CONTINUOUS TIME BAYESIAN NETWORKS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Uri D. Nodelman
June 2007

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Daphne Koller) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Eric Horvitz)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Andrew Ng)

Approved for the University Committee on Graduate Studies.

To Becky Brooks and Lyn Cox

Abstract

Many domains require us to reason about system change. Examples include life history data analysis, financial risk modelling, fault diagnosis, and the study of evolution. Reasoning about such systems involves asking questions about event timing, e.g., when will a person find employment. Our answers, best expressed as probability distributions over time, must account for many factors that are, themselves, changing. Unfortunately, as the number of variables increases, the state space over which we must maintain a distribution grows exponentially. Such exponential growth makes modelling these domains difficult.

We introduce the framework of *continuous time Bayesian networks* (CTBNs) to address this problem. The approach is based on the framework of finite state, homogeneous Markov processes, but uses ideas from Bayesian networks (BNs) to define continuous time models over a structured state space. The CTBN framework uses cyclic graphs that encode conditional independencies in the distribution over the evolution of the system. It explicitly represents temporal dynamics and allows us to query the network for distributions over the times when particular events of interest occur.

We specify the class of processes representable by CTBNs and prove there is a unique minimal CTBN structure to encode any representable process. We provide algorithms for learning parameters and structure of CTBN models from both fully observed and partially observed data. We prove that the structure learning problem for CTBNs is easier than for traditional BNs or dynamic Bayesian networks (DBNs). We develop an inference algorithm for CTBNs which is a variant of expectation propagation and leverages domain structure and the explicit model of time for computational

advantage. We also show how to use CTBNs to model a rich class of distributions over time. Finally, we compare our framework to DBNs. Importantly, our framework does not require that we choose some fixed temporal granularity; hence, we avoid the DBN requirement that we represent and reason about the process at the finest granularity. Finally, we demonstrate on a real, life history data set that CTBNs with non-exponential duration distributions achieve better performance than DBNs.

Acknowledgments

There are many people who have made this thesis possible.

First, I would like to thank my advisor Daphne Koller whose dedication to excellence and clarity of vision are an inspiration. Her enthusiasm and belief in my potential are what drew me to the Computer Science department. I am especially appreciative of her sharp insight into complex technical issues and her patient support through some challenging periods.

I am deeply indebted to Christian Shelton, with whom I spent countless hours discussing and working on the CTBN framework. My collaboration with him was instrumental to my development as a researcher. I would also like to thank Eric Horvitz, who invited me to spend a fruitful and productive summer at Microsoft Research. My time at MSR and collaboration with Eric led to significant improvements in the expressive power of CTBNs and helped sharpen my thoughts about evaluation metrics. It also opened the door to some exciting, and as yet unexplored, future directions.

I am grateful for comments and feedback from Andrew Ng, Sebastian Thrun, and Ross Schachter before, after, and during my defense. I would also like to thank Suchi Saria, who has helped me clarify many hitherto murky aspects of this work. Her patience and insight have led us to make significant progress on CTBN inference beyond the scope of this thesis.

While at Stanford, I have been privileged to discuss ideas with a number of people. I am especially grateful for many influential conversations with Carlos Guestrin, Avi Pfeffer, Kevin Leyton-Brown, Nir Friedman, Carolyn Talcott, Sol Feferman, Grigori Mints, and Johan van Benthem. I would also like to acknowledge John McCarthy,

Balaji Prabhakar, Eran Segal, Ben Taskar, Dragomir Anguelov, Uri Lerner, Lise Getoor, Urszula Chajewska, Brian Milch, Xavier Boyen, David Vickrey, and Haidong Wang. While at Microsoft Research, I had many productive conversations with Evgeniy Gabrilovich and received useful comments from Max Chickering. I have also benefitted from the opportunity to discuss this work with Oscar Kipersztock from the Boeing Company.

At the University of Maryland, College Park, I would like to thank Jeffrey Adams, Michael Brin, Jim Lesher, and Christopher Cherniak for their support and encouragement.

I would also like to thank my various sources of funding: ONR contract N00014-00-1-0637 under the MURI program "Decision Making under Uncertainty", DARPA's EPCA program, under subcontract to SRI International, the Boeing Company, and Microsoft Research.

Over the course of my graduate studies, the Stanford Encyclopedia of Philosophy has been a second home for me. It has challenged me, supported me, and given me space when needed. But it would not exist without the dedication, integrity, acumen, and deep analytical skills of Edward Zalta. Over the years that I have worked with Ed, he has become a close friend. He and Susanne Riehemann have opened their hearts and home, and I am deeply indebted them.

I am also grateful to all my friends who have helped to sustain me. I would particularly like to mention: Mike Poston, Daniel Lev, Chris and Judy Tate, Paul Oppenheimer, Peter J. Sampson, Melinda Brooks, and Kimberly Bradshaw.

Finally, this work would never have happened without the support and care of my family. They have faithfully sustained and supported me through innumerable challenges. I am ever grateful to the Cox family; Edith Weiss; Ilana Nodelman, Greg Bowman, and Isaac; Laura and Joseph Nodelman; and, above all, Becky Brooks and Lyn Cox, to whom I dedicate this thesis.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Change is ubiquitous. Whether we are playing catch, cooking a meal, investing money, or pursuing a scientific understanding of some corner of the universe, we are reasoning about change. In doing so, we must have some understanding about which changes are likely to occur and when they will happen.

The time and nature of any particular change is influenced by many factors which are, themselves, changing. Consider a medical situation where we have administered a drug to a patient and wish to know how long it will take for the drug to take effect. The answer to this question is likely to depend on various factors, such as how recently the patient has eaten. We want to reason about the temporal process for the effect of the drug and how its dynamics depend on these other factors. As another example, we might want to predict the amount of time that a person remains unemployed, which can depend on the state of the local economy where they live, on their own financial situation, and more.

Although these questions touch on a wide variety of issues, they are all questions about distributions over time. Standard ways of approaching such questions — event history analysis (Blossfeld et al., 1988; Blossfeld & Rohwer, 1995; Andersen et al., 1993) and Markov process models (Aalen, 1975; Andersen & Borgan, 1985; Duffie et al., 1996; Lando, 1998) — work well, but there is a catch. Including more factors

in our model comes at the price of an exponential increase in the size of our state space. We need only two states to describe whether or not a person is employed. But that doubles to four states if we include information about whether or not the person is married. And it doubles again, to eight states, if we want to include information about whether or not the person has children. While eight states is not, in itself, a problem, additional factors will quickly lead us to models that would require millions of states and more.

This exponential explosion in the state space makes investigating these questions more difficult. One approach is to pretend that these other factors never change. So we can learn a two-state model for the employment of a person who is married without children and a separate two-state model for the employment of a person who is married with children. But this is unsatisfactory on several levels. If I want a true estimate of how long a person who is married without children will remain employed (or employed at their current job), I should include the possibility that that person might have children. And someone else might want to ask a slightly different question, such as when a person is likely to have children. Should we have to learn an entirely new set of models?

The solution is to allow the specification of models with structured state spaces where some variables do not directly depend on others. For example, the distribution over how fast a drug takes effect might be mediated by how fast it reaches the bloodstream which may itself be affected by how recently the person has eaten. Also, if the drug is supposed to alleviate pain a person feels in their joints, the drug's influence may be affected by aggravating factors such as whether the barometric pressure is changing. But barometric pressure is not likely to have a direct effect on how recently the person has eaten.

Unfortunately, the previous approaches to answering these kinds of questions have not allowed us to make use of this kind of structure. Bayesian networks (Pearl, 1988) are a standard approach for modelling structured domains. With such a representation we can be explicit about the direct dependencies which are present and use the independencies to our advantage computationally. However, Bayesian networks are designed to reason about static situations, and cannot be used directly to answer the

types of questions that concern us here.

Dynamic Bayesian networks (DBNs) (Dean & Kanazawa, 1989) are the standard extension of Bayesian networks to temporal processes. DBNs model a dynamic system by discretizing time at some *granularity*, Δt, and providing a Bayesian network fragment that represents the probabilistic transition of the state at time t to the state at time $t + \Delta t$. Thus, DBNs represent the state of the system at different points in time, but do not represent time explicitly. As a consequence, it is very difficult to query a DBN for a distribution over the time at which a particular event takes place.

Moreover, the DBN representation forces us to choose one particular granularity for the entire process — it must always propagate the joint distribution over the variables at the same, regular rate. And once we have chosen to use some particular granularity, we must use it to model all the factors in a system — even if some parts of the system change more slowly.

This is not just a representation issue. It leads to computational inefficiencies in inference tasks, where we are computing the answer to some query given some evidence. If we obtain observations which are irregularly spaced in time, we must still represent and perform computations over the intervening time slices at which no evidence is obtained.

Finally, many processes do not have any natural granularity. Does barometric pressure change over minutes, hours, days, or weeks? Will pain medication take effect over minutes or hours? Will a person stay employed for months, years, or decades? If we want to capture all the possibilities we imagine, we may have to model the system at the finest possible granularity (i.e., the smallest Δt). So the inefficiencies described above are unavoidable.

The purpose of this thesis is to address these issues by providing a method of using continuous time models over a structured state space. By modelling time directly we avoid the requirement to choose some granularity for the entire system. We can also answer queries, such as when some medication will take effect, directly with distributions over time. By using a structured state space, we avoid the exponential blow-up in our representation, which we then leverage for computational advantage in learning models from data and using models to answer queries.

1.2 Contributions

The central contribution of this thesis is to introduce the framework of *continuous time Bayesian networks* (CTBNs). The approach is based on the framework of *homogeneous Markov processes*, but utilizes ideas from Bayesian networks to provide a graphical representation language for these systems. In particular, the graphical structure encodes conditional indepencies in the distribution over the evolution of the system.

Moreover, this framework explicitly represents temporal dynamics and allows us to query the network for the distribution over the time when particular events of interest occur. Unlike DBNs, it does not require that we choose some temporal granularity. Given sequences of observations spaced irregularly through time, we can propagate the joint distribution from observation to observation.

More specifically, the contributions of this thesis are:

- We introduce CTBNs, a new framework for modelling finite-state, continuous time processes over a factored state. The graphical representation allows for natural, cyclic dependency graphs. There is no need to specify a temporal granularity for a model.
- We provide two alternative semantics for CTBNs and prove their equivalence.
- We specify the class of processes representable by CTBNs and prove that for any representable process, there is a unique minimal structure among CTBNs that encode it.
- We provide algorithms for learning CTBN parameters from complete data, including maximum likelihood and Bayesian approaches.
- We show how to use score-based structure search to learn CTBN structure from complete data. This includes the derivation of a Bayesian score for evaluating CTBN structures.
- We prove that, when restricted to networks with a fixed maximum number of parents per node, finding the highest scoring CTBN structure can be done in

polynomial time in the number of variables and the size of the dataset. The equivalent problem for regular, dynamic Bayesian networks is NP-hard.

- We show how to use expectation maximization (EM) and structural expectation maximization (SEM) to learn CTBN parameters and structure from incomplete data.
- We develop an inference algorithm for CTBNs — a variant of expectation propagation (EP) — that uses message-passing in a cluster graph. It takes computational advantage of the conditional independencies and the explicit temporal model describing how quickly each node is expected to change. That way, computational resources are focused on parts of the system that are likely to change more rapidly.
- We extend the basic CTBN representation to allow for more complex distributions over when the next state transition will occur.
- We provide a comparison between the CTBN and DBN frameworks and demonstrate on a real, life history data set that CTBNs with non-exponential duration distributions achieve better performance than DBNs.

1.3 Overview

In the next chapter, we review the basic background on Bayesian networks, dynamic Bayesian networks, and continuous time Markov processes. The discussion there about the two forms of parameterization for continuous time Markov processes introduces some new terminology, but is not, in itself, novel.

The contribution of this thesis begins with Chapter 3, covering representation and semantics. It begins with the new definition of *conditional Markov processes*, which form the basic building block of CTBNs. It then defines the CTBN representation and gives two alternative semantics which are shown to be equivalent. After a brief discussion of conditional independence, we provide some results describing the class of Markov processes representable by CTBNs.

Chapter 4 covers the form of the likelihood function and sufficient statistics for CTBNs. It follows from the definitions of Chapter 3 and sets the stage for the next chapters.

Chapter 5 treats the problem of learning CTBNs, providing an algorithm for use when we have complete data. This chapter includes a Bayesian approach to learning parameters and structure. It also contains the key complexity result for model search.

Chapter 6 compares the CTBN and DBN frameworks including some experimental results.

Chapter 7 shows how to use expectation maximization to extend CTBN parameter and structure learning to situations where we have missing data.

Chapter 8 looks at the problem of inference and provides a message-passing cluster graph algorithm that is a variant of Expectation Propagation (Minka, 2001a).

Chapter 9 provides a powerful and general method for increasing the representational power of CTBNs. It also revisits the comparison with DBNs.

Chapter 10 discusses related work and Chapter 11 concludes the thesis with a summary and discussion of future work.

1.4 Publications

Much of the content of this thesis has appeared in previous publications. The publications are

- Nodelman, U., Shelton, C. R., & Koller, D. (2002). Continuous time Bayesian networks. *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (pp. 378–387).

- Nodelman, U., Shelton, C. R., & Koller, D. (2003). Learning continuous time Bayesian networks. *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence* (pp. 451–458).

- Nodelman, U., & Horvitz, E. (2003). *Continuous time Bayesian networks for inferring users' presence and activities with extensions for modeling and evaluation* (Technical Report MSR-TR-2003-97). Microsoft Research.

- Nodelman, U., Koller, D., & Shelton, C. R. (2005a). Expectation propagation for continuous time Bayesian networks. *Proceedings of the Twenty-first Conference on Uncertainty in Artificial Intelligence* (pp. 431–440).

- Nodelman, U., Shelton, C. R., & Koller, D. (2005b). Expectation maximization and complex duration distributions for continuous time Bayesian networks. *Proceedings of the Twenty-first Conference on Uncertainty in Artificial Intelligence* (pp. 421–430).

Chapter 3 includes material from Nodelman et al. (2002; 2003) and Nodelman et al. (2005a). Chapter 4 includes material from Nodelman et al. (2003; 2005b). Chapter 5 and Chapter 6 are largely drawn from Nodelman et al. (2003). The material in Chapter 7 is from Nodelman et al. (2005b). Chapter 8 includes material from from Nodelman et al. (2002; 2003; 2005a). Finally, Chapter 9 is from Nodelman and Horvitz (2003) and Nodelman et al. (2005b).

Chapter 2

Preliminaries

This chapter provides background material about Bayesian networks (BNs) (Pearl, 1988), dynamic Bayesian networks (DBNs) (Dean & Kanazawa, 1989), and Markov processes. The mathematical underpinnings of continuous time Bayesian networks (CTBNs) come from continuous time Markov processes but the key idea of factored state representation and the form of the learning and inference algorithms come from the theory of Bayesian networks. In this way, CTBNs provide an alternative to DBNs for modelling factored state stochastic processes.

We begin with a brief description of some notational conventions we will follow. We then introduce the BN framework, DBNs, and finally continuous time Markov processes.

2.1 Notation

We begin this chapter with a brief discussion of the notation we use for the thesis. This includes reference to some concepts that have not yet been introduced, but is placed here to provide for quick reference.

Process variables, which can be viewed as a continuous time indexed collection of random variables are also denoted by upper case letters — e.g., E, X_i, Z. As a result, regular random variables are generally written with a particular time index — e.g., $E(t)$, $X_i(t)$. In a few cases where we are discussing BNs instead of CTBNs, we

will use non-indexed upper case letters for random variables.

Values of variables are denoted by the corresponding lower case letters, sometimes with subscripted numbers to represent different values — e.g., e_1, e_2, z. The set of instantiations to a variable Z is denoted $\mathit{Val}(Z)$. The number of instantiations is denoted $\mathit{Card}(Z)$ (i.e., the cardinality of the set of instantiations).

Sets of variables are denoted by bold-face upper case letters — e.g., $\boldsymbol{V}$, $\boldsymbol{X}$ — and instantiations to the set are denoted by the corresponding bold-face lower case letter — e.g., $\boldsymbol{v}$, $\boldsymbol{x}$. The set of instantiations to a set of variables $\boldsymbol{X}$ is denoted $\mathit{Val}(\boldsymbol{X})$ and the number of these instantiations is denoted $\mathit{Card}(\boldsymbol{X})$.

The probability distribution over the value of $X(t)$ is denoted $P(X(t))$ or sometimes more simply $P_X(t)$. The initial distribution (at time $t = 0$) is denoted P_X^0. The conditional distribution of $X(t)$ given $X(s)$ is denoted $P(X(t) \mid X(s))$. The expectation of a function f given a distribution P is denoted $E_P[f]$.

In a graph $\mathcal{G}$, the parent set of a variable X_i is denoted $\mathbf{Pa}_{\mathcal{G}}(X_i)$ or, more simply, $\mathbf{U}_i$. The set of children of X_i is denoted $\mathbf{Cd}_{\mathcal{G}}(X_i)$, and the set of co-parents of X_i (i.e., the set of other parents of children of X_i) is denoted

So, for a fixed t, $X(t)$ is simply a random variable. However, Sometimes it is convenient to write X_t for $X(t)$ and $X_{s:t}$ for the trajectory of X on the time interval $[s, t)$.

2.2 Bayesian Networks

We now provide a brief overview of Bayesian networks whose factored representation is leveraged to enable more efficient learning (constructing models from data) and inference (answering probabilistic queries).

2.2.1 Representation

The key insight behind the Bayesian network representation is the importance of *conditional independence*. To explain this, we begin with the definition of *independence* for random variables.

Definition 2.2.1 *Two random variables B_1 and B_2 are* independent *iff*

$$P(B_1, B_2) = P(B_1)P(B_2) \ .$$

That is, iff their joint probability is equal to the probability of the marginals. ∎

Another way to think of this is to observe that if B_1 and B_2 are independent, then $P(B_1 \mid B_2) = P(B_1)$. Intuitively, this says that having evidence about B_2 does not change our distribution over B_1.

If we wish to model a complex domain, using some set of variables, it is unlikely that any of the variables will be independent of each other. *Conditional independence* is a weaker notion of independence, but it is more common.

Definition 2.2.2 *Two random variables B_1 and B_2 are conditionally independent given a set of random variables $\boldsymbol{C}$ iff*

$$P(B_1 \mid \boldsymbol{C}, B_2) = P(B_1 \mid \boldsymbol{C}) \ .$$

∎

This means that if we have evidence over the variables $\boldsymbol{C}$, then B_1 and B_2 do not influence each other. Intuitively, the influence B_1 and B_2 have on each other is mediated through $\boldsymbol{C}$.

It is easier to believe that, in a given domain, most variables will not *directly* affect most other variables. Instead, for each variable, only a limited set of other variables influence it.

That is the key idea of a factored representation and with it, we now define the Bayesian network representation.

Definition 2.2.3 *A* Bayesian network*, $\mathcal{B}$, over a set of random variables $\boldsymbol{B}$ is a compact representation of a joint probability distribution. It is specified in terms of*

- *A* directed acyclic graph *(DAG) $\mathcal{G}$ whose nodes correspond to the random variables $B_i \in \boldsymbol{B}$.*

- *A* conditional probability distributions *(CPDs) for each B_i, specifying the conditional distribution $P(B_i \mid \boldsymbol{Pa}_{\mathcal{G}}(B_i))$ of B_i as a function of its parent set in $\mathcal{G}$.*

For each $B \in \boldsymbol{B}$ with parent set $\mathbf{U}$, we write the probability that $P(B = b \mid \mathbf{U} = \mathbf{u}) = \theta_{b|\mathbf{u}}$, i.e., this how we write the relevant multinomial distribution parameter from the CPD for B. ∎

The CPDs form a set of local probability models that can be combined to describe the full joint distribution over the variables $\boldsymbol{B}$ via the *chain rule*:

$$P(B_1, B_2, ..., B_n) = \prod_{i=1}^{n} P(B_i \mid \mathbf{Pa}_{\mathcal{G}}(B_i)) \ .$$

The graph $\mathcal{G}$ of a Bayesian network encodes a set of conditional indepence assumptions. In particular a variable $B \in \boldsymbol{B}$ is independent of its non-descendents given its parents $\mathbf{Pa}_{\mathcal{G}}(B)$.

There is a useful notion of separation that allows us to characterize independencies in a Bayesian network. We can construct an undirected graph $\mathcal{G}^M$ called the *moralized* graph, such that sets of variables $\boldsymbol{Y}$ and $\boldsymbol{W}$ are conditional independent given some set of variables $\boldsymbol{Z}$ if variables $\boldsymbol{Z}$ separate $\boldsymbol{Y}$ and $\boldsymbol{W}$ in $\mathcal{G}^M$ (Lauritzen et al., 1990). Separation in $\mathcal{G}^M$ simply means that all paths from any node in $\boldsymbol{Y}$ to any node in $\boldsymbol{W}$ must go through a node in $\boldsymbol{Z}$. The construction of the moralized graph is as follows.

Definition 2.2.4 *We construct the* moralized graph*, $\mathcal{G}^M$, which is undirected, from the directed graph $\mathcal{G}$ in two steps. Starting with an empty graph $\mathcal{G}^M$, we add an* undirected *edge between any two nodes X and Y that share a common child in $\mathcal{G}$. Then, we add an undirected edge to $\mathcal{G}^M$ between any two nodes X and Y connected by an arc in $\mathcal{G}$.* ∎

If we want to separate a variable Y from all other variables in the network, it is natural to look at the set of variables that are contained in CIMs with Y. Clearly this includes the parents of Y, $\mathbf{Pa}_{\mathcal{G}}(Y)$, and the children of Y, $\mathbf{Cd}_{\mathcal{G}}(Y)$. But it also includes variables that are co-parents with Y of some other variable because the CIM

of the shared child contains both Y and the co-parent. This leads us to the following definition (Pearl, 1988).

Definition 2.2.5 *Suppose we have a directed graph $\mathcal{G}$ over variables $\boldsymbol{X}$. The* Markov blanket $\boldsymbol{B_Y}$ *of a set of variables $\boldsymbol{Y} \subseteq \boldsymbol{X}$ is the set of variables of $\boldsymbol{X}$ that are not members of $\boldsymbol{Y}$ but are parents or children of some $Y \in \boldsymbol{Y}$ or co-parents with some variable $Y \in \boldsymbol{Y}$. (We say that Y and V are co-parents if they share a child.) More formally, we write*

$$\boldsymbol{B_Y} = \left(\bigcup_{Y \in \boldsymbol{Y}} \boldsymbol{Pa}_{\mathcal{G}}(Y) \cup \boldsymbol{Cd}_{\mathcal{G}}(Y) \cup \boldsymbol{CoPa}_{\mathcal{G}}(Y) \right) \setminus \boldsymbol{Y} .$$

∎

The Markov blanket $\boldsymbol{B_Y}$ separates the nodes $\boldsymbol{Y}$ from the rest of the nodes $\boldsymbol{W}$ in the moralized graph $\mathcal{G}^M$, so the set $\boldsymbol{Y}$ is independent of $\boldsymbol{W}$ given its Markov blanket $\boldsymbol{W}$.

2.2.2 Learning

The data $\mathcal{D}$ we need to learn a BN is a set of *instantiations* of its variables. As a member of the exponential family of probabilistic models, the relevant part of the data for learning can be summarized by aggregating *sufficient statistics*.

The sufficient statistics for a Bayesian network $\mathcal{B}$ are counts of the co-occurrence of a particular value for a variable under different possible instantiations of its parents. So, for variable $B \in \mathcal{B}$ with parent set $\mathbf{U}$, $M[b|\mathbf{u}]$ be the number of occurrences in $\mathcal{D}$ of the co-occurrence of $B = b$ with $\mathbf{U} = \mathbf{u}$.

Significantly, the *likelihood* of the data given the model decomposes by family and can be written as a product of local likelihoods for each variable B

$$L_B(\boldsymbol{\theta} : \mathcal{D}) = \prod_{\mathbf{u}} \prod_{b} \theta_{b|\mathbf{u}}^{M[b|\mathbf{u}]}$$

where $\boldsymbol{\theta}$ is the set of CPD parameters. The log-likelihood is thus a sum of local

log-likelihoods for each variable B,

$$\ell_B(\boldsymbol{\theta} : \mathcal{D}) = \sum_{\mathbf{u}} \sum_{b} M[b|\mathbf{u}] \ln(\theta_{b|\mathbf{u}}) \ .$$

If we want to learn maximum likelihood parameters for the Bayesian network $\mathcal{B}$, we can set

$$\hat{\theta}_{b|\mathbf{u}} = \frac{M[b|\mathbf{u}]}{M[\mathbf{u}]}$$

where $M[\mathbf{u}] = \sum_b M[b|\mathbf{u}]$.

More details on BN learning can be found in Heckerman (1995).

2.2.3 Inference

Theoretically, since a Bayesian network defines a full joint distribution over its variables, we can answer any probabilistic query by forming the joint distribution and then marginalizing out any variables as appropriate. Unfortunately, in most cases this is impractical because the size of the full joint distribution is exponential in the number of variables and once we have more than a few variables we cannot form the joint distribution.

Message passing algorithms work by creating a secondary structure called a *cluster graph*. In cluster graph algorithms, we construct a graph whose nodes correspond to clusters of variables, and pass messages between these clusters to produce an alternative parameterization, in which the marginal distribution of the variables in each cluster can be read directly from the cluster. In discrete graphical models, when the cluster graph is a clique tree, two passes of message passing produce exact marginals. In *generalized belief propagation* (Yedidia et al., 2000), message passing is applied to a graph which is not a clique tree, in which case the algorithm may not converge, and produces only approximate solutions. There are several forms of message passing algorithm. We briefly review the multiply-marginalize-divide scheme of Lauritzen and Spiegelhalter (1988). But first, we must define the notion of a *factor* and its basic operations.

Definition 2.2.6 *A* factor*,* $\phi(\boldsymbol{V})$*, over set of variables or* scope $\boldsymbol{V}$*, is a function from Val*$(\boldsymbol{V})$ *to the set of non-negative real numbers.*

If $\boldsymbol{V}_1$*,* $\boldsymbol{V}_2$*, and* $\boldsymbol{V}_3$ *are disjoint sets of variables, and we have* $\phi_1(\boldsymbol{V}_1, \boldsymbol{V}_2)$ *and* $\phi_2(\boldsymbol{V}_2, \boldsymbol{V}_3)$ *then we define the* factor product *as* $\phi_1 \times \phi_2$ *as a new factor* ψ *where*

$$\psi(\boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3) = \phi_1(\boldsymbol{V}_1, \boldsymbol{V}_2) \cdot \phi_2(\boldsymbol{V}_2, \boldsymbol{V}_3)\,,$$

that is, the product is a new factor over the union of the variables defined by multiplying the value of ϕ_1 *according to the particular instantiation of* $\boldsymbol{V}_1 \cup \boldsymbol{V}_2$ *by the value of* ϕ_2 *according to the the instantiation of* $\boldsymbol{V}_2 \cup \boldsymbol{V}_3$*.*

If $\boldsymbol{V}_1$ *and* $\boldsymbol{V}_2$ *are disjoint sets of variables, and we have* $\phi_1(\boldsymbol{V}_1, \boldsymbol{V}_2)$ *and* $\phi_2(\boldsymbol{V}_2)$ *then the* factor division *of* ϕ_1/ϕ_2 *is a new factor* ψ *where*

$$\psi(\boldsymbol{V}_1, \boldsymbol{V}_2) = \frac{\phi_1(\boldsymbol{V}_1, \boldsymbol{V}_2)}{\phi_2(\boldsymbol{V}_2)}\,.$$

We define the factor marginalization *of* $\boldsymbol{V}_1$ *in* $\phi(\boldsymbol{V}_1, \boldsymbol{V}_2)$ *as a factor* ψ *over* $\boldsymbol{V}_2$ *defined as*

$$\psi(\boldsymbol{V}_2) = \sum_{\boldsymbol{v}_2 \in Val(\boldsymbol{V}_2)} \phi(\boldsymbol{V}_1, \boldsymbol{V}_2)\,.$$

∎

At a high level, a cluster graph is defined in terms of a set of clusters $\boldsymbol{C}_i$, whose *scope* is some subset of the variables $\boldsymbol{X}$. Clusters are connected to each other by edges, along which messages are passed. The edges are annotated with a set of variables called a *sepset* $\boldsymbol{S}_{i,j}$ which is the set of variables in $\boldsymbol{C}_i \cap \boldsymbol{C}_j$. The messages passed over an edge $\boldsymbol{C}_i$—$\boldsymbol{C}_j$ are *factors* over the scope $\boldsymbol{S}_{i,j}$.

Definition 2.2.7 *We say that a cluster graph has the* running intersection property *if, whenever there is a variable* X *such that* $X \in \boldsymbol{C}_i$ *and* $X \in \boldsymbol{C}_j$*, then* X *is also in every cluster in the path between* $\boldsymbol{C}_i$ *and* $\boldsymbol{C}_j$*.* ∎

Each cluster $\boldsymbol{C}_i$ maintains a *potential* π_i, a factor which reflects its current beliefs over the variables in its scope. Each edge similarly maintains a message $\mu_{i,j}$, which

encodes the last message sent over the edge. The potentials are initialized with a product of some subset of factors parameterizing the model (CIMs in our setting). Messages are initialized to be uninformative. Clusters then send messages to each other, and use incoming messages to update their beliefs over the variables in their scope. The message $\delta_{i\to j}$ from $\boldsymbol{C}_i$ to $\boldsymbol{C}_j$ is the marginal distribution $\boldsymbol{S}_{i,j}$ according to π_i. The neighboring cluster $\boldsymbol{C}_j$ assimilates this message by multiplying it into π_i, but avoids double-counting by first dividing by the stored message $\mu_{i,j}$. Thus, the message update takes the form $\pi_j \leftarrow \pi_j \cdot \frac{\delta_{i\to j}}{\mu_{i,j}}$.

More details on message passing algorithms for inference in Bayesian networks can be found in Huang and Darwiche (1996).

2.2.4 Dynamic Bayesian Networks

The dynamic Bayesian network (DBN) framework (Dean & Kanazawa, 1989) is a method for modelling discrete time, or *time-sliced*, stochastic processes. Essentially, it provides a way to define a regular Bayesian network over a countably infinite collection of time indexed random variables.

We have a set of a random variables $\boldsymbol{D}$ which are used to describe the state of the system at a single time-slice. The entire collection of random variables in the underlying Bayesian network are all variables in the set $\{\boldsymbol{D}_0, \boldsymbol{D}_1, \boldsymbol{D}_2, ...\}$.

Definition 2.2.8 *A* dynamic Bayesian network *(DBN) is a pair* $\langle \mathcal{B}_0, \mathcal{B}_{2T} \rangle$ *where* $\mathcal{B}_0$ *is a Bayesian network over* $\boldsymbol{D}_0$*, and* $\mathcal{B}_{2T}$ *is called a 2-TBN, or* 2 time slice Bayesian network*. It provides a transition model from time slice* i *to* $i+1$ *as a Bayesian network over the variables of* $\boldsymbol{D}_{i+1}$ *conditional on the instantiation of* $\boldsymbol{D}_i$*. Two sorts of arcs are allowed:*

- Intra-time-slice arcs *are allowed between the variables within a time slice — e.g., between* D *and* D' *for* $D, D' \in \boldsymbol{D}_{i+1}$*.*
- Inter-time-slice arcs *are allowed from a variable in one time-slice to any variable in the next time slice — e.g., between* $D \in \boldsymbol{D}_i$ *and* $D \in \boldsymbol{D}_{i+1}$ *or between* $D \in \boldsymbol{D}_i$ *and* $D' \in \boldsymbol{D}_{i+1}$*.*

The 2-TBN specifies the CPDs for the variables in $\boldsymbol{D}_{i+1}$ *but not in* $\boldsymbol{D}$. ∎

More details on dynamic Bayesian networks can be found in Murphy (2002).

2.3 Continuous Time Markov Processes

2.3.1 Background

We can think of a continuous time random process X as a collection of random variables indexed by time $t \in [0, \infty)$. it is often more convenient to view X across all values of t as a single *process variable* or PV. The instantiation of a particular set of values for $X(t)$ for all t is called a *trajectory*.

Definition 2.3.1 *The* Markov assumption *is the assumption that the future of a process is independent of its past given its present. More explicitly, the process* X *satisfies the Markov assumption iff* $P(X(t+\Delta t) \mid X(t), X(s)) = P(X(t+\Delta t) \mid X(t))$ *for all* s, t *such that* $s < t < \infty$. *(Norris, 1997)*

2.3.2 Definitions

We begin the discussion of continuous time with finite state, continuous time Markov processes. Such a process is generally defined by an initial distribution and a matrix of transition *intensities* where the (i, j) entry gives the intensity of transitioning from state i to state j and the entries along the main diagonal make each row sum to zero. As will be seen later, the CTBN framework will be based on *homogeneous Markov processes* — in which the transition intensities do not depend on time.

Definition 2.3.2 *Let* X *be a process variable whose state changes over continuous time. Let the domain of* X *be* $Val(X) = \{x_1, x_2, \ldots, x_N\}$. *Then* X *is a homogeneous Markov process iff its behavior can be specified in terms of an initial distribution* P_0^X *over* $Val(x)$ *and a Markovian transition model usually presented as an intensity*

matrix:

$$\mathbf{Q}_X = \begin{bmatrix} -q_1 & q_{12} & \cdots & q_{1N} \\ q_{21} & -q_2 & \cdots & q_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ q_{N1} & q_{N2} & \cdots & -q_N \end{bmatrix} ,$$

where $q_i = \sum_{j \neq i} q_{ij}$ *and* $q_i, q_{ij} \geq 0$. ∎

Intuitively, the intensity q_i gives the 'instantaneous probability' of leaving state x_i and the intensity q_{ij} gives the 'instantaneous probability' of transitioning from x_i to x_j. More formally,

$$\begin{aligned} \lim_{\Delta t \to 0} P(X(t + \Delta t) = x_j \mid X(t) = x_i) &= \lim_{\Delta t \to 0} q_{ij} \Delta t + O(\Delta t^2), \text{for } i \neq j \\ \lim_{\Delta t \to 0} P(X(t + \Delta t) = x_i \mid X(t) = x_i) &= \lim_{\Delta t \to 0} 1 - q_i \Delta t + O(\Delta t^2) \ . \end{aligned} \tag{2.1}$$

As a whole, the matrix $\mathbf{Q}_X$ defines the instantaneous behavior of the process X and makes X satisfy the Markov assumption. We can see this because the future behavior of X is defined by $\mathbf{Q}_(X)$ solely in terms of its current state. Thus $P(X(t + \Delta t) \mid X(t), X(s)) = P(X(t + \Delta t) \mid X(t))$ for $0 < s < t < \infty$.

The instantaneous specification of the transition model of X induces a probability distribution over its trajectories. To see the way the distribution is induced, we must first introduce a matrix function.

Definition 2.3.3 *The* matrix exponential *for a matrix* $\mathbf{Q}$ *is defined as*

$$\exp \mathbf{Q} = \sum_{k=0}^{\infty} \frac{\mathbf{Q}^k}{k!} \ .$$

Note that though the matrix exponential is defined with this equation, it is not generally a good way to compute it — see Moler and Loan (2003). ∎

Now, analogously to Equation 2.1, we have

$$\begin{aligned}\lim_{\Delta t \to 0} P(X(t+\Delta t) \mid X(t)) &= \lim_{\Delta t \to 0} \exp(\mathbf{Q}_X \Delta t) \\ &= \lim_{\Delta t \to 0} \mathbf{I} + \mathbf{Q}_X \Delta t + O(\Delta t^2) \ . \end{aligned} \tag{2.2}$$

The relationship between the intensity matrix $\mathbf{Q}_X$ and the probability distribution over the trajectory of X is discussed further in Section 2.3.5.

Given the $\mathbf{Q}_X$ matrix we can describe the transient behavior of $X(t)$ as follows. If $X(0) = x_i$ then it stays in state x_i for an amount of time exponentially distributed with parameter q_i. Thus, the probability density function f and corresponding distribution function F for $X(t)$ remaining equal to x_i are given by

$$\begin{aligned} f(t) &= q_i \exp(-q_i t), \qquad t \geq 0 \\ F(t) &= 1 - \exp(-q_i t), \quad t \geq 0 \ . \end{aligned}$$

The expected time of transitioning is $1/q_i$. Upon transitioning, X shifts to state x_j with probability q_{ij}/q_i.

Example 2.3.4 *Assume that we want to model the behavior of the barometric pressure $B(t)$ discretized into three states (b_1 = falling, b_2 = steady, and b_3 = rising), we could write the intensity matrix as*

$$\mathbf{Q}_B = \begin{bmatrix} -.21 & .2 & .01 \\ .05 & -.1 & .05 \\ .01 & .2 & -.21 \end{bmatrix} \ .$$

If we view units for time as hours, this means that if the pressure is falling, we expect that it will stop falling in a little less than 5 hours ($1/.21$ hours). It will then transition to being steady with probability $.2/.21$ and to falling with probability $.01/.21$.

It is also useful to note the following definitions.

Definition 2.3.5 *Two Markov processes are said to be* stochastically equivalent *if they have the same state space and transition probabilities (Gihman & Skorohod,*

1973).

Definition 2.3.6 *Two Markov processes X and Y are* independent *if $P(X,Y) = P(X)P(Y)$. That is, if the distribution over their joint trajectories is equal to the product of the marginal distributions over their trajectories.*

2.3.3 Pure Intensity and Mixed Intensity Parameterizations

When the transition model is defined solely in terms of an intensity matrix (as above), we refer to it as using a *pure intensity* parameterization. The parameters for an N state process are $\{q_i, q_{ij} \in \mathbf{Q}_X : 1 \leq i, j \leq N, i \neq j\}$.

This is not the only way to parameterize a Markov process. Note that the distribution over transitions of X factors into two pieces: an exponential distribution over *when* the next transition will occur and a multinomial distribution over *where* the state transitions — the next state of the system.

Definition 2.3.7 *The* mixed intensity *parameterization for a homogeneous Markov process X with N states is given by the two sets of parameters*

$$\boldsymbol{q}_X = \{q_i : 1 \leq i \leq N\},$$
$$\boldsymbol{\theta}_X = \{\theta_{ij} : 1 \leq i, j \leq N, i \neq j\},$$

where $\boldsymbol{q}_X$ is a set of intensities parameterizing the exponential distributions over when *the next transition occurs and $\boldsymbol{\theta}_X$ is a set of probabilities parameterizing the distribution over* where *the state transitions.* ∎

In both cases we use N^2 parameters, although there are only $N^2 - N$ free parameters. To relate these two parametrizations we note the following theorem.

Theorem 2.3.8 *Let X and Y be two Markov processes over the same state space with the same initial distribution. If X is defined by the intensity matrix $\mathbf{Q}_X$ and Y is defined by the mixed intensity parameterization $\boldsymbol{q}_Y$, $\boldsymbol{\theta}_Y$ then X and Y are stochastically*

equivalent if and only if $q_i \in \boldsymbol{q}_Y$ *is identical to* $q_i \in \mathbf{Q}_X$ *and*

$$\theta_{ij} = \frac{q_{ij}}{q_i} \ .$$

Proof: X and Y share the same state space and stay in state i for time exponentially distributed with parameter q_i after which they transition to state $j \neq i$ with probability q_{ij}/q_i. So, by Definition 2.3.5 they are stochastically equivalent. If any $q_i \in \boldsymbol{q}_Y$ or if any $\theta_{ij} \in \boldsymbol{\theta}_Y$ is set differently then the corresponding transition probability would be different so X and Y would not be stochastically equivalent. ∎

Another way to see the relationship between these two parameterizations is by considering the transitions made between two consecutive different states, ignoring the time spent at each state. Specifically, we can define the embedded Markov chain E which is formed by ignoring the amount of time X spends in its states and noting only the sequence of transitions it makes from state to state. We can write out the $N \times N$ transition probability matrix $\boldsymbol{P}_E$ for this chain, by putting zeros along the main diagonal and θ_{ij} in the (i, j) entry. We can also consider the distribution over the amount of time X spends in a state before leaving again, ignoring the particular transitions X makes. We can write out the $N \times N$ state duration matrix $\boldsymbol{M}$ (which is often called the *completion rate matrix* or *holding rate matrix*), by putting the q_i values along the main diagonal and zeros everywhere else. It is easy to see that we can describe the original intensity matrix in terms of these two matrices:

$$\mathbf{Q} = \boldsymbol{M}(\boldsymbol{P}_E - \mathbf{I}) \ .$$

Example 2.3.9 *For our barometric pressure process* B*,*

$$\mathbf{Q}_B = \begin{bmatrix} .21 & 0 & 0 \\ 0 & .1 & 0 \\ 0 & 0 & .21 \end{bmatrix} \left(\begin{bmatrix} 0 & \frac{20}{21} & \frac{1}{21} \\ \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{20}{21} & \frac{1}{21} & 0 \end{bmatrix} - \mathbf{I} \right) \ .$$

Depending on the task, sometimes one parameterization is more convenient and we will use them interchangeably.

2.3.4 Subsystems

It is often useful to consider *subsystems* of a Markov process. A subsystem, S, describes the behavior of the process over a subset of the full state space — i.e., $Val(S) \subset Val(X)$. In such cases we can form the intensity matrix of the subsystem, $\mathbf{U}_S$, by using only those entries from $\mathbf{Q}_X$ that correspond to states in S.

Example 2.3.10 *If we want the subsystem of the barometric pressure process, B, corresponding to the pressure being steady or rising ($S = \{b_2, b_3\}$), we get*

$$\mathbf{U}_S = \begin{bmatrix} -.1 & .05 \\ .2 & -.21 \end{bmatrix} .$$

Note that, for a subsystem, the sums of entries along a row are not, in general, zeros. This is because a subsystem is not a closed system — i.e, from each state, there can be a positive probability of entering states not in S and thus not represented in the transition matrix for the subsystem.

Once we have formed a subsystem S of X, we can also talk about the complement subsystem $\overline{S}$, which is a subsystem over the other states — i.e., $Val(\overline{S}) = Val(X) - Val(S)$. In general, when examining the behavior of a subsystem, we consider the *entrance* and *exit* distributions for the subsystem. An *entrance distribution* is a distribution over the states of S, where the probability of a state s is the probability that s is the state to which we first transition when entering the subsystem S. An *exit distribution* describes the first state not in $Val(S)$ to which we transition when we leave the subsystem.

2.3.5 Reasoning with Markov processes

If we have an intensity matrix, $\mathbf{Q}_X$, for a homogeneous Markov process $X(t)$ and an initial distribution over the value of X at time 0, P_X^0, there are many questions about the process which we can answer.

The conditional distribution over the value of X at time t given the value at an

earlier time s is

$$P(X(t) \mid X(s)) = \exp(\mathbf{Q}_X(t-s)) \text{, for } s < t \ .$$

Thus, the distribution over the value of $X(t)$ is given by

$$P_X(t) = P_X^0 \exp(\mathbf{Q}_X t) \ .$$

As t grows, $P_X(t)$ approaches the stationary distribution π for X which can be computed by an eigenvalue analysis. Additionally, we can form the joint distribution over any two time points using the above two formulas:

$$P_X(s,t) = P_X(s) \exp(\mathbf{Q}_X(t-s)) \ .$$

Suppose we are interested in some subsystem S of X. Given an entrance distribution P_S^0 into S, we can calculate the distribution over the amount of time that we remain within the subsystem. This distribution function is called a *phase distribution* (Neuts 1975; 1981), and is given by

$$F_S(t) = 1 - P_S^0 \exp(\mathbf{U}_S t)\mathbf{e} \ .$$

where $\mathbf{U}_S$ is (as above) the subsystem intensity matrix and $\mathbf{e}$ is a vector of ones. The expected time to remain within the subsystem is given by $P_S^0(-\mathbf{U}_S)^{-1}\mathbf{e}$.

Example 2.3.11 *In our barometric pressure example, if we have a uniform entrance distribution for the subsystem S in Example 2.3.10, then the distribution in time over when the pressure begins to fall is given by*

$$\begin{aligned} F_S(t) &= 1 - \begin{bmatrix} .5 & .5 \end{bmatrix} \exp\left(\begin{bmatrix} -.1 & .05 \\ .2 & -.21 \end{bmatrix} t \right) \mathbf{e} \\ &\approx 1 - 1.0476(0.960^t) + 0.0476(0.7641^t) \ . \end{aligned}$$

Finally, given an entrance distribution, P_S^0, to a subsystem S of X, we can calculate

the exit distribution. To do so, we construct a new process X' by setting all intensities to zero within rows corresponding to states not in S. This transformation, in effect, makes every state which is not in the subsystem an absorbing state. (Once the system has entered an absorbing state, it can never leave that state.) If we use our entrance distribution over the states of S for our initial distribution to X' (setting the probability of starting in other states to zero), we can see that the exit distribution is given by the stationary distribution of X'. This is because the only way that we can enter the newly constructed absorbing states is by leaving S and so the probability with which we end up in an absorbing state is the probability that we entered that state by exiting the subsystem.

Chapter 3

Representation and Semantics

The key step in bringing the representational power of the factored state Bayesian network to Markov process modelling is the definition of the local probability model. We will need a way for the dynamics of a local Markov process to depend on some *limited* set of other variables — not on the state of the entire system.

This chapter starts by showing how to define these local probability models and how to compose them to create a continuous time Bayesian network. We then describe how to understand the CTBN as a single global process defined by the set of local models. We show that the global process is itself a homogeneous Markov process.

We examine the conditional indepencies encoded by a CTBN and clarify the class of homogeneous Markov processes representable as CTBNs. We then show that, for any representable process, there is a unique minimal structure to encode it.

3.1 Continuous Time Bayesian Networks

3.1.1 Conditional Markov Processes

In order to compose Markov processes in a larger network, we need to introduce the notion of a *conditional Markov process*. This is a type of inhomogeneous Markov process where the intensities vary with time, but not as a direct function of time. Rather, the intensities are a function of the current values of a set of other variables,

which also evolve as Markov processes.

Let Y be a variable whose domain is $Val(Y) = \{y_1, y_2, \ldots, y_m\}$. Assume that Y evolves as a Markov process $Y(t)$ whose dynamics are conditioned on a set $\boldsymbol{V}$ of variables, each of which also can also evolve over time. Then we have a *conditional intensity matrix* (CIM) which can be written

$$\mathbf{Q}_{Y|\boldsymbol{V}} = \begin{bmatrix} -q^y_1(\boldsymbol{V}) & q^y_{12}(\boldsymbol{V}) & \cdots & q^y_{1m}(\boldsymbol{V}) \\ q^y_{21}(\boldsymbol{V}) & -q^y_2(\boldsymbol{V}) & \cdots & q^y_{2m}(\boldsymbol{V}) \\ \vdots & \vdots & \ddots & \vdots \\ q^y_{m1}(\boldsymbol{V}) & q^y_{m2}(\boldsymbol{V}) & \cdots & -q^y_m(\boldsymbol{V}) \end{bmatrix} .$$

Equivalently, we can view a CIM as set of intensity matrices, one for each instantiation of values $\boldsymbol{v}$ to the variables $\boldsymbol{V}$. The set of variables $\boldsymbol{V}$ are called the *parents* of Y, and denoted $\mathbf{Pa}(Y)$. Note that, if the parent set $\mathbf{Pa}(Y)$ is empty, then the CIM is simply a standard intensity matrix. Just as a regular intensity matrix, a CIM induces a distribution over the dynamics of Y given the behavior of $\mathbf{Pa}(Y) = \boldsymbol{V}$. If $\boldsymbol{V}$ is instantiated as $\boldsymbol{v}$ on some interval $[t, t+\epsilon), \epsilon > 0$, then, as in Equation 2.2,

$$\begin{aligned} \lim_{\Delta t \to 0} \Pr\{Y_{t+\Delta t} \mid Y_t, \boldsymbol{v}\} &= \lim_{\Delta t \to 0} \exp(\mathbf{Q}_{Y|\boldsymbol{v}} \Delta t) \\ &= \lim_{\Delta t \to 0} \mathbf{I} + \mathbf{Q}_{Y|\boldsymbol{v}} \Delta t + O(\Delta t^2) \ . \end{aligned} \tag{3.1}$$

If we specify an initial distribution over Y then we have defined a Markov process whose behavior depends on the instantiation of values to $\mathbf{Pa}(Y)$.

Example 3.1.1 *Consider a variable $E(t)$ which models whether or not a person is eating (e_1 = not eating, e_2 = eating) and is conditional on a variable $H(t)$ which models whether or not a person is hungry (h_1 = not hungry, h_2 = hungry). Then we can specify the CIM for $E(t)$ as*

$$\mathbf{Q}_{E|h_1} = \begin{bmatrix} -.01 & .01 \\ 10 & -10 \end{bmatrix} \quad \mathbf{Q}_{E|h_2} = \begin{bmatrix} -2 & 2 \\ .01 & -.01 \end{bmatrix} \ .$$

Given this model, we expect a person who is hungry and not eating to begin eating in

*half an hour. We expect a person who is not hungry and is eating to stop eating in 6 minutes (*1/10 *hour).*

3.1.2 The CTBN Model

Conditional intensity matrices provide a way of modelling the local dependence of one variable on a set of others. By putting these local models together, we can define a single joint structured model. As is the case with dynamic Bayesian networks, there are two central components to define: the initial distribution and the dynamics with which the system evolves through time.

Definition 3.1.2 *Let $\boldsymbol{X}$ be a set of local variables $X_1, \ldots, X_n$. Each X_i has a finite domain of values Val(X_i). A* continuous time Bayesian network $\mathcal{N}$ *over $\boldsymbol{X}$ consists of two components: The first is an* initial distribution $P^0_{\boldsymbol{X}}$*, specified as a Bayesian network $\mathcal{B}$ over $\boldsymbol{X}$. The second is a* continuous transition model*, specified as*

- *A directed (possibly cyclic) graph $\mathcal{G}$ whose nodes are $X_1, \ldots, X_n$; $\boldsymbol{Pa}(X_i)$ denotes the parents of X_i in $\mathcal{G}$.*
- *A conditional intensity matrix, $\mathbf{Q}_{X|\boldsymbol{Pa}(X)}$, for each variable $X_i \in \boldsymbol{X}$.* ∎

Unlike traditional Bayesian networks, there is no problem with cycles in the graph $\mathcal{G}$. An arc $X \to Y$ in the graph implies that the dynamics of Y's evolution in time depends on the value of X. There is no reason why the dynamics of X's evolution cannot simultaneously depend on the value of Y. This dependency is analogous to a DBN model where we have arcs $X^t \to Y^{t+1}$ and $Y^t \to X^{t+1}$.

Example 3.1.3 *Figure 3.1 shows the graph structure for a sample CTBN modelling the drug effect situation we described in Section 1.1. There are nodes for the uptake of the drug and for the resulting concentration of the drug in the bloodstream. The concentration is affected by the how full the patient's stomach is. The drug is supposed to alleviate joint pain, which may be aggravated by falling pressure. The drug may also cause drowsiness. The model contains a cycle, indicating that whether a person is hungry depends on how full their stomach is, which depends on whether or not they are eating.*

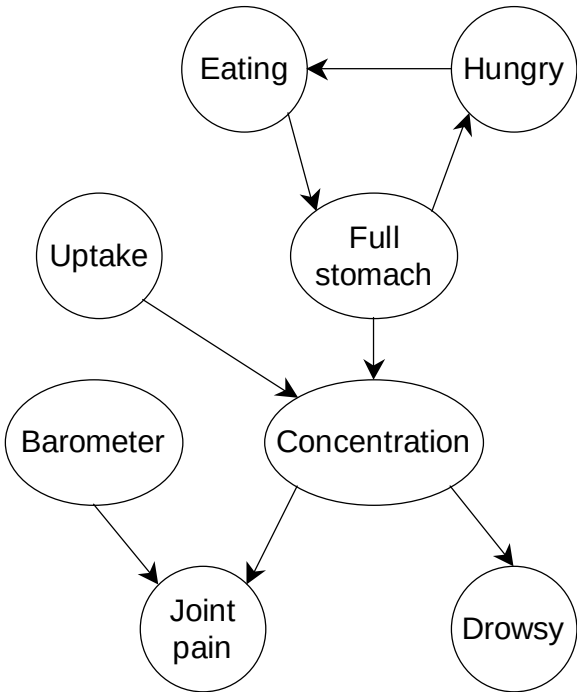

Figure 3.1: Drug effect network

3.2 Generative Semantics

In order to define the semantics of a CTBN, we must show how to view the entire system as a single process.

It is important to note that we make a fundamental assumption in the construction of the CTBN model: two variables cannot transition at the same time. This can be viewed as a formalization of the view that variables must represent distinct aspects of the world. We should not, therefore, model a domain in which we have two variables that functionally and deterministically change simultaneously. For example, in the drug effect network, we should not add a variable describing the type of food, if any, a person is eating. We could, however, change the value space of the Eating variable from a binary yes/no to a more descriptive set of possibilities.

In this section, we define the semantics of CTBNs by specifying a generative process, a procedure that takes the initial distribution and the description of each

Procedure *CTBN-Sample*(t_{end})
1. $t \leftarrow 0$, $\sigma \leftarrow \emptyset$
2. For each variable $X \in \boldsymbol{X}$
   Choose state $x(0)$ according to $\boldsymbol{\theta}^{\mathcal{B}}_{X|\mathbf{pa}_{\mathcal{B}}(X)}$.

Loop:

3. For each variable X such that $Time(X)$ is undefined:
   Choose Δt for next X transition from an exponential dist. with parameter $q_{x(t)|\mathbf{u}_X(t)}$.
   Define $Time(X) \leftarrow t + \Delta t$
4. Let $X = \arg\min_{Y \in \boldsymbol{X}}[Time(Y)]$
5. If $t \geq t_{end}$
   return σ
6. Update $t \leftarrow Time(X)$
7. Choose $x(t)$ the next value of X from the multinomial with parameters $\boldsymbol{\theta}_{x(t)|\mathbf{u}_X(t)}$.
   Add $\langle X \leftarrow x(t), t\rangle$ to σ.
   Undefine $Time(X)$ and $Time(Y)$ for all variables Y for which $X \in \mathbf{U}_Y$.

Figure 3.2: Forward sampling semantics for a CTBN

local variable and randomly samples a trajectory for the system.

The procedure, shown in Fig. (3.2), takes an end time and returns a sampled trajectory σ ending at that time. For each variable $X \in \boldsymbol{X}$, it maintains $x(t)$ — the state of X at time t, and $Time(X)$ — the next potential transition time for X. We will use $\mathbf{u}_X(t)$ to represent the instantiation to parents of X at time t. The initial distribution is defined by the Bayesian network $\mathcal{B}$. Recall from Theorem 2.3.8 that for every $X \in \boldsymbol{X}$ and every $x \in Val(X)$, the parameters of the multinomial distribution, $\boldsymbol{\theta}_{x|\mathbf{u}_X}$ are set such that

$$\theta_{xx'|\mathbf{u}_X} = \frac{q_{xx'|\mathbf{u}_X}}{\sum_{x'} q_{xx'|\mathbf{u}_X}}\,,\ x' \neq x\,. \tag{3.2}$$

3.3 Amalgamation Semantics

An alternative method is to construct a single, homogeneous Markov process which describes the behavior of the CTBN. To do this, we introduce an operation called

amalgamation on CIMs. This operation combines two CIMs to produce a single, larger CIM. The new CIM contains the intensities for the variables in $\boldsymbol{S}$ conditioned on those of $\boldsymbol{C}$. As noted above, a basic assumption is that, as time is continuous, variables cannot transition at the same instant. Thus, all intensities corresponding to two simultaneous changes are zero.

As we will be discussing the structure of single matrix over multiple variables, we need a function to map rows of the larger matrix to instantiations of the variables. To do this we use an ordering of the variables which, in turn, defines a unique sequence of instantiations to the set.

With that, we will be able to define the *expansion* of a CIM (defined as a set of matrices) into a single matrix over a larger set of variables. Then we will describe the amalgamation operation which leads to the joint intensity matrix expresses the behavior of a CTBN as a homogeneous Markov process.

We will conclude this section by showing the equivalence of this approach to the generative semantics given in the previous section.

3.3.1 Variable Orderings

As noted above, when discussing matrices over multiple variables, we need a mapping between row or column numbers and instantiations of the variables. This mapping is provided by a variable ordering.

Definition 3.3.1 *Let $\xi_{\boldsymbol{S}}$ be a* variable ordering *for variables in the set $\boldsymbol{S}$. Then $\xi_{\boldsymbol{S}}$ can be viewed as a mapping from row numbers to instantiations $\boldsymbol{s}$ of $\boldsymbol{S}$ by adopting the convention that we iterate over the all the values of the first variable in the ordering before iterating to the second value of the second variable, and continuing in that way to iterate over the values of all the variables in turn. We write $\xi_{\boldsymbol{S}}[i]$ for the ith instantiation.* ∎

We also define notions of consistency between orderings and, separately, between instantiations.

Definition 3.3.2 *For two variable ordering $\xi_{\boldsymbol{S}}$ and $\xi_{\boldsymbol{S}'}$ over (possibly) distinct sets of variables $\boldsymbol{S}$ and $\boldsymbol{S}'$, we say that the orderings are* consistent *if they have the same order for the variables of the intersection $\boldsymbol{S} \cap \boldsymbol{S}'$.* ∎

Note that for any subset $\boldsymbol{S}' \subseteq \boldsymbol{S}$, the ordering $\xi_{\boldsymbol{S}}$ defines a unique, consistent sub-ordering $\xi_{\boldsymbol{S}'}$ by simply leaving out the extra variables, $\boldsymbol{S} \setminus \boldsymbol{S}'$.

Definition 3.3.3 *We say the instantiations $\xi_{\boldsymbol{S}}[j]$ and $\xi_{\boldsymbol{S}'}[k]$ are* consistent *if they assign the same values to the variables in the intersection $\boldsymbol{S} \cap \boldsymbol{S}'$ and write this $\xi_{\boldsymbol{S}}[j] \cong \xi_{\boldsymbol{S}'}[k]$. We note that instantiations can be consistent even if the orderings $\xi_{\boldsymbol{S}}$ and $\xi_{\boldsymbol{S}'}$ are inconsistent.* ∎

Example 3.3.4 *Consider the ordering $\xi_{\boldsymbol{S}} = \langle X, Y, Z \rangle$, where all variables are binary. Then the state sequence defined by $\xi_{\boldsymbol{S}}$ is*

$$\langle x_1, y_1, z_1 \rangle, \langle x_2, y_1, z_1 \rangle, \langle x_1, y_2, z_1 \rangle, \langle x_2, y_2, z_1 \rangle,$$
$$\langle x_1, y_1, z_2 \rangle, \langle x_2, y_1, z_2 \rangle, \langle x_1, y_2, z_2 \rangle, \langle x_2, y_2, z_2 \rangle .$$

Thus, $\xi_{\boldsymbol{S}}[2] = \langle x_2, y_1, z_1 \rangle$ and $\xi_{\boldsymbol{S}}[7] = \langle x_1, y_2, z_2 \rangle$. Also, note that $\xi_{\boldsymbol{S}}$ is consistent with the ordering $\langle Y, W, Z \rangle$ but inconsistent with $\langle W, Z, X \rangle$. Finally, $\xi_{\boldsymbol{S}}[2] \cong \langle y_1, w_2, z_1 \rangle$ and $\xi_{\boldsymbol{S}}[7] \cong \langle w_1, z_2, x_1 \rangle$. ∎

3.3.2 CIM Expansion

As noted above, before we can combine CIMs by amalgamation, we need a way to write them as single matrices over the same set of variables. A CIM $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ over variables $\boldsymbol{S} \subseteq \boldsymbol{X}$ conditioned on $\boldsymbol{C} \subset \boldsymbol{X}$ defines the dynamics of $\boldsymbol{S}$ given $\boldsymbol{C}$. We can rewrite $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ as a single block matrix over the joint space $\boldsymbol{S} \times \boldsymbol{C}$:

$$\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}} = \begin{bmatrix} \mathbf{Q}_{S|c_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_{S|c_2} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{Q}_{S|c_N} \end{bmatrix} .$$

We call this a CIM which is *expanded* over the variables $\boldsymbol{S} \cup \boldsymbol{C}$. Again, the CIM written in this way induces a distribution over the dynamics of the variables $\boldsymbol{S}$ given the trajectory of the variables $\boldsymbol{C}$. Here, we have

$$\begin{aligned} \lim_{\Delta t \to 0} \Pr\{\boldsymbol{S}_{t+\Delta t} \mid \boldsymbol{S}_t, \boldsymbol{C}\} &= \lim_{\Delta t \to 0} \exp(\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}} \Delta t) \\ &= \lim_{\Delta t \to 0} \mathbf{I} + \mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}} \Delta t + O(\Delta t^2) \ . \end{aligned} \tag{3.3}$$

Note that, unlike Equation 3.1, we do not need to worry about the particular instantiation $\boldsymbol{c}$ to variables $\boldsymbol{C}$ because when written as a single matrix, $\exp(\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}})$ includes a copy of $\exp(\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}})$ for every instantiation $\boldsymbol{c}$.

More formally, we define CIM expansion in terms of a set of *expansion matrices*.

Definition 3.3.5 *Fix an ordering $\xi_{\boldsymbol{S},\boldsymbol{C}}$ and let $\xi_{\boldsymbol{S}}$ and $\xi_{\boldsymbol{C}}$ be the consistent sub-orderings. Let $\boldsymbol{c}_i = \xi_{\boldsymbol{C}}[i]$. Then, the* expansion matrix $\mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i}$ *is a rectangular matrix of dimension Card$(\boldsymbol{S} \cup \boldsymbol{C}) \times$ Card$(\boldsymbol{S})$ where the $[j,k]$ entry is defined as*

$$\mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i}[j,k] = \begin{cases} 1 & \text{if } \xi_{\boldsymbol{S},\boldsymbol{C}}[j] \cong \xi_{\boldsymbol{C}}[i] \text{ and } \xi_{\boldsymbol{S},\boldsymbol{C}}[j] \cong \xi_{\boldsymbol{S}}[k] \\ 0 & \text{otherwise}. \end{cases}$$

That is, $\mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i}$ is everywhere 0 except on the square submatrix defined by the set of rows consistent with $\boldsymbol{c}_i$. On that square submatrix, $\mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i}$ is the identity matrix — i.e., a matrix that is everywhere 0 except along the main diagonal corresponding to entries where the row and column correspond to consistent instantiations of $\boldsymbol{S}$. ∎

Example 3.3.6 *Suppose we have two binary variables X and Y and the ordering $\langle X, Y \rangle$. So, the state sequence for the rows is $\langle x_1, y_1 \rangle$, $\langle x_2, y_1 \rangle$, $\langle x_1, y_2 \rangle$, $\langle x_2, y_2 \rangle$. Then,*

$$\mathbf{N}_{X|y_1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \mathbf{N}_{X|y_2} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \ .$$

We also have

$$\mathbf{N}_{Y|x_1} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \quad \mathbf{N}_{Y|x_2} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} .$$

Note that for $\mathbf{N}_{Y|x_2}$ only the second and fourth rows correspond to $X = x_2$, so all other rows have only zeroes. In the second row, which corresponds to $\langle x_2, y_1 \rangle$, we have a 1 in column one which corresponds to the matching instantiation $Y = y_1$. Similarly, in the fourth row we have a 1 in column two, corresponding to $Y = y_2$. ∎

Example 3.3.7 *Suppose we have a binary variable A, a three-valued variable B and the ordering $\langle A, B \rangle$. So, the state sequence for the rows is*

$$\langle a_1, b_1 \rangle, \langle a_2, b_1 \rangle, \langle a_1, b_2 \rangle, \langle a_2, b_2 \rangle, \langle a_1, b_3 \rangle, \langle a_2, b_3 \rangle.$$

Then,

$$\mathbf{N}_{A|b_1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \mathbf{N}_{A|b_2} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \mathbf{N}_{A|b_3} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} .$$

We also have

$$\mathbf{N}_{B|a_1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad \mathbf{N}_{B|a_2} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} .$$

∎

We are now ready to define the expansion of a CIM $\mathbf{Q}_{\boldsymbol{S|C}}$ into a single matrix

over $\boldsymbol{S} \times \boldsymbol{C}$.

Definition 3.3.8 *The* expanded form *of* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ *given the ordering* $\xi_{\boldsymbol{S},\boldsymbol{C}}$ *is a Card*$(\boldsymbol{S} \cup \boldsymbol{C}) \times$ *Card*$(\boldsymbol{S} \cup \boldsymbol{C})$ *matrix defined in terms of the Card*$(\boldsymbol{S}) \times$ *Card*$(\boldsymbol{S})$ *matrices* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ *as*

$$\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}} = \sum_{\boldsymbol{c}_i \in Val(\boldsymbol{C})} \mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i} \mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i} \mathbf{N}'_{\boldsymbol{S}|\boldsymbol{c}_i} ,$$

where $\mathbf{N}'_{\boldsymbol{S}|\boldsymbol{c}_i}$ *is the transpose of* $\mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i}$*. Essentially, we iterate over the instantiations of the ordering* $\xi_{\boldsymbol{C}}$*, expanding each matrix* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ *over* $\boldsymbol{S}$ *into a matrix over* $\boldsymbol{S} \times \boldsymbol{C}$ *and adding them to each other.*

More generally, if a CIM $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ *is originally conditioned on a limited set of variables* $\boldsymbol{C}$*, it can be expanded over a larger set of conditioning variables* $\boldsymbol{D}$ *where* $\boldsymbol{C} \subseteq \boldsymbol{D}$*. We simply use multiple copies of each* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_j}$ *in the following way: as we iterate over instantiations* $\boldsymbol{d}_i$ *we use the matrix* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_j}$*, where* $\boldsymbol{c}_j$ *is consistent with* $\boldsymbol{d}_i$*. More formally, to expand* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ *into a matrix over* $\boldsymbol{S} \times \boldsymbol{D}$*, where* $\boldsymbol{C} \subseteq \boldsymbol{D}$ *we let*

$$\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{D}} = \sum_{\boldsymbol{d}_i \in Val(\boldsymbol{D})} \mathbf{N}_{\boldsymbol{S}|\boldsymbol{d}_i} \mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_j} \mathbf{N}'_{\boldsymbol{S}|\boldsymbol{d}_i} \text{, where } \boldsymbol{c}_j \cong \boldsymbol{d}_i \,.$$

We note that $\boldsymbol{d}_i$ *determines a unique, consistent* $\boldsymbol{c}_j$ *since* $\boldsymbol{C} \subseteq \boldsymbol{D}$*.* ∎

It may seem that we have abused our notation by using $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ for both the single expanded matrix for $\boldsymbol{S}$ over the space $\boldsymbol{S} \times \boldsymbol{C}$ and for the set of intensity matrices $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$, where $\boldsymbol{c}_i \in Val(\boldsymbol{c})$. Especially so, since the single expanded matrix is formed by summing over the copies of the members of the set. However, we justify this dual usage by noting, in the following theorem, that we can recover the set of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ from the single expanded matrix.

Theorem 3.3.9 *Each intensity in the fully expanded matrix* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ *corresponds to a single intensity from* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ *for some* $\boldsymbol{c}_i \in Val(\boldsymbol{C})$*. Moreover, by choosing an appropriate submatrix from* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$*, we can recover* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ *for each* $\boldsymbol{c}_i \in Val(\boldsymbol{C})$*.*

Proof: The key insight to the proof of this theorem is to show that each matrix $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ is mapped into a unique submatrix of the expanded form. That means that when

adding the individually expanded matrices, it never happens that an intensity from $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ is added to an intensity from $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_j}$ for any pair $\boldsymbol{c}_i, \boldsymbol{c}_j \in \mathit{Val}(\boldsymbol{C})$. Moreover, none of the intensities within a single $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ are added to each other.

We begin by noting that for each $\boldsymbol{c}_i \in \mathit{Val}(\boldsymbol{C})$, the individual expanded matrix is computed as $\mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i}\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}\mathbf{N}'_{\boldsymbol{S}|\boldsymbol{c}_i}$. By Definition 3.3.5 (the definition of the expansion matrix), $\mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i}$ has a $\mathit{Card}(\boldsymbol{S}) \times \mathit{Card}(\boldsymbol{S})$ identity matrix as the submatrix corresponding to rows consistent with $\boldsymbol{c}_i$ (and is, otherwise, 0). Thus, multiplying $\mathbf{N}_{\boldsymbol{S}|\boldsymbol{c}_i}\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ yields a $\mathit{Card}(\boldsymbol{S} \cup \boldsymbol{C}) \times \mathit{Card}(\boldsymbol{S})$ matrix that is everywhere 0 except on the submatrix corresponding to rows consistent with $\boldsymbol{C} = \boldsymbol{c}_i$. That submatrix is an exact copy of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$. When, in turn, we right-multiply by the transpose, $\mathbf{N}'_{\boldsymbol{S}|\boldsymbol{c}_i}$, the result is a $\mathit{Card}(\boldsymbol{S} \cup \boldsymbol{C}) \times \mathit{Card}(\boldsymbol{S} \cup \boldsymbol{C})$ matrix that is everywhere zero except on the square submatrix corresponding to rows and columns consistent with $\boldsymbol{C} = \boldsymbol{c}_i$. Again, that submatrix is an exact copy of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$.

Any other intensity matrix $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_j}$ for $\boldsymbol{c}_j \neq \boldsymbol{c}_i$, is mapped to a non-overlapping submatrix — namely, the one constructed by choosing the rows and columns of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ consistent with $\boldsymbol{C} = \boldsymbol{c}_j$. ∎

Henceforth, we will generally use $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ to refer to the single, expanded matrix.

Corollary 3.3.10 *Each positive intensity of the fully expanded matrix $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ corresponds to a single transition of a single variable of $\boldsymbol{S}$ and the negative intensities of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ all fall on the main diagonal and make the row sum to zero.*

Proof: By Theorem 3.3.9, each positive intensity of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ comes directly from a positive intensity in a single $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ for some instantiation $\boldsymbol{c}_i$. But each CIM over $\boldsymbol{S}$ for a fixed $\boldsymbol{c}_i$ only has positive intensities for single transitions of single variables of $\boldsymbol{S}$. Therefore, each positive intensity of the fully expanded matrix $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ corresponds to a single transition of a single variable of $\boldsymbol{S}$.

Also, the submatrix of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ corresponding to $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ uses the set of rows and columns whose corresponding instantiations are consistent with $\boldsymbol{C} = \boldsymbol{c}_i$. Because the rows and columns use the same ordering $\xi_{\boldsymbol{S}\cup\boldsymbol{C}}$, it follows that entries of the main diagonal of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ fall on the main diagonal of $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$. Since all negative intensities of

$\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{c}_i}$ are on its main diagonal, they will be on the main diagonal of the expanded form, $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$, as well. ∎

To further ground the notion of matrix expansion, we provide a few examples.

Example 3.3.11 *Consider our variables A and B with ordering $\langle A, B\rangle$ from Example 3.3.7. Let us define the CIMs for these as*

$$\mathbf{Q}_A \qquad \mathbf{Q}_{B|a_1} \qquad \mathbf{Q}_{B|a_2}$$
$$\begin{bmatrix} -1 & 1 \\ 2 & -2 \end{bmatrix} \quad \begin{bmatrix} -5 & 2 & 3 \\ 2 & -6 & 4 \\ 2 & 5 & -7 \end{bmatrix} \quad \begin{bmatrix} -7 & 3 & 4 \\ 3 & -8 & 5 \\ 3 & 6 & -9 \end{bmatrix}.$$

The first step in expanding $\mathbf{Q}_A$ as a matrix over $A \times B$ is

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 2 & -2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}' = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

When we finish the expansion, we get

$$\mathbf{Q}_A = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix}.$$

The first step in expanding $\mathbf{Q}_{B|A}$ *is*

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -5 & 2 & 3 \\ 2 & -6 & 4 \\ 2 & 5 & -7 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}' = \begin{bmatrix} -5 & 0 & 2 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & -6 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 5 & 0 & -7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Finishing this expansion yields

$$\mathbf{Q}_{B|A} = \begin{bmatrix} -5 & 0 & 2 & 0 & 3 & 0 \\ 0 & -7 & 0 & 3 & 0 & 4 \\ 2 & 0 & -6 & 0 & 4 & 0 \\ 0 & 3 & 0 & -8 & 0 & 5 \\ 2 & 0 & 5 & 0 & -7 & 0 \\ 0 & 3 & 0 & 6 & 0 & -9 \end{bmatrix}.$$

∎

3.3.3 CIM Amalgamation

We can now define the amalgamation operation in terms of the expanded form for CIMs that we have just defined.

Definition 3.3.12 Amalgamation *is an operation which takes two CIMs* $\mathbf{Q}_{\boldsymbol{S}_1|\boldsymbol{C}_1}$, $\mathbf{Q}_{\boldsymbol{S}_2|\boldsymbol{C}_2}$, *and forms the new CIM* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ *where* $\boldsymbol{S} = \boldsymbol{S}_1 \cup \boldsymbol{S}_2$ *and* $\boldsymbol{C} = (\boldsymbol{C}_1 \cup \boldsymbol{C}_2) \setminus \boldsymbol{S}$. *Given a fixed ordering* $\xi_{\boldsymbol{S},\boldsymbol{C}}$, *we expand* $\mathbf{Q}_{\boldsymbol{S}_1|\boldsymbol{C}_1}$ *and* $\mathbf{Q}_{\boldsymbol{S}_2|\boldsymbol{C}_2}$ *into single matrices over* $\boldsymbol{S} \times \boldsymbol{C}$ *and then define the amalgamated matrix as the sum* $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}} = \mathbf{Q}_{\boldsymbol{S}_1|\boldsymbol{C}_1} + \mathbf{Q}_{\boldsymbol{S}_2|\boldsymbol{C}_2}$.

∎

Definition 3.3.13 *The* inverse of amalgamation *is computed by matrix subtraction.*

∎

Lemma 3.3.14 *The amalgamation operation satisfies the following properties:*

$$\mathbf{Q}_{\boldsymbol{S}_1|\boldsymbol{C}_1} + \mathbf{Q}_{\boldsymbol{S}_2|\boldsymbol{C}_2} = \mathbf{Q}_{\boldsymbol{S}_2|\boldsymbol{C}_2} + \mathbf{Q}_{\boldsymbol{S}_1|\boldsymbol{C}_1} \qquad \textit{(commutativity)}$$

$$(\mathbf{Q}_{\boldsymbol{S}_1|\boldsymbol{C}_1} + \mathbf{Q}_{\boldsymbol{S}_2|\boldsymbol{C}_2}) + \mathbf{Q}_{\boldsymbol{S}_3|\boldsymbol{C}_3} = \mathbf{Q}_{\boldsymbol{S}_1|\boldsymbol{C}_1} + (\mathbf{Q}_{\boldsymbol{S}_2|\boldsymbol{C}_2} + \mathbf{Q}_{\boldsymbol{S}_3|\boldsymbol{C}_3}) \qquad \textit{(associativity)}$$

Proof: This follows directly from the commutativity and associativity of matrix addition. ∎

Example 3.3.15 *Consider a CTBN $A \rightarrow B$ over the variables from Example 3.3.11. In that example, we expanded both $\mathbf{Q}_A$ and $\mathbf{Q}_{B|A}$ into a single matrices over the space $A \times B$ using the variable ordering $\langle A, B \rangle$.*

The amalgamation of these two CIMs is given by the matrix addition $\mathbf{Q}_A + \mathbf{Q}_{B|A}$ (computed using the expanded forms), producing the matrix

$$\mathbf{Q}_{AB} = \begin{bmatrix} -6 & 1 & 2 & 0 & 3 & 0 \\ 2 & -9 & 0 & 3 & 0 & 4 \\ 2 & 0 & -7 & 1 & 4 & 0 \\ 0 & 3 & 2 & -10 & 0 & 5 \\ 2 & 0 & 5 & 0 & -8 & 1 \\ 0 & 3 & 0 & 6 & 2 & -11 \end{bmatrix} .$$

∎

The use of addition for amalgamation of CIMs is very natural when we consider the following theorem which relates the sum of the intensity matrices to the product of the induced distributions.

Theorem 3.3.16 *Consider two CIMs, $\mathbf{Q}_{\boldsymbol{S}'|\boldsymbol{C}'}$ and $\mathbf{Q}_{\boldsymbol{S}''|\boldsymbol{C}''}$. The instantaneous probability over the trajectory of the joint system induced by the amalgamated matrix $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}} = \mathbf{Q}_{\boldsymbol{S}'|\boldsymbol{C}'} + \mathbf{Q}_{\boldsymbol{S}''|\boldsymbol{C}''}$ is equal to the product of the instantaneous probability induced by $\mathbf{Q}_{\boldsymbol{S}'|\boldsymbol{C}'}$ and the instantaneous probability induced by $\mathbf{Q}_{\boldsymbol{S}''|\boldsymbol{C}''}$. Formally,*

$$\lim_{\Delta t \to 0} \Pr\{\boldsymbol{S}_{t+\Delta t} \mid \boldsymbol{S}_t, \boldsymbol{C}\} = \lim_{\Delta t \to 0} \Pr\{\boldsymbol{S}'_{t+\Delta t} \mid \boldsymbol{S}'_t, \boldsymbol{C}'\} \cdot \Pr\{\boldsymbol{S}''_{t+\Delta t} \mid \boldsymbol{S}''_t, \boldsymbol{C}''\} .$$

Proof: Using Equation 3.3, one can see that it suffices to show that

$$\begin{aligned}
&\lim_{\Delta t \to 0} \mathbf{I} + \mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}\Delta t + O(\Delta t^2) \\
&\qquad = \lim_{\Delta t \to 0} \left(\mathbf{I} + \mathbf{Q}_{\boldsymbol{S}'|\boldsymbol{C}'}\Delta t + O(\Delta t^2)\right)\left(\mathbf{I} + \mathbf{Q}_{\boldsymbol{S}''|\boldsymbol{C}''}\Delta t + O(\Delta t^2)\right) \\
&\qquad = \lim_{\Delta t \to 0} \mathbf{I} + \mathbf{Q}_{\boldsymbol{S}'|\boldsymbol{C}'}\Delta t + \mathbf{Q}_{\boldsymbol{S}''|\boldsymbol{C}''}\Delta t + O(\Delta t^2) \\
&\qquad = \lim_{\Delta t \to 0} \mathbf{I} + (\mathbf{Q}_{\boldsymbol{S}'|\boldsymbol{C}'} + \mathbf{Q}_{\boldsymbol{S}''|\boldsymbol{C}''})\Delta t + O(\Delta t^2)
\end{aligned}$$

But, by definition of amalgamation, $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}} = \mathbf{Q}_{\boldsymbol{S}'|\boldsymbol{C}'} + \mathbf{Q}_{\boldsymbol{S}''|\boldsymbol{C}''}$ so the left- and right-hand sides of the equation are equal. ∎

3.3.4 Joint Intensity Matrix

We can use the amalgamation operation to define a single homogeneous Markov process that defines the dynamics of the entire system.

Definition 3.3.17 *Consider a CTBN $\mathcal{N}$ over variables $\boldsymbol{X}$. Fix an ordering $\xi_{\boldsymbol{X}}$ of the entire set of variables. The* joint intensity matrix*, $\mathbf{Q}_{\mathcal{N}}$ is a homogeneous Markov process intensity matrix defined as*

$$\mathbf{Q}_{\mathcal{N}} = \sum_{X \in \boldsymbol{X}} \mathbf{Q}_{X|\mathbf{U}_X}\,, \tag{3.4}$$

that is, the amalgamation of all the CIMs of the CTBN.

With the initial distribution, the joint intensity matrix defines $\mathcal{N}$ as a homogeneous Markov process. We can briefly characterize the matrix as follows. $\mathbf{Q}_{\mathcal{N}}$ is a square matrix over the entire joint state space — i.e., it has $Card(\boldsymbol{X}) = \prod_i Card(X_i)$ rows and columns. In general, many of the entries on each row are 0. Consider the following theorem.

Theorem 3.3.18 *Each positive intensity (off-diagonal) in the fully amalgamated matrix $\mathbf{Q}_{\mathcal{N}}$ corresponds to a transition of a single variable and comes from exactly one of the CIMs $\mathbf{Q}_{X|\mathbf{u}}$ for some instantiation $\mathbf{u}$ to $\mathbf{U}_X$, the parents of X. Each diagonal entry makes the row sum to zero.*

Proof: By definition of the joint intensity matrix (Definition 3.3.17), we have a fixed ordering $\xi_{\boldsymbol{X}}$ of the variables $\boldsymbol{X}$ of $\mathcal{N}$. Note that we can construct the joint intensity matrix by expanding each CIM $\mathbf{Q}_{X|\mathbf{U}}$ into a single intensity matrix over the entire state space, $\mathbf{Q}_{X|\boldsymbol{X}}$ and setting

$$\begin{aligned}\mathbf{Q}_{\mathcal{N}} &= \sum_{X\in\boldsymbol{X}} \mathbf{Q}_{X|\mathbf{U}_X} \\ &= \sum_{X\in\boldsymbol{X}} \sum_{\boldsymbol{c}_i\in \mathit{Val}(\boldsymbol{X}\setminus\mathbf{U}_X)} \mathbf{N}_{X|\boldsymbol{c}_i}\mathbf{Q}_{X|\mathbf{u}_j}\mathbf{N}'_{X|\boldsymbol{c}_i}\,, \text{where } \mathbf{u}_j \cong \boldsymbol{c}_i = \xi_{\boldsymbol{X}}[i]\,. \end{aligned} \tag{3.5}$$

Now, consider an entry $[m,n]$, $m \neq n$ of $\mathbf{Q}_{\mathcal{N}}$ that has a positive intensity $q_{m,n}$. This corresponds to a transition from the instantiation $\xi_{\boldsymbol{X}}[m]$ to the instantiation $\xi_{\boldsymbol{X}}[n]$. We know, from Theorem 3.3.9, that no intensities get added together in the inner sum of Eq. (3.5). So every positive intensity in the expanded form of $\mathbf{Q}_{X|\mathbf{U}_X}$ comes from a single $\mathbf{Q}_{X|\mathbf{u}_j}$ for some $\mathbf{u}_j \in \mathit{Val}(\mathbf{U}_X)$. Moreover, we know that each of these correspond to a single transition of the variable X. Since $q_{m,n}$ is positive and came from the sum in Eq. (3.5), it follows that there is at least one matrix of the form $\mathbf{N}_{X|\boldsymbol{c}_i}\mathbf{Q}_{X|\mathbf{u}_j}\mathbf{N}'_{X|\boldsymbol{c}_i}$ that contributed to $q_{m,n}$. But that implies, by Corollary 3.3.10, that $\xi_{\boldsymbol{X}}[m]$ differs from $\xi_{\boldsymbol{X}}[n]$ by only a single value of a single variable $X \in \boldsymbol{X}$. Furthermore, no other CIM $\mathbf{Q}_{X'|\mathbf{U}_{X'}}$ can contribute to $q_{m,n}$ because all of its positive intensities correspond to single transitions of the variable X'.

By Corollary 3.3.10, all the expanded matrices have the property that the diagonal entry makes the row sum to zero. This property is preserved under matrix addition since the negative intensities that are added to the main diagonal entry of each row exactly balance the sum of the additional positive intensities. ∎

Each row and column corresponds to some instantiation $\langle x_1, \ldots, x_n\rangle$, so using the above theorem we have a simple consequence.

Corollary 3.3.19 *Each row of $\mathbf{Q}_{\mathcal{N}}$ over variables $\boldsymbol{X}$ has one and only one positive intensity for every possible transition of the system and a negative intensity on the diagonal. Each row has at most $1 + \sum_i(\mathit{Card}(X_i) - 1)$ non-zero entries.*

Proof: Consider the row corresponding to some arbitrary instantiation $\langle x_1, \ldots, x_n\rangle$. According to Theorem 3.3.18, the only positive values in $\mathbf{Q}_{\mathcal{N}}$ and, hence, on the row

correspond to single transitions of single variables. For each variable X_i there are at most $Card(X_i)-1$ possible values to which it can transition. (Some of the $Card(X_i)-1$ potential transitions of X_i may happen to have a 0 intensity.) Considering all the variables and adding one for the non-zero diagonal (which is negative) leads to the above formula. ∎

3.3.5 Equivalence to Generative Semantics

In this section, we show that the two approaches we have defined for the semantics of a CTBN coincide.

Theorem 3.3.20 *The Markov process determined by the generative semantics is stochastically equivalent to the Markov process determined by the joint intensity matrix.*

Before proving this theorem, it is helpful to recall a couple of basic properties about the exponential distribution (Karlin & Taylor, 1998). Namely,

1. If T_1 and T_2 are distributed exponentially with parameters q_1 and q_2 then (a) $\Pr\{T_1 < T_2\} = \frac{T_1}{T_1+T_2}$ and $\Pr\{T_1 > T_2\} = \frac{T_2}{T_1+T_2}$. Moreover, (b) if we define $T_c = \min\{T_1, T_2\}$, then T_c is distributed exponentially with parameter $q^* = q_1 + q_2$.

2. For T_1 and T_2 as above, define $T_d = |T_1 - T_2|$. Conditioned on $T_1 > T_2$, T_d is distributed exponentially with parameter q_1 and conditioned on $T_2 > T_1$, T_d is distributed exponentiall with parameter q_2.

With these, we can prove Theorem 3.3.20.

Proof: Suppose we have a CTBN $\mathcal{N}$ over variables $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ with initial distribution $P^0_{\boldsymbol{X}}$. Let $\boldsymbol{G}$ be the Markov process defined by the generative semantics and let $\boldsymbol{A}$ be the Markov process defined by the amalgamation semantics (i.e., by $\mathbf{Q}_{\mathcal{N}}$). By Definition 2.3.5 it suffices to show that $\boldsymbol{G}$ and $\boldsymbol{A}$ have the same state space and transition probabilities. Clearly they have the same state space because

$$Val(\boldsymbol{G}) = Val(\boldsymbol{X}) = Val(\boldsymbol{A}) \ .$$

They also have the same initial distribution, $P^0_{\boldsymbol{X}}$. It remains only to show that the transition probabilities are equal.

Suppose, without loss of generality, that we start in state $\boldsymbol{x} = \langle x_1, x_2, \ldots, x_n \rangle \in \mathit{Val}(\boldsymbol{X})$. According to $\boldsymbol{G}$ we sample a time T_i for the next transition of each variable $X_i \in \boldsymbol{X}$, according the local parameter $q_{x_i|\mathbf{u}_i}$ associated with the instantiation $X_i = x_i$ and parent set $\mathbf{U}_i = \mathbf{u}_i$. We then select the earliest next transition time T_c, where $c = \arg\min_i\{T_i\}$. This process is equivalent (by Property 1(b) above) to drawing T_c from an exponential distribution with parameter

$$q^* = \sum_{X_i} q_{x_i|\mathbf{u}_i} \ .$$

Now, for process $\boldsymbol{A}$, we must look at the row of $\mathbf{Q}_{\mathcal{N}}$ corresponding to state $\boldsymbol{X} = \boldsymbol{x}$. Given that we have some fixed ordering $\xi_{\boldsymbol{X}}$, there is a row j such that $\boldsymbol{x} = \xi_{\boldsymbol{X}}[j]$. We draw the next transition time from an exponential distribution parametrized by the diagonal element of the row j. The diagonal element of that row is $-q_{j,j}$ where $q_{j,j}$ is the sum of all other intensities on that row. By Theorem 3.3.18 and Corollary 3.3.19, the diagonal element of that row is the sum of the main diagonal entry of row j for each CIM $\mathbf{Q}_{X_i|\mathbf{U}_i}$ expanded over all variables $\boldsymbol{X}$. For variable $X_i \in \boldsymbol{X}$, row j of its expanded CIM corresponds to the instantiation of parent set $\mathbf{U}_X$ consistent with the current full instantiation $\xi_{\boldsymbol{X}}[j] = \boldsymbol{x}$. Without loss of generality, we denoted this $\mathbf{u}_i$ above — i.e., for all X_i, we denoted $\mathbf{u}_i$ as the parent instantiation such that $\mathbf{u}_i \cong \xi_{\boldsymbol{X}}[j]$. Similarly, the instantiation of X_i consistent with $\boldsymbol{x} = \xi_{\boldsymbol{X}}[j]$ is denoted x_i. So, we can write the $[j, j]$ entry of the expanded CIM for X_i as $q_{x_i|\mathbf{u}_i}$ and we have

$$\begin{aligned} q_{j,j} &= \sum_{X_i} q_{x_i|\mathbf{u}_i} \\ &= q^* \, . \end{aligned}$$

Thus, the transition time in $\boldsymbol{A}$ is drawn from an exponential distribution with parameter q^* and the distribution over the transition time in $\boldsymbol{G}$ is identical.

We must now show that the distributions over where the system goes are equivalent. In $\boldsymbol{G}$, once we have determined the transition time T_c as the minimum of the potential transition times T_i for each X_i, we select X_c to transition next. In $\boldsymbol{G}$, according to Property 1(a) above, X_c is selected with probability $q_{x_c|\mathbf{u}_c}/q^*$ and we transition from $X_c = x_c$ to $X_c = x'_c$ with probability $\theta_{x_c x'_c|\mathbf{u}_c}$ which, by Equation 3.2 equals $q_{x_c x'_c|\mathbf{u}_c}/q_{x_c|\mathbf{u}_c}$. So the probability of the transition is

$$\frac{q_{x_c|\mathbf{u}_c}}{q^*} \cdot \frac{q_{x_c x'_c|\mathbf{u}_c}}{q_{x_c|\mathbf{u}_c}} = \frac{q_{x_c x'_c|\mathbf{u}_c}}{q^*} \ .$$

But this is exactly the probability of the transition from $X_c = x_c$ to $X_c = x'_c$ in $\boldsymbol{A}$.

There is one further subtlety. After transitioning in $\boldsymbol{G}$, we resample the variable X_c which transitioned and all its children but we leave the sampled times for other variables unchanged. Since the sampled times are absolute (not relative) times, it would seem that the effective sampled times of these transitions are reduced. But, according to Property 2, each non-resampled time is distributed exponentially with the parameter originally used to sample the time which yields a transition time distribution identical to $\boldsymbol{A}$. Thus $\boldsymbol{G}$ and $\boldsymbol{A}$ are stochastically equivalent. ∎

3.4 Conditional Independence

As in a Bayesian network, the graph structure of a CTBN can be viewed in two different yet closely related ways. The first is as a data structure with which we can associate parameters to define a joint distribution. The second is as a qualitative description of the independence properties of the distribution.

Definition 2.3.6, for independence between Markov processes, implies that independencies specified by the CTBN graph are between distributions over entire trajectories of its variables. For example, in our drug effect network, the joint pain is independent of taking the drug given the moment by moment concentration of the drug in the bloodstream.

To investigate these independencies, we consider a way in which we can separate groups of variables in the CTBN graph. We start by noting that variables are bound

together by CIMs. The CIM $\mathbf{Q}_{X|\mathbf{Pa}_{\mathcal{G}}(X)}$ *contains* or *connects* the variables $\{X\} \cup \mathbf{Pa}_{\mathcal{G}}(X)$.

Recall from Definition 2.2.5, that the Markov blanket of of a set of nodes $\boldsymbol{Y}$ is:

$$\boldsymbol{B}_{\boldsymbol{Y}} = \left(\bigcup_{Y \in \boldsymbol{Y}} \mathbf{Pa}_{\mathcal{G}}(Y) \cup \mathbf{Cd}_{\mathcal{G}}(Y) \cup \mathbf{CoPa}_{\mathcal{G}}(Y) \right) \setminus \boldsymbol{Y} .$$

The idea is that the Markov blanket $\boldsymbol{B}_{\boldsymbol{Y}}$ separates the variables $\boldsymbol{Y}$ from the from the remaining variables of $\mathcal{N}$ in the moralized graph (see Definition 2.2.4). It turns out that the characterization of conditional independencies encoded by the graph of the CTBN are similar because we still have that separation in the moralized graph implies conditional independence.

Theorem 3.4.1 *For a CTBN $\mathcal{N}$ with graph $\mathcal{G}$ over variables $\boldsymbol{X}$, consider three groups of disjoint variables $\boldsymbol{Y}$, $\boldsymbol{V}$, and $\boldsymbol{W}$. Now, suppose that $\boldsymbol{V}$ separates $\boldsymbol{Y}$ from $\boldsymbol{W}$ in the moralized graph $\mathcal{G}^M$. Then $(\boldsymbol{Y} \perp \boldsymbol{W} \mid \boldsymbol{V})$, i.e., $\boldsymbol{Y}$ is conditionally independent of $\boldsymbol{W}$ given $\boldsymbol{V}$.*

Proof: Suppose we are given the entire trajectory for each variable $V \in \boldsymbol{V}$ and that we have a fixed ordering $\xi_{\boldsymbol{X}}$ over all variables $\boldsymbol{X}$ of $\mathcal{N}$. Consider the separately amalgamated CIMs

$$\begin{aligned}
\mathbf{Q}_{\boldsymbol{Y}|\mathbf{Pa}_{\mathcal{G}}(\boldsymbol{Y})} &= \sum_{Y \in \boldsymbol{Y}} \mathbf{Q}_{Y|\mathbf{Pa}_{\mathcal{G}}(Y)} , \\
\mathbf{Q}_{\boldsymbol{V}|\mathbf{Pa}_{\mathcal{G}}(\boldsymbol{V})} &= \sum_{V \in \boldsymbol{V}} \mathbf{Q}_{B|\mathbf{Pa}_{\mathcal{G}}(V)} , \\
\mathbf{Q}_{\boldsymbol{W}|\mathbf{Pa}_{\mathcal{G}}(\boldsymbol{W})} &= \sum_{W \in \boldsymbol{W}} \mathbf{Q}_{W|\mathbf{Pa}_{\mathcal{G}}(W)} ,
\end{aligned}$$

and note that because $\boldsymbol{V}$ separates $\boldsymbol{Y}$ and $\boldsymbol{W}$ in $\mathcal{G}^M$ there is no variable contained in $\mathbf{Q}_{\boldsymbol{W}|\mathbf{Pa}_{\mathcal{G}}(\boldsymbol{W})}$ that is contained in $\mathbf{Q}_{\boldsymbol{Y}|\mathbf{Pa}_{\mathcal{G}}(\boldsymbol{Y})}$. In other words, according to $\mathbf{Q}_{\boldsymbol{Y}|\mathbf{Pa}_{\mathcal{G}}(\boldsymbol{Y})}$, no change in the instantiation of $\boldsymbol{W}$ affects the dynamics of $\boldsymbol{Y}$ and, according to $\mathbf{Q}_{\boldsymbol{W}|\mathbf{Pa}_{\mathcal{G}}(\boldsymbol{W})}$, no change in the instantiation of $\boldsymbol{Y}$ affects the dynamics of $\boldsymbol{W}$.

Now, consider the probability density over the trajectory of the joint process $p(\boldsymbol{Y}, \boldsymbol{V}, \boldsymbol{W})$. From the analysis above, we can see that it factors into two pieces.

Hence, we can write

$$p(\boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{V}) = \frac{1}{Z} p_{\boldsymbol{YV}}(\boldsymbol{Y}, \boldsymbol{V}) \cdot p_{\boldsymbol{VW}}(\boldsymbol{V}, \boldsymbol{W}) \,.$$

where Z is the normalization factor. But, if we are given full information about the trajectory of variables $\boldsymbol{V}$, then at every moment in time, we have an instantiation $\boldsymbol{v}$ of $\boldsymbol{V}$. But this means we can write the density over the trajectory of the joint process as

$$\begin{aligned} p(\boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{v}) &= \frac{1}{Z'} p_{\boldsymbol{YV}}(\boldsymbol{Y}, \boldsymbol{v}) \cdot p_{\boldsymbol{VW}}(\boldsymbol{v}, \boldsymbol{W}) \\ &= \frac{1}{Z'} p_{\boldsymbol{Yv}}(\boldsymbol{Y}) \cdot p_{\boldsymbol{Wv}}(\boldsymbol{W}) \,. \end{aligned}$$

But that means that, under the assumption we are given $\boldsymbol{V}$, the trajectory over $\boldsymbol{Y}$ and the trajectory over $\boldsymbol{W}$ are independent. ∎

3.5 Representational Ability

We begin by considering the scope of the CTBN representation: Which underlying distributions can we represent using a CTBN? Recall (from Definition 2.3.5) that two Markov processes are *stochastically equivalent* if they have the same state space and transition probabilities. Note that we need the initial distribution for a complete description of the process, but we are here interested in the distribution over the evolution of the process as governed by the intensity matrix. Since any initial distribution can be represented by a CTBN, we can always assume that the CTBN exists where $P^0_{\mathcal{N}} = P^0_{\boldsymbol{X}}$. So the initial distribution will not put any limitations on the set of Markov processes we can represent. As a result, we allow ourselves to refer to an intensity matrix as if it were a Markov process.

Now, consider a homogeneous stochastic process over $\mathit{Val}(\boldsymbol{X})$, defined with an intensity matrix $\mathbf{Q}_{\boldsymbol{X}}$. We would like to determine when there is a CTBN which is stochastically equivalent to $\mathbf{Q}_{\boldsymbol{X}}$.

In section 3.3, we provided a semantics for a CTBN in terms of an *amalgamation*

operation, which takes a CTBN and converts it into a single intensity matrix that specifies a homogeneous stochastic process. For a CTBN $\mathcal{N}$, let $\mathbf{Q}_{\mathcal{N}}$ be the induced joint intensity matrix. We can now define

Definition 3.5.1 *A CTBN structure $\mathcal{G}$ is an* S-map *for a homogeneous Markov process $\mathbf{Q}_{\boldsymbol{X}}$ if there exists a CTBN $\mathcal{N}$ over the graph $\mathcal{G}$ such that $\mathbf{Q}_{\mathcal{N}}$ is stochastically equivalent to $\mathbf{Q}_{\boldsymbol{X}}$.*

As discussed in section 3.3, a basic assumption in the semantics of CTBNs is that, as time is continuous, variables cannot transition at the same instant. Thus, in the joint intensity matrix, all intensities that correspond to two simultaneous changes are zero. More precisely:

Definition 3.5.2 *A homogeneous Markov process whose intensity matrix is $\mathbf{Q}_{\boldsymbol{X}}$ with entries $q_{\boldsymbol{x}\boldsymbol{x}'}$ is said to be* variable-based *if, for any two assignments $\boldsymbol{x}$ and $\boldsymbol{x}'$ to $\boldsymbol{X}$ that differ on more than one variable, $q_{\boldsymbol{x}\boldsymbol{x}'} = 0$.*

It turns out that this condition is the only restriction on the CTBN expressive power. Let $\mathcal{G}^{\top}$ be the fully connected directed graph. Then we can show that

Theorem 3.5.3 *The graph $\mathcal{G}^{\top}$ is an S-map for any variable-based homogeneous Markov process with intensity matrix $\mathbf{Q}_{\boldsymbol{X}}$.*

Proof: Fix any arbitrary variable-based homogeneous Markov process with intensity matrix $\mathbf{Q}_{\boldsymbol{X}}$ over variables $\boldsymbol{X}$. It is sufficient to construct a CTBN $\mathcal{N}$ over $\mathcal{G}^{\top}$ such that $\mathbf{Q}_{\mathcal{N}} = \mathbf{Q}_{\boldsymbol{X}}$. Consider variable $X \in \boldsymbol{X}$ and let $\mathbf{Q}_{X \in \boldsymbol{X}}$ be the result of setting intensities for transitions of all variables other than X to zero and increasing the diagonal entry of each row by the total of the zeroed intensities on that row (thus decreasing the magnitude of the diagonal and preserving the row sums). Since $\mathbf{Q}_{\boldsymbol{X}}$ is variable-based, we do not need to set to zero any intensities corresponding to multiple transitions. Thus

$$\mathbf{Q}_{\boldsymbol{X}} = \sum_{X \in \boldsymbol{X}} \mathbf{Q}_{X \in \boldsymbol{X}} \ .$$

Now we define a CTBN $\mathcal{N}$ with parent sets $\mathbf{U}_X = \boldsymbol{X} \setminus \{X\}$ so that it has the graph $\mathcal{G}^{\top}$. Next we set each parameter $q_{xx'|\mathbf{u}_x}$ as intensity from $\mathbf{Q}_{X \in \boldsymbol{X}}$ corresponding to

the transition from $X = x$ to $X = x'$ with the rest of the variables instantiated $\mathbf{U}_X = \mathbf{u}_X$. Then, by construction, we have that the expanded CIM $\mathbf{Q}_{X|\mathbf{U}_X} = \mathbf{Q}_{X \in \boldsymbol{X}}$ which means that

$$\mathbf{Q}_{\boldsymbol{X}} = \sum_{X \in \boldsymbol{X}} \mathbf{Q}_{X|\mathbf{U}_X} = \mathbf{Q}_{\mathcal{N}}$$

and our construction is complete. ∎

Thus, we can represent every variable-based homogeneous process as some parameterization over the graph $\mathcal{G}^\top$. In fact, this parameterization is unique:

Theorem 3.5.4 *Let $\mathcal{N}$ and $\mathcal{N}'$ be two CTBNs over variables $\boldsymbol{X}$ with graph $\mathcal{G}^\top$. Then $\mathbf{Q}_{\mathcal{N}}$ and $\mathbf{Q}_{\mathcal{N}'}$ are stochastically equivalent if and only if their conditional intensity matrices are identical.*

Proof: Suppose $\mathcal{N}$ and $\mathcal{N}'$ have identical CIMs so $\mathbf{Q}_{X|\mathbf{U}_X} = \mathbf{Q}'_{X|\mathbf{U}_X}$ for all $X \in \boldsymbol{X}$. It is sufficient to show that $\mathbf{Q}_{\mathcal{N}} = \mathbf{Q}_{\mathcal{N}'}$ because then the transition probabilities must be identical. But, we have

$$\mathbf{Q}_{\mathcal{N}} = \sum_{X \in \boldsymbol{X}} \mathbf{Q}_{X|\mathbf{U}_X} = \sum_{X \in \boldsymbol{X}} \mathbf{Q}'_{X|\mathbf{U}_X} = \mathbf{Q}_{\mathcal{N}'} \ .$$

Conversely, suppose $\mathbf{Q}_{\mathcal{N}}$ and $\mathbf{Q}_{\mathcal{N}'}$ are stochastically equivalent and hence have identical transition probabilities. Since each entry governs a particular transition probability, we have $\mathbf{Q}_{\mathcal{N}} = \mathbf{Q}_{\mathcal{N}'}$. Suppose their conditional intensity matrices are not identical. By Theorem 3.3.18, all CIMs have a zero intensity for multiple simultaneous transitions and the diagonals simply make the rows sum to zero. So there must be some entry $q_{xx'|\mathbf{u}_X}$ (where $x \neq x'$) in the CIM for X in $\mathcal{N}$ that is different than $q'_{xx'|\mathbf{u}_X}$, the corresponding entry in the CIM for X in $\mathcal{N}'$. But the CIM over X is the only source for the intensity (and hence probability) of the transition from $X = x$ to $X = x'$ which means that $q_{xx'|\mathbf{u}_X}$ is the entry in $\mathbf{Q}_{\mathcal{N}}$ corresponding to that transition and $q'_{xx'|\mathbf{u}_X}$ is the corresponding entry in $\mathbf{Q}_{\mathcal{N}'}$. But this is a contradiction, so the conditional intensity matrices must be identical. ∎

Let $\mathcal{N}_{\mathbf{Q}_{\boldsymbol{X}}}$ represent the unique CTBN over $\mathcal{G}^\top$ which is stochastically equivalent to $\mathbf{Q}_{\boldsymbol{X}}$. Although capturing a stochastic process using a fully-connected CTBN is not very interesting, it provides us with the tools for proving our main result.

Theorem 3.5.5 *A CTBN structure $\mathcal{G}$ is an S-map for a variable-based process $\mathbf{Q}_{\boldsymbol{X}}$ if and only if $\mathcal{N}_{\mathbf{Q}_{\boldsymbol{X}}}$ satisfies the following condition:*

For every variable X, consider a partition of $\boldsymbol{X}$ into X, $\boldsymbol{Pa}_{\mathcal{G}}(X) = \mathbf{U}$ and remaining variables $\boldsymbol{W}$. Denote $x\mathbf{u}\boldsymbol{w}$ as the instantiation to $\boldsymbol{X}$ that sets $X = x$, $\mathbf{U} = \mathbf{u}$, $\boldsymbol{W} = \boldsymbol{w}$. Now consider four assignments to $\boldsymbol{X}$: $x\mathbf{u}\boldsymbol{w}$, $x'\mathbf{u}\boldsymbol{w}$, $x\mathbf{u}\boldsymbol{w}'$, and $x'\mathbf{u}\boldsymbol{w}'$. Then $q_{x\mathbf{u}\boldsymbol{w}\to x'\mathbf{u}\boldsymbol{w}} = q_{x\mathbf{u}\boldsymbol{w}'\to x'\mathbf{u}\boldsymbol{w}'}$.

Proof: Note that $q_{x\mathbf{u}\boldsymbol{w}\to x'\mathbf{u}\boldsymbol{w}}$ and $q_{x\mathbf{u}\boldsymbol{w}'\to x'\mathbf{u}\boldsymbol{w}'}$ both represent the same transition single-variable transition from $X = x$ to $X = x'$ under a fixed parent instantiation $\mathbf{u}$ but under (possibly) differet instantiations to the other variables $\boldsymbol{W}$. They are only guaranteed to overlap on the instantiation of the variables $\mathbf{Pa}_{\mathcal{G}}(X)$. According to the condition, this is enough to guarantee that the transition intensities are identical. Thus the condition is equivalent to saying that in $\mathcal{N}_{\mathbf{Q}_X}$, the CIM $\mathbf{Q}_{X|\mathbf{U}_X}$ over $\boldsymbol{X} = \{X\} \cup \mathbf{U}_X$ consists of copies of $\mathbf{Q}_{X|\mathbf{Pa}_{\mathcal{G}}(X)}$ — i.e., it is equivalent to a CIM $\mathbf{Q}_{X|\mathbf{Pa}_{\mathcal{G}}(X)}$ expanded to be over $\boldsymbol{X}$.

Now, if $\mathcal{G}$ is an S-map of $\mathbf{Q}_{\boldsymbol{X}}$ then, by definition, there is a CTBN $\mathcal{N}$ with CIMs $\mathbf{Q}_{X|\mathbf{Pa}_{\mathcal{G}}(X)}$ such that $\mathbf{Q}_{\mathcal{N}} = \sum_X \mathbf{Q}_{X|\mathbf{Pa}_{\mathcal{G}}(X)}$. Then clearly $\mathcal{N}_{\mathbf{Q}_X}$ satisfies the condition because $\mathbf{Q}_{X|\mathbf{U}_X}$ is equal to the expanded $\mathbf{Q}_{X|\mathbf{Pa}_{\mathcal{G}}(X)}$ for every X.

Conversely, suppose $\mathcal{N}_{\mathbf{Q}_X}$ satisfies the condition. Then, as noted according to the equivalent condition above, for every X, $\mathbf{Q}_{X|\mathbf{U}_X}$ is equivalent to a CIM $\mathbf{Q}_{X|\mathbf{Pa}_{\mathcal{G}}(X)}$ so we can define a CTBN $\mathcal{N}$ with structure $\mathcal{G}$ with these CIMs and it follows that $\mathcal{G}$ is an S-map of $\mathbf{Q}_{\boldsymbol{X}}$. ∎

Thus, we cannot represent the same process using two fundamentally different CTBN structures. We can only add spurious edges, corresponding to vacuous dependencies.

Definition 3.5.6 *In a CTBN $\mathcal{N}$ with graph $\mathcal{G}$, an arc $A \to B$ is* spurious *under the following condition. Let $\boldsymbol{Pa}_{\mathcal{G}}(B) = \{A\} \cup \boldsymbol{W}$. Then, for all $a, a' \in Val(A)$ and some fixed instantiation $\boldsymbol{w}$ to the other parents $\boldsymbol{W}$, $\mathbf{Q}_{B|\boldsymbol{w},a} = \mathbf{Q}_{B|\boldsymbol{w},a'}$.*

Theorem 3.5.7 *For any variable-based process $\mathbf{Q}_{\boldsymbol{X}}$, there exists an S-map $\mathcal{G}^*$ such that, for any S-map $\mathcal{G}$ for $\mathbf{Q}_{\boldsymbol{X}}$, $\mathcal{G}^* \subseteq \mathcal{G}$.*

Proof: Consider any two structures $\mathcal{G}$ and $\mathcal{G}'$ which are S-maps of $\mathbf{Q}_{\boldsymbol{X}}$. Then we have corresponding CTBNs $\mathcal{N}$ and $\mathcal{N}'$ such that $\mathbf{Q}_{\mathcal{N}}$ and $\mathbf{Q}_{\mathcal{N}'}$ are stochastically equivalent to $\mathbf{Q}_{\boldsymbol{X}}$. We first modify $\mathcal{N}$ by adding spurious edges to $\mathcal{G}$ until we have a fully connected graph. We similarly modify $\mathcal{N}'$ by adding spurious edges to $\mathcal{G}'$ until we have a fully connected graph. But the resulting CTBNs $\mathcal{N}$ and $\mathcal{N}'$ must be identical to $\mathcal{N}_{\mathbf{Q}_X}$ by Theorem 3.5.4. But that means that the CIMs of $\mathcal{N}$ and $\mathcal{N}'$ must be identical. Since the definition of a spurious edge (Definition 3.5.6) is in terms of a condition on the CIMs, any spurious edges in the modified $\mathcal{N}$ are also spurious in the modified $\mathcal{N}'$. That means that any spurious edges we added to $\mathcal{G}$ that were already in $\mathcal{G}'$ were already spurious in $\mathcal{G}'$. Likewise, any spurious edges we added to $\mathcal{G}'$ that were already in $\mathcal{G}$ were already spurious in $\mathcal{G}$.

Let us define $\mathcal{G}_\cap$ as the graph containing all and only those edges that are in $\mathcal{G}$ and in $\mathcal{G}'$ — that is, the intersection of the set of edges in the two graphs. Since $\mathcal{G}_\cap$ can be constructed by deleting only spurious edges from $\mathcal{G}$ (or by deleting only spurious edges from $\mathcal{G}'$), it follows that $\mathcal{G}_\cap$ is an S-map of $\mathbf{Q}_{\boldsymbol{X}}$. Furthermore $\mathcal{G}_\cap \subseteq \mathcal{G}$ and $\mathcal{G}_\cap \subseteq \mathcal{G}'$. Since there are only finitely many possible structures, this is enough to guarantee the existence of a unique $\mathcal{G}^*$. ∎

Thus, we have a unique minimal S-map.

3.6 Discussion

In this chapter, we have defined the CTBN framework — showing how to construct a continuous time Markov process over a factored state space analogous to a Bayesian network.

We provided two alternative semantics for this process and showed their equivalence. We provided a characterization of the joint intensity matrix for the CTBN in terms of the local CIM probability models.

We explored issues of conditional independence in a CTBN, showing that the Markov blanket of a set of variables renders them conditionally independent of the remaining variables.

Our final result of this chapter shows that we cannot represent the same process

using two essentially different CTBN structures — we have a unique minimal S-map. Let us compare this result to the case of Bayesian networks. There, any distribution has many minimal I-maps; indeed, many distributions even have several perfect maps, each of which captures the structure of the distribution perfectly. In the case of CTBNs, we have a unique minimal S-map. To obtain some intuition for this difference, consider the simple example of a two-variable CTBN $\mathcal{N}$ with the graph $X \rightarrow Y$. Unless the edge between X and Y is vacuous, this graph cannot give rise to the same transition probabilities as any CTBN $\mathcal{N}'$ with the graph $X \leftarrow Y$. To see that, recall that in $\mathcal{N}$, the variable Y is an inhomogeneous Markov process whose transition probabilities vary over time as a function of the changing value of X. But, in $\mathcal{N}'$, the variable Y is a homogeneous Markov process whose transition probabilities never change.

Further implications of these definitions and results will be explored in depth over the rest of this thesis.

Chapter 4

Likelihood and Sufficient Statistics

In the last chapter we defined CTBNs. As we look towards the next chapters which will cover the task of learning CTBNs from data, we must first answer a few basic questions. What do we mean by data? What is the probability or *likelihood* of some set of data, given a CTBN model? If our data is, in some sense, incomplete how do we calculate the *expected* likelihood? These are the questions explored in this chapter.

Since CTBNs are a member of the exponential family of models, the likelihood of a data set can be expressed in terms of *sufficient statistics* aggregated over the data. We will show the form these statistics have and that, just as with BNs, the likelihood of a CTBN decomposes as an aggreggate of local likelihoods.

We also provide a method for calculating expected sufficient statistics and discuss computational issues associated with that calculation.

4.1 Data

At its core, a CTBN models the joint trajectories of its variables.

4.1.1 Complete Data

In order to have complete data we need a set of one or more full trajectories of all the variables of the CTBN. So, if $\mathcal{D}$ represents a complete data set, then $\mathcal{D} = \{\sigma_1, \ldots, \sigma_h\}$

where each σ_i is a complete set of state transitions and the times at which they occurred. In other words, for each point in time of each trajectory, we know the full instantiation to all variables.

4.1.2 Incomplete Data

Given the description of complete data, we can see that an incomplete data set is a set of one or more *partial* trajectories.

Example 4.1.1 *Suppose we have a process X whose state space is*

$$Val(X) = \{y_1, y_2\} \times \{z_1, z_2\}.$$

Consider the following example σ^+ of a fully observed trajectory over the time interval $[0, 2)$: X starts in $\langle y_1, z_2\rangle$ at time 0; at time 0.5 it transitions to $\langle y_2, z_2\rangle$; at time 1.7 it transitions to $\langle y_2, z_1\rangle$.

Note we can write $\langle \cdot, z_i\rangle$ for the subsystem (see Section 2.3.4) consisting of the states $\langle y_1, z_i\rangle$ and $\langle y_2, z_i\rangle$. An example partially observed trajectory σ over the interval $[0, 2)$ is: X starts in $\langle \cdot, z_2\rangle$; at time 1.7 it transitions to $\langle \cdot, z_1\rangle$. Note that σ^+ is a completion of σ. Another possible completion of σ is: X starts in $\langle y_2, z_2\rangle$ at time 0; at time 1.0 it transitions to $\langle y_1, z_2\rangle$; at time 1.7 it transitions to $\langle y_1, z_1\rangle$.

We can describe a partially observed trajectory σ' with point evidence at time 0.7 and 1.8: X starts in $\langle \cdot, \cdot\rangle$; at time 0.7 we observe $\langle \cdot, z_2\rangle$; from time 0.7 to 1.8 we observe X in $\langle \cdot, \cdot\rangle$; at time 1.8 we observe X in $\langle \cdot, z_1\rangle$; from time 1.8 on, we observe X in $\langle \cdot, \cdot\rangle$. Note that σ^+ is also a completion of σ'.

So, for a general Markov process, a partially observed trajectory σ is given as a sequence of N subsystems (see Section 2.3.4) so that the state is restricted to subsystem S_i during interval $[t_i, t_{i+1})$ for $0 \leq i \leq (N-1)$. Without loss of generality, we can assume that σ begins at time 0 and ends at time τ so $t_0 = 0$ and $t_N = \tau$.

For subsystem S, let $\mathbf{Q}_S$ be the $n \times n$ intensity matrix $\mathbf{Q}_X$ with all intensities zeroed out except those corresponding to transitions within the subsystem S (and associated diagonal elements). For subsystems S_1, S_2, let $\mathbf{Q}_{S_1 S_2}$ be the $n \times n$ intensity

matrix $\mathbf{Q}_X$ with all intensities zeroed out except those corresponding to transitions from S_1 to S_2. Note that this means all the intensities corresponding to transitions within S_1 and within S_2 are also zeroed out.

Sometimes it is convenient to refer to evidence by providing an arbitrary time interval. Let $\sigma_{t_1:t_2}$ denote the evidence provided by σ over the interval $[t_1, t_2)$. (So, $\sigma_{0:\tau}$ is the evidence provided by all of σ.) Let $\sigma_{t_1:t_2^+}$ denote the evidence over the interval $[t_1, t_2]$, and $\sigma_{t_1^+:t_2}$ the evidence over the interval (t_1, t_2).

4.2 Single Markov Process

We begin by considering the likelihood of *complete*, or fully observed, data. Consider a homogeneous Markov process $X(t)$. As all the transitions are observed, the likelihood of $\mathcal{D}$ can be decomposed as a product of the likelihoods for individual transitions d. Let $d = \langle x_d, t_d, x'_d \rangle \in \mathcal{D}$ be the transition where X transitions to state x'_d after spending the amount of time t_d in state x_d. Using the *mixed intensity parameterization*, we can write the likelihood for the single transition d as

$$L_X(\boldsymbol{q}, \boldsymbol{\theta} : d) = L_X(\boldsymbol{q} : d) L_X(\boldsymbol{\theta} : d) \tag{4.1}$$

$$= \left(q_{x_d} \exp(-q_{x_d} t_d)\right) \left(\theta_{x_d x'_d}\right). \tag{4.2}$$

By multiplying the likelihoods for each transition d, we see that we can summarize our data $\mathcal{D}$ in terms of sufficient statistics.

Definition 4.2.1 *The* sufficient statistics *for the transition dynamics of a single Markov process X are*

- *$T[x]$, the amount of time spent in each state $x \in Val(X)$, and*
- *$M[x, x']$, the number of transitions from x to x', where $x \neq x'$.*

∎

If we write $M[x] = \sum_{x'} M[x, x']$, the total number of transitions leaving the state $X = x$ then we have

$$\begin{aligned} L_X(\boldsymbol{q}, \boldsymbol{\theta} : \mathcal{D}) &= \left(\prod_{d \in \mathcal{D}} L_X(\boldsymbol{q} : d) \right) \left(\prod_{d \in \mathcal{D}} L_X(\boldsymbol{\theta} : d) \right) \\ &= \left(\prod_x q_x^{M[x]} \exp(-q_x T[x]) \right) \left(\prod_x \prod_{x' \neq x} \theta_{xx'}^{M[x,x']} \right). \end{aligned} \tag{4.3}$$

4.3 Exponential Family

We can write the sufficient statistics of our data $\mathcal{D}$ as a vector $\boldsymbol{v}_{s[\mathcal{D}]} = \{T[x], M[x, x']\}$ consisting of the the amount of time spent in each state x followed by the number of transitions from every state x to every other x' where $x, x' \in \mathit{Val}(X)$.

Similarly, using the *pure intensity parameterization*, we can write a vector of the parameters of $\mathbf{Q}$, the intensity matrix, $\boldsymbol{v}_p = \{q_x, ln(q_{xx'})\}$ consisting of the intensity of leaving each state x followed by the log of the intensity of transitioning from state x to state x'.

Then the likelihood of the data can then be written as the dot product of the two vectors

$$\begin{aligned} P(\sigma) &= \frac{1}{Z} \exp(\langle \boldsymbol{v}_{s[\mathcal{D}]}, \boldsymbol{v}_p \rangle) \\ &= \frac{1}{Z} \exp\left(\sum_x q_x T[x] + \sum_x \sum_{x' \neq x} \ln q_{xx'} M[x, x'] \right) , \end{aligned}$$

where Z is the partition function.

4.4 Computing Expected Sufficient Statistics

In many situations we do not have access to continuous observation of every part of the system. If there are gaps in our observation of the system, then we are unable to compute the sufficient statistics — we do not know how many transitions may have occurred, nor do we know when they occurred. So, we are unable to compute $T[x]$

or $M[x, x']$. But we do have a model, which gives us a distribution over trajectories. Hence, we compute *expected* sufficient statistics: the expected amount of time we spend in each state, $\mathbf{E}[T[x]]$, and the expected number of transitions from one state to another, $\mathbf{E}[M[x, x']]$.

So, given an n-state homogeneous Markov process X_t with intensity matrix $\mathbf{Q}_X$, our task is to compute the expected sufficient statistics with respect to the posterior probability density over completions of the data given the observations and the current model. For simplicity, we omit the explicit dependence on the parameters $\boldsymbol{q}^k, \boldsymbol{\theta}^k$.

This computation is not straightforward. The space of trajectories is infinite in the times at which an unknown transition may have occurred. Moreover, we do not even know how many missing transitions there are.

4.4.1 Notation

In order to compute the expected sufficient statistics over $\mathcal{D}$, we compute them for each partially observed trajectory $\sigma \in \mathcal{D}$ separately and then combine the results.

Because we will be dealing with a number of matrix operations, it will be convenient to use row numbers when discussing states. So, assuming the $\mathit{Val}(X) = \{x_1, \ldots, x_n\}$, we will, for purposes of the discussion in this section, denote $x_i = \xi_X[i]$ simply by i. Likewise, we denote $x_j = \xi_X[j]$ simply as j.

Let $\mathbf{e}$ be a (column) n-vector of ones. Let $\mathbf{e}_j$ be an n-vector of zeros with a one in position j. Let $\boldsymbol{\Delta}_{j,k}$ be an $n \times n$ matrix of zeros with a one in position j, k. (So $\boldsymbol{\Delta}_{j,k} = \mathbf{e}_j \mathbf{e}_k'$.) Note that all multiplications below are standard vector and matrix multiplications as opposed to factor multiplications.

Define the vectors $\boldsymbol{\alpha}_t^-$ and $\boldsymbol{\beta}_t^+$ component-wise as

$$\boldsymbol{\alpha}_t^-[i] = p(X_{t^-} = i, \sigma_{0:t})$$
$$\boldsymbol{\beta}_t^+[i] = p(\sigma_{t^+:\tau} \mid X_{t^+} = i),$$

where, if X transitions at t, X_{t^-} is the value of X just prior to the transition, and X_{t^+} the value just afterward. (If there is no transition, $X_{t^-} = X_{t^+}$.) Moreover, recall that $\sigma_{0:t}$ represents the evidence over interval $[0, t)$ not including t and $\sigma_{t^+:\tau}$ represents

evidence over interval (t, τ) not including t. Thus, neither of these vectors include evidence of a transition at time t. We also define the vectors

$$\begin{aligned}\boldsymbol{\alpha}_t[i] &= p(X_t = i, \sigma_{0:t^+}) \\ \boldsymbol{\beta}_t[i] &= p(\sigma_{t:\tau} \mid X_t = i)\end{aligned}$$

both of which include evidence of any transition at time t.

4.4.2 Expected Amount of Time

The sufficient statistic $T[j]$ is the amount of time that X spends in state j over the course of trajectory σ. We can write the expectation of $T[j]$ according to the posterior probability density given the evidence as

$$\begin{aligned}\mathbf{E}[T[j]] &= \int_0^\tau p(X_t \mid \sigma_{0:\tau})\mathbf{e}_j dt \\ &= \sum_{i=0}^{N-1} \int_{t_i}^{t_{i+1}} p(X_t \mid \sigma_{0:\tau})\mathbf{e}_j dt \\ &= \frac{1}{p(\sigma_{0:\tau})} \sum_{i=0}^{N-1} \int_{t_i}^{t_{i+1}} p(X_t, \sigma_{0:\tau})\mathbf{e}_j dt \ .\end{aligned}$$

The constant fraction at the beginning of the last line serves to make the total expected time over all j sum to τ.

We must show how to compute the above integrals over intervals of constant evidence. Let $[v, w)$ be such an interval, and let S be the subsystem to which the state is restricted on this interval. Then we have

$$\begin{aligned}\int_v^w p(X_t, \sigma_{0:\tau})\mathbf{e}_j dt &= \int_v^w p(X_t, \sigma_{0:t})\boldsymbol{\Delta}_{j,j} p(\sigma_{t:\tau} \mid X_t) dt \\ &= \int_v^w \boldsymbol{\alpha}_v p(X_t, \sigma_{v:t} \mid X_v)\boldsymbol{\Delta}_{j,j} p(X_w, \sigma_{t:w} \mid X_t)\boldsymbol{\beta}_w dt \\ &= \int_v^w \boldsymbol{\alpha}_v \exp(\mathbf{Q}_S(t-v))\boldsymbol{\Delta}_{j,j} \exp(\mathbf{Q}_S(w-t))\boldsymbol{\beta}_w dt \,. \qquad (4.4)\end{aligned}$$

We can calculate the total expected time, $\mathbf{E}[T[j]]$, by summing the above expression over all intervals of constant evidence. The numerical integration is done via the Runge-Kutta method which is described in Section 4.4.4.

4.4.3 Expected Number of Transitions

The sufficient statistic $M[j,k]$ $(j \neq k)$ is the number of times X transitions from state j to state k over the course of trajectory σ. We consider discrete time approximations of $M[j,k]$ and take the limit as the size of our discretization goes to zero, yielding an exact equation. For $\epsilon > 0$, let

$$M_\epsilon[j,k] = \sum_{t=0}^{\tau/\epsilon-1} \mathbf{1}\{X_{t\epsilon} = j, X_{(t+1)\epsilon} = k\} \ .$$

Note that our discrete approximation is dominated by the actual value, i.e.,

$$|M_\epsilon[j,k]| \leq M[j,k] \, ,$$

and also that as $\epsilon \downarrow 0$, $M_\epsilon[j,k] \to M[j,k]$. Hence, by dominated convergence for conditional expectations (Billingsley, 1995), we have

$$\mathbf{E}[M[j,k]] = \lim_{\epsilon \downarrow 0} \mathbf{E}[M_\epsilon[j,k]] \ .$$

This last expectation can be broken down as

$$\mathbf{E}[M_\epsilon[j,k]] = \sum_{t=0}^{\tau/\epsilon-1} \frac{p(X_{t\epsilon} = j, X_{(t+1)\epsilon} = k, \sigma_{0:\tau})}{p(\sigma_{0:\tau})} \ .$$

Note that, for ϵ small enough, we observe at most one transition per interval. Thus, each of the intervals in the sum falls into one of two categories: either the interval contains a (partially observed) transition, or the evidence is constant over the interval. We treat each of these cases separately.

Let $[t\epsilon, (t+1)\epsilon)$ be an interval containing a partially observed transition at time

t_i. We observe only that we are transitioning from one of the states of S_i to one of the states of S_{i+1}. We can calculate the contribution of this interval to the expected sufficient statistics (ignoring the constant $1/p(\sigma_{0:\tau})$) as

$$\begin{aligned}
&p(X_{t\epsilon} = j, X_{(t+1)\epsilon} = k, \sigma_{0:\tau}) \\
&\quad = p(X_{t\epsilon} = j, \sigma_{0:t\epsilon}) p(X_{(t+1)\epsilon} = k, \sigma_{t\epsilon:(t+1)\epsilon} \mid X_{t\epsilon} = j) p(\sigma_{(t+1)\epsilon:\tau} \mid X_{(t+1)\epsilon} = k) \ .
\end{aligned}$$

As $\epsilon \downarrow 0$, we have the probability of the state and the evidence up to, but not including, time t_i, times the instantaneous probability of transitioning from state j to state k, times the probability of the evidence given the state just after t_i. Thus, at the limit, this transition's contribution is

$$\boldsymbol{\alpha}^-_{t_i} \mathbf{e}_j q_{jk} \mathbf{e}'_k \boldsymbol{\beta}^+_{t_i} = q_{jk} \boldsymbol{\alpha}^-_{t_i} \boldsymbol{\Delta}_{j,k} \boldsymbol{\beta}^+_{t_i} \ . \tag{4.5}$$

Now consider the case when we are within an interval $[v, w) = [t_i, t_{i+1})$ of constant evidence — i.e., it does not contain a partially observed transition and will generally be of length much larger than ϵ. Let $\Delta t = w - v$ and let S be the subsystem to which the state is restricted on this interval. As ϵ grows small, the contribution of this interval to the sum (again, ignoring $1/p(\sigma_{0:\tau})$) is

$$\begin{aligned}
&\sum_{T=0}^{\Delta t/\epsilon - 1} p(X_{v+t\epsilon} = j, X_{v+(t+1)\epsilon} = k, \sigma_{0:\tau}) \\
&\qquad = \sum_{t=0}^{\Delta t/\epsilon - 1} \sum_{X_v, X_w} \begin{bmatrix} p(X_v, \sigma_{0:v}) p(X_{v+t\epsilon} = j, \sigma_{v:v+t\epsilon} \mid X_v) \\ p(X_{v+(t+1)\epsilon} = k, \sigma_{v+t\epsilon:v+(t+1)\epsilon} \mid X_{v+t\epsilon} = j) \\ p(X_w, \sigma_{v+(t+1)\epsilon:w} \mid X_{v+(t+1)\epsilon} = k) p(\sigma_{w:\tau} \mid X_w) \end{bmatrix} \\
&\qquad = \sum_{t=0}^{\Delta t/\epsilon - 1} \begin{bmatrix} \boldsymbol{\alpha}_v \exp(\mathbf{Q}_S t\epsilon) \mathbf{e}_j \\ \mathbf{e}'_j p(X_{v+(t+1)\epsilon}, \sigma_{v+t\epsilon:v+(t+1)\epsilon} \mid X_{v+t\epsilon}) \mathbf{e}_k \\ \mathbf{e}'_k \exp(\mathbf{Q}_S (w - (v + (t+1)\epsilon))) \boldsymbol{\beta}_w \end{bmatrix} \ .
\end{aligned}$$

As in the case with observed transitions, as $\epsilon \downarrow 0$, the middle term becomes $q_{jk} dt$, the instantaneous probability of transitioning. Since $\exp(\mathbf{Q}_S t)$ is continuous, we can

express the limit as a sum of integrals of the form

$$q_{jk}\int_v^w \boldsymbol{\alpha}_v \exp(\mathbf{Q}_S(t-v))\boldsymbol{\Delta}_{j,k}\exp(\mathbf{Q}_S(w-t)\boldsymbol{\beta}_w dt \tag{4.6}$$

We have one such term for each interval of constant evidence. Essentially we are integrating the instantaneous probability of transitioning from state j to k over the interval given the evidence. Note that this is very similar to the form of Eq. (4.4) — the only difference is the matrix $\boldsymbol{\Delta}_{j,k}$ and the term q_{jk}.

To obtain the overall sufficient statistics, we have a sum with two types of terms: a term as in Eq. (4.5) for each observed transition, and an integral as in Eq. (4.6) for each interval of constant evidence. The overall expression is

$$\frac{q_{jk}}{p(\sigma_{0:\tau})}\left[\sum_{i=1}^{N-1}\boldsymbol{\alpha}^-_{t_i}\boldsymbol{\Delta}_{j,k}\boldsymbol{\beta}^+_{t_i} + \sum_{i=0}^{N-1}\int_v^w \boldsymbol{\alpha}_v \exp(\mathbf{Q}_S(t-v))\boldsymbol{\Delta}_{j,k}\exp(\mathbf{Q}_S(w-t))\boldsymbol{\beta}_w dt\right] .$$

4.4.4 Runge-Kutta

As we can see from above, computing the expected sufficient statistics involves a set of integrals of the form

$$c_{ij} = \int_v^w \boldsymbol{\alpha}_v \exp(\mathbf{Q}_S(t-v))\boldsymbol{\Delta}_{i,j}\exp(\mathbf{Q}_S(w-t))\boldsymbol{\beta}_w dt\,.$$

We calculate the set of these integrals for all j and k simultaneously via the Runge-Kutta method of fourth order with adaptive step size.

As with many methods of numerical integration, Runge-Kutta makes use of the idea that

$$f(t+\Delta t) \approx f(t) + \frac{df}{dt}\Delta t\ .$$

If we are computing the integral over an interval $[v, w)$, we break it into a series of smaller sub-intervals in which we approximate the value of the function this way. There are two important questions to answer: (1) how do we choose a value to use for the derivative over the sub-interval $[t, t+\Delta t)$ given that the derivative is not generally constant over that interval; (2) how do we decide on the step-size, i.e., the

partitioning of the interval $[v, w)$ into sub-intervals. We look at these two issues, in turn.

Over each sub-interval, the Runge-Kutta method of fourth order uses a weighted average of 4 slopes for the derivative. These correspond to the slope at the beginning and the end of the sub-interval as well as two possible slopes at the mid-point of the interval. We can see from this that we would prefer that the derivative change as little as possible over the course of the sub-interval. While this puts some pressure on us to use smaller sub-intervals, that would mean more steps which would make the computation more complex and could increase rounding errors.

Importantly, the matrix $\mathbf{Q}_S$ gives us a model of how fast the derivative is changing. The intensities represent rates of evolution for states of the system, so larger intensities mean a faster rate of change which usually requires a smaller step size. Suppose q_* is the parameter in the intensity matrix with the greatest absolute value. Then the fastest possible change of state (and hence, change of derivative) occurs, on expectation, after time $1/q_*$. So, we might consider simply choosing a step-size of $1/q_*$. Note that this choice of step-size is context-sensitive — it varies across different subsystems and different evidence.

But if many states transition more slowly or the fast transitioning states have low probability according to the current distribution, this step-size may still be smaller than we want. So, following Press et al. (1992), we use a standard adaptive procedure that allows larger steps to be taken when possible based on error estimates. Since we begin the method with $1/q_*$ as our step-size, there are two ways in which the integration uses a step size that is *adaptive* and not fixed — through the initial choice of step-size based on the maximum intensity and through the standard error-estimate based adaptive procedure.

As we calculate the expected sufficient statistics — over an interval of duration $T = w - v$ with an $n \times n$ intensity matrix — using Runge-Kutta, we take $O(q_* T)$ steps. Each step within Runge Kutta is of complexity $O(n^3)$ (due to matrix multiplication) so the entire call to Runge Kutta has complexity $O(n^3 q_* T)$.

4.4.5 Computing $\boldsymbol{\alpha}_t$ and $\boldsymbol{\beta}_t$

One method of computing $\boldsymbol{\alpha}_t$ and $\boldsymbol{\beta}_t$ is via a forward-backward style algorithm (Rabiner & Juang, 1986) over the entire trajectory to incorporate evidence and get distributions over the state of the system at every time t_i.

We have already defined the forward and backward probability vectors, $\boldsymbol{\alpha}_t$ and $\boldsymbol{\beta}_t$. To initialize the vectors, we simply let $\boldsymbol{\alpha}_0$ be the initial distribution over the state and $\boldsymbol{\beta}_\tau = \mathbf{e}$, a vector of ones. To update the vectors from their previously computed values, we calculate

$$\begin{aligned}\boldsymbol{\alpha}_{t_{i+1}} &= \boldsymbol{\alpha}_{t_i} \exp(\mathbf{Q}_{S_i}(t_{i+1} - t_i))\mathbf{Q}_{S_i S_{i+1}} \\ \boldsymbol{\beta}_{t_i} &= \mathbf{Q}_{S_{i-1} S_i} \exp(\mathbf{Q}_{S_i}(t_{i+1} - t_i))\boldsymbol{\beta}_{t_{i+1}}\end{aligned}$$

To exclude incorporation of the evidence of the transition from either forward or backward vector (or if the time in question is not a transition time), one can simply remove the subsystem transition intensity matrix ($\mathbf{Q}_{S_i S_{i+1}}$) from the calculation. We can compute

$$\begin{aligned}\boldsymbol{\alpha}^-_{t_{i+1}} &= \boldsymbol{\alpha}_{t_i} \exp(\mathbf{Q}_{S_i}(t_{i+1} - t_i)) \\ \boldsymbol{\beta}^+_{t_i} &= \exp(\mathbf{Q}_{S_i}(t_{i+1} - t_i))\boldsymbol{\beta}_{t_{i+1}} \ .\end{aligned}$$

Moreover, as time 0 and τ are not transition times, we have

$$\begin{aligned}\boldsymbol{\alpha}_\tau &= \boldsymbol{\alpha}_{t_{N-1}} \exp(\mathbf{Q}_{S_{N-1}}(\tau - t_{N-1})) \\ \boldsymbol{\beta}_0 &= \exp(\mathbf{Q}_{S_0}(t_1 - 0))\boldsymbol{\beta}_{t_1} \ .\end{aligned}$$

We can then write the distribution over the state of the system at time t given all the evidence as

$$P(X_t = j \mid \sigma_{0:\tau}) = \frac{1}{p(\sigma_{0:\tau})} \boldsymbol{\alpha}^-_t \boldsymbol{\Delta}_{j,j} \boldsymbol{\beta}_t \ .$$

4.5 CTBNs

In a CTBN $\mathcal{N}$, each variable $X \in \boldsymbol{X}$ is conditioned on its parent set $\mathbf{U}$, and each transition of X must be considered in the context of the instantiation to $\mathbf{U}$. With complete data, we know the value of $\mathbf{U}$ during the entire trajectory, so we know at each point in time precisely which homogeneous intensity matrix $\mathbf{Q}_{X|\mathbf{u}}$ governed the dynamics of X.

Thus, the likelihood decomposes by variable as

$$\begin{aligned} L_{\mathcal{N}}(\boldsymbol{q}, \boldsymbol{\theta} : \mathcal{D}) &= \prod_{X_i \in \boldsymbol{X}} L_{X_i}(\boldsymbol{q}_{X_i|\mathbf{U}_i}, \boldsymbol{\theta}_{X_i|\mathbf{U}_i} : \mathcal{D}) \\ &= \prod_{X_i \in \boldsymbol{X}} L_{X_i}(\boldsymbol{q}_{X_i|\mathbf{U}_i} : \mathcal{D}) L_{X_i}(\boldsymbol{\theta}_{X_i|\mathbf{U}_i} : \mathcal{D}) \ . \end{aligned}$$

The term $L_X(\boldsymbol{\theta}_{X|\mathbf{U}} : \mathcal{D})$ is the probability of the sequence of state transitions, disregarding the times between transitions. These state changes depend only on the value of the parents at the instant of the transition. Therefore, with the sufficient statistic $M[x, x'|\mathbf{u}]$, we have

$$L_X(\boldsymbol{\theta} : \mathcal{D}) = \prod_{\mathbf{u}} \prod_{x} \prod_{x' \neq x} \theta_{xx'|\mathbf{u}}^{M[x,x'|\mathbf{u}]} \ .$$

The computation of $L_X(\boldsymbol{q}_{X|\mathbf{U}} : \mathcal{D})$ is more subtle. Consider a particular transition d where a state in which $X = x, \mathbf{U} = \mathbf{u}$ transitioned to another state $X = x, \mathbf{U} = \mathbf{u}'$ after time t. In other words, the duration in the state was terminated not due to a transition of X, but due to a transition of one of its parents. Intuitively, these transitions still depend on X's dynamics, as they can only occur if X stayed at the value x for at least a duration of t. The probability that X stayed at x for this duration is $1 - F(q_{x|\mathbf{u}}, t) = \exp(-q_{x|\mathbf{u}} t)$.

More formally, we define the sufficient statistics for a CTBN as follows.

Definition 4.5.1 *The* sufficient statistics *for the transition dynamics of a CTBN over variables $\boldsymbol{X}$ decompose as a set for each variable $X \in \boldsymbol{X}$ as follows*

- $T[x|\mathbf{u}]$*, the amount of time that* $X = x$ *while* $\mathbf{U}_X = \mathbf{u}$*, and*
- $M[x, x'|\mathbf{u}]$*, the number of transitions* $X = x$ *to* $X = x'$*, while* $\mathbf{U}_X = \mathbf{u}$*.*

The sufficient statistic $T[x|\mathbf{u}]$, the total amount of time where $X = x$ and $\mathbf{U} = \mathbf{u}$, can be decomposed into two different kinds of durations: $T[x|\mathbf{u}] = T_r[x|\mathbf{u}] + T_c[x|\mathbf{u}]$, where $T_r[x|\mathbf{u}]$ is the total over durations t_d that terminate with X remaining equal to x (these include transitions where $\mathbf{U}$ changed value, as well as the end of a trajectory), and $T_c[x|\mathbf{u}]$ is the total over durations t_d that terminate with a change in the value of X. It is easy to see that the terms for the different transitions that comprise $T_r[x|\mathbf{u}]$ combine as do $T_c[x|\mathbf{u}]$, so that we have

$$\begin{aligned} L_X(\boldsymbol{q}_{X|\mathbf{U}} : \mathcal{D}) &= \left(\prod_{\mathbf{u}} \prod_{x} q_{x|\mathbf{u}}^{M[x|\mathbf{u}]} \exp(-q_{x|\mathbf{u}} T_c[x|\mathbf{u}]) \right) \times \left(\prod_{\mathbf{u}} \prod_{x} \exp(-q_{x|\mathbf{u}} T_r[x|\mathbf{u}]) \right) \\ &= \prod_{\mathbf{u}} \prod_{x} q_{x|\mathbf{u}}^{M[x|\mathbf{u}]} \exp(-q_{x|\mathbf{u}} T[x|\mathbf{u}]) \ . \end{aligned}$$

Thus, we do not need to maintain the distinction between $T_c[x|\mathbf{u}]$ and $T_r[x|\mathbf{u}]$. Instead, we can simply use $T[x|\mathbf{u}]$ as the sufficient statistic.

We can now write the log likelihood as a sum of local variable likelihoods of the form

$$\begin{aligned} &\ell_X(\boldsymbol{q}, \boldsymbol{\theta} : \mathcal{D}) \\ &\quad = \ell_X(\boldsymbol{q} : \mathcal{D}) + \ell_X(\boldsymbol{\theta} : \mathcal{D}) \\ &\quad = \left[\sum_{\mathbf{u}} \sum_{x} M[x|\mathbf{u}] \ln(q_{x|\mathbf{u}}) - q_{x|\mathbf{u}} \cdot T[x|\mathbf{u}] \right] + \left[\sum_{\mathbf{u}} \sum_{x} \sum_{x' \neq x} M[x, x'|\mathbf{u}] \ln(\theta_{xx'|\mathbf{u}}) \right] \ . \end{aligned} \tag{4.7}$$

4.6 Discussion

In this chapter we showed how to compute the likelihood of a data given a CTBN model. This infrastructure will be crucial to the algorithms we develop over the next few chapters to support learning and inference tasks.

Chapter 5

Learning with Complete Data

In the last chapter, we saw how to compute the likelihood of a data set given a CTBN model. But what if we do not have a CTBN model? The likelihood function can still help us because there is a clear relationsip, by Bayes' theorem, between the likelihood (i.e., probability of a data set given a model) and the probability of a model given a data set.

In this chapter we will show how to learn CTBN parameters and structure from complete data. The basic methods are analogous to standard Bayesian network techniques, but the formulae and derivations are substantially different. One significant point of departure between BNs and CTBNs is model search: we will see that the search space over CTBN structures is substantially simpler than that of BN or DBN structures.

For now, we will assume that we have complete data. This assumption will be relaxed in the next chapter.

5.1 Parameter Estimation

We first consider the problem of estimating the parameters of a CTBN with a fixed structure $\mathcal{G}$. As usual, this problem is not only useful on its own, but also as a key component in the structure learning task.

Theorem 5.1.1 *To maximize the likelihood given in Equation 4.7, the parameters* $q_{x|\mathbf{u}}$ *and* $\theta_{xx'|\mathbf{u}}$ *can be written as a function of the sufficient statistics as*

$$\hat{q}_{x|\mathbf{u}} = \frac{M[x|\mathbf{u}]}{T[x|\mathbf{u}]} ,$$
$$\hat{\theta}_{xx'|\mathbf{u}} = \frac{M[x,x'|\mathbf{u}]}{M[x|\mathbf{u}]} . \tag{5.1}$$

These are the the maximum-likelihood (MLE) parameters.

Proof: Taking the partial derivative of Equation 4.7 with respect to $q_{x|\mathbf{u}}$ and setting the result to 0 gives us

$$\frac{M[x|\mathbf{u}]}{q_{x|\mathbf{u}}} - T[x|\mathbf{u}] = 0 .$$

Solving for $q_{x|\mathbf{u}}$ leads to the formula above. Alternatively, as a member of the exponential family, we can derive this formula by matching moments. According to the data, the mean time to transition from $X = x$ and $\mathbf{U} = \mathbf{u}$ is the total time divided by the number of transitions from that state, i.e., $T[x|\mathbf{u}]/M[x|\mathbf{u}]$. The mean of the exponential distribution with parameter $q_{x|\mathbf{u}}$ is $1/q_{x|\mathbf{u}}$. By setting these means equal to each other, we get the maximum likelihood formula above.

For each multinomial distribution, we must maximize the likelihood subject to the constraint that its parameters sum to 1. Using the method of Lagrange multipliers, we set the constraint function g to

$$g = \sum_{x' \neq x} \theta_{xx'|\mathbf{u}} - 1 = 0$$

and maximize

$$\sum_{x' \neq x} M[x,x'|\mathbf{u}] \ln(\theta_{xx'|\mathbf{u}}) + \lambda \cdot g .$$

Taking the partial derivative with respect to $\theta_{xx'|\mathbf{u}}$ and setting it to 0 yields

$$\frac{M[x,x'|\mathbf{u}]}{\theta_{xx'|\mathbf{u}}} - \lambda = 0 .$$

which implies that

$$\theta_{xx'|\mathbf{u}} = \frac{M[x, x'|\mathbf{u}]}{\lambda} \ .$$

Taking the partial derivative with respect to λ and setting it to 0 yields the original constraint function. Substitution yields

$$\sum_{x' \neq x} \frac{M[x, x'|\mathbf{u}]}{\lambda} = 1 \ .$$

Solving for λ, we get $\lambda = \sum_{x' \neq x} M[x, x'|\mathbf{u}] = M[x|\mathbf{u}]$. From this, we can conclude

$$\theta_{xx'|\mathbf{u}} = \frac{M[x, x'|\mathbf{u}]}{M[x|\mathbf{u}]} \ .$$

as desired. Note that this can also be derived by matching moments because the mean number of transitions out of the state $X = x$ and $\mathbf{U} = \mathbf{u}$ that led to the state $X = x'$ according to the data is exactly $M[x, x'|\mathbf{u}]/M[x|\mathbf{u}]$. ∎

5.1.1 Bayesian Approach

To perform Bayesian parameter estimation, and to define a Bayesian score for our structure search, we need to define a prior distribution over the parameters of our CTBN. As usual, for computational efficiency, we want to use a *conjugate prior* — one where the posterior (after conditioning on the data) is in the same parametric family as the prior.

Let us begin with constructing an appropriate prior for a single Markov process. Recall that a Markov process has two sets of parameters: a multinomial distribution parameterized by $\boldsymbol{\theta}$, and an exponential distribution parameterized by q. An appropriate conjugate prior for the exponential parameter q is the *Gamma distribution* $P(q) = Gamma(\alpha, \tau)$, where

$$P(q) = \frac{(\tau)^{\alpha+1}}{\Gamma(\alpha + 1)} q^{\alpha} \exp(-q\tau) \ .$$

The multinomial distribution is familiar from traditional Bayesian networks where the

standard conjugate prior is a Dirichlet distribution (Heckerman et al., 1995; Geiger & Heckerman, 1995).

If we assume that

$$\begin{aligned} P(\boldsymbol{\theta}) &= Dir(\alpha_{xx_1}, \ldots, \alpha_{xx_k}) \\ P(q) &= Gamma(\alpha_x, \tau_x) \\ P(\boldsymbol{\theta}, q) &= P(\boldsymbol{\theta})P(q), \end{aligned}$$

then, after conditioning on the data, we have

$$\begin{aligned} P(\boldsymbol{\theta} \mid \mathcal{D}) &= Dir(\alpha_{xx_1} + M[x, x_1], \ldots, \alpha_{xx_k} + M[x, x_k]) \\ P(q \mid \mathcal{D}) &= Gamma(\alpha_x + M[x], \tau_x + T[x]) \ . \end{aligned}$$

We generalize this idea to a parameter prior for an entire CTBN by making two standard assumptions for parameter priors in Bayesian networks (Heckerman et al., 1995), *global parameter independence*:

$$P(\boldsymbol{q}, \boldsymbol{\theta}) = \prod_{X \in \boldsymbol{X}} P(\boldsymbol{q}_{X|\mathbf{Pa}(X)}, \boldsymbol{\theta}_{X|\mathbf{Pa}(X)})$$

and *local parameter independence*:

$$P(\boldsymbol{q}_{X|\mathbf{U}}, \boldsymbol{\theta}_{X|\mathbf{U}}) = \left(\prod_x \prod_{\mathbf{u}} P(q_{x|\mathbf{u}}) \right) \left(\prod_x \prod_{\mathbf{u}} P(\boldsymbol{\theta}_{x|\mathbf{u}}) \right).$$

If our parameter prior satisfies these assumptions, so does our posterior, as it belongs to the same parametric family. Thus, we can maintain our parameter distribution in closed form, and update it using the obvious sufficient statistics: $M[x, x'|\mathbf{u}]$ for $\boldsymbol{\theta}_{x|\mathbf{u}}$, and $M[x|\mathbf{u}], T[x|\mathbf{u}]$ for $q_{x|\mathbf{u}}$.

Given a parameter distribution, we can use it to predict the next event, averaging out the event probability over the possible values of the parameters. As usual, this prediction is equivalent to using "expected" parameter values, which have the

same form as the MLE parameters, but account for the "imaginary counts" of the hyperparameters:

$$\hat{q}_{x|\mathbf{u}} = \frac{\alpha_{x|\mathbf{u}} + M[x|\mathbf{u}]}{\tau_{x|\mathbf{u}} + T[x|\mathbf{u}]} ,$$
$$\hat{\theta}_{xx'|\mathbf{u}} = \frac{\alpha_{xx'|\mathbf{u}} + M[x,x'|\mathbf{u}]}{\alpha_{x|\mathbf{u}} + M[x|\mathbf{u}]} . \tag{5.2}$$

Note that, in principle, this choice of parameters is only valid for predicting a single transition, after which we should update our parameter distribution accordingly. However, as is often done in other settings, we can approximate the exact Bayesian computation by "freezing" the parameters to these expected values, and use them for predicting an entire trajectory.

5.2 Learning Structure

We now turn to the problem of learning the structure of a CTBN. We take a score-based approach to this task, defining a Bayesian score for evaluating different candidate structures, and then using a search algorithm to find a structure that has high score.

5.2.1 Score Function

The *Bayesian score* over structures $\mathcal{G}$ is defined as

$$\mathbf{score}_B(\mathcal{G} : \mathcal{D}) = \ln P(\mathcal{D} \mid \mathcal{G}) + \ln P(\mathcal{G}) . \tag{5.3}$$

We can significantly increase the efficiency of our search algorithm if we assume that our prior satisfies certain standard assumptions. We assume that our structure prior $P(\mathcal{G})$ satisfies *structure modularity*, so that $P(\mathcal{G}) = \prod_i P(\mathbf{Pa}(X_i) = \mathbf{Pa}_{\mathcal{G}}(X_i))$. We also assume that our parameter prior satisfies *parameter modularity*: For any two structures $\mathcal{G}$ and $\mathcal{G}'$ such that $\mathbf{Pa}_{\mathcal{G}}(X) = \mathbf{Pa}_{\mathcal{G}'}(X)$, we have that $P(\boldsymbol{q}_X, \boldsymbol{\theta}_X \mid \mathcal{G}) = P(\boldsymbol{q}_X, \boldsymbol{\theta}_X \mid \mathcal{G}')$. Combining parameter modularity and parameter independence, we

have

$$P(\boldsymbol{q}_{\mathcal{G}}, \boldsymbol{\theta}_{\mathcal{G}} \mid \mathcal{G}) = \prod_{X_i} P(\boldsymbol{q}_{X_i|\mathbf{U}_i} \mid \mathbf{Pa}(X_i) = \mathbf{Pa}_{\mathcal{G}}(X_i)) P(\boldsymbol{\theta}_{X_i|\mathbf{U}_i} \mid \mathbf{Pa}(X_i) = \mathbf{Pa}_{\mathcal{G}}(X_i)).$$

As $P(\mathcal{G})$ does not grow with the amount of data, the significant term in Eq. (5.3) is the *marginal likelihood* $P(\mathcal{D} \mid \mathcal{G})$. This term incorporates our uncertainty over the parameters by integrating over all of their possible values:

$$P(\mathcal{D} \mid \mathcal{G}) = \int_{\boldsymbol{q}_{\mathcal{G}}, \boldsymbol{\theta}_{\mathcal{G}}} P(\mathcal{D} \mid \boldsymbol{q}_{\mathcal{G}}, \boldsymbol{\theta}_{\mathcal{G}}) P(\boldsymbol{q}_{\mathcal{G}}, \boldsymbol{\theta}_{\mathcal{G}} \mid \mathcal{G}) d\boldsymbol{q}_{\mathcal{G}} d\boldsymbol{\theta}_{\mathcal{G}}.$$

As in Eq. (4.3), the likelihood decomposes as a product:

$$\begin{aligned} P(\mathcal{D} \mid \boldsymbol{q}_{\mathcal{G}}, \boldsymbol{\theta}_{\mathcal{G}}) &= \prod_{X_i} L_{X_i}(\boldsymbol{q}_{X_i|\mathbf{U}_i} : \mathcal{D}) L_{X_i}(\boldsymbol{\theta}_{X_i|\mathbf{U}_i} : \mathcal{D}) \\ &= \underbrace{\left(\prod_{X_i} L_{X_i}(\boldsymbol{q}_{X_i|\mathbf{U}_i} : \mathcal{D}) \right)}_{L(\boldsymbol{q}:\mathcal{D})} \underbrace{\left(\prod_{X_i} L_{X_i}(\boldsymbol{\theta}_{X_i|\mathbf{U}_i} : \mathcal{D}) \right)}_{L(\boldsymbol{\theta}:\mathcal{D})} . \end{aligned}$$

Using this decomposition, and global parameter independence, we now have

$$\begin{aligned} P(\mathcal{D} \mid \mathcal{G}) &= \int_{\boldsymbol{q}_{\mathcal{G}}, \boldsymbol{\theta}_{\mathcal{G}}} L(\boldsymbol{q}_{\mathcal{G}} : \mathcal{D}) L(\boldsymbol{\theta} : \mathcal{D}) P(\boldsymbol{\theta}_{\mathcal{G}}) P(\boldsymbol{q}_{\mathcal{G}}) d\boldsymbol{q}_{\mathcal{G}} d\boldsymbol{\theta}_{\mathcal{G}} \\ &= \left(\int_{\boldsymbol{q}_{\mathcal{G}}} L(\boldsymbol{q}_{\mathcal{G}} : \mathcal{D}) P(\boldsymbol{q}_{\mathcal{G}}) d\boldsymbol{q}_{\mathcal{G}} \right) \qquad (5.4) \\ &\quad \times \left(\int_{\boldsymbol{\theta}_{\mathcal{G}}} L(\boldsymbol{\theta}_{\mathcal{G}} : \mathcal{D}) P(\boldsymbol{\theta}_{\mathcal{G}}) d\boldsymbol{\theta}_{\mathcal{G}} \right) . \qquad (5.5) \end{aligned}$$

Using local parameter independence, the term (5.4) can be decomposed as

$$\begin{aligned}
&\prod_{X\in\boldsymbol{X}}\prod_{\mathbf{u}}\prod_{x}\int_0^\infty P(q_{x|\mathbf{u}})L_X(q_{x|\mathbf{u}}:\mathcal{D})dq_{x|\mathbf{u}} \\
&= \prod_{X\in\boldsymbol{X}}\prod_{\mathbf{u}}\prod_{x}\int_0^\infty \begin{array}{l}\frac{(\tau_{x|\mathbf{u}})^{\alpha_{x|\mathbf{u}}+1}}{\Gamma(\alpha_{x|\mathbf{u}}+1)}(q_{x|\mathbf{u}})^{\alpha_{x|\mathbf{u}}}\exp(-q_{x|\mathbf{u}}\tau_{x|\mathbf{u}}) \\ \times(q_{x|\mathbf{u}})^{M[x|\mathbf{u}]}\exp(-q_{x|\mathbf{u}}\cdot T[x|\mathbf{u}])dq_{x|\mathbf{u}}\end{array} \\
&= \prod_{X\in\boldsymbol{X}}\prod_{\mathbf{u}}\prod_{x}\int_0^\infty \left[\begin{array}{l}\frac{(\tau_{x|\mathbf{u}})^{\alpha_{x|\mathbf{u}}+1}}{\Gamma(\alpha_{x|\mathbf{u}}+1)}(q_{x|\mathbf{u}})^{\alpha_{x|\mathbf{u}}+M[x|\mathbf{u}]} \\ \times\exp(-q_{x|\mathbf{u}}(\tau_{x|\mathbf{u}}+T[x|\mathbf{u}]))\end{array}\right]dq_{x|\mathbf{u}} \\
&= \prod_{X\in\boldsymbol{X}}\prod_{\mathbf{u}}\prod_{x}\frac{\Gamma(\alpha_{x|\mathbf{u}}+M[x|\mathbf{u}]+1)(\tau_{x|\mathbf{u}})^{\alpha_{x|\mathbf{u}}+1}}{\Gamma(\alpha_{x|\mathbf{u}}+1)(\tau_{x|\mathbf{u}}+T[x|\mathbf{u}])^{\alpha_{x|\mathbf{u}}+M[x|\mathbf{u}]+1}} \\
&= \prod_{X\in\boldsymbol{X}} MargL^q(X,\mathbf{Pa}_{\mathcal{G}}(X):\mathcal{D})\ .
\end{aligned} \tag{5.6}$$

As the distributions over the parameters $\boldsymbol{\theta}$ are Dirichlet, the analysis of the term Eq. (5.5) is analogous to traditional Bayesian networks, simplifying to

$$\prod_{X\in\boldsymbol{X}}\prod_{\mathbf{u}}\prod_{x}\frac{\Gamma(\alpha_{x|\mathbf{u}})}{\Gamma(\alpha_{x|\mathbf{u}}+M[x|\mathbf{u}])}\times\prod_{x'\neq x}\frac{\Gamma(\alpha_{xx'|\mathbf{u}}+M[x,x'|\mathbf{u}])}{\Gamma(\alpha_{xx'|\mathbf{u}})} = \prod_{X\in\boldsymbol{X}} MargL^\theta(X,\mathbf{Pa}_{\mathcal{G}}(X):\mathcal{D})\ .$$

Using this decomposition, and the assumption of structure modularity, the Bayesian score in Eq. (5.3) can now be decomposed as a sum of family scores — individually denoted $\mathbf{FamScore}(X,\mathbf{Pa}_{\mathcal{G}}(X):\mathcal{D})$ — that each measure the quality of $\mathbf{Pa}_{\mathcal{G}}(X)$ as a parent set for X given data $\mathcal{D}$:

$$\begin{aligned}
\mathbf{score}_B(\mathcal{G}:\mathcal{D}) &= \sum_{X_i\in\boldsymbol{X}}\mathbf{FamScore}(X_i,\mathbf{Pa}_{\mathcal{G}}(X_i):\mathcal{D}) \\
&= \sum_{X_i\in\boldsymbol{X}}\ln P(\mathbf{Pa}(X) \\
&= \mathbf{Pa}_{\mathcal{G}}(X_i)) + \ln MargL^q(X_i,\mathbf{U}_i:\mathcal{D}) + \ln MargL^\theta(X_i,\mathbf{U}_i:\mathcal{D})\ .
\end{aligned}$$

5.2.2 Model Search

Given the score function, it remains to find a structure $\mathcal{G}$ that maximizes the score. This task is an optimization problem over possible CTBN network structures. Interestingly, the search space over CTBN structures is significantly simpler than that of BNs or DBNs.

Chickering et al. (1994) show that the problem of learning a optimal Bayesian network structure is NP-hard. Specifically, they define the problem **k-Learn**: Finding the highest scoring Bayesian network structure, when each variable is restricted to have at most k parents. The problem **k-Learn** is NP-hard even for $k = 2$. Intuitively, the reason is that we cannot determine the optimal parent set for each node individually; due to the acyclicity constraint, the choice of parent set for one node restricts our choices for other nodes. The same NP-hardness result clearly carries over to DBNs, if we allow edges within a time slice.

However, this problem does not arise in the context of CTBN learning. Here, all edges are across time — representing the effect of the current value of one variable on the next value of the other. Thus, we have no acyclicity constraints, and we can optimize the parent set for each variable independently. Specifically, if we restrict the maximum number of parents to k, we can simply exhaustively enumerate each of the possible parent sets $\mathbf{U}$ for $|\mathbf{U}| \leq k$ and compute $\mathbf{FamScore}(X \mid \mathbf{U} : \mathcal{D})$. We then choose as $Pa(X)$ the set $\mathbf{U}$ which maximizes the family score. For fixed k, this algorithm is polynomial in n. Therefore,

Theorem 5.2.1 *The problem **k-Learn** for CTBNs, for fixed k, can be solved in polynomial time in the number of variables n and the size of the data set $\mathcal{D}$.*

In practice, we do not wish to exhaustively enumerate the possible parent sets for each variable X. We can therefore use a greedy hill-climbing search with operators that add and delete edges in the CTBN graph. However, due to the lack of interactions between the families of different variables, we can perform this greedy search separately for each variable X, selecting a locally optimal family for it. Thus, this heuristic search can be performed much more efficiently than for BNs or DBNs.

5.3 Structure Identifiability

So far, we have focused on the problem of learning a CTBN that provides a good fit to some training data $\mathcal{D}$. As shown in section 3.5 any variable-based stochastic process has a unique minimal CTBN representation, so an important question is whether we can identify this CTBN from data. More precisely, assume that our data $\mathcal{D}$ is generated from some process $\mathbf{Q}_{\boldsymbol{X}}$, and let $\mathcal{G}^*$ be the minimal S-map for $\mathbf{Q}_{\boldsymbol{X}}$. We would like our learning algorithm to return a network whose structure is $\mathcal{G}^*$. Our learning algorithm searches for the network structure that maximizes the Bayesian score. Thus, the key property (ignoring possible limitations of our search procedure) is the following.

Definition 5.3.1 *A scoring function is said to be* consistent *if, as the amount of data $|\mathcal{D}| \to \infty$, the following holds with probability that approaches 1: The structure $\mathcal{G}^*$ will maximize the score, and the score of all structures $\mathcal{G} \neq \mathcal{G}^*$ will have a strictly lower score.*

Once again, compare this situation to that of Bayesian networks. There, the best we can hope for is that all and only structures that are I-equivalent to the "true" network will maximize the score.

To prove that our score is consistent, it helps to consider its behavior as the amount of data increases.

Theorem 5.3.2 *As the amount of data $|\mathcal{D}| \to \infty$,*

$$\boldsymbol{score}_B(\mathcal{G} : \mathcal{D}) = \ell(\hat{\boldsymbol{q}}_{\mathcal{G}}, \hat{\boldsymbol{\theta}}_{\mathcal{G}} : \mathcal{D}) - \frac{\ln |\mathcal{D}|}{2} \boldsymbol{Dim}[\mathcal{G}] + O(1) \tag{5.7}$$

where $\boldsymbol{Dim}[\mathcal{G}]$ is the number of independent parameters in $\mathcal{G}$, and $\hat{\boldsymbol{q}}_{\mathcal{G}}$ and $\hat{\boldsymbol{\theta}}_{\mathcal{G}}$ are the MLE parameters of Eq. (5.1).

Eq. (5.7) is simply the standard BIC approximation to the Bayesian score (Lam & Bacchus, 1994), which carries over to CTBNs. It shows that, asymptotically, the CTBN Bayesian score trades off fit to data and model complexity. We are more likely to add an arc if it represents a strong connection between the variables. Moreover,

as the amount of data grows, we obtain more support for weak connections, and are more likely to introduce additional arcs.

Proof: Both $\mathbf{score}_B(\mathcal{G} : \mathcal{D})$ and $\ell(\hat{\boldsymbol{q}}_{\mathcal{G}}, \hat{\boldsymbol{\theta}}_{\mathcal{G}} : \mathcal{D})$ decompose into two components — one for the parameters $\boldsymbol{\theta}$ which are exactly analogous to the standard Bayesian network formulae, and one for the parameters $\boldsymbol{q}$. We will only show the argument for the latter, though both make similar use of Stirling's Approximation,

$$\ln \Gamma(x) = \frac{1}{2}\ln(2\pi) + (x - \frac{1}{2})\ln x - x + O(1) \; .$$

As the score function decomposes by variable and parent instantiation, fix any arbitrary instantiation of $X = x$ with parents $\mathbf{U} = \mathbf{u}$ and simply write α, τ, M, and T instead of including all the subscripts. Then, it suffices to show that, as $|\mathcal{D}| \to \infty$, the log of the expression in Eq. (5.6) approaches $M \ln \hat{q} - T\hat{q} - \frac{1}{2}\ln T + O(1)$ where

$$\hat{q} = \frac{M}{T} \, .$$

So, we have

$$\begin{aligned}
&\ln\Gamma(\alpha + M + 1) - \ln \Gamma(\alpha + 1) + (\alpha + 1)\ln(\tau) - (\alpha + M + 1)\ln(\tau + T) \\
&\quad \approx (M + \alpha + \frac{1}{2})\ln(M + \alpha) - M - (\alpha + M + 1)\ln(\tau + T) + O(1) \\
&\quad = (M + \alpha + \frac{1}{2})\ln(M + \alpha) - M - (\alpha + M)\ln(\tau + T) - \ln(\tau + T) + O(1) \\
&\quad \to M \ln M + (\alpha + \frac{1}{2})\ln M - M - M \ln T - \alpha \ln T - \ln T + O(1) \\
&\quad = M \ln M - M \ln T - M + (\alpha + \frac{1}{2})\ln(T\hat{q}) - \alpha \ln T + O(1) \\
&\quad = M \ln \frac{M}{T} - T\frac{M}{T} - \frac{1}{2}\ln T + O(1) \\
&\quad = M \ln \hat{q} - T\hat{q} - \frac{1}{2}\ln T + O(1) \; .
\end{aligned}$$

We combine the results of each local computation, we get the desired result. ∎

Theorem 5.3.3 *$\boldsymbol{score}_B(\mathcal{G} : \mathcal{D})$ is consistent.*

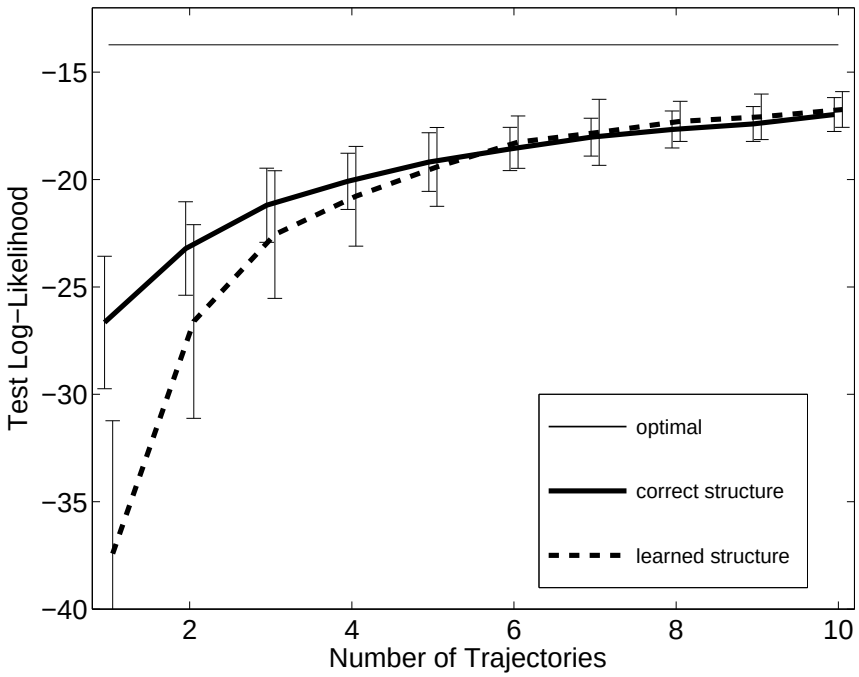

Figure 5.1: Log-likelihoods of test data for CTBN with learned parameters and structure and CTBN with learned parameters. The CTBNs were learned from varying amounts of data generated from the drug effect network. Each trajectory corresponds to 6 units of time, and about 18 transitions. The thin line shows the likelihood for the true network.

Proof: The proof shows that the BIC score is consistent; as consistency is an asymptotic property, it suffices to show the consistency of the Bayesian score. The argument for the consistency of the BIC score is a standard one: If $\mathcal{G}$ is a superset of $\mathcal{G}^*$, it can represent $\mathbf{Q}_{\boldsymbol{X}}$ exactly; thus, with enough data, the difference between the log-likelihood components of the score of $\mathcal{G}$ and $\mathcal{G}^*$ will go to zero. But, $\mathcal{G}$ has more parameters, leading to a higher penalty and thus a lower score. If $\mathcal{G}$ is not a superset of $\mathcal{G}^*$, it follows from Theorem 3.5.7 that it is not capable of representing $\mathbf{Q}_{\boldsymbol{X}}$. In this case, as the amount of data grows, the likelihood portion of the score will dominate and $\mathcal{G}^*$ will have the higher score. ∎

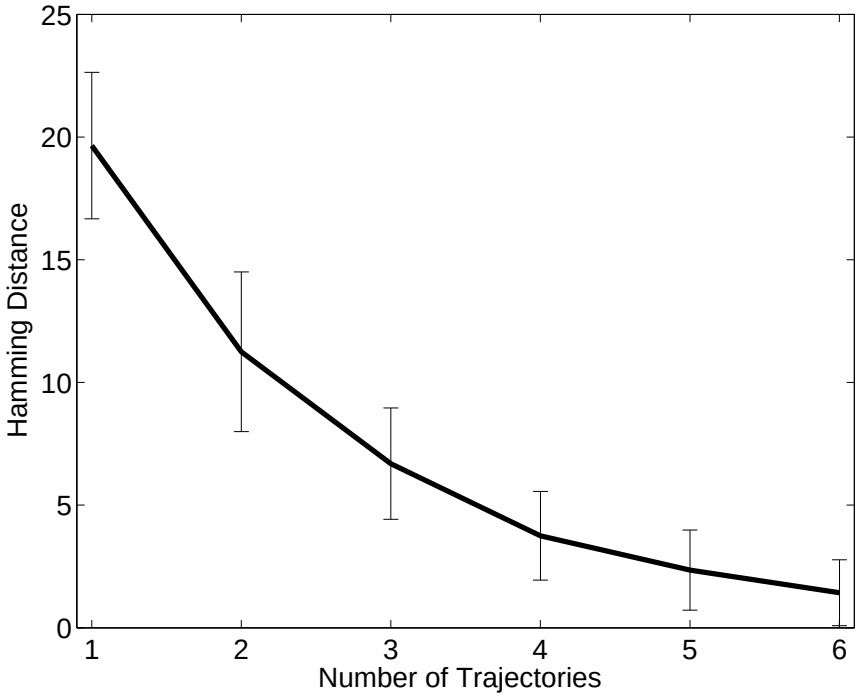

Figure 5.2: Hamming distance between the true structure and the highest scoring structure for randomly generated 10-node CTBNs, for varying amounts of data. Each trajectory corresponds to 150 units of time, and about 1000 transitions.

5.4 Results

We first tested our ability to recover complex structures with our learning algorithm. We generated different amounts of data from the drug effect network of Figure 3.1, and used it to learn two models: one where we learned both the CTBN structure and the parameters, and the other where we simply estimated parameters for the correct network structure. We then computed the log-likelihood of test data for all networks, including the generating network. In all cases where we used a learned network, we used the expected parameters of Eq. (5.2) throughout the entire test trajectory. The results are shown in Figure 5.3. Even for fairly small amounts of data, our results with unknown structure are essentially identical to those with the correct structure.

To further test the ability of our algorithm to recover structure, we generated 100

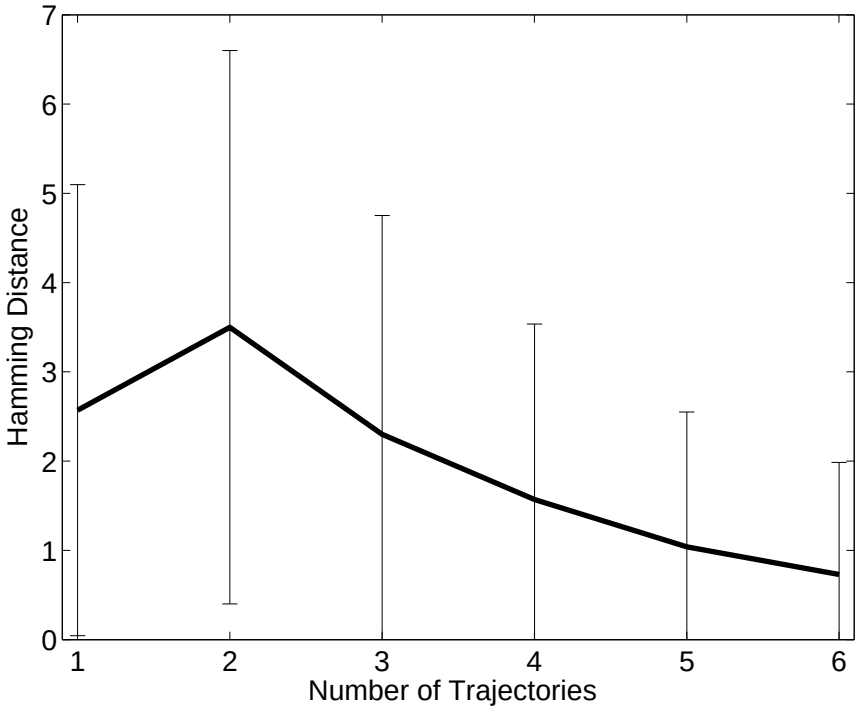

Figure 5.3: Hamming distance between the highest scoring structure and the structure learned by greedy search for random CTBNs for randomly generated 10-node CTBNs, for varying amounts of data. Each trajectory corresponds to 150 units of time, and about 1000 transitions.

random networks of 10 binary processes. We fixed a maximum parent set size of 4 and generated a random graph structure obeying this constraint. We then drew the multinomial parameters of the network from Dirichlet distributions (with parameters all 1) and the exponential parameters from a Gamma distribution (with both parameters equal to 1). In Figure 5.3 we compared the maximum-score structure (with the same constraint on parent sets) to the true structure. The Hamming distance measured is the number of arcs present in only one of the graphs. As predicted by Theorem 5.3.3, as the amount of data grows, the correct structure has the highest score. Indeed, this happens even for very reasonable amounts of data. More interestingly, in a very large fraction of the cases, the simple greedy search algorithm recovers the highest scoring network very reliably. Figure 5.3 shows the difference

between the maximum-score structure and the one As we can see, the local minima in the search space are less frequent as the amount of data grows, and in general the difference between the exhaustive and greedy search techniques is small (roughly one edge difference for reasonable amounts of data).

5.5 Discussion

We have presented maximum likelihood and Bayesian approaches to parameter estimation for CTBNs and a Bayesian structure learning algorithm. As we showed, learning temporal processes as a CTBN has several important advantages. As we are not discretizing time, we do not need to choose some single time granularity in which to model the process. The model for each variable can reflect its own time granularity, better representing its evolution. Additionally, structure search under the contraint of a fixed limit to the number of parents per node can find the highest scoring structure in polynomial time.

We continue the discussion of these algorithms in the next chapter where we will use these algorithms to compare the CTBN framework with the DBN framework.

Chapter 6

CTBNs vs DBNs

Now that we have defined the basic representation and delved into some basic questions of learning CTBNs from data, a natural question is: how do CTBNs compare to the established DBN framework?

As we will see, there is an even more basic question to address first — namely, how *can* we compare them. We begin with a discussion of differences between CTBNs and DBNs, explain why they are difficult to compare, and then provide some experiments comparing them. Some additional experimental results comparing CTBNs and DBNs are given in Section 9.5.

6.1 Comparison of the Frameworks

To compare the CTBN and DBN frameworks, suppose we start with a non-trivial CTBN — that is, one with at least one arc in the graph. For any finite amount of time, probabilistic influence can flow between any variables connected by a path in the CTBN graph. Thus, if we want to construct an "equivalent" DBN, the 2-TBN must be fully connected regardless of the Δt we choose for each time slice. We can construct a DBN that approximates the CTBN by picking a subset of the connections — e.g., those which have the strongest influence.

Once we have an approximate DBN, we still have the standard problem of exponential blowup in performing inference over time (Boyen & Koller, 1998). So we

would be led to perform approximate DBN inference in an approximate DBN. While this could form the basis of an approximation algorithm for CTBNs, we chose to work directly with continuous time, making a direct approximation.

It is also interesting to consider the mapping from a DBN to an "equivalent" CTBN. If we start with a DBN defined by a 2-TBN and try to construct a CTBN, we have two sorts of arcs to consider. *Intra-time-slice* arcs in the DBN have a temporally immediate effect. In some sense, this is faster than any arc in a CTBN, but we could attempt to model it with a high-intensity arc. On the other hand, *inter-time-slice* arcs are more complex. These have an influence that jumps over a time interval of length Δt with no clear description of what happens during the interval. Such arcs can describe any arbitrary probabilistic relationship between the state at time t and the state at time $t + \Delta t$. Potentially, that can include probabilistic relationships that could not have arisen through a continuously Markovian process such as a CTBN. The Markov assumption for DBNs holds only at the level of the granularity of the model — it has the capability of modelling non-Markovian behavior between time-slices.

There is another way of looking at this same issue. Consider the data that is used to learn DBNs. The core datum is the *observed state*. Complete data for a DBN is a sequence of state observations. This means that when we learn a DBN, we are learning a probabilistic model that accounts for a series of snapshots of system.

Now, compare the data that is used to learn a CTBN. The core datum is the *observed transition*. Complete data for a CTBN is a sequence of transitions. This means that when we learn a CTBN, we are learning a model of the way the system transitions from state to state. This is a direct probabilistic model of the structure of the process.

A DBN can be *interpreted* as a model of a process, but it is, more naturally, a model of an observation sequence. It should not, then, be surprising to find that the learned model can be quite sensitive to the granularity that is chosen. When one learns an arc from data in a DBN, there is a question of whether that arc is a feature of the process or a product of the granularity of the observation sequence.

By contrast, a CTBN is a natural model of the process. Because the nature of the CTBN and DBN models are so different, it is challenging to evaluate their

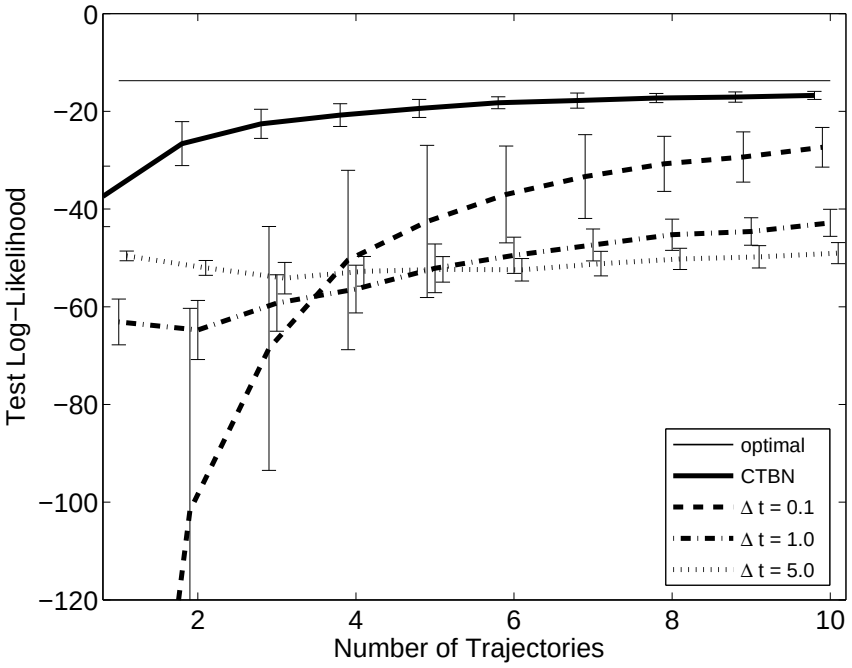

Figure 6.1: Log-likelihoods of test data for learned CTBN model and DBN models with differing time granularity. The networks were learned from varying amounts of data generated from the drug effect network. Each trajectory corresponds to 6 units of time, and about 18 transitions. The thin line shows the likelihood for the true network.

relative performance. We can interpret the DBN as a process model — but to do so and compare to the CTBN process model, we must add some framework to the DBN to make explicit what happens between time slices. This has the side effect of making the DBN more expressive than a CTBN in its ability to model non-Markovian processes. It also has more free parameters because of the possible intra-time-slice arcs. For example, if there are 2 binary variables, a fully-connected DBN has 12 free parameters and a fully-connected CTBN has only 8. Thus, the DBN can represent certain transition models that do not arise from a purely Markovian continuous-time process.

Alternatively, we can use a CTBN to model a sequence of state observations. But this does not make use of a CTBN to its full potential, because it explicitly ignores

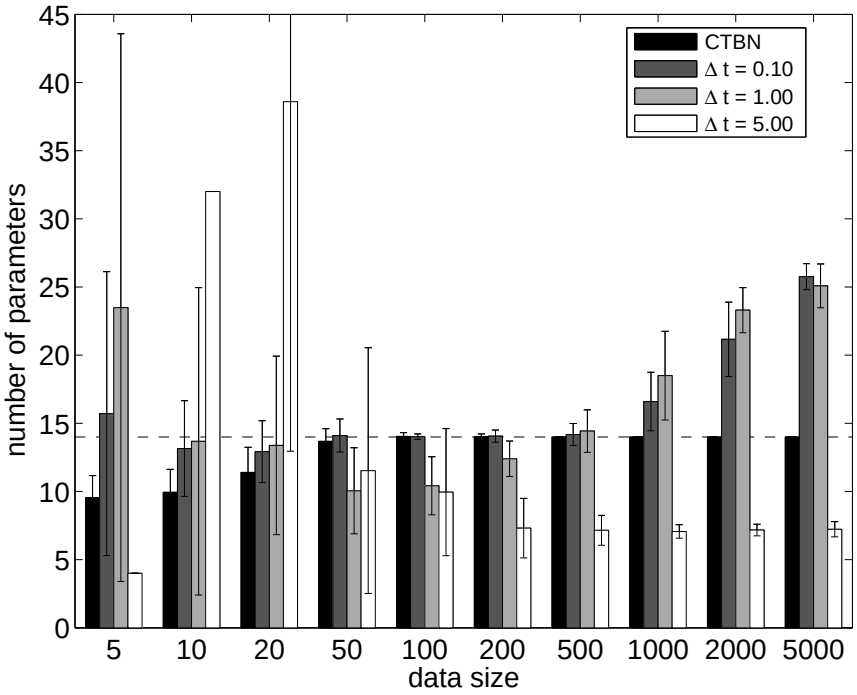

Figure 6.2: For a 4-node chain network, the number of parameters of the learned structures as a function of the amount of time the data was collected, for CTBNs and DBNs with varying time granularity.

the direct process model.

In the first set of results below, we interpret the DBN as a process model. We also use comparison of parameter counts as a way of gaining insight into the relative complexity of learned models

6.2 Results

First, we wanted to compare the generalization performance of learned CTBNs with those of learned DBNs. To do so, we extended the DBN model to include distributions over when, within a time slice, a given transition occurred. We assumed a uniform distribution within the time slice, augmenting it with a parameter for each variable that determines the probability that the value of the variable transitions more than

once within a time slice. The value of this parameter was also learned from data. Figure 6.1 compares the generalization ability of learned CTBNs and learned DBNs with varying time granularity. As expected, the correct DBN structure exhibits entanglement due to the temporal discretization, and therefore requires more edges to approximate the distribution well. Even for small Δt, the amount of data required to estimate the much larger number of parameters is significantly greater. As Δt grows large, the performance of the DBN decreases rapidly. Interestingly, for large values of Δt, the DBNs simply cannot capture the transition dynamics accurately enough to converge to competitive performance.

We further tested our CTBN learning framework on various synthetic data sets, generated from CTBNs. We used a simple greedy hill-climbing algorithm over the space of structures, optimizing the family for each variable separately. For comparison, we also learned DBNs using different time granularities. To allow a fair comparison, we used the same greedy hill-climbing algorithm there.

We tested the ability of CTBNs and DBNs to capture very simple dependencies. We constructed a CTBN model with four binary variables arranged in a chain. The first variable randomly switches between its states with an expected transition rate of once per 1 time unit and each of the other variables follows its predecessor on the same time scale. In total, there are 14 parameters in this network. We learned a CTBN structure with increasing amounts of data, and DBNs with varying time granularities. The number of parameters learned can be seen in Figure 6.2. The CTBN learning converges very quickly to the correct number of parameters, and, indeed, to the correct structure. Moreover, as we can see from the error bars, there is very low variance in the structures produced. By contrast, the DBN learning algorithm fluctuates significantly, and does not converge to the right number of parameters even with a large amount of data.

To understand what is happening, we examine some typical structures as shown in Figure 6.3. As we can see, a DBN with time slices much shorter than the average rate of change of the system does converge to a reasonable structure; however, for large amounts of data, the structure still becomes more complex than the corresponding CTBN due to entanglement. For a time granularity on the same order as the time

scale of the system, things become more difficult for the DBN, as it must model multiple transitions in a single time step, leading to entanglement which increases with the amount of available data. Finally, if the time-slicing is too coarse, the DBN learns a model of the steady-state distribution without any model of the transition probabilities.

The problem is that the "right" number of parameters is being determined by our understanding of the structure of the process. But with enough data the DBN will attempt to learn an "equivalent" distribution and the only way it can do that is to eventually learn a fully connected model. Eventually, with enough data, even the $\Delta t = 5$ granularity DBN should be able to support the arcs necessary for a fully connected graph.

6.3 Discussion

In this chapter we have examined the differences between the CTBN and DBN frameworks. While a CTBN is a fully Markovian, direct model of a process transitioning over time, a DBN is more accurately characterized as model of a sequence of state observations. This allows the DBN (with more free parameters) to represent processes that have a non-Markovian character in their evolution between time slices.

DBNs are a good choice for domains where the data is naturally time-sliced and where questions about events occuring between time points are not relevant. However, there are domains where the data has no natural time-slices — e.g., computer system monitoring, life history data analysis, the study of evolution. Such domains are more naturally modelled as CTBNs than DBNs and the estimation of fewer parameters make CTBNs simpler to learn.

It is also worth noting that as CTBNs do not aggregate multiple transitions over the course of a time slice, they avoid entanglement due to aggregation. Thus, they allow us to learn a model that more directly reflects the dependencies in the process.

Finally, as we discussed above, the difference between the nature of the CTBN and DBN frameworks creates a difficulty in evaluating their relative performance. We explored the possibility of interpreting the DBN as a process model in some of our

experiments above.

We will return to the topic of CTBN and DBN comparison briefly in Section 8.7 where we discuss inference and, more deeply in Chapter 9 after we have extended the CTBN representation in Chapter 9. We provide results (in Section 9.5) of an experiment where we compare CTBNs and DBNs by using the CTBN to model a sequence of state observations.

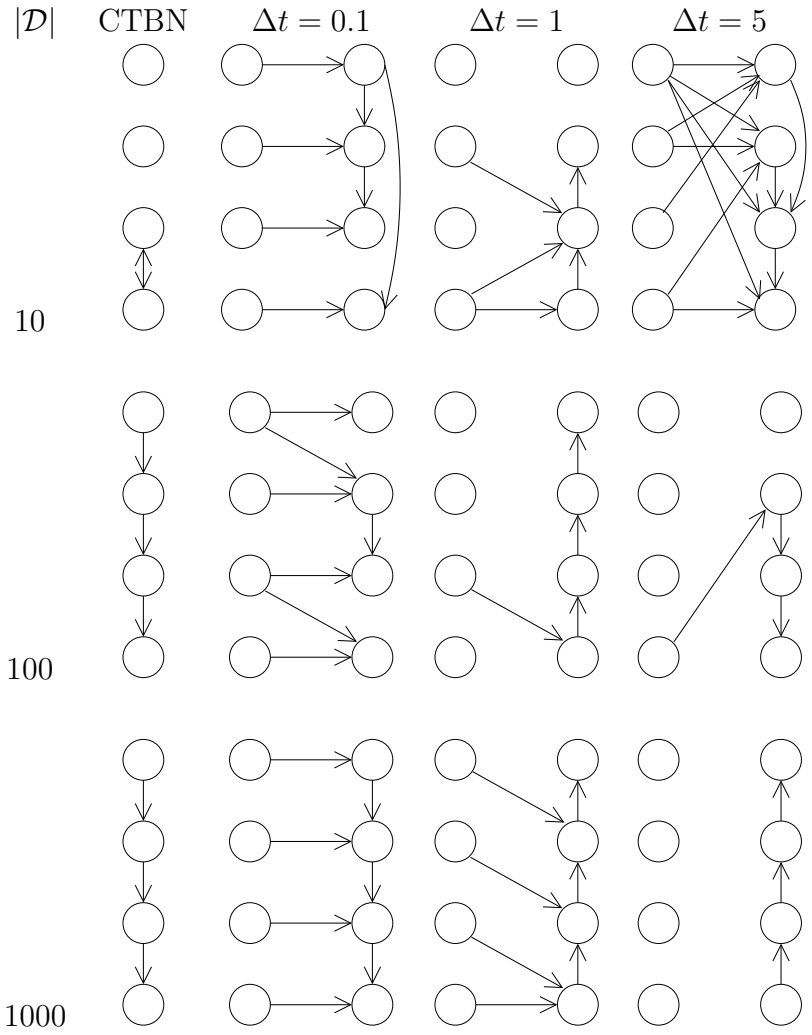

Figure 6.3: Example learned structures for the 4-node chain network.

Chapter 7

Learning with Incomplete Data

As we have seen, there are stringent requirements for having a complete dataset for a CTBN. We must know the value of every variable continuously through time. There are many applications in which we do not have so much information. So, we would like to be able to handle situations where we have incomplete data. As we will see later (in Chapter 9), being able to learn from incomplete data will enable us to learn richer models even in situations where we have complete data.

In BNs, the standard algorithms for dealing with incomplete data are expectation maximization (EM) for learning parameters and structural expectation maximization (SEM) for learning structures. In this chapter, we show how to extend these algorithms to CTBNs. As we will see, the mathematical core of this work is computing expected sufficient statistics, which was described previously in Section 4.4.

7.1 EM for Markov Processes

Though we discussed incomplete data in Section 4.1.2, it will be useful to begin the discussion with the notion of dataset *completion*.

7.1.1 Completions of Data

For any partial trajectory $\sigma[i]$ we can consider a space $\boldsymbol{H}[i]$ of possible completions of that trajectory. Each completion $h[i] \in \boldsymbol{H}[i]$ specifies, for each transition of $\sigma[i]$, which underlying transition of X occurred and also specifies all the entirely unobserved transitions of X. Combining $\sigma[i]$ and $h[i]$ gives us a complete trajectory $\sigma^+[i]$ over X. Note that, in a partially observed trajectory, the number of possible unobserved transitions is unknown. Moreover, there are uncountably many times at which each transition can take place. Thus, the set of possible completions of a partial trajectory σ is, in general, the union of a countably infinite number of spaces, which are real-valued spaces of unbounded dimension. Nevertheless, the notion of all possible completions is well-defined. We can define the set $\mathcal{D}^+ = \{\sigma^+[1], \ldots, \sigma^+[w]\}$ of completions of all of the partial trajectories in $\mathcal{D}$.

7.1.2 Expected Sufficient Statistics and Likelihood

Recall from Chapter 4 that the sufficient statistics of a set of complete trajectories $\mathcal{D}^+$ for a Markov process are $T[x]$ — the total amount of time that $X = x$, and $M[x, x']$ — the number of times X transitions from x to x'. If we let $M[x] = \sum_{x'} M[x, x']$ we can write the log-likelihood for X:

$$\begin{aligned}\ell_X(\boldsymbol{q}, \boldsymbol{\theta} : \mathcal{D}^+) &= \ell_X(\boldsymbol{q} : \mathcal{D}^+) + \ell_X(\boldsymbol{\theta} : \mathcal{D}^+) \\ &= \sum_x \left(M[x] \ln(q_x) - q_x T[x] + \sum_{x' \neq x} M[x, x'] \ln(\theta_{xx'}) \right) .\end{aligned}$$

Let r be a probability density over each completion $\boldsymbol{H}[i]$ which, in turn, yields a density over possible completions of the data $\mathcal{D}^+$. We can write the expectations of the sufficient statistics with respect to the probability density over possible completions of the data: $\bar{T}[x]$, $\bar{M}[x, x']$, and $\bar{M}[x]$. These expected sufficient statistics allow us to

write the expected log-likelihood for X as

$$\begin{aligned}\mathbf{E}_r[\ell_X(\boldsymbol{q}, \boldsymbol{\theta} : \mathcal{D}^+)] &= \mathbf{E}_r[\ell_X(\boldsymbol{q} : \mathcal{D}^+)] + \mathbf{E}_r[\ell_X(\boldsymbol{\theta} : \mathcal{D}^+)] \\ &= \sum_x \left(\bar{M}[x] \ln(q_x) - q_x \bar{T}[x] + \sum_{x' \neq x} \bar{M}[x, x'] \ln(\theta_{xx'}) \right) .\end{aligned}$$

7.1.3 The EM Algorithm

We use the expectation maximization (EM) algorithm (Dempster et al., 1977) to find maximum likelihood parameters $\boldsymbol{q}$, $\boldsymbol{\theta}$ of X. The EM algorithm begins with an arbitrary initial parameter assignment, $\boldsymbol{q}^0$, $\boldsymbol{\theta}^0$. It then repeats the two steps below, updating the parameter set, until convergence. After the kth iteration we start with parameters $\boldsymbol{q}^k$, $\boldsymbol{\theta}^k$:

Expectation Step. Using the current set of parameters, we define for each $\sigma[i] \in \mathcal{D}$, the probability density

$$r^k(h[i]) = p(h[i] \mid \sigma[i], \boldsymbol{q}^k, \boldsymbol{\theta}^k) \tag{7.1}$$

We then compute expected sufficient statistics $\bar{T}[x]$, $\bar{M}[x, x']$, and $\bar{M}[x]$ according to this posterior density over completions of the data given the data and the model.

Maximization Step. Using the expected sufficient statistics we just computed as if they came from a complete data set, we set $\boldsymbol{q}^{k+1}$, $\boldsymbol{\theta}^{k+1}$ to be the new maximum likelihood parameters for our model as follows

$$\begin{aligned} q_x^{k+1} &= \frac{\bar{M}[x]}{\bar{T}[x]} , \\ \theta_{xx'}^{k+1} &= \frac{\bar{M}[x, x']}{\bar{M}[x]} . \end{aligned} \tag{7.2}$$

The difficult part in this algorithm is the Expectation Step. As we discussed, the space over which we are integrating is highly complex, and it is not clear how we can compute the expected sufficient statistics in a tractable way. This problem is the focus of the next section.

7.2 CTBNs

We can now extend the EM algorithm to continuous time Bayesian networks which are a factored representation for homogeneous Markov processes.

7.2.1 Expected Log-likelihood

Extending the EM algorithm to CTBNs involves making it sensitive to a factored state space. Our incomplete data, $\mathcal{D}$, are now partially observed trajectories describing the behavior of a dynamic system factored into a set of state variables $\boldsymbol{X}$.

As shown in Section 4.5, the log-likelihood decomposes as a sum of local log-likelihoods for each variable. Specifically, given variable X, let $\mathbf{U}$ be its parent set in $\mathcal{N}$. Then the sufficient statistics of $\mathcal{D}^+$ for our model are $T[x|\mathbf{u}]$, the total amount of time that $X = x$ while $\mathbf{U} = \mathbf{u}$, and $M[x, x'|\mathbf{u}]$, the number of times X transitions from x to x' while $\mathbf{U} = \mathbf{u}$. If we let $M[x|\mathbf{u}] = \sum_{x'} M[x, x'|\mathbf{u}]$ the likelihood for each variable X further decomposes as

$$\begin{aligned}\ell_X(\boldsymbol{q}, \boldsymbol{\theta} : \mathcal{D}^+) &= \ell_X(\boldsymbol{q} : \mathcal{D}^+) + \ell_X(\boldsymbol{\theta} : \mathcal{D}^+) \\ &= \left[\sum_{\mathbf{u}} \sum_{x} M[x|\mathbf{u}] \ln(q_{x|\mathbf{u}}) - q_{x|\mathbf{u}} \cdot T[x|\mathbf{u}]\right] \\ &\quad + \left[\sum_{\mathbf{u}} \sum_{x} \sum_{x' \neq x} M[x, x'|\mathbf{u}] \ln(\theta_{xx'|\mathbf{u}})\right] . \end{aligned} \tag{7.3}$$

By linearity of expectation, the expected log-likelihood function also decomposes in the say way, and we can write the expected log-likelihood $\mathbf{E}_r[\ell(\boldsymbol{q}, \boldsymbol{\theta} : \mathcal{D}^+)]$ as a sum of terms (one for each variable X) in the same form as Eq. (7.3), except using the expected sufficient statistics $\bar{T}[x|\mathbf{u}]$, $\bar{M}[x, x'|\mathbf{u}]$, and $\bar{M}[x|\mathbf{u}]$.

7.2.2 EM for CTBNs

The EM algorithm for CTBNs is essentially the same as for homogeneous Markov processes. We need only specify how evidence in the CTBN induces evidence on the

induced Markov process, and how expected sufficient statistics in the Markov process give us the necessary sufficient statistics for the CTBN.

A CTBN is a homogeneous Markov process over the joint state space of its constituent variables. Any assignment of values to a subset of the variables forms a subsystem of the CTBN — it restricts us to a subset of the joint state space (as shown with binary "variables" Y and Z in Example 4.1.1). Just as before, our evidence can be described as a sequence of subsystems S_i, each with an associated duration.

Recall that, in a CTBN, the expected sufficient statistics have the form $\bar{T}[x|\mathbf{u}]$ and $\bar{M}[x, x'|\mathbf{u}]$. We can thus replace Eq. (7.2) in the maximization step of EM with:

$$\begin{aligned} q_{x|\mathbf{u}}^{k+1} &= \frac{\bar{M}[x|\mathbf{u}]}{\bar{T}[x|\mathbf{u}]}, \\ \theta_{xx'|\mathbf{u}}^{k+1} &= \frac{\bar{M}[x, x'|\mathbf{u}]}{\bar{M}[x|\mathbf{u}]} \ . \end{aligned} \tag{7.4}$$

The expectation step of EM can, in principle, be done by flattening the CTBN into a single homogeneous Markov process with a state space exponential in the number of variables and following the method described above. In this case, we can compute $\bar{T}[x|\mathbf{u}]$ by summing up all of the expected sufficient statistics $\bar{T}[j]$ for any state j consistent with $X = x, \mathbf{U} = \mathbf{u}$. Similarly, $\bar{M}[x|\mathbf{u}]$ can be computed by summing up all of $\bar{M}[j, k]$ for state j consistent with $X = x, \mathbf{U} = \mathbf{u}$ and k consistent with $X = x', \mathbf{U} = \mathbf{u}$.

However, as the number of variables in the CTBN grows, that process becomes intractable, so we are forced to use approximate inference. The approximate inference algorithm must be able to compute approximate versions of the forward and backward messages $\boldsymbol{\alpha}_t, \boldsymbol{\beta}_w$. It must also be able to extract the relevant sufficient statistics — themselves a sum over an exponentially large space — from the approximate messages efficiently.

In the next chapter, we provide a cluster graph inference algorithm which can be used to perform this type of approximate inference. For each segment $[t_i, t_{i+1})$ of continuous fixed evidence, we construct a cluster graph data structure, whose nodes

correspond to clusters of variables $\boldsymbol{C}_k$, each encoding a distribution over the trajectories of the variables $\boldsymbol{C}_k$ for the duration $[t_i, t_{i+1})$. A message-passing process calibrates the clusters. We can then extract from the cluster $\boldsymbol{C}_k$ both beliefs about the momentary state of the variables $\boldsymbol{C}_k$ at time t_i and t_{i+1}, as well as a distribution over the trajectories of $\boldsymbol{C}_k$ during the interval. The former provide a factored representation of our forward message $\boldsymbol{\alpha}_{t_{i+1}}$ and backward message $\boldsymbol{\beta}_{t_i}$, and are incorporated into the cluster graphs for the adjoining cluster in a forward-backward message passing process. The cluster distributions are represented as local intensity matrices, from which we can compute the expected sufficient statistics over families $X_i, \mathbf{U}_i$, as above. This, this algorithm allows us to perform the steps required for the E-step, and the M-step can be performed easily as described above.

7.2.3 Structural EM for CTBNs

We can also learn structure from incomplete data by applying the structural EM (SEM) algorithm of Friedman (1997) to our setting. We start with some initial graph structure $\mathcal{G}^0$, initial parameters $\boldsymbol{q}^0$, $\boldsymbol{\theta}^0$, and dataset $\mathcal{D}$.

As with SEM for Bayesian networks, there are three steps and we can iteratively alternate among them in any order. They are:

Inference Step. Run an inference algorithm to compute the expected sufficient statistics.

Parameter Update Step. Update the parameters as in Eq. (7.4) according to the currently computed expected sufficient statistics.

Structure Modification Step. Using the current parameterization and expected sufficient statistics, choose a structure modification that increases the score. SEM is used with a variety of scores, most commonly the BIC score or the Bayesian score with expected sufficient statistics as if real. In both cases, the score can be written in terms of expected sufficient statistics, allowing SEM to be used with our algorithm above.

SEM leaves unspecified the issue of how many greedy search steps one takes before recomputing the expected sufficient statistics and parameters. Secction 5.2.2 showed

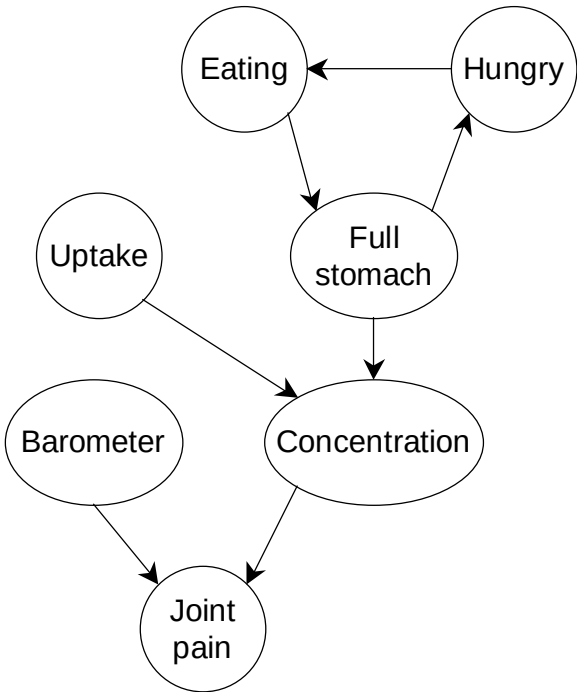

Figure 7.1: Simplified drug effect network.

that, for CTBNs, structure search for a fixed number of parents per node can be done in polynomial time. Thus, it is possible, in this setting, to find the globally optimal structure given the current parametrization in the structure modification step. If one does this, SEM for CTBNs becomes an iterated optimization algorithm with a full maximization step for both structure and parameters.

7.3 Results

We implemented the EM and SEM algorithms described above. We used exact inference by constructing the flattened state space. While this prevents us from solving large problems, it also keeps the analysis of EM separate from that of approximate inference. To verify our algorithms' correctness, we used the a simplified version of the drug effect network as shown in Figure 7.2.3, where all of the variables were binary-valued, for tractability. We sampled increasing numbers of trajectories of 5

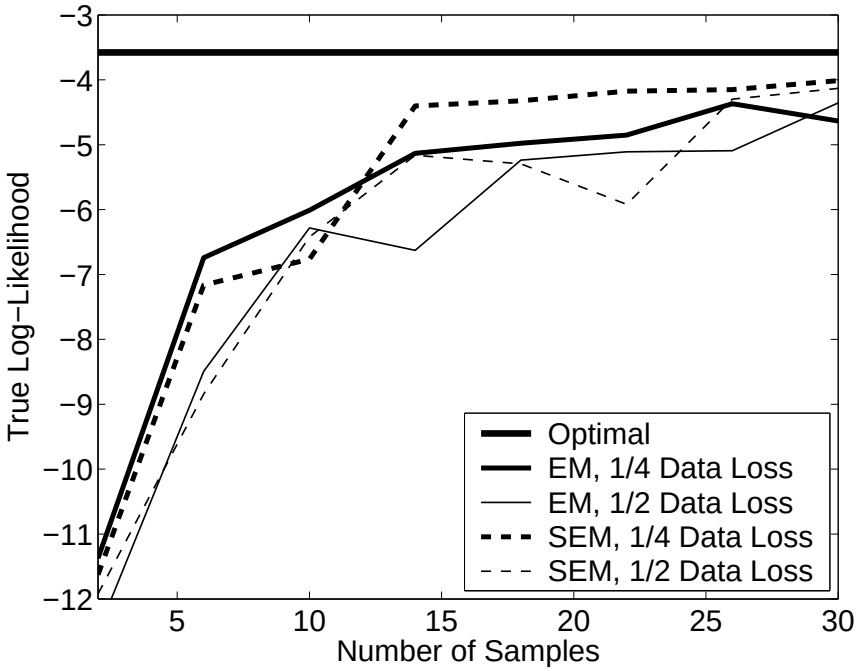

Figure 7.2: Learning results for drug effect net.

time lengths. We ran both EM and SEM, giving the former the true network structure and hiding the structure from the latter. For each data example, we hid parts of the trajectory by selecting time windows of length 0.25 uniformly at random and hiding the value of one variable during the window. We kept dropping data in this fashion until all variables had lost either 1/4 or 1/2 of their total trajectory, depending on the experiment. The results of these experiments are shown in Figure 7.2.3. In some cases SEM outperforms EM because the true structure with learned parameters yields a lower log-likelihood given the amount of data at those points. Note that the horizontal axis represents the amount of data prior to dropping any of it.

The SEM algorithm worked well in this setting. As noted, removing the restriction on acyclicity allows CTBN structure search to decompose. Therefore, at each iteration, we performed a full structure search, which provided marked improvement over a greedy one-step optimization.

7.4 Discussion

In this chapter, we provided algorithms for learning parameters and structure of CTBNs from partially observed data. Given the stringent requirements on fully observed data for CTBNs, this is should significantly increase the range of possible applications for the CTBN framework.

Even in cases where we have fully observed data, the algorithms of this chapter will prove crucial to the wider applicability of CTBNs. But the details must wait until Chapter 9 where we explore the role of adding hidden state to CTBNs in order to greatly increase their expressive power.

Chapter 8

Inference

We have now defined the CTBN framework and shown how to learn models from data. This chapter covers the key remaining task we have yet to explore: inference. We begin with a discussion of the types of queries we might wish to answer and the difficulties of exact inference.

This discussion leads us to approximate inference which can take computational advantage of the independencies encoded in our factored CTBN representation. We present a cluster-graph based message-passing algorithm which is a variant of expectation propagation (Minka, 2001a).

8.1 Queries over a CTBN

In Chapter 3, we showed that we can view a CTBN as a compact representation of a joint intensity matrix for a homogeneous Markov process. Thus, at least in principle, we can use a CTBN to answer any query that we can answer using an explicit representation of a Markov process: We can form the joint intensity matrix and then answer queries just as we do for any homogeneous Markov process, as described in Section 2.3.5.

For example, in the drug effect network, we can set the initial distribution such that the drug was administered at $t = 0$ *hours*, compute the joint distribution over the state of the system at $t = 5$, and then marginalize it to obtain a distribution over

joint pain at $t = 5$. Additionally, because we have the full joint distribution at this point in time, we can calculate for $t = 5$ the distribution over drowsiness given that the concentration of the drug is high.

Now, assume that we have a series of observations. We can compute the joint distribution over the system state for any point in time at or after the time of the last observation. We calculate the new joint distribution at the time of the first observation, condition on the observation, and use that as the initial distribution from which to compute the joint distribution at the next observation time. This process can be executed for an entire series of observations. For example, assume that our patient took the drug at $t = 0$, ate after an hour ($t = 1$) and felt drowsy three hours after eating ($t = 4$). We can compute the distribution over joint pain six hours after taking the drug ($t = 6$) by computing the joint distribution at time 1, conditioning that distribution on the observation of eating, and using that as an initial distribution with which to compute the joint distribution 3 hours later. After conditioning on the observation of drowsiness, we use the result as an initial distribution with which to calculate the joint distribution 2 hours after that. That joint distribution can be marginalized to give the distribution over joint pain given the sequence of evidence. The key is that, unlike in DBNs, we need only do one propagation for each observation time, even if the observations are irregularly spaced.

As noted in section 2.3.5 we can compute the joint distribution between any two points in time. By conditioning on evidence at the later time point, we can propagate evidence backwards in time. Even more interestingly, we can calculate the distribution over the first time a variable X takes on a particular value x: X taking the value x is simply a subsystem of the joint intensity matrix, and we can compute the distribution over the entrance time into the subsystem. For example, we could set our initial distribution to one where the patient takes the drug and has joint pain. We could then directly compute the distribution over the time at which the joint pain goes away. Note that this type of query could also be computed for the time after some sequence of evidence.

8.2 Difficulties with Exact Inference

The obvious flaw in our discussion above is that our approach for answering these queries requires that we generate the full joint intensity matrix for the system as a whole, which is exponential in the number of variables. The graphical structure of the CTBN immediately suggests that we perform the inference in a decomposed way, as in Bayesian networks. Unfortunately, as we now show, the problems are significantly more complex in this setting.

Consider a simple chain $X \rightarrow Y \rightarrow Z$. It might appear that, at any point in time, Z is independent of X given Y. Unfortunately, this is not the case. Even though the transition intensity for Z depends only on the value of Y at any instant in time, as soon as we consider temporal evolution, their states become correlated. This problem is completely analogous to the entanglement problem in DBNs (Boyen & Koller, 1998), where all variables in the DBN typically become correlated over some number of time slices. The primary difference is that, in continuous time, even the smallest time increment Δt results in the same level of entanglement as we would gain from an arbitrary number of time slices in a DBN.

In fact, as discussed in Section 3.4, the only conclusion we can make about a structure $X \rightarrow Y \rightarrow Z$ is that Z is independent of X given the *full trajectory* of Y. As a consequence, we can fully reconstruct the distribution over trajectories of Z, ignoring X, if we are given the full distribution over trajectories for Y. Of course, a full distribution over continuous time processes is a fairly complex structure. One might hope that we can represent it compactly, e.g., using an intensity matrix. Unfortunately, even when the distribution over the joint X, Y process is a homogeneous Markov process, its projection over Y is not a homogeneous Markov process — clearly the transition model for Y changes over time as X changes, hence it can not be homogeneous.

A second potential avenue is the fairly natural conjecture that we do not always need the full distribution over trajectories. Perhaps, if our goal is only to answer certain types of queries, we can make do with some summary over Y. Most obviously, suppose we want to compute the stationary distribution over Z. It seems reasonable

to assume that Z's stationary behavior might depend only on the stationary behavior of Y. After all, the transitions for Z are governed by two matrices $\mathbf{Q}_{Z|y_1}$ and $\mathbf{Q}_{Z|y_2}$. As long as we know the stationary distribution for Y, we know which fraction of the time Z uses each of its transition matrices. So, we should be able to compute the stationary distribution for Z from this information. Unfortunately, this assumption turns out to be unfounded.

Example 8.2.1 *Consider the following intensity matrices.*

$$\mathbf{Q}_Y = \begin{bmatrix} -1 & 1 \\ 2 & -2 \end{bmatrix} \qquad \mathbf{Q}_{Y'} = \begin{bmatrix} -10 & 10 \\ 20 & -20 \end{bmatrix}$$

$$\mathbf{Q}_{Z|y_1} = \begin{bmatrix} -3 & 3 \\ 15 & -15 \end{bmatrix} \qquad \mathbf{Q}_{Z|y_2} = \begin{bmatrix} -5 & 5 \\ 4 & -4 \end{bmatrix}$$

Note that Y and Y' both have the same stationary distribution, $\begin{bmatrix} .667 & .333 \end{bmatrix}$. If we look at the CTBN with the graph $Y \rightarrow Z$ we get a stationary distribution for Z of $\begin{bmatrix} .7150 & .2850 \end{bmatrix}$. But, if we look at the CTBN with graph $Y' \rightarrow Z$, we get a stationary distribution for Z of $\begin{bmatrix} .7418 & .2582 \end{bmatrix}$.

Thus, even the stationary behavior of Z depends on the specific trajectory of Y and not merely the fraction of time it spends in each of its states. We can gain intuition for this phenomenon by thinking about the intensity matrix as an infinitesimal transition matrix. To determine the behavior of Z, we can imagine that for each infinitesimal moment of time we multiply it to get to the next time instance.

At each instant, we check the value of Y and select which matrix we multiply for that instant. The argument that we can restrict attention to the stationary distribution of Y implicitly assumes that we care only about "how many" times we use each matrix. Unfortunately, matrix multiplication does not commute. If we are rapidly switching back and forth between different values of Y we get a different product at the end than if we switch between the values more slowly. The product is different because the order in which we multiplied was different — even if the total number of times we used one matrix or the other were same in both cases. The

product integral can be used to make this argument mathematically precise (Gill & Johansen, 1990).

Thus, there seems to be no easy way in which we can summarize enough information about the distribution over Y's trajectories in order to allow us to reason about Z while ignoring X. More generally, the exact marginal distributions that would form the messages to describe the true joint distribution can be arbitrarily complex, requiring a number of parameters which grows exponentially with the size of the network. Thus, we cannot pass messages exactly without giving up the computational efficiency of the algorithm. We address this issue using the *expectation propagation (EP)* approach (Minka, 2001a; Minka, 2001b; Heskes & Zoeter, 2002), which performs approximate message passing in cluster graphs.

8.3 Algorithm Overview

In the previous section, we argued that we can not generally use exact inference in CTBNs. So, in the remainder of this chapter, we focus on approximate inference. We restrict our attention to task of filtering only because complications arise when attempting to run a version of this algorithm for smoothing.

Our algorithm uses message passing in cluster graphs, of which clique tree algorithms are a special case.

In our algorithm, the clusters do not represent factors over values of random variables. Rather, cluster potentials and messages both encode measures over entire trajectories of the variables in their scope.

EP addresses the problem where messages can be too complex to represent and manipulate by using approximate messages, projecting each message $\delta_{i \to j}$ into a compactly representable space so as to minimize the KL-divergence between $\delta_{i \to j}$ and its approximation $\hat{\delta}_{i \to j}$. In a prototypical example (Minka, 2001a), the cluster potentials and therefore the sepset marginals are mixtures of Gaussians, which are projected into the space of Gaussian distributions in the message approximation step. For messages in the exponential family, $\arg\min_{\hat{\delta}_{i \to j}} \mathbf{D}(\delta_{i \to j} || \hat{\delta}_{i \to j})$ can be obtained by matching moments of the distribution. EP can be applied to clique trees or to general cluster

graphs. Note that, even in clique trees, the algorithm does not generally converge after two passes of message passing (as it does in exact inference), so that multiple iterations are generally required, and convergence is not guaranteed.

In our application of EP, we use conditional intensity matrices (CIMs), reduced to match the evidence, to encode the cluster potentials; we approximate the messages in the cluster graph as homogeneous Markov processes, using a KL-divergence projection.

We note that EP is based on multiply-marginalize-divide scheme of Lauritzen and Spiegelhalter (1988). Since our algorithm performs approximate marginalization so as to minimize KL-divergence, we can use the iterative EP algorithm for message propagation, improving the quality of approximation. As an instance of EP, our algorithm has the property that it converges to fixed points of the approximate free energy function, subject to calibration constraints on the approximate messages.

8.4 Basic Operations

To apply the EP algorithm to clusters whose potentials are encoded as CIMs, we need to define basic operations over CIMs. Recall, from Section 2.2.3 that the basic operations are: multiplication, division, incorporation of evidence, and marginalization. As described above, we will need to rely on *approximate* marginalization.

The first two of these CIM product and division are amalgamation and its inverse — see Definitions 3.3.12 and 3.3.13. But we have not yet discussed incorporation of evidence and approximate marginalization.

8.4.1 Incorporating Evidence into CIMs

Point observations about the system state affect our distribution over the state at a single point in time, which in turn, affects the distribution over the behavior of the system. But they do not affect our distribution over the dynamics as parameterized by the CIMs. By contrast, consider continuous evidence, as in Example 3.3.15. If we condition on the continuous evidence that $A = a_1$ for all $t \in [0, 1]$, then the

dynamics of Z during that interval is described solely by $\mathbf{Q}_{B|a_1}$ rather than a mixture of $\mathbf{Q}_{B|a_1}$ and $\mathbf{Q}_{B|a_2}$. An observation over an interval restricts our transition dynamics to remain within a subset of the full state space for the duration of the interval.

To account for such evidence, we *reduce* the CIM — eliminate the rows and columns of the CIM that correspond to states inconsistent with the evidence. In the special case where we are conditioning $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C}}$ on evidence $\mathbf{e}$ over some variable(s) in the conditioning set $\boldsymbol{C}$, the result is a CIM $\mathbf{Q}_{\boldsymbol{S}|\boldsymbol{C},\mathbf{e}}$ that represents the conditional distribution $\phi(\boldsymbol{S}|\boldsymbol{C},\mathbf{e})$. More generally, when we have evidence $\mathbf{e}_1$ within $\boldsymbol{S}$ and $\mathbf{e}_2$ within $\boldsymbol{C}$, the reduced CIM represents the unnormalized conditional distribution $\phi(\boldsymbol{S},\mathbf{e}_1|\boldsymbol{C},\mathbf{e}_2)$. In this case, the reduced intensity matrix $\mathbf{Q}_{\boldsymbol{S},\mathbf{e}_1|\boldsymbol{C},\mathbf{e}_2}$ will have rows that sum to negative numbers. These negative numbers represent "extra" intensity with which we would normally leave the subsystem (if not for the evidence), and represent the probability flowing out of the subsystem. Note that a reduced intensity matrix $\phi(\boldsymbol{S},\mathbf{e})$ cannot, in general, be normalized and represented as an intensity matrix.

Example 8.4.1 *Consider the system over A and B with graph $A \to B$ described in Example 3.3.15. Recall that the joint intensity matrix using the ordering $\langle A, B\rangle$ for the process is*

$$\mathbf{Q}_{AB} = \begin{bmatrix} -6 & 1 & 2 & 0 & 3 & 0 \\ 2 & -9 & 0 & 3 & 0 & 4 \\ 2 & 0 & -7 & 1 & 4 & 0 \\ 0 & 3 & 2 & -10 & 0 & 5 \\ 2 & 0 & 5 & 0 & -8 & 1 \\ 0 & 3 & 0 & 6 & 2 & -11 \end{bmatrix}.$$

If we want to incorporate the continuous evidence that $B = b_1$ for time $t \in [0, 1]$, we must consider the submatrix corresponding to instantiations consistent with $B = b_1$. Given the variable ordering, that means we must use the first two rows and columns. So the reduced intensity matrix is

$$\mathbf{Q}_{A,b_1} = \begin{bmatrix} -6 & 1 \\ 2 & -9 \end{bmatrix}.$$

As described above, the rows sum to negative numbers, whose magnitude corresponds to the intensity with which we would normally leave the subsystem when $B = b_1$.

8.4.2 Marginalizing CIMs

Clusters in our cluster tree are associated with unnormalized CIMs, perhaps reduced by the incorporation of continuous evidence. In most cases, the marginal dynamics of such a CIM over a subset of variables cannot be described using an unconditional intensity matrix. Indeed, in general, the marginal distribution over a single variable X can only be correctly described by constructing the entire joint intensity matrix, and considering its marginal distribution over X. However, we can approximately marginalize factors — products of (reduced) CIMs — by projecting them into the space of distributions represented as unconditional intensity matrices.

More precisely, suppose we have an unnormalized intensity matrix over $\boldsymbol{S}$ and we are trying to find the projection over the subset of variables $\boldsymbol{V}$. Now, consider the distribution $\phi(\boldsymbol{S}) \propto P^0_{\boldsymbol{S}} \exp(\mathbf{Q}_{\boldsymbol{S}} t)$ described by a (possibly reduced) intensity matrix $\mathbf{Q}_{\boldsymbol{S}}$. This distribution induces a marginal distribution $\phi(\boldsymbol{V})$ over the dynamics of $\boldsymbol{V}$ for any subset $\boldsymbol{V} \subset \boldsymbol{S}$. We would like to project $\phi(\boldsymbol{V})$ onto the space of distributions representable by the intensity matrix $\hat{\mathbf{Q}}_{\boldsymbol{V}}$, by minimizing the Küllback-Leibler divergence; specifically, we want to compute $\arg\min_{\hat{P}_{\boldsymbol{V}}} \mathbf{D}(P_{\boldsymbol{V}} || \hat{P}_{\boldsymbol{V}})$ where $\hat{P}_{\boldsymbol{V}}(t) = P^0_{\boldsymbol{V}} \exp(\hat{\mathbf{Q}}_{\boldsymbol{V}} t)$. As the set of distributions representable by an intensity matrix is in the exponential family, we can minimize the KL-divergence over an interval $[t_1, t_2)$ by choosing $\hat{P}_{\boldsymbol{V}}(t)$ to match the moments of $P_{\boldsymbol{V}}(t)$ over $[t_1, t_2)$.

Importantly, even an unreduced intensity matrix does not define a distribution over trajectories. To define a distribution and the requisite moments, we need an initial state distribution $P^0_{\boldsymbol{V}}$ at time t_1 and the duration of the interval $[t_1, t_2)$. Given a reduced CIM $\phi(\boldsymbol{V}, \mathbf{e})$ over the interval $[t_1, t_2]$, we can obtain the conditional distribution over the system behavior by normalizing the distribution: $\phi(\boldsymbol{V}|\mathbf{e}) = \frac{1}{Z} \exp(\mathbf{Q}_{\boldsymbol{V},\mathbf{e}} \cdot (t_2 - t_1))$, where Z is the *partition function* representing the probability of the evidence: $Z = \int_{t_1}^{t_2} P^0_{\boldsymbol{V}} \exp(\mathbf{Q}_{\boldsymbol{V},\mathbf{e}} \cdot t) dt$. Note that Z is a function of the amount of time the evidence persists and of the distribution $P^0_{\boldsymbol{V}}$ over the state at the beginning

of the evidence. Unlike the case with Gaussians, the integral is always finite.

To match moments, we must compute the expected sufficient statistics over the interval $[t_1, t_2)$ for the variables in $\boldsymbol{S}$. These expected sufficient statistics are $\mathbf{E}[T[j]]$, the expected amount of time in each state j, and $\mathbf{E}[M[j,k]]$, the expected number of transitions from j to k. For simplicity, assume that the evidence is constant throughout the interval. We can compute sufficient statistics for the more general case using a forward-backward algorithm (see chapter 7 for details and derivations — though it is based on the discussion in Section 4.4).

Given the expected sufficient statistics over $\boldsymbol{S}$, we can calculate $\mathbf{E}[T[\boldsymbol{v}]]$, the expected amount of time in each instantiation $\boldsymbol{v}$ of $\boldsymbol{V}$, and $\mathbf{E}[M[\boldsymbol{v}, \boldsymbol{v}']]$, the expected number of transitions from $\boldsymbol{v}$ to $\boldsymbol{v}'$. We also compute the total number of expected transitions from $\boldsymbol{v}$, $\mathbf{E}[M[\boldsymbol{v}]] = \sum_{\boldsymbol{v}'} M[\boldsymbol{v}, \boldsymbol{v}']$. We can now match moments, setting the parameters of $\hat{\mathbf{Q}}_{\boldsymbol{V}}$ to be the maximum likelihood parameters (as in equation 5.1),

$$\begin{aligned} q_{\boldsymbol{v}} &= \frac{\mathbf{E}[M[\boldsymbol{v}]]}{\mathbf{E}[T[\boldsymbol{v}]]}\,, \\ \theta_{\boldsymbol{v}\boldsymbol{v}'} &= \frac{\mathbf{E}[M[\boldsymbol{v}, \boldsymbol{v}']]}{\mathbf{E}[M[\boldsymbol{v}]]}\,. \end{aligned} \tag{8.1}$$

We write $\hat{\phi}(\boldsymbol{V}) = \mathrm{marg}^{P^0,T}_{\boldsymbol{S}\setminus\boldsymbol{V}}(\phi(\boldsymbol{S}))$ for the distribution parameterized by $\hat{\mathbf{Q}}_{\boldsymbol{V}}$.

Example 8.4.2 *Consider the system over A and B described in Example 3.3.15. If we assume a uniform initial distribution and that we want to use this approximation for unit time ($T = 1$), then the matrix of expected sufficient statistics*

$$\bar{M}[(a,b),(a',b')] = \begin{bmatrix} - & .18 & .36 & 0 & .54 & 0 \\ .24 & - & 0 & .35 & 0 & .47 \\ .45 & 0 & - & .23 & .91 & 0 \\ 0 & .42 & .28 & - & 0 & .70 \\ .41 & 0 & 1.03 & 0 & - & .21 \\ 0 & .39 & 0 & .78 & .26 & - \end{bmatrix},$$

and $\bar{T}[(a,b)] = \begin{bmatrix} .18 & .12 & .23 & .14 & .21 & .13 \end{bmatrix}$. *Combining sufficient statistics for* b_1

(rows 1,2), b_2 (rows 3,4), and b_3 (rows 5,6) we get the following matrix of expected sufficient statistics over B

$$\bar{M}[b, b'] = \begin{bmatrix} - & .71 & 1.01 \\ .87 & - & 1.61 \\ .80 & 1.81 & - \end{bmatrix}.$$

and $\bar{T}[b] = \begin{bmatrix} .30 & .37 & .33 \end{bmatrix}$. *With the expected sufficient statistics over B, we can compute the parameters of* $\hat{\phi}(B) = marg_A^{P^0,1}(\phi(A, B))$,

$$\hat{\mathbf{Q}}_B = \begin{bmatrix} -5.73 & 2.37 & 3.36 \\ 2.35 & -6.70 & 4.35 \\ 2.42 & 5.49 & -7.91 \end{bmatrix}.$$

There is an additional subtlety to the computation if $\mathbf{Q}_{\boldsymbol{S}}$ is conditioned on continuous evidence and has negative row sums (representing the probability of the evidence as discussed in Section 8.4.1). In this case, we must account for the extra intensity of leaving the subsystem entirely when computing the expected number of transitions out of each state $\mathbf{E}[M[\boldsymbol{v}]]$. To do so, we add an extra state ι to $\mathbf{Q}_{\boldsymbol{S}}$ before computing the expected sufficient statistics. For each instantiation $\boldsymbol{s}$ of $\boldsymbol{S}$, the intensity of entering the extra state — $q_{\boldsymbol{v}\iota}$ — makes the row sum to zero. Then, when we compute $\mathbf{E}[M[\boldsymbol{v}]]$, we also include $\mathbf{E}[M[\boldsymbol{v}, \iota]]$ the expected number of transitions to the extra state, and use Eq. (8.1).

Note that, as ι does not correspond to any instantiation $\boldsymbol{v}$, we have that $\sum_{\boldsymbol{v}'} \theta_{\boldsymbol{v}\boldsymbol{v}'} < 1$, and therefore the row sums in the resulting intensity matrix will also be negative. This corresponds to the fact that our marginalized intensity matrix approximates the marginal of $P(\mathbf{e}, \boldsymbol{S} \mid \boldsymbol{C})$ in this case. As a result, after computing the expected sufficient statistics, we find that the sum t_* over the time spent in actual states — i.e., not including ι — is less than the total time we are supposed to account for — namely, $t_2 - t_1$. So, we set our normalization c to $(t_2 - t_1)/t_*$.

Example 8.4.3 *Continuing Example 8.4.1, we add a new state ι, resulting in a new*

CIM:

$$\mathbf{Q}_{A,b_1} = \begin{bmatrix} -6 & 1 & 5 \\ 2 & -9 & 7 \\ 0 & 0 & 0 \end{bmatrix},$$

where the last row/column correspond to the absorbing state ι. Assume a uniform initial distribution over the states of A and that we are in this subsystem for total time $T = 1$. Then, calculating the integrals by Runge-Kutta without normalizing yields the unnormalized matrix over transitions of A including the additional state ι,

$$\begin{bmatrix} - & .105 & .526 \\ 0.134 & - & .470 \\ 0 & 0 & - \end{bmatrix},$$

and the unnormalized vector $\begin{bmatrix} .105 & .067 & .828 \end{bmatrix}$ representing the amount of time in each state. The normalization constant $c = 1/(.105 + .067) = 5.81$. So the expected sufficient statistics (given that we spend no time in ι) are

$$\bar{M}[a, a'] = \begin{bmatrix} - & 0.61 & 3.05 \\ 0.78 & - & 2.73 \\ 0 & 0 & - \end{bmatrix},$$

and $\bar{T}[a] = \begin{bmatrix} .61 & .39 & 0 \end{bmatrix}$. When we compute parameters with these statistics, we find that we get back the same $\mathbf{Q}_{A,b_1}$ as above because we have not incorporated any additional evidence. Incorporating evidence will generally lead to a different intensity matrix, as in Example 8.5.1.

8.5 Expectation Propagation

Based on these operations, we can describe a message propagation algorithm for CTBNs. As discussed above, when using approximate projection, we can apply the algorithm iteratively, as in expectation propagation, with the goal of improving our

estimates.

8.5.1 EP for Segments

We first consider the message propagation algorithm for one segment of our trajectory, with constant continuous evidence. The generalization to multiple segments follows.

We first construct the cluster tree for the graph $\mathcal{G}$. This procedure is exactly the same as in Bayesian networks — cycles do not introduce new issues. We simply moralize the graph, connecting all parents of a node with undirected edges, and then make all the remaining edges undirected. If we have a cycle, it simply turns into a loop in the resulting undirected graph. We then select a set of clusters $\boldsymbol{C}_i$. These clusters can be selected so as to produce a clique tree for the graph, using any standard method for constructing such trees. Or, we can construct a loopy cluster graph, and use generalized belief propagation. The message passing scheme is the same in both cases.

Let $\boldsymbol{A}_i \subseteq \boldsymbol{C}_i$ be the set of variables whose factors we associate with cluster $\boldsymbol{C}_i$. Let N_i be the set of neighboring clusters for $\boldsymbol{C}_i$ and let $\boldsymbol{S}_{i,j}$ be the set of variables in $\boldsymbol{C}_i \cap \boldsymbol{C}_j$. We also compute, for each cluster $\boldsymbol{C}_i$, the initial distribution $P^0_{\boldsymbol{C}_i}$ using standard BN inference on the network $\mathcal{B}$. After initialization, the algorithm is

Procedure *CTBN-Segment-EP*$(P^0, T, \mathbf{e}, \mathcal{G})$

1. For each cluster $\boldsymbol{C}_i$
   $\pi_i \leftarrow \prod_{X \in \boldsymbol{A}_i} \phi(X, \mathbf{U}_X, \mathbf{e})$
2. For each edge $\boldsymbol{C}_i$—$\boldsymbol{C}_j$
   $\mu_{i,j} \leftarrow \mathbf{1}$

Loop until convergence:

3. Choose $\boldsymbol{C}_i$—$\boldsymbol{C}_j$
4. *Send-Message*$(i, j, P^0_{\boldsymbol{C}_i}, T)$

Procedure *Send-Message*(i, j, P^0, T)

1. $\delta_{i \to j} \leftarrow \text{marg}^{P^0,T}_{\boldsymbol{C}_i \setminus \boldsymbol{S}_{i,j}}(\pi_i)$
2. $\pi_j \leftarrow \pi_j \cdot \frac{\delta_{i \to j}}{\mu_{i,j}}$
3. $\mu_{i,j} \leftarrow \delta_{i \to j}$

It takes the initial distributions over the clusters P^0, an amount of time T, and possibly some continuous evidence $\mathbf{e}$ which holds for the total time T. We use $\phi(\cdot, \mathbf{e})$ to denote the CIM reduced by continuous evidence $\mathbf{e}$ if applicable. The algorithm iteratively selects an edge (i, j) in the cluster graph, and passes a message from $\boldsymbol{C}_i$ to $\boldsymbol{C}_j$. In clique tree propagation, we might select edges so as to iteratively perform an upward and downward pass. In generalized belief propagation, we might use a variety of message passing schemes. Convergence occurs when messages cease to affect the potentials which means that neighboring clusters $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$ agree on the approximate marginals over the variables $\boldsymbol{S}_{i,j}$.

The basic factor operations are performed as described in Section 8.4. Specifically, let $\rho(\cdot)$ be a function taking factors to their CIM parameterization. For the initial potentials, $\rho(\pi_i)$ is computed by adding the intensity matrices $\mathbf{Q}_{X|\mathbf{U}_X}$ reduced by evidence $\mathbf{e}$ for $X \in \boldsymbol{A}_i$. Also, $\rho(\mathbf{1})$ is an intensity matrix of zeros. Factor product is implemented as addition of intensity matrices, and factor division as subtraction, so that $\rho(\pi_j \cdot \frac{\delta_{i\to j}}{\mu_{i,j}}) = \rho(\pi_j) + \rho(\delta_{i\to j}) - \rho(\mu_{i,j})$. Marginalization is implemented by computing the expected sufficient statistics, using the evidence $\mathbf{e}$, the time period T, and the initial distribution P^0, as described in Section 8.4.2.

Example 8.5.1 *Assume we have a CTBN with 4 binary variables and graph*

$$A \to B \to C \to D$$

with CIMs

$$\begin{array}{ccc} \mathbf{Q}_A & \mathbf{Q}_{B|a_1} & \mathbf{Q}_{B|a_2} \\ \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} & \begin{bmatrix} -1 & 1 \\ 10 & -10 \end{bmatrix} & \begin{bmatrix} -10 & 10 \\ 1 & -1 \end{bmatrix} \end{array},$$

where $\mathbf{Q}_{C|B}$ and $\mathbf{Q}_{D|C}$ have the same parameterization as $\mathbf{Q}_{B|A}$. So A switches randomly between states a_1 and a_2, and each child tries to match the behavior of its parent. Suppose we have a uniform initial distribution over all variables except D which starts in state d_1 and remains in that state for unit time (T=1). Our cluster

tree is AB—BC—CD and our initial potentials are:

$$\rho(\pi_1) = \mathbf{Q}_{AB} = \begin{bmatrix} -2 & 1 & 1 & 0 \\ 1 & -11 & 0 & 10 \\ 10 & 0 & -11 & 1 \\ 0 & 1 & 1 & -2 \end{bmatrix},$$

$$\rho(\pi_2) = \mathbf{Q}_{BC} = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -10 & 0 & 10 \\ 10 & 0 & -10 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix},$$

$$\rho(\pi_3) = \mathbf{Q}_{Cd_1} = \begin{bmatrix} -1 & 0 \\ 0 & -10 \end{bmatrix}.$$

Our initial messages are

$$\rho(\delta_{1\to 2}) = \begin{bmatrix} -2.62 & 2.62 \\ 2.62 & -2.62 \end{bmatrix} \qquad \rho(\delta_{3\to 2}) = \begin{bmatrix} -1 & 0 \\ 0 & -10 \end{bmatrix}$$

These messages leave π_1, π_3 unchanged and give us:

$$\rho(\pi_2) = \begin{bmatrix} -4.62 & 2.62 & 1 & 0 \\ 2.62 & -13.62 & 0 & 10 \\ 10 & 0 & -22.62 & 2.62 \\ 0 & 1 & 2.62 & -13.62 \end{bmatrix}.$$

Our next messages are:

$$\rho(\delta_{2\to 1}) = \begin{bmatrix} -5.02 & 2.62 \\ 2.62 & -8.57 \end{bmatrix} \qquad \rho(\delta_{2\to 3}) = \begin{bmatrix} -4.42 & 3.42 \\ 3.62 & -13.62 \end{bmatrix}.$$

These leave π_2 unchanged and give us

$$\rho(\pi_1) = \begin{bmatrix} -4.40 & 1 & 1 & 0 \\ 1 & -13.40 & 0 & 10 \\ 10 & 0 & -16.94 & 1 \\ 0 & 1 & 1 & -7.94 \end{bmatrix},$$

$$\rho(\pi_3) = \begin{bmatrix} -4.42 & 3.42 \\ 3.62 & -13.62 \end{bmatrix}.$$

Now $\delta_{3\to 2}$ would have no effect on π_2, however,

$$\rho(\delta_{1\to 2}) = \begin{bmatrix} -5.34 & 2.95 \\ 3.31 & -9.26 \end{bmatrix}$$

which changes π_2 so that

$$\rho(\pi_2) = \begin{bmatrix} -4.95 & 2.95 & 1 & 0 \\ 3.31 & -14.31 & 0 & 10 \\ 10 & 0 & -22.95 & 2.95 \\ 0 & 1 & 3.31 & -14.31 \end{bmatrix}.$$

Our next messages are

$$\rho(\delta_{2\to 1}) = \begin{bmatrix} -5.39 & 2.95 \\ 3.31 & -9.16 \end{bmatrix} \qquad \rho(\delta_{2\to 3}) = \begin{bmatrix} -4.43 & 3.43 \\ 3.76 & -13.76 \end{bmatrix}.$$

This gives us

$$\rho(\pi_1) = \begin{bmatrix} -4.45 & 1 & 1 & 0 \\ 1 & -13.45 & 0 & 10 \\ 10 & 0 & -16.85 & 1 \\ 0 & 1 & 1 & -7.85 \end{bmatrix},$$

$$\rho(\pi_3) = \begin{bmatrix} -4.43 & 3.43 \\ 3.76 & -13.76 \end{bmatrix}.$$

At this point we have converged. If we use π_1 to compute the distribution over A at time 1, we get $\begin{bmatrix} .703 & .297 \end{bmatrix}$. *If we do exact inference by amalgamating all the factors and exponentiating, we get* $\begin{bmatrix} .738 & .262 \end{bmatrix}$.

Theoretically, we should be able to generalize this algorithm to do smoothing over trajectories, however there are a number of complications that arise. For example, message passing can lead to a potential with a negative intensity in an off-diagonal entry. When that happens, it is not clear how to proceed. This issue is addressed in later work — see Saria et al. (2007).

8.5.2 Energy Functional

As for any EP algorithm over the exponential family, we can show that the convergence points of the EP algorithm in Section 8.5.1 are fixed points of the constrained optimization of the Kikuchi free energy functional, subject to calibration constraints on the projected marginals.

The Kikuchi free energy function for a cluster graph $\mathcal{G}$ satisfying the running intersection property is

$$\hat{\boldsymbol{F}}[P_{\mathcal{N}}, \hat{P}] = \sum_{\phi \in \mathcal{N}} \mathbf{E}_{\pi_\phi}[\ln \phi] + \sum_{\boldsymbol{C}_i \in \mathcal{G}} \boldsymbol{H}_{\pi_i}(\boldsymbol{C}_i) - \sum_{\boldsymbol{C}_i — \boldsymbol{C}_j \in \mathcal{G}} \boldsymbol{H}_{\mu_{i,j}}(\boldsymbol{S}_{i,j}) \tag{8.2}$$

subject to the constraints:

$$\mu_{i,j} = \text{marg}^{P^0,T}_{\boldsymbol{C}_i \setminus \boldsymbol{S}_{i,j}}(\pi_i) \ . \tag{8.3}$$

Theorem 8.5.2 *A set of potentials π_i, $\mu_{i,j}$ is a stationary point of the maximization of Eq. (8.2) subject to Eq. (8.3) if and only if, for every edge $\boldsymbol{C}_i$—$\boldsymbol{C}_j$ there are potentials of the form $\delta_{i \to j}(\boldsymbol{S}_{i,j})$ such that*

$$\begin{aligned}
\delta_{i \to j} &\propto marg^{P^0,T}_{\boldsymbol{C}_i \setminus \boldsymbol{S}_{i,j}} \left(\pi_i^0 \times \prod_{k \in N_i - \{j\}} \delta_{k \to i} \right) \\
\pi_i &\propto \pi_i^0 \times \prod_{j \in N_i} \delta_{j \to i} \\
\mu_{i,j} &= \delta_{j \to i} \times \delta_{i \to j}
\end{aligned}$$

Corollary 8.5.3 *Convergence points of the procedure* CTBN-Segment-EP *are stationary points of maximizing Eq. (8.2) subject to Eq. (8.3).*

The proof of these results is a special case of the general result on convergence of EP, which applies to any class of distributions in the exponential family.

8.6 Results

In our experiments, we used the drug effect network shown in Figure 3.1 allowing us to compare to the previous inference algorithm. We ran the expectation propagation filtering algorithm and we compared the results of our implementation of expectation propagation with exact inference. We chose an example small enough to allow us to perform exact inference for comparison.

We ran three scenarios. In each one, at $t = 0$, the person modelled by the system experiences joint pain due to falling barometric pressure and takes the drug to alleviate the pain, is not eating, has an empty stomach, is not hungry, and is not drowsy. The drug is uptaking and the current concentration is 0. All scenarios ended

Avg KL-div from Exact	
	EP
No Evidence	
1 segment	0.0629
6 segments	0.0077
Point Evidence	
3 segments	0.0086
6 segments	0.0076

Table 8.1: Average KL-div. between the exact joint distribution and EP, our inference algorithm. The average is computed over 60 time points.

at $t = 6$ (after 6 hours). We compare to exact inference by computing the average KL-divergence as discussed below.

In the first scenario, there was no evidence after the given initial distribution. We ran the algorithms viewing the entire trajectory as a single segment. We tried using one approximation to describe the dynamics over the system and also broke it down into 6 evenly spaced segments. In the second second scenario, we observe at $t = 1$ that the person is not hungry and at $t = 3$, that he is drowsy. We ran the algorithms with 3 segments and again with 6 segments.

Table 8.1 shows the average KL-divergence between exact joint distribution and the approximate joint distribution averaged over 60 evenly spaced time points between $t = 0$ and $t = 6$ for the experiments described above. From the table, one can see the expectation propagation does a reasonable job. We note that there is a significant advantage to breaking the trajectory down into multiple segments.

In the third scenario, we have continuous observations over the variables representing hunger, eating, and drowsiness. After the initial distribution given above, these three variables persisted in their initial state until $t = 0.5$, (half an hour later), after which the person became hungry. At $t = 1$ the person begins to eat. At $t = 1.5$ the hunger is gone and at $t = 2$ the person stops eating. At $t = 2.5$ the person becomes drowsy and these three variables maintain their final value to the end of the trajectory at $t = 6$. We ran the EP forward filter with one segment for each interval of continuous evidence — a total of 6 segments (not evenly spaced). We

again measured the average KL-divergence between the actual and approximate joint distributions as above, measuring at 60 evenly spaced time points between $t = 0$ and $t = 6$. The average KL-divergence was 0.00122. Allowing EP to run for only a single pass instead of going until convergence had a negligible effect — worsening the average KL-divergence by 6.7×10^{-7}. This is not surprising, as we found EP to converge rapidly: of the 6 segments we ran for the continuous evidence, all but one converged within a single pass.

8.7 Discussion

In this chapter we have discussed inference for CTBNs. Exact inference is intractable and compactly summarizing the distributions over trajectories appears infeasible. However, we have presented a well-founded, approximate inference algorithm for CTBNs that allows us to answer a full range of queries including the ability to handle continuous observations. Furthermore, we showed how we can compute a KL-divergence minimizing approximate marginalization of the distribution parametrized by the CIM.

These results enabled us to provide an expectation propagation algorithm for CTBNs which, subject to our approximate marginalizations, converges to stationary points of the approximate free energy function.

One of the most appealing properties of this inference algorithm is that it adaptively selects the time granularity used for reasoning about a cluster based on the rate at which the cluster evolves. Different clusters will be discretized at different granularities, automatically selected by the integration algorithm. The same cluster may be discretized at one granularity in one interval of continuous evidence, and differently in another. By contrast, in DBNs, all variables in the system must be modeled at the time granularity of the variable that evolves most quickly.

Finally, we note that we chose to use one cluster graph for each segment of fixed continuous evidence. As a consequence, each cluster will approximate the trajectory of the variables it contains as a homogeneous Markov process, for the duration of the segment. We can modify the quality of the approximation by either refining or

coarsening our choice of segments. In particular, if a set of variables is changing rapidly, we might want to partition a segment into subsegments, even if the evidence remains constant. Alternatively, we can reduce computational cost by collapsing several intervals of continuous evidence, approximating the trajectory distribution over the entire interval as a homogeneous Markov process. This step requires a more complex computation of sufficient statistics over the combined interval, but is not substantially different. The decision of how to partition time into intervals is analogous to a situation where we are approximating a distribution over continuous variables as a set of Gaussians, each defined over a subset of the space. The choice of how to partition the space into subsets determines the quality of our approximation.

One of the main limitations of this algorithm, aside from the restriction to filtering, is that while time segments can be of different length, all the clusters of variables are broken up over the same segment boundaries. Thus, if one cluster evolves more rapidly than others, requiring a finer-grained approximation, inference in the entire system will have to be approximated with the same segmentation. Moreover, a person must choose how many segments to use in advance.

These issues are explored and handled in later work beyond the scope of this thesis — see Saria et al. (2007).

Chapter 9

Complex Duration Distributions

Accurate modelling of many data sets requires an extension of the work presented thus far. The Markovian assumption restricts the expressive power of CTBNs to modelling exponential distributions over time. The constitutes a significant limitation to the expressive power of our framework since the distribution between state transitions for even a single variable are often more complex than the exponential distribution. However, with the extension of the EM to CTBNs, we can now address this limitation.

9.1 Phase-type Distributions

Phase-type distributions are a rich, semi-parametric class of distributions over durations that use the exponential distribution as a building block. A phase-type distribution is modeled as a set of *phases*, through which a process evolves. Each of these phases is associated with an exponential distribution, which encodes the duration for the process to remain in that phase. That is, we enter a phase k, and then leave in time t exponentially distributed with the parameter q_k associated with that phase. We can view the process as moving over a directed, possibly cyclic graph of nodes each of which represents a phase. Thus, we can create combinations of chains, mixtures, and loops of such exponentially distributed phases linked together in a variety of ways. We spend some amount of time going from one phase to another, but eventually we leave the set of phases altogether. The distribution over when we leave such

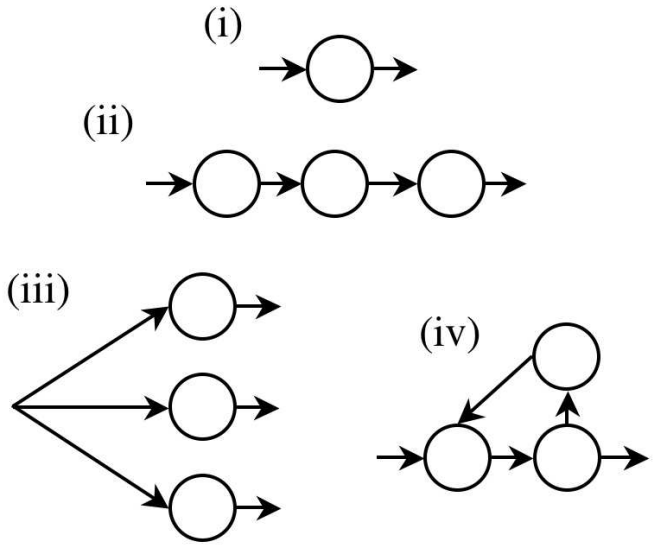

Figure 9.1: Phase transition diagrams for (i) a single exponential phase, (ii) an Erlang (chain), (iii) a mixture, and (iv) a loop.

a system of phases is called a *phase-type distribution.*

As shown by Johnson and Taaffe (1988), phase-type distributions are dense in the set of continuous distributions meaning that, if allowed any number of phases, one can use a phase-type distribution to approximate any continuous distribution arbitrarily closely.

Example 9.1.1 *Consider a 4-state homogeneous Markov process PH_t with intensity matrix*

$$\mathbf{Q}_{PH} = \begin{bmatrix} -q_1 & q_{12} & q_{13} & q_{14} \\ q_{21} & -q_2 & q_{23} & q_{24} \\ q_{31} & q_{32} & -q_3 & q_{34} \\ 0 & 0 & 0 & 0 \end{bmatrix} .$$

If the intensities of states 1, 2, and 3 are non-zero, then regardless of the initial

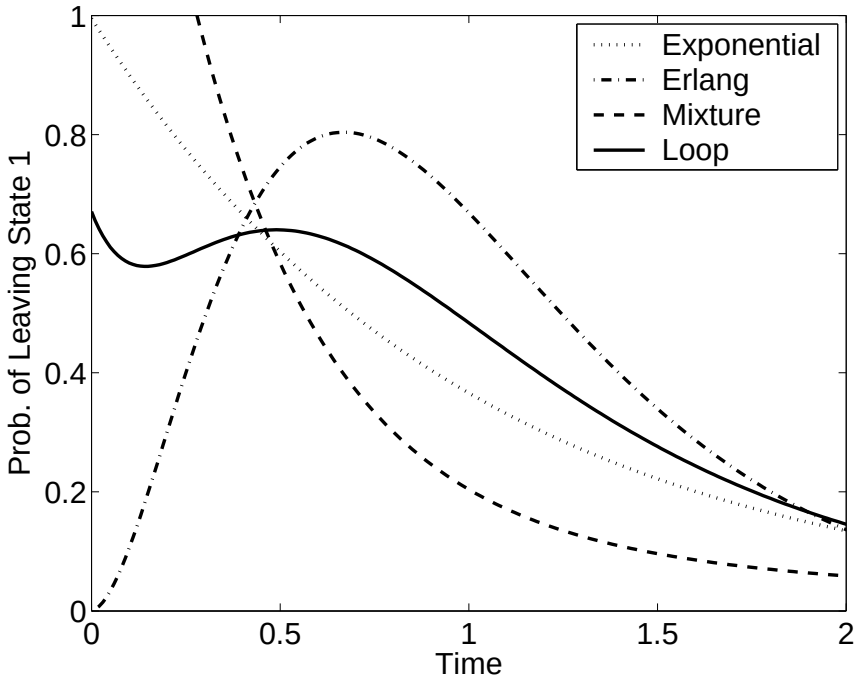

Figure 9.2: The probability density of the first time to leave state 1 (the phase-type distribution), for three example binary variable with 3 phases for both states. All examples have an expected time of 1 to leave state 1.

phase, PH_t will end up in state 4 and remain there. Thus, state 4 is called an absorbing *state and the others are called* transient *states. We call the transient states* phases *and the distribution over when PH_t reaches state 4 is called a* phase *distribution. In this particular case it has 3 phases. If we wanted to encode a chain $1 \rightarrow 2 \rightarrow 3$, we would have all off-diagonal entries equal to 0, except q_{12}, q_{23}, q_{34}. If we wanted to encode a loop, we would also allow $q_{31} \neq 0$. Figure 9 shows some different possible phase transition diagrams and Figure 9 shows some simple distribution shapes that can be formed with 3 phases. Note that, while a chain distribution always begin in phase 1 and ends in some final phase p, general phase-type distributions might start and end in any phase (e.g., the mixture distribution shown).*

Definition 9.1.2 *A* phase-type distribution of p phases *is defined as the distribution over time when a homogeneous Markov process with a single absorbing state and p*

transient phases reaches absorption (Neuts 1975; 1981).

We can specify a p-phase distribution with a $p \times p$ matrix, $\mathbf{Q}_P$, by including only the subsystem of transient phases without losing any information. The rows of the new intensity matrix will have a (possibly) negative row sum, where the missing intensity corresponds exactly to the intensity with which we leave the entire system — i.e., the intensity with which we enter the absorbing state. Phase-type distributions with a single phase are simply exponential distributions.

Definition 9.1.3 *The* Erlangian-p *is a commonly used subclass of phase-type distributions that can be constructed with a chain of p phases, where each phase is visited exactly once — in order — and all phases have the same exit intensity. It thus has a single parameter q. The mean of the Erlangian-p with parameter q is p/q and the variance is p/q^2. (Lipsky, 1992).* ∎

Example 9.1.4 *In an* Erlangian-3 *distribution with the parameter $q = 2$, there are 3 phases and each phase is visited exactly once, in order. Each phase is exponentially distributed with intensity 2, so the intensity matrix for this distribution is*

$$\mathbf{Q}_{E_3} = \begin{bmatrix} -2 & 2 & 0 \\ 0 & -2 & 2 \\ 0 & 0 & -2 \end{bmatrix} .$$

To ensure this intensity matrix will yield an Erlangian-3 distribution we must set the initial distribution to $\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$. Otherwise, we may not go through all 3 phases.

9.2 CTBN Durations as Phase-type Distributions

One can directly model the distribution in state x of variable X in CTBN $\mathcal{N}$ as any *phase-type distribution* instead of an exponential distribution.

When using this *phase modelling* method, the structure of the intensity matrix must be altered by adding *phases* as additional rows (and columns). We use the term *phases* to distinguish additional hidden state in the intensity matrix from states of

the variable. Thus, a subsystem of several phases is used to implement a single state of a variable. In this context there is no absorption and the "final" transition of the phase-type distribution is the transition of the variable to its the next state.

Example 9.2.1 *To make a binary variable W have the duration in each of its 2 states be* Erlangian-3 *distributions (this first with intensity 1 for exiting each of its 3 phases and the second with intensity 2 for exiting each of its 3 phases), we write its intensity matrix as*

$$\mathbf{Q}_W = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & -2 & 2 \\ 2 & 0 & 0 & 0 & 0 & -2 \end{bmatrix} .$$

The top three rows correspond to state w_1 and the bottom three to state w_2. Note that when restricted to modelling with Erlang distributions for a fixed number of phases, the number of free parameters is the same as a regular (exponentially distributed) CTBN.

Using *phase modelling* greatly extends the expressive power of CTBNs and fits naturally within the existing CTBN framework. The basic structure of existing algorithms for CTBNs remains unaltered.

The child of a variable with complex durations sees only the state of its parent and so does not, in general, depend on the current phase of a parent. There are a number of design choices to make in implementing phase-type distributions for durations in CTBNs. Different choices may be appropriate for different applications. One can add phases in a uniform way to each state of each variable — as in example 9.2.1 where each state gets three phases. Alternatively, one might allow some states of some variables to be modelled with more phases than others.

When the parent instantiation changes, one might allow the child in its current state to stay in the same phase or to reset. But there are subtleties with either option.

Suppose we have a phase modelled variable Z with binary parent Y. Suppose we have $Y = y_1$ and $Z = z_1$, and the behavior of Z is controlled by the intensity matrix $\mathbf{Q}_{Z|y_1}$. When the value of Y changes to y_2, the relevant intensity matrix for Z is $\mathbf{Q}_{Z|y_2}$.

Now, suppose $Z = z_1$ has 3 phases. If we allow Z to stay in the same phase when Y transitions, we must be able to map the row of the current phase of $Z = z_1$ in $\mathbf{Q}_{Z|y_1}$ to the corresponding row for $Z = z_1$ in $\mathbf{Q}_{Z|y_2}$. The clearest way to ensure that there are "corresponding" rows is to force consistency in the number of phases allowed for each state across all parent instantiations.

On the other hand, if we want to "reset" the phase for Z's state when Y transitions, we must handle the question about which phase of $Z = z_2$ we start in when transitioning from $Z = z_1$. In particular, there might be a fixed distribution over phases with which one always enters a particular state or that distribution might depend upon the previous state or the current parent instantiation.

In general, we need a function that tells us how the CIM for Z should handle the transition from $Y = y_1$ to $Y = y_2$. The choice of this function can be selected by hand or automatically by EM.

9.3 Using Hidden Variables

An alternate method of allowing complex duration distributions for the states of a variable X in CTBN $\mathcal{N}$ is to introduce a special hidden variable H_X as a parent of X. This has the advantage of being a very clean way to add expressive power to CTBNs. Without this technique, the parents of X must be other variables that we are modelling in our domain which means that the intensities which control the evolution of X can only change when a regular modelled variable changes. The addition of a hidden parent allows us to describe more complex distributions over trajectories of X by allowing the intensities which control the evolution of X to change more frequently.

There are different ways to add a hidden variable H_X as a parent of X. We might force H_X to have no parents, or allow it to have parents in addition to X having parents. However, while adding an explicit hidden variable with clear semantics

might be useful, it is important to realize that all complex duration distributions expressible through use of hidden variables are expressible by direct phase modelling of the state. For example, suppose H_X has 3 states and X has 2 states. Using direct phases, we can rewrite a 6×6 intensity matrix for X having 3 phases for each of its 2 states corresponding to the hidden state of H_X. More generally, we can amalgamate a set of hidden parents $\boldsymbol{H}_X$ and X into a single cluster node $\boldsymbol{S}$, whose parents are X's parents other than $\boldsymbol{H}_X$ along with the parents of $\boldsymbol{H}_X$. Each state of X now corresponds to a set of instantiations to $\boldsymbol{S}$. We can reinterpret the amalgamated CIM for $\boldsymbol{S}$ (given its parents) as a phase-type distribution for X, with $|Val(\boldsymbol{H}_X)|$ phases per state of X.

Conversely, many complex duration distributions expressible by direct phase modelling cannot be expressed using hidden variables. In particular, the joint behavior of H_X and X is restricted in that X and H_X cannot transition simultaneously. This corresponds to the constraint that $\mathbf{Q}_{\boldsymbol{S}}$ must have zeros in locations that correspond to a simultaneous shift in the state of X and H_X. No such constraint holds in direct phase modelling which means that we have more free parameters to describe the distribution.

Thus, we have shown:

Theorem 9.3.1 *For a fixed number of phases p, a CTBN variable X with direct phase modelling for durations, i.e., using $|X| \cdot p$ rows for $\mathbf{Q}_X$, is strictly more expressive than a variable with complex durations modelled by using a hidden parent H_X with p states.*

9.4 Parameter Estimation

We have now described the representational issues involved in modelling complex duration distributions within a CTBN. In general, using a more complex duration distribution involves adding extra parameters to our model. This leads naturally to the question of how we can estimate these parameters.

Our observations of a phase-distributed variable are always partial in that we might observe the current *state* of a variable but never its associated *phase*.

If we restrict our attention to the Erlangian-p distribution for a fixed p, then we have the same number of parameters and we can derive a closed-form equation for the maximum likelihood parameters. (Recall that an Erlangian-p has only only one intensity parameter.)

Theorem 9.4.1 *If we model the duration in the state $X = x$ with parent instantiation $\mathbf{U} = \mathbf{u}$ as an Erlangian-p distribution (for a fixed value of p), Then the maximum likelihood (MLE) parameters can be written as a function of the sufficient statistics as*

$$\begin{aligned} \hat{q}_{x|\mathbf{u}} &= \frac{M[x|\mathbf{u}]}{p \cdot T[x|\mathbf{u}]} \,, \\ \hat{\theta}_{xx'|\mathbf{u}} &= \frac{M[x, x'|\mathbf{u}]}{M[x|\mathbf{u}]} \,. \end{aligned} \tag{9.1}$$

where $\hat{q}_{x|\mathbf{u}}$ is the intensity with which we identically exit each of the p phases.

Proof: As a member of the exponential family, we can derive the formula for the maximum likelihood intensity parameter by matching moments. According to the data, the mean time to transition from $X = x$ given $\mathbf{U} = \mathbf{u}$ is the total time divided by the number of transitions from that state, i.e., $T[x|\mathbf{u}]/M[x|\mathbf{u}]$. The mean of the Erlangian-p distribution with parameter $q_{x|\mathbf{u}}$ is $p/q_{x|\mathbf{u}}$. By setting these means equal to each other, we get the maximum likelihood formula above. Note that this proof is wholly analogous to the alternative derivation given in the proof of Theorem 5.1.1. Moreover, the use of the Erlangian distribution to model the duration does not affect the multinomial distribution over the state to which we transition. Thus, for the multinomial, the formula (and the proof) for the maximum likelihood estimation is identical to the one given earlier. ∎

For more general phase-type distributions, we use EM for parameter estimation. This algorithm has been discussed in Chapter 7. However, we can sometimes get away with using a simpler variant of the full EM algorithm. In particular, there are many situations where the system we want to learn is actually fully observed. However, exponential duration distributions may not be a good fit, and so we may want to use general purpose phase modelling. In such instances we can use *local-EM*.

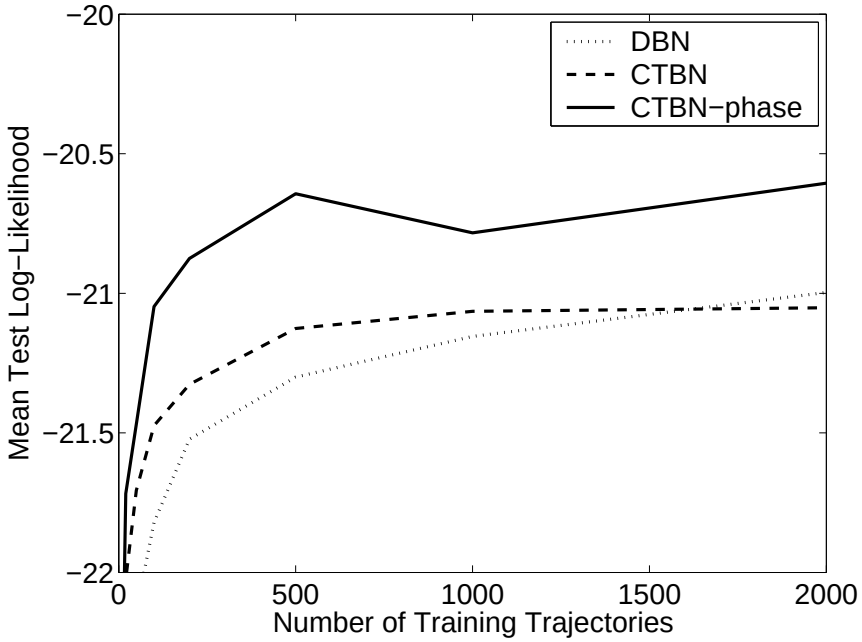

Figure 9.3: Learning results for British Household Panel Survey.

Local-EM is a computationally lightweight variant of EM in which we do not need to run inference over the whole network for the E-step — we can just look at each variable along with its parents. Essentially, if we have N variables we run EM over N small, independent networks. Each separate network includes one variable with its parents.

If our observations of the system are not complete, then we must use the regular version of the EM algorithm, viewing it as a regular partially observed process.

9.5 Results

We ran SEM on the British Household Panel Survey (Economic and Social Research Council (ESRC) Research Centre on Micro-social Change, 2003). This data is collected by the British government by yearly asking thousands of residents of Britain about important events in their lives. They track when these events occurred and provide this data anonymously and separately for each member of the survey. Over

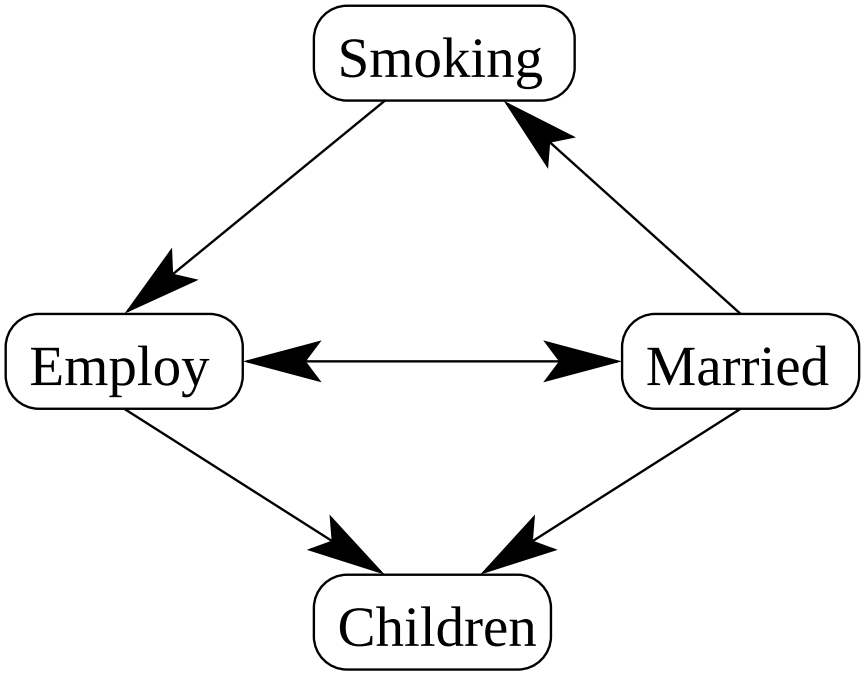

Figure 9.4: Learned BHPS network (200 training points).

8000 people are currently in the dataset. We randomly divided this set into 4000 training examples and 4000 testing examples (each example is a trajectory of a different person). Because we are employing exact inference, we had to keep the variable set small and chose 4 variables: employ (ternary: student, employed, unemployed), married (binary: not married, married), children (ternary: 0, 1, 2+), and smoking (binary: non-smoker, smoker). The average number of events per person is 5.6.

We learned structures and parameters for a time-sliced DBN with a time-slice width of 1 year, a standard CTBN, and a CTBN with 2 phases for every state of every variable. No restrictions were placed on the structure of these 2 phases (so, in general, they form a loop) and the phase for a variable resets when the parent changes.

In order to compare CTBNs to DBNs, (a model that cannot predict the exact timing of events), we sampled the testing data at the same yearly rate and calculated the probability of these sampled trajectories, a value both CTBNs and DBNs can

predict. The results are shown in Figure 9.4. The DBN and plain CTBN models are comparable, with the DBN doing better with more data due to its increased flexibility (due to intra-time slice arcs, DBNs have more potential parameters). However, the phase-type distributions increase the performance of the CTBN model; the trajectories are approximately twice as likely as with the other two models.

Figure 9.4 shows a learned exponential CTBN network. The parameters are interesting. For example, the rate (intensity) with which a person stops smoking given that they have two or more children is three times the rate at which a childless person quits smoking. The rate at which a person begins smoking given that they have no children is 300 times the rate of a person with two or more children. The rate at which a person becomes unemployed (after having been employed) tends to decrease with more children. But that "job security" is most apparent in people who are married. The rate of becoming unemployed also tends to be less if one smokes.

9.6 Discussion

This chapter addresses one of the primary limitations of the CTBN model. Although reasonable in some cases, an exponential model is a poor fit for many real-life domains (such as the interval between getting married and having children). Indeed, our experimental results show that CTBNs parameterized with the richer class of phase-type distributions significantly outperform both CTBNs and DBNs on a real-world domain.

Although one can learn discrete-time models that use the finest level of granularity, the geometric duration distribution of such models (which is particularly marked in fine-grained time models) can be a poor fit to the true duration distribution. Conversely, as we showed in Chapter 6, using an overly coarse granularity can also lead to artifacts in the learned model.

We have shown that it is possible to learn, in closed form, the parameters for the class of Erlangian-p distributions. We have described local-EM which is powerful and computationally lightweight method for learning fully general phase distributions in the context of complete data.

Full EM as described in Chapter 7 allows us to learn general phase distributions even when we have only partial information about a system.

Chapter 10

Related Work

In this chapter, we discuss work related to this thesis. This presentation is organized into three sections. We begin with other work on modelling time with Bayesian networks. We continue with work on event history analysis and Markov process models. The chapter concludes with a discussion of related work that bears directly on the CTBN framework.

Some of the related work arguably belongs in more than one section. In such cases, we have listed the work only once.

10.1 Bayesian Networks and Time

A number of methodologies have been developed to handle temporal reasoning using the general BN framework. Chief among these is the DBN framework (Dean & Kanazawa, 1989) which we have discussed and compared with CTBNs in Chapter 6 and Section 9.5.

Hanks et al. (1995) present another discrete time approach to temporal reasoning — related to DBNs — which they extend with a rule-based formalism to model endogenous changes to variables that occur between exogenous events. These endogenous changes add a flavor of continuous time to the framework since the number of endogenous changes that can occur between the discrete time exogenous events is not fixed. However, there is no unified graphical model and there is no direct method for

calculating state transition probabilities. The only approach to inference discussed is stochastic simulation that explicitly focuses on likely scenarios.

Arroyo-Figueroa and Sucar (1999) present *Temporal Nodes Bayesian Networks* (TNBNs), which are specifically designed for fault diagnosis and prediction in specific domains and are not presented as a general temporal model. The approach is based on time intervals — where events can only happen in one of a small number intervals. A TNBN is made up of a set of nodes, each representing an event that can happen only once. Each event has a "normal" or default state and a (small) set of relative time intervals in which the event may occur. Arcs represent causal temporal relationships; so, nodes without parents represent possible trigger events which can cause other events associated with a fault or some "disturbance". A child event in the network has different distributions over its values (i.e., over the possible time intervals in which it might occur) depending on which cause activated it. Inference is done by instantiating known events within known time intervals (where the timing of a trigger event gives a basis for determining the actual times for the relative intervals) and propagating the evidence to generate a posterior distribution. However, if an event were to occur at a time outside any of intervals specified for it, the corresponding node would have to be instantiated as simply not occuring in the required time frame. Each TNBN is engineered by domain experts for a particular problem.

Tawfik and Neufeld (1994) present *Temporal Bayesian Networks* which allow time to be modelled as discrete or continuous. Probabilities are represented as functions of time. However, this framework only allows the calculation of distributions over the state at a given time — not distributions over time. Inference is simplified by making the assumption that the different instants in time are independent. That is, observations at time t do not affect the distribution at t' for $t \neq t'$. Possible methods of relaxing this assumption are only described at a high level. Also, the problem of learning probabilities as functions over time is not addressed.

Colbry et al. (2002) present *Quantitative Temporal Bayesian Networks* (QTBNs). This framework augments standard DBNs with a methodology based on Time Nets (Kanazawa, 1991) to allow for the representation of qualitative time interval constraints on the occurence of events. These do not constrain what happens, only what

is *supposed* to happen as we monitor the behavior of a system. Regular DBN inference is used to propagate the distribution forward in time and there is some additional computational overhead to check whether the system is behaving according to the desired constraints once one has the distribution over a new instant in time. The problem of learning these models is not addressed.

Finally, Tawfik and Neufeld (2000) provide an overview of techniques used for modelling time with Bayesian networks. They focus on questions of representation. Inference is discussed in terms of patterns of reasoning with a few small examples but no general algorithms. The most notable exception is in the case where one assumes that distinct time points are independent. Learning tasks are not discussed. Significantly, they include mention of continuous time survival analysis as a method of modelling the time to occurrence of significant events in causal models. However, temporal distributions generated by these survival functions are not used for modelling state durations of local variables. The resulting models are analogous to standard event history analysis.

10.2 Event History Analysis and Markov Process Models

There are many different methodologies used for continuous time modelling. We have discussed (finite state) continuous time Markov proceses extensively as the basis of the CTBN framework.

There is a broad class of related methods we refer to as event history analysis (Blossfeld et al., 1988; Blossfeld & Rohwer, 1995; Andersen et al., 1993) though other terms — e.g., survival analysis, hazard models, failure analysis, duration analysis, etc. — are used in the literature as well. Such models are used to examine and model the distribution over time between two significant events. For example, these events might be the birth and death of a person or the manufacture and failure of a part. While external factors (e.g., does the person smoke) may be included for more

accurate analysis, the focus is modelling some particular duration. Note that phase-type duration distributions have been used extensively with these methods (Aalen, 1995).

A general theme with these models is that there is some distinguished event that is of central interest as opposed to CTBNs where we are modelling the transient (i.e., moment-to-moment) dynamics of a structured stochastic system.

In *counting process* models (Aalen, 1975; Andersen et al., 1993), the distinguished event is repeatable and we are interested in the distribution over how many times it has occurred. There are multivariate versions of these models where we are interested in the distribution over how many times a set of events has occurred (Andersen & Borgan, 1985). Bayesian statistical methods have been used for inference in these models (Hjort, 1986; Aven, 1986) but, crucially, they have not used a factored representation which would allow modelling of conditional independencies.

In *queueing theory* (Gross & Harris, 1998; Lipsky, 1992), there are two distinguished, repeatable events: arrivals to and departures from a queue. The amount of time between an arrival and the corresponding departure from the queue is the service time. Queues are often characterized by the distribution over the occurrence of arrival events and the distribution over the service time. Individual queues or service centers can be combined into a queueing network (Gross & Harris, 1998) that has structure — i.e., some service centers lead only to certain other service centers. But this is not a dependency structure. In particular, the behavior of each service center does not depend on the current state of any other service center.

10.3 CTBNs

We now turn to works that are more directly related to the contributions of this thesis.

Our definition of a conditional Markov process (in Section 3.1.1) is similar to a model introduced and used by Lando (1998), but the conditioning variables there were static — they were not viewed as Markov processes, nor were they built into any larger structured model, as in our framework.

The method we present in Section 4.4 for computing expected sufficient statistics is a generalization the work of Asmussen et al. (1996) who utilize numerical integration via the Runge-Kutta method. However, their work is limited to the case of a stand-alone phase-type distribution with an absorbing state. Holmes and Rubin (2002) provide an alternative approach to computing expected sufficient statistics using the eigenvalue decomposition of the intensity matrix.

Gopalratnam et al. (2005) extend the basic CTBN representation by allowing durations to be modelled as Erlang-Coxian distributions, which is a limited subclass of general phase distributions. In particular, it does not allow for the exponential phases to be looped. Restriction to subclasses makes learning from data easier; in particular, EM is not required necessary to learn distributions of this class (as we saw in Section 9.4 for the Erlangian subclass of phase distributions). However, these subclasses have several drawbacks, including reduced expressivity, especially with small numbers of phases (Asmussen et al., 1996). Since our general method from Chapter 9 is based on the EM algorithm, it allows the use of general phase-type distributions in CTBNs without restriction.

In DBNs and HMMs, the general idea of representing complex state durations by the use of multiple (geometrically distributed) phases leads to a form of *semi-Markov* models as described by Durbin et al. (1998) and Murphy (2002). These techniques have not typically allowed the phases to be looped back on each other and such feedback loops are an important part of the expressive power of phase-type distributions (Asmussen et al., 1996).

More generally, semi-Markov models (Lévy, 1954; Howard, 1971; Blossfeld et al., 1988) can include arbitrary distributions over the time until some variable changes state. These types of models are compelling because, in many domains, simple distributions over durations do not provide a good fit to data. If we are only interested in modelling one variable, choosing an arbitrary distribution over when it will transition is less of a problem. We can even allow the arbitrary distribution to be a function of other factors — if they are, themselves, fixed. The difficulty is how to define a globally coherent probabilistic semantics over a process where we have different variables transitioning with arbitrary distributions. It is unclear how to determine which

variable in the global system will transition next and when that transition will occur.

Ng et al. (2005) extend the CTBN representation to allow for a hybrid state model — i.e., one that includes nodes with finite state spaces and nodes with continuous state spaces. They use the hybrid CTBN to model a remote rover which they use as a testbed for a continuous time particle filtering algorithm which they introduce. The computational advantages of the continuous time model over discrete time are discussed — in particular, the way that computation is only necessary when observations arrive instead of being required on a fixed schedule.

Boudali and Dugan (2006) use CTBNs to develop a framework for analysis of reliability. They develop a closed-form solution for the reliability of a sample system and compare it favorably as a generalization of earlier work using a discrete time model.

Friedman and Kupferman (2006) consider CTBNs over systems with components that evolve over widely differing time scales — i.e., some components evolve very quickly compared to others. They show a theoretically useful method for reducing the size of such CTBNs for purposes of approximate inference.

Finally, El-Hay et al. (2006) develop a new framework of *continuous time Markov nets* (CTMNs) based on CTBNs. The CTMN framework factors a continuous time Markov process over an undirected graph. Two nodes, X and Y, that are connected by an undirected arc mutually depend upon one another, so X—Y in the CTMN can be viewed as the cycle $X \leftrightarrows Y$ in a corresponding CTBN. Thus, CTMNs represent a subclass of the processes representable by CTBNs. But, through a clever parametrization, they guarantee that the stationary distribution is representable as a factored state — unlike the situation in standard CTBNs (where the stationary distribution is generally entangled). They also provide a reduction of CTMNs to CTBNs in order to make use of CTBN algorithms for inference, allowing them to focus on presenting learning algorithms for the new framework.

Chapter 11

Conclusion

11.1 Brief Summary

This thesis has introduced CTBNs, a new framework for modelling continuous time over a factored state. It allows us to model processes without needing to choose an arbitrary temporal granularity. It allows us to use cyclic graphs which are a natural way to model mutual influence and can make knowledge engineering easier.

We have developed principled learning algorithms that learn CTBN models from both fully and partially observed data. We also have a polynomial time structure search algorithm (when limited to some fixed maximum number of parents per node).

We have explored the difficulties of exact inference and provided a inference algorithm which leverages our factored state to gain computational advantage. It takes additional advantage of the fact that each node has a model of how fast it expects to change. This enables us to focus computational resources on parts of the system that need it most.

We have shown how to model state durations for individual CTBN variables as phase-type distributions — an extraordinarily expressive class. Finally, we have compared our framework with DBNs, noting that there are complex issues involved in evaluating them comparatively.

11.2 Future Work

The introduction of a new continuous time framework for BN reasoning opens up a substantial amount of potential future work. There are many open questions that can be fruitfully explored to expand upon the work of this thesis.

Over the years the basic BN representation has been augmented with hierarchical structure, objects and relations, actions and rewards. These extensions could and should eventually be incorporated into CTBNs. Continuous state CTBNs and hybrid continuous state / discrete state CTBNs, pose an interesting potentially useful line of future work. Many other inference algorithms can be explored, including sampling based methods.

There is more work to be done investigating the ways that phase-type distributions can be used to model complex durations. That includes more work examining the impact of phase distributions on learning and inference.

In the current CTBN model, every *event* is simply the transition of a state variable from one value to another. But in many systems of interest, there would seem to be a useful, somewhat broader, notion of events. On the one hand, these may be one-time or "single-shot" events that have far reaching consequences on the behavior of a system. We could conceivably model these as binary variables but they would have many children and that does not seem to capture the intent of such variables. This might include the more direct incorporation of survival analysis, counting processes, or even queueing theory models with the CTBN framework.

Another approach might be to allow the CTBN to be a factored *inhomogeneous* Markov process instead. Here the local intensity might be expressed as functions of time instead of constant. This adds significant complexity to the model and much work would likely be needed to make design choices — e.g., limiting the set of available functions over time — in order to construct a usable model. However, such work would open the door to periodic CTBNs as well. Periodicity would be useful in capturing recurring events that have significant impact on timing of events, such as rush hour traffic.

Another useful direction would be working to develop a better framework for

understanding the relationships between different models of temporal reasoning and allowing for clearer evaluation metrics to assess the circumstances under which different types of models are best used. As we discussed in Chapter 6, this is particularly difficult when attempting to compare the accuracy of inference or quality of a learned model between a continuous time and discrete time methodology.

A clearer theory of evaluation might allow us to develop a single framework which would allow the use of continuous time nodes, discrete time nodes (of varying granularities), periodic nodes, one time event nodes, and static nodes — any of which with discrete or continuous state. Such a framework would be even more powerful if supported by learning algorithms capable of selecting the appropriate type for each node.

Ultimately, such a framework might allow us to get meaningful answers to questions about temporal processes directly from data without the need to choose a specific flavor of model. That would, in turn, put these tools into the hands of far more people.

Bibliography

Aalen, O. O. (1975). *Statistical inference for a family of counting processes*. Doctoral dissertation, Department of Statistics, University of California, Berkeley.

Aalen, O. O. (1995). On phase-type distributions in survival analysis. *Scandinavian Journal of Statistics*, *22*, 447–463.

Andersen, P. K., & Borgan, Ø. (1985). Counting process models for life history data: a review (with discussion). *Scandinavian Journal of Statistics*, *12*, 97–158.

Andersen, P. K., Borgan, Ø., Gill, R. D., & Keiding, N. (1993). *Statistical models based on counting processes*. New York: Springer-Verlag.

Arroyo-Figueroa, G., & Sucar, L. (1999). Temporal bayesian network for diagnosis and prediction. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (pp. 13–20).

Asmussen, S., Nerman, O., & Olsson, M. (1996). Fitting phase-type distributions via the EM algorithm. *Scandinavian Journal of Statistics*, *23*, 419–441.

Aven, T. (1986). Bayesian inference in a parametric counting process model. *Scandinavian Journal of Statistics*, *13*, 87–97.

Billingsley, P. (1995). *Probability and measure*. New York: John Wiley and Sons, Inc. Third edition.

Blossfeld, H.-P., Hamerle, A., & Mayer, K. U. (1988). *Event history analysis: statistical theory and application in the social sciences*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Blossfeld, H.-P., & Rohwer, G. (1995). *Techniques of event history modeling — new approaches to causal analysis*. Mahwah, New Jersey: Lawrence Erlbaum Associates.

Boudali, H., & Dugan, J. B. (2006). A continuous-time Bayesian network reliability modeling, and analysis framework. *IEEE transactions on reliability*, 86–97.

Boyen, X., & Koller, D. (1998). Tractable inference for complex stochastic processes. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (pp. 33–42).

Chickering, D. M., Geiger, D., & Heckerman, D. (1994). *Learning Bayesian Networks is NP-Hard* (Technical Report MSR-TR-94-17). Microsoft Research.

Colbry, D., Peintner, B., & Pollack, M. E. (2002). Execution monitoring with quantitative temporal bayesian networks. *Proceedings of the Sixth International Conference on AI Planning and Scheduling*.

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

Dempster, A., N.M.Laird, & D.B.Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, *39*, 1–38.

Duffie, D., Schroder, M., & Skiadas, C. (1996). Recursive valuation of defaultable securities and the timing of resolution of uncertainty. *The Annals of Applied Probability*, *6*, 1075–1090.

Durbin, R., Eddy, S., Krogh, A., & Mitchison, G. (1998). *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge University Press.

Economic and Social Research Council (ESRC) Research Centre on Micro-social Change (2003). British household panel survey. Computer Data File and Associated Documentation. http://iserwww.essex.ac.uk/bhps. Colchester: The Data Archive.

El-Hay, T., Friedman, N., Koller, D., & Kupferman, R. (2006). Continuous time markov networks. *Proceedings of the Twenty-second Conference on Uncertainty in AI (UAI)*. Boston, Massachussetts.

Friedman, N. (1997). Learning belief networks in the presence of missing values and hidden variables. *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning* (pp. 125–133). Morgan Kaufmann Publishers Inc.

Friedman, N., & Kupferman, R. (2006). Dimension reduction in singularly perturbed continuous-time Bayesian networks. *Proceedings of the Twenty-second Conference on Uncertainty in AI (UAI)*. Boston, Massachussetts.

Geiger, D., & Heckerman, D. (1995). *A characterization of the dirichlet distribution with application to learning Bayesian networks* (Technical Report MSR-TR-94-16). Microsoft Research.

Gihman, I. I., & Skorohod, A. V. (1973). *The theory of stochastic processes II*. New York: Springer-Verlag.

Gill, R. D., & Johansen, S. (1990). A survey of product-integration with a view towards applications in survival analysis. *The Annals of Statistics*, *18*, 1501–1555.

Gopalratnam, K., Kautz, H., & Weld, D. S. (2005). Extending continuous time Bayesian networks. *AAAI05: Proceedings of the Twentieth National Conference on Artificial Intelligence*.

Gross, D., & Harris, C. M. (1998). *Fundamentals of queueing theory*. New York: John Wiley and Sons, Inc. Third edition.

Hanks, S., Madigan, D., & Gavrin, J. (1995). Probabilistic temporal reasoning with endogenous change. *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*.

Heckerman, D. (1995). *A tutorial on learning with Bayesian networks* (Technical Report MSR-TR-95-06). Microsoft Research.

Heckerman, D., Geiger, D., & Chickering, D. M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, *20*, 197–243.

Heskes, T., & Zoeter, O. (2002). Expectation propagation for approximate inference in dynamic Bayesian networks. *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*.

Hjort, N. L. (1986). Bayes estimators and asymptotic efficiency in parametric counting process models. *Scandinavian Journal of Statistics*, *13*, 63–85.

Holmes, I., & Rubin, G. M. (2002). An expectation maximization algorithm for training hidden substitution models. *J. Mol. Biol.*, *317*, 753–764.

Howard, R. (1971). *Dynamic probabilistic systems*, vol. I & II. New York: John Wiley and Sons, Inc.

Huang, C., & Darwiche, A. (1996). Inference in belief networks: A procedural guide. *International Journal of Approximate Reasoning*, *15*, 225–263.

Johnson, M. A., & Taaffe, M. R. (1988). *The denseness of phase distributions* (Research Memoranda 88-20). Purdue School of Industrial Engineering.

Kanazawa, K. (1991). A logic and time nets for probabilistic inference. *AAAI91: Proceedings of the Ninth National Conference on Artificial Intelligence*.

Karlin, S., & Taylor, H. M. (1998). *An introduction to stochastic modeling*. San Diego, California: Academic Press. Third edition.

Lam, W., & Bacchus, F. (1994). Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, *10*, 269–293.

Lando, D. (1998). On Cox processes and credit risky securities. *Review of Derivatives Research*, *2*, 99–120.

Lauritzen, S., Dawid, A., Larsen, B., & Leimer, H.-G. (1990). Independence properties of directed Markov fields. *Networks*, *20*, 579–605.

Lauritzen, S., & Spiegelhalter, D. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, *50*, 157–224.

Lévy, P. (1954). Processus semi-Markoviens. *Proceedings of the International Congress of Mathematicians* (pp. 416–426). Amsterdam, North-Holland.

Lipsky, L. R. (1992). *Queueing theory: A linear algebraic approach.* New York: Macmillan Publishing Company.

Minka, T. P. (2001a). Expectation propagation for approximate bayesian inference. *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence* (pp. 362–369). Morgan Kaufmann Publishers Inc.

Minka, T. P. (2001b). *A family of algorithms for approximate Bayesian inference.* Doctoral dissertation, MIT.

Moler, C., & Loan, C. V. (2003). Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, *45*, 3–49.

Murphy, K. (2002). *Dynamic Bayesian networks: representationm, inference, and learning.* Doctoral dissertation, University of California, Berkeley.

Neuts, M. F. (1975). Probability distributions of phase type. *Liber Amicorum Prof. Emeritus H. Florin* (pp. 173–206).

Neuts, M. F. (1981). *Matrix-geometric solutions in stochastic models — an algorithmic approach.* Baltimore: John Hopkins University Press.

Ng, B., Pfeffer, A., & Dearden, R. (2005). Continuous time particle filtering. *Proceedings of the Nineteenth International Joint Conference on AI.* Edinburgh, UK.

Nodelman, U., & Horvitz, E. (2003). *Continuous time Bayesian networks for inferring users' presence and activities with extensions for modeling and evaluation* (Technical Report MSR-TR-2003-97). Microsoft Research.

Nodelman, U., Koller, D., & Shelton, C. R. (2005a). Expectation propagation for continuous time Bayesian networks. *Proceedings of the Twenty-first Conference on Uncertainty in Artificial Intelligence* (pp. 431–440).

Nodelman, U., Shelton, C. R., & Koller, D. (2002). Continuous time Bayesian networks. *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (pp. 378–387).

Nodelman, U., Shelton, C. R., & Koller, D. (2003). Learning continuous time Bayesian networks. *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence* (pp. 451–458).

Nodelman, U., Shelton, C. R., & Koller, D. (2005b). Expectation maximization and complex duration distributions for continuous time Bayesian networks. *Proceedings of the Twenty-first Conference on Uncertainty in Artificial Intelligence* (pp. 421–430).

Norris, J. R. (1997). *Markov chains*. Cambridge: Cambrdige University Press.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems*. Morgan Kauffman.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipes in C*, chapter 16. Cambridge University Press. Second edition.

Rabiner, L. R., & Juang, B. H. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine*, 4–16.

Saria, S., Nodelman, U., & Koller, D. (2007). Reasoning at the right time granularitty. *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence*.

Tawfik, A. Y., & Neufeld, E. M. (1994). Temporal Bayesian networks. *Proceedings of the First International Workshop on Temporal Representation and Reasoning*.

Tawfik, A. Y., & Neufeld, E. M. (2000). Temporal reasoning and bayesian networks. *Computational Intelligence*, *16*, 349–377.

Yedidia, J. S., Freeman, W. T., & Weiss, Y. (2000). Generalized belief propagation. *NIPS* (pp. 689–695).