's state is represented by a set of variables, $\mathbf{X}_t$ for each time $t \in \mathbb{Z}^*$ and the DBN represents the joint distribution over the variables $\bigcup_{t=0}^{\infty} \mathbf{X}_t$. Typically we assume that the

system's dynamics do not change over time, so the joint distribution is captured by a 2-TBN (2-Timeslice Bayesian Network), which is a compact graphical representation of the state prior $P(\mathbf{X}_0)$ and the stochastic dynamics $P(\mathbf{X}_{t+1}|\mathbf{X}_t)$. In turn, the dynamics are represented in factored form via a collection of local conditional models $P(X^i_{t+1}|\pi(X^i_{t+1}))$, where $\pi(X^i_{t+1})$ are the parent variables of X^i_{t+1} in slice t or $t+1$. Henceforth, we will consider all X^i_t to be discrete.

Consider a simple one-variable DBN, where the variable (say X) can be in one of k discrete states. Let $p_{i,j} = Pr(X_{t+1}=j|X_t=i)$. We define the *timescale of X for state i*, T^i_X, as the expected number of time-steps that the variable spends in state i before changing state, given that it is currently in state i. It can be shown that $T^i_X = 1/(1-p_{i,i})$. Thus, the overall timescale of variable X is actually a range of timescales from $\min_i T^i_X$ to $\max_i T^i_X$.

In a general DBN, let $\hat{\pi}(X_{t+1})$ denote the parents of variable X_{t+1} in the 2-TBN representation *other than* X_t. Then the conditional probability table (CPT) for X_{t+1} is given by $p^k_{i,j} = Pr(X_{t+1}=j|X_t=i, \hat{\pi}(X_{t+1})_t=k)$. We also define $T^{i,k}_X = 1/(1-p^k_{i,i})$ to be the timescale of variable X in state i when its parents are in state k, where k is any state in the joint state space of $\hat{\pi}(X_{t+1})$.

Now, consider two variables X_1 and X_2 in a general DBN. Let $l_{X_1} = \min_{i,k} T^{i,k}_{X_1}$ and $h_{X_2} = \max_{i,k} T^{i,k}_{X_2}$. If $l_{X_1} \gg h_{X_2}$, then X_1 and X_2 are said to be *slow* and *fast* variables (respectively) with respect to each other. Their *timescale separation* is defined as the ratio l_{X_1}/h_{X_2}. For a set of variables $C=\{X_1,\ldots,X_n\}$, the lower timescale bound is defined as $l_C = \min_{X_i \in C} l_{X_i}$, with h_C also defined in a similar fashion. The existence of significant timescale separation in a DBN is crucial in allowing accuracy-preserving model simplifications.

A continuous-time analog of the DBN is the continuous-time Markov chain. Formally, it is defined as a Markov stochastic process $\{\mathbf{x}_t\}_{t\in\mathbb{R}^+}$ with state space $\mathbb{I}$. Let $\mathbb{I}=\{1,2,\ldots,k\}$. The transition matrix for the interval from 0 to t, $P_{ij}(t) = Pr(\mathbf{X}_t = j|\mathbf{X}_0 = i), (i,j) \in \mathbb{I}\times\mathbb{I}$, is given by $P(t) = e^{Lt}$, where the matrix L is called the *generator* of the Markov chain. L has the following properties: (i) $\sum_j l_{ij} = 0, \forall i \in \mathbb{I}$ (this makes L *conservative*); (ii) $l_{ij} \in [0,\infty), \forall (i,j) \in \mathbb{I}\times\mathbb{I}$ with $i \neq j$. L can be computed by:

$$L = \lim_{t\to 0} \frac{P(t) - I}{t} \tag{3}$$

We will assume that the transition matrix $P(t)$ is *standard* (i.e., $\lim_{t\to 0} P_{ii}(t) = 1, \forall i \in \mathbb{I}$).

3 A Motivating Example: Human pH Regulation System

pH is a measure of the concentration of hydrogen ions in a solution or substance. Measured on a log scale of 0–14, higher numbers represent alkaline nature and lower numbers are characteristic of acids. The pH balance of the human bloodstream is one of the most important biochemical balances in the human body since it controls the speed of our body's biochemical reactions (Guyton and Hall, 1997).

The human body has a complex system to maintain body pH around a setpoint ($\simeq 7.4$) under normal circumstances. Generally, metabolism leads to CO_2 production, thereby producing carbonic acid, which lowers the pH of blood (an abnormal lowering leads to "acidosis"). On the other hand, respiration brings O_2 into the system and also removes CO_2, which neutralizes the acid in the blood, thus raising the pH. These are the two main compensatory mechanisms that we will consider in our model. Chemical acid–base buffer systems of the body fluids (which provide the first line of defense against fluctuations in blood pH) are not modeled. Metabolism rate increases with increasing temperature and higher levels of exertion. The respiratory rate (measured as "Minute Volume", which is the volume of air inhaled in a minute) is raised by a lower pH value, if the pH setpoint is normal. However, an overdose of certain narcotics might lower the pH setpoint of the body. In this case, a low pH will not trigger a rise in the Minute Volume, causing the blood to become more and more acidic and possibly resulting in death. Figure 2 shows the DBN model of the system which controls the body pH. Variables with similar shades have comparable timescales. The darkest shaded variable (i.e., "Minute Volume") has the fastest dynamics, while the lightest shaded variable ("pH setpoint") has the slowest dynamics. Details of each variable in the model are provided in Table 1.

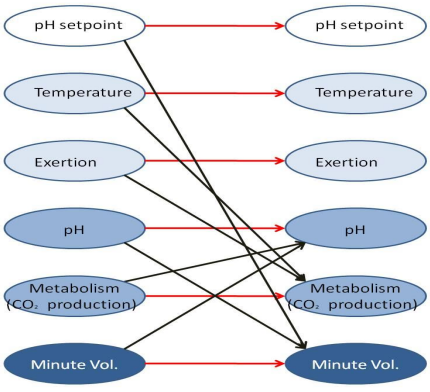

Figure 2: Exact model for the pH control system for a small time-step δ.

We chose this model as a motivating example, since there are interacting variables in this system which evolve at very different timescales. We will construct

Table 1: Information about the variables in the DBN (including their state space and timescales)

Details of the pH control system DBN		
Variable Name	State Space	Timescale (Seconds)
pH setpoint (pHst)	{ Normal, Low }	$l_{pHst} = 3.3e6$ $h_{pHst} = 1e7$
Temperature (Temp)	{ Hot, Warm, Cool, Cold }	$l_{Temp} = 8e3$ $h_{Temp} = 1e4$
Exertion (Ex)	{ High, Normal, Low}	$l_{Ex} = 5e3$ $h_{Ex} = 1e4$
pH (pH)	{ Acid, Neutral, Alkaline }	$l_{pH} = 100$ $h_{pH} = 300$
Metabolism (Meta)	{ High, Normal, Low }	$l_{Meta} = 70$ $h_{Meta} = 150$
Minute Volume (MV)	{ High, Normal, Low }	$l_{MV} = 1.1$ $h_{MV} = 5$

approximate, sparsely connected models for this system over large time-steps in Section 6.

4 Approximation scheme

Let us consider the general 2-variable DBN in Figure 3(a). (Although one might expect a link between $s_{t+\delta}$ and $f_{t+\delta}$, we can reduce δ appropriately to drop the intra-time-slice links since *any differential equation system can be represented without contemporaneous edges.*) For simplicity of presentation, we will assume that s and f are binary random variables (although all the results presented in this section are generally applicable to any finite discrete state space for the two variables). Since we assume there is a timescale separation in the dynamics of s and f, their CPTs should have the following structure:

$$p(s_{t+1}|s_t, f_t) = \begin{bmatrix} 1-\epsilon x_1 & \epsilon x_1 \\ 1-\epsilon x_2 & \epsilon x_2 \\ \epsilon x_3 & 1-\epsilon x_3 \\ \epsilon x_4 & 1-\epsilon x_4 \end{bmatrix} \quad (4)$$

$$p(f_{t+1}|s_t, f_t) = \begin{bmatrix} 1-a_1 & a_1 \\ a_2 & 1-a_2 \\ 1-a_3 & a_3 \\ a_4 & 1-a_4 \end{bmatrix} \quad (5)$$

where $\epsilon \ll 1$, $0 < a_i, x_i < 1$ and $a_i, x_i \gg \epsilon$. The rows correspond to $(s_t, f_t) = \{(0,0),(0,1),(1,0),(1,1)\}$ respectively. The first column corresponds to s_{t+1} (or f_{t+1}) $= 0$. Using the definitions in Section 2, $l_s = \min_i 1/\epsilon x_i$ and $h_f = \max_i 1/a_i$. Therefore $l_s/h_f = O(1/\epsilon)$. It should be noted that ϵ is a redundant parameter in this specification—hence we have an option to choose an ϵ. This choice has to be made such that the order of the x_i's and a_i's are similar and they also satisfy the previous constraints.

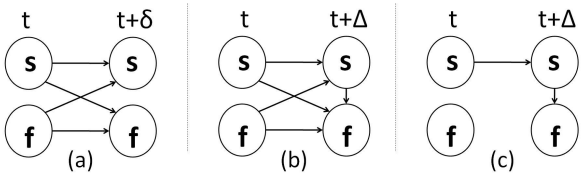

Figure 3: Two variable general DBN: The slow variable s is also dependent on the fast variable f. (a) Exact model for small time-step δ. (b) Exact model for large time-step Δ. (c) Approximate model for large time-step Δ.

The exact transition model for a larger time-step Δ is shown in Figure 3(b). Without loss of generality, let us assume $\delta = 1$. As shown in Figure 3(c), for the large time-step Δ, the distribution of $f_{t+\Delta}$ becomes (approximately) independent of the value of s_t and f_t. The key observation is that irrespective of the value of f_t, the distribution of f_{t+i}, for i $\in [1, \Delta]$ will exponentially converge to the equilibrium distribution of f for the current value of s. Once it does so (approximately), we can compute an exact expression for $\hat{P}(s_{t+(N+1)}|s_{t+N})$, where N is large enough for f to approximately reach its equilibrium distribution. This expression will be equal to

$$\hat{P}(s_{t+1}|s_t) = \sum_{f_t} p(s_{t+1}|s_t, f_t) \times P_\infty(f_t|s_t) \quad (6)$$

where $P_\infty(f_t|s_t)$ is the equilibrium distribution of f. Since f (nearly) reaches its equilibrium in a short fraction of Δ, we ignore that portion of f's trajectory, and simply assign $(\hat{P}(s_{t+(N+1)}|s_{t+N}))^\Delta$ to be the CPT of s for the large time-step Δ. Equation 6 is analogous to Forward Euler integration since we use s_t only to determine the equilibrium distribution of f.

The CPT $\hat{P}(f_{t+\Delta}|s_{t+\Delta})$ is simply the invariant distribution of f for the fixed value of $s_{t+\Delta}$.

4.1 Correctness of the approximation scheme

The above approximation heuristic is analogous to elimination of the fast variable through averaging for the continuous-time Markov chain. (For a complete treatment of the continuous-time case, see Pavliotis and Stuart (2007).) In particular, let the state spaces of s and f be $\mathbb{I}_s$ and $\mathbb{I}_f$ respectively. Let $q((i,k),(j,l))$ denote the element of the generator matrix associated with transition from $(i,k) \in \mathbb{I}_s \times \mathbb{I}_f$ to $(j,l) \in \mathbb{I}_s \times \mathbb{I}_f$. $B_0(i)$ is a generator with entries $b_0(k,l;i)$ where the indices indicate transition from $k \in \mathbb{I}_f$ to $l \in \mathbb{I}_f$ for given fixed $i \in \mathbb{I}_s$. For each $i \in \mathbb{I}_s$, $B_0(i)$ generates an ergodic Markov chain on $\mathbb{I}_f$. Let $\rho^B_\infty(k;i)_{k\in\mathbb{I}_f}$ be the invariant distribution of a Markov chain on $\mathbb{I}_f$, indexed by $\mathbb{I}_s$. Similarly, let $B_1(k)$ with indices $b_1(i,j;k)$ denote transition from $i \in \mathbb{I}_s$ to $j \in \mathbb{I}_s$, for each fixed $k \in \mathbb{I}_f$. Let us introduce generators Q_0 and Q_1 of

Markov chains on $\mathbb{I}_s \times \mathbb{I}_f$ by:

$$q_0((i,k),(j,l)) = b_0(k,l;i)\delta_{ij},$$
$$q_1((i,k),(j,l)) = b_1(i,j;k)\delta_{kl},$$

where δ_{ij} and δ_{kl} are Kronecker delta functions. Now, let us define another generator $\bar{Q}$ of a Markov chain on $\mathbb{I}_s$ by $\bar{q}(i,j) = \sum_k \rho^B_\infty(k;i) b_1(i,j;k)$.

Lemma 1

If Q, the generator of a Markov chain, takes the form $Q = \frac{1}{\epsilon}Q_0 + Q_1$, then for $\epsilon \ll 1$ and times t up to $O(1)$, the finite-dimensional distribution of $s \in \mathbb{I}_s$ is approximated by a Markov chain with generator $\bar{Q}$ with an error of $O(\epsilon)$.

The proof (Pavliotis and Stuart, 2007, Section 9.4) bounds the error on a vector $v_i(t) = \mathbb{E}(\phi_{x(t)}|x(0) = i)$, where $\phi : \mathbb{I} \mapsto \mathbb{R}$. $v(t)$ satisfies the backward Kolmogorov equation (i.e., $dv/dt = Lv$; $v(0) = \phi$). Thus $v(t) = P(t)\phi$, where $P(t)$ is the transition matrix for the interval from 0 to t. Since this is true for any mapping ϕ, the approximation error in any element of $P(t)$ is necessarily bounded by $O(\epsilon)$ too.

We now show how our approximation scheme for the discrete time-step is equivalent to their solution for continuous time and thereby shares the same order of approximation error. The first step towards proving equivalence is to show that the generator matrix corresponding to the discrete-time process also has a similar structure (i.e., $Q = \frac{1}{\epsilon}Q_0 + Q_1$). Firstly, for $\delta \ll 1$ and an integer $n > 0$,

$$\begin{bmatrix} 1-\delta x & \delta x \\ \delta y & 1-\delta y \end{bmatrix}^n \approx \begin{bmatrix} 1-n\delta x & n\delta x \\ n\delta y & 1-n\delta y \end{bmatrix} \quad (7)$$

when we ignore δ^2 terms. If we consider $1/\delta$ to be a large integer, Equation 7 implies

$$\begin{bmatrix} 1-x & x \\ y & 1-y \end{bmatrix}^\delta \approx \begin{bmatrix} 1-\delta x & \delta x \\ \delta y & 1-\delta y \end{bmatrix} \quad (8)$$

Let P^δ_f (similarly P^δ_s) be the CPT for the dynamics of f (s) over an infinitesimal time-step δ when we freeze s (f). Using Equation 8, we get:

$$P^\delta_s \approx \begin{bmatrix} 1-\delta\epsilon x_1 & \delta\epsilon x_1 \\ 1-\delta\epsilon x_2 & \delta\epsilon x_2 \\ \delta\epsilon x_3 & 1-\delta\epsilon x_3 \\ \delta\epsilon x_4 & 1-\delta\epsilon x_4 \end{bmatrix}; P^\delta_f \approx \begin{bmatrix} 1-\delta a_1 & \delta a_1 \\ \delta a_2 & 1-\delta a_2 \\ 1-\delta a_3 & \delta a_3 \\ \delta a_4 & 1-\delta a_4 \end{bmatrix}$$

We now combine P^δ_s and P^δ_f to form $P^\delta_{s,f}$.

$$P^\delta_{s,f} \approx \begin{bmatrix} (1-\delta\epsilon x_1)(1-\delta a_1) & \cdots & \cdots & \delta\epsilon x_1 \delta a_1 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \delta\epsilon x_4 \delta a_4 & \cdots & \cdots & (1-\delta\epsilon x_4)(1-\delta a_4) \end{bmatrix}$$

The generator corresponding to this discrete-time process can be computed by the formula $L = \lim_{\delta\to 0}(P^\delta_{s,f} - I)/\delta$. Since we divide by δ when taking the limit, ignoring the higher order terms of δ in the previous step(s) becomes inconsequential. The generator matrix L thus obtained is:

$$L = \begin{bmatrix} -a_1 & a_1 & 0 & 0 \\ a_2 & -a_2 & 0 & 0 \\ 0 & 0 & -a_3 & a_3 \\ 0 & 0 & a_4 & -a_4 \end{bmatrix} + \epsilon \begin{bmatrix} -x_1 & 0 & x_1 & 0 \\ 0 & -x_2 & 0 & x_2 \\ x_3 & 0 & -x_3 & 0 \\ 0 & x_4 & 0 & -x_4 \end{bmatrix}$$

Hence, the generator has the form $L_0 + \epsilon L_1$. Thus, $L = O(\epsilon Q)$. Since $P(t) = e^{Lt}$, the behavior of L at time T/ϵ is similar to the behavior of Q at time T. Thus we can use Lemma 4.1 to say the following:

If L, the generator of a Markov chain, takes the form $L = L_0 + \epsilon L_1$, then for $\epsilon \ll 1$ and times t up to $O(1/\epsilon)$, the finite-dimensional distribution of $s \in I_s$ is approximated by a Markov chain with generator $\bar{L}$ with an error of $O(\epsilon)$, where $\bar{L}(i,j) = \sum_k \rho^L_\infty(k;i) l_1(i,j;k)$.

It is easy to see that the generator corresponding to the transition matrix $\hat{P}(s_{t+1}|s_t)$ (Equation 6) is exactly equal to the generator $\bar{L}$, and hence we arrive at the following result.

Result 1

If a discrete time Markov process has conditional probability tables given by Equations 4 and 5, then for $\epsilon \ll 1$ and times Δ up to $O(1/\epsilon)$, the finite-dimensional distribution of $s \in \mathbb{I}_s$ is approximated by $\hat{P}(s_{t+1}|s_t)^\Delta$ (given by Equation 6), with error $O(\epsilon)$.

For very small values of Δ (like $O(1)$), the error for f decays exponentially ($O(|\lambda|^\Delta)$) where λ is the maximum singular value of $p(f_{t+1}|s_t, f_t)$. However, for $\Delta = O(1/\epsilon)$, the error for f essentially replicates the error for s, since the fast ergodic dynamics of f has almost reached a quasi-static equilibrium.

4.2 Special case

In the exact model, if $p(s_{t+1}|s_t, f_t) = p(s_{t+1}|s_t)$ (i.e., the dynamics of s is independent of f as shown in Figure 1), then the dynamics of s is tracked exactly by the above scheme. In Equation 6, the $p(s_{t+1}|s_t, f_t)$ term goes outside the summation, and the invariant distribution of f sums to 1.

4.3 Other approaches

There is another line of work (Yin and Zhang, 2004) where discrete time transition models of the form $P_\epsilon = P + \epsilon Q$ are considered. Here, P is a stochastic matrix and Q is a generator matrix. The approximation error for P^k_ϵ can be made $O(\epsilon^{n+1})$ by constructing a series of approximation functions. While this approach has the benefit of a potentially much smaller error, the functions are much more expensive to compute and the

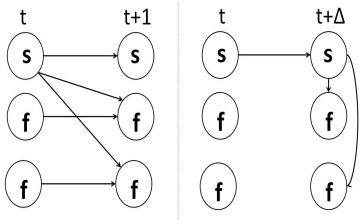

Figure 4: Structural transformation in the large time-step model when f_1 and f_2 have **no** cross links in the small time-step model

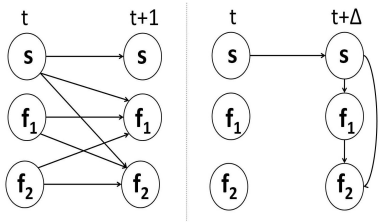

Figure 5: Structural transformation in the large time-step model when f_1 and f_2 have cross links in the small time-step model

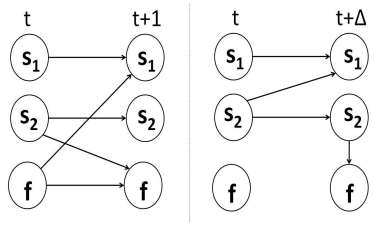

Figure 6: A slow cluster s_1 has a new parent s_2 in the larger time-step model when s_2 is a parent of f in the smaller time-step model

approximate model has no simple or intuitive mapping to the original model.

5 General Rules of Construction

This section describes the general algorithm for constructing an approximate DBN for a larger time-step when given the exact DBN for a small time-step δ. We will use the approximate CPT construction discussed in section 4. Let the DBN have n variables X_1, X_2, ...,X_n. This algorithm will produce a sequence of approximate DBNs for various increasing values of Δ (the larger time-step). The algorithm is as follows:

1. For each variable X_i, determine l_{X_i} and h_{X_i}.
2. Cluster the variables into $\{C_1, C_2, \ldots, C_m\}$ ($m \leq n$), such that $\epsilon_i \approx \frac{h_{C_i}}{l_{C_{i+1}}} \ll 1$, $\forall i \in \{1, 2, \ldots, m-1\}$, i.e., there is a significant timescale separation between successive clusters. C_1 is the cluster of fastest variables, while C_m is the cluster of slowest variables. In the worst case, m can be 1, when all variables have very comparable timescales.
3. Repeat the following steps for $i = \{1, \ldots, m-1\}$. Let $\Delta_0 = 1$.
   (a) Select $\Delta_i = \Delta_{i-1} \times O(1/\epsilon_i)$
   (b) For each configuration of the slower parents of C_i, compute the stationary point (equilibrium distribution) of C_i to fill in the CPT of $p((C_i)_{t+\Delta_i} | \pi(C_i)_{t+\Delta_i})$ in the approximate model for time-step Δ_i. If the fast variables in C_i are conditionally independent given the slow parents C_j ($j > i$), then the individual equilibrium are calculated (see Figure 4). However, if the variables in C_i are not conditionally independent given C_j ($j > i$), we have to compute the joint equilibrium of C_i (as in Figure 5).
   (c) While C_i only has parents in the same time-slice in the approximate model, C_j ($j > i$) will have parents from the previous time-slice. If there were no links to C_j from C_i in the exact model, then the CPT of C_j is exactly equal to $C\hat{P}T_{C_j}^{(\Delta_i/\Delta_{i-1})}$ (as mentioned in section 4.2), where $C\hat{P}T_{C_j}$ is the joint CPT of C_j and its parents for time-step Δ_{i-1}. In the worst case, all C_j's ($j > i$) can become fully connected.
   However, if there are links from the fast cluster C_i to the slow cluster C_j, then we have to use Equation 6 to compute the CPT of C_j for time-step Δ_i. *Since the equilibrium distribution of C_i is parameterized by the parents of C_i, these variables now become additional parents of C_j* (see Figure 6).
   (d) Now we have an approximate model for time-step Δ_i. This model only has links across time-slices for C_j ($j > i$). This approximate model is used as the exact model for the next iteration (since using the exact model would result in the same approximations).

The sequence of DBNs produced by this algorithm become more and more sparse. This makes exact inference on these approximate models much more feasible than the fully connected exact model for the same time-step. Let D_s be the number of variables in the slow clusters and D_f be the number of variables in the fast clusters and let all variables be binary. Then, the complexity of exact inference (per time step) in the fully connected, exact model is $O(2^{2\times(D_s+D_f)})$ while that in the approximate model is $O(2^{2\times D_s} + 2^{D_s+D_f})$. This complexity is for the projection of the joint state space distribution vector. Also, particle filters will run much faster on these approximate models because particles are only needed to estimate the joint state space of D_s and not $D_s \cup D_f$ (as is the case in the exact model). In the next section, we return to the pH regulation model from Section 3 and create approximate models for appropriate values of Δ.

6 Experiment

As mentioned previously, the pH regulation model exhibits a wide range of timescales. Minute Volume can

potentially change every second, depending on the current needs of the body. The pH of the body and the rate of metabolism have much slower dynamics relative to Minute Volume. Temperature and Exertion are even slower, while pH setpoint (which only changes upon a heavy overdose of narcotics, a very rare event) is the slowest of all. (Timescales are specified in Table 1.) Thus, there are four separate clusters of variables on which we can apply the algorithm of Section 5.

The three approximate models created for $\Delta = 20$, $\Delta = 1000$ and $\Delta = 50000$ are shown in Figure 7. For the first approximate model M_{20}, only Minute Volume is the fast variable. Hence, its parents are pH and pH-setpoint from the same time-slice. Also, since pH-setpoint determines the equilibrium distribution of Minute Volume, which in turn is a parent of pH (in the exact model), pH now has an additional parent, pH-setpoint (according to step 3.(c) in Section 5). For the second approximate model M_{1000}, pH and Metabolism are the new fast variables. Since Metabolism is a parent of pH, we have to consider the joint equilibrium of the two variables, given each configuration of their slow parents (i.e., pH-setpoint, Temperature and Exertion). For the third approximate model M_{50000}, Temperature and Exertion also become fast variables. Since these were independently evolving variables, they do not have any parents in M_{50000}.

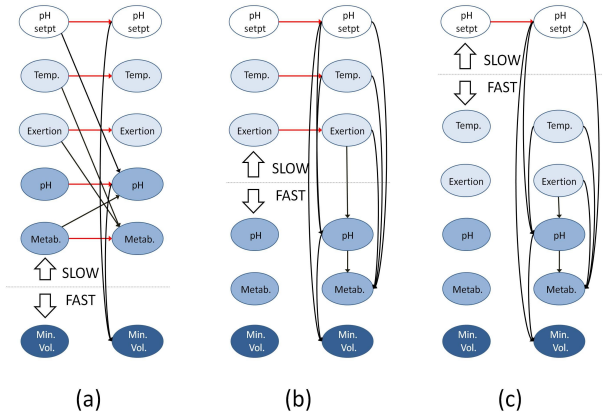

Figure 7: Approximate models of the pH regulation system of the human body. (a) Approximate model for $\Delta = 20$. (b) Approximate model for $\Delta = 1000$. (c) Approximate model for $\Delta = 50000$

For evaluation purposes, we implemented the exact model and the three approximate models in MATLAB. We chose 10 random starting configurations of the variables and simulated the exact trajectory of the belief vector for each of these initial configurations for 1,000,000 time-steps. Then we used the same starting configurations and M_{20} to simulate the trajectories at regular intervals of 20 steps over 1 million time-steps. We repeated the same procedure with M_{1000} and M_{50000} to simulate the trajectories at intervals of 1000 and 50000 steps respectively. The L2-error (per time step) of the joint state space belief vector was averaged over all 10 instances and plotted in Figure 8. As expected, the error increases with the level of approximation—although all three models perform well.

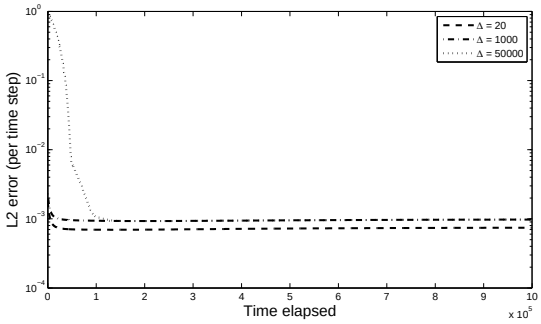

Figure 8: Comparison of the average L2-error(per time-step) of the belief vector of the joint state space for M_{20}, M_{1000} and M_{50000}.

Table 2: Computational speed-up in different models

Model	Avg. Simulation Time (sec)	Speedup Factor
Exact	385.44	1
$\Delta = 20$	24.87	15.5
$\Delta = 1000$	.0889	4300
$\Delta = 50000$	.0006327	> 600000

The performance speed-up details of the different models are summarized in Table 2. The speed-up factor for M_{20} was less than 20 because the exact model required only matrix multiplication (very efficient in MATLAB), while M_{20} needed some indexing work to compute the distribution of the fast variable Minute Volume even though its matrix multiplication requirements were less. The benefit of a much simpler (sparser) model was evident for both M_{1000} and M_{50000}, as the speed-up factor exceeded the size of the time-step (Δ).

Since this is a model for pH regulation, we also decided to check the performance of the approximate models on the marginal distribution of pH. Since pH is a slow variable in M_{20}, it is simulated exactly in that model and hence is not relevant to this experiment. As we can see from Figure 9, both M_{1000} and M_{50000} perform very well with an error of less than 0.04%.

7 Conclusion

We have shown how DBNs that are naturally sparse for a small time step may be converted to (different)

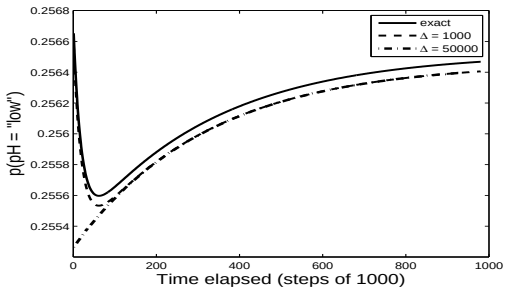

Figure 9: Accuracy of M_{1000} and M_{50000} in tracking the marginal distribution of pH

sparse DBNs for large time steps, even though an exact conversion methods would yield a fully connected model. The sparse approximation becomes more and more accurate with increasing separation of timescales among variables. Our error analysis also supports a quantitative trade-off between accuracy and speed-up. The methods accommodate models with widely varying timescales and/or intermittent observations and should be applicable to a broad range of chemical, biological, and social systems with these properties.

Aleks *et al.* (2009) noted that specifying a DBN may be quite easy for a small time-step but much harder for a larger time-step. This construction automates the conversion. Thus, it allows the user to build a DBN at a natural time-step, yet run it at much larger time-steps to reduce computational cost.

Further work along these lines includes extending the results to handle continuous variables; adding the possibility of replacing a state variable by another variable corresponding to its long-term average value (e.g., replacing instantaneous blood pressure by its one-minute average); and adding the possibility of replacing a set of variables by functions of those variables (e.g., replacing two Boolean variables by their XOR, or two continuous variables by linear combinations thereof). These two latter ideas both create additional scope for clean separation of timescales.

Acknowledgements

We would like to acknowledge Intel Corporation and the UC Discovery Program for support of the project "Center for Biomedical Informatics in Critical Care" and NSF for support of the project "Hierarchical Decision Making for Physical Agents". We would also like to thank Jaijeet Roychowdhury, Emma Brunskill, Jason Wolfe, Aastha Jain and Nick Hay for their comments and suggestions.

References

Aleks, N., Russell, S., Madden, M. G., Morabito, D., Staudenmayer, K., Cohen, M., and Manley, G. T. (2009). Probabilistic detection of short events, with application to critical care monitoring. In Koller, D., Schurmans, D., Bengio, Y., and Bottou, L. (Eds.), *Advances in Neural Information Processing Systems 21*, pp. 49–56. MIT Press.

Baum, L. E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, *41*.

Dean, T. and Kanazawa, K. (1989). A model for reasoning about persistence and causation. *Computational Intelligence*, *5*(3), 142–150.

Fine, S., Singer, Y., and Tishby, N. (1998). The hierarchical hidden markov model: Analysis and applications. In *Machine Learning*, pp. 41–62.

Friedman, N., Murphy, K., and Russell, S. J. (1998). Learning the structure of dynamic probabilistic networks. In *Uncertainty in Artificial Intelligence: Proceedings of the Fourteenth Conference*, Madison, Wisconsin. Morgan Kaufmann.

Ghahramani, Z. and Jordan, M. I. (1997). Factorial hidden Markov models. *Machine Learning*, *29*, 245–274.

Gómez-Uribe, C. A., Verghese, G. C., and Tzafriri, A. (2008). Enhanced identification and exploitation of time scales for model reduction in stochastic chemical kinetics. *The Journal of Chemical Physics*, *129*(24).

Guyton, A. and Hall, J. (1997). *Human Physiology and Mechanisms of Disease*. W.B. Saunders Company.

Iwasaki, Y. and Simon, H. A. (1994). Causality and model abstraction. *Artif. Intell.*, *67*(1), 143–194.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, *82*, 35–46.

Michaelis, L. and Menten, M. (1913). Die kinetik der invertinwirkung. *Biochem.*, *49*, 333–369.

Nodelman, U., Shelton, C., and Koller, D. (2002). Continuous time Bayesian networks. In *Uncertainty in Artificial Intelligence: Proceedings of the Eighteenth Conference*, Edmonton, Alberta. Morgan Kaufmann.

Pavliotis, G. A. and Stuart, A. M. (2007). *Multiscale methods: Averaging and homogenization*. Springer-Verlag.

Yin, G. G. and Zhang, Q. (2004). *Discrete-Time Markov Chains: Two-Time-Scale Methods and Applications*. Applications of Mathematics Stochastic Modelling and Applied Probability 55. Springer-Verlag.