International Journal of Approximate Reasoning 90 (2017) 253–271

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Disjunctive interaction in continuous time Bayesian networks [☆]

Logan Perreault [*], Monica Thornton, John Sheppard [**], Joseph DeBruycker

Computer Science Department, Montana State University, Bozeman, MT 59717, United States

ARTICLE INFO

Article history:
Received 15 October 2016
Received in revised form 4 July 2017
Accepted 17 July 2017
Available online 9 August 2017

Keywords:
Continuous time Bayesian networks
Disjunctive interaction
Noisy-OR
Conditional intensity matrix

ABSTRACT

A continuous time Bayesian network is a probabilistic graphical model capable of describing discrete state systems that evolve in continuous time. Unfortunately, the number of parameters required for each node in the graph is exponential in the number of parents of the node, which can be prohibitively large for many real-world systems. To mitigate this problem, disjunctive interaction is proposed as a method for reducing the number of required parameters from exponential to linear. In this work, the relation between disjunctive interaction and standard parameterization techniques is explored both theoretically and experimentally. Experimental results demonstrate that inference over models with disjunctive interaction exhibits greater scalability with no degradation in accuracy.

1. Introduction

Continuous time Bayesian networks (CTBNs) are a probabilistic graphical model that provide a powerful framework for modeling systems of discrete variables that evolve in continuous time [1]. Using a CTBN, it is possible to obtain information regarding the expected probability distribution over the states of a variable as a function of time. Work has been done to improve both the representational power of the model, as well as the supporting algorithms that aid in reasoning over complex and uncertain domains [2]. To date, CTBNs have been employed successfully in a number of diverse, real-world applications [3–5]. The positive results for the model have increased interest, and its popularity as a temporal model has increased in recent years [6]. This is largely due to the explicit continuous time representation, which has advantages over its discrete-time counterparts such as the dynamic Bayesian network [7]. Despite their demonstrated utility, many domains have yet to adopt the CTBN framework for practical use. This may be due to the computational complexity associated with CTBNs, which can become unmanageable when working with large models [8].

A CTBN represents a system of variables as the nodes in a directed graph. The transition behavior of each variable is encoded using parameters that describe the transition rates between states over time. Let $\mathbf{U}$ be the set of parent nodes for a node X in the graph. The behavior of X depends on the current state of the variables $\mathbf{U}$. Furthermore, the number of parameters required to specify X is exponential in the size of $\mathbf{U}$. While this is a non-issue for toy problems with few variables, many real-world problems consist of variables whose behavior is dictated by many other factors. These types of

[☆] This paper is part of the Virtual special issue on Uncertainty Reasoning, Edited by Robert E. Mercer and Salem Benferhat.
[*] Principal corresponding author.
[**] Corresponding author.
E-mail addresses: logan.perreault@montana.edu (L. Perreault), monica.thornton@msu.montana.edu (M. Thornton), john.sheppard@montana.edu (J. Sheppard), joseph.debruycker@msu.montana.edu (J. DeBruycker).

http://dx.doi.org/10.1016/j.ijar.2017.07.011
0888-613X/© 2017 Elsevier Inc. All rights reserved.

problems containing large numbers of dependencies correspond to models with large parent sets. As the size of the parent sets increase, the required number of parameters may make the tasks of learning and inference intractable.

For example, consider a node X with five parents $\mathbf{U} = \{A, B, C, D, E\}$. Transition behavior for X must be specified for each combination of state assignments to $\mathbf{U}$, including $\{a_0, b_0, c_0, d_0, e_0\}$, $\{a_0, b_0, c_0, d_0, e_1\}$, etc. If each parent has three states, then the total number of unique state assignments to the parents is $3^5 = 243$. Each of these instantiations require a separate parameterization of X to describe its transition behavior under the given parent states. Even with this relatively small number of parents and states, it is apparent that scalability is a concern with respect to the size of parent sets.

The computational complexity associated with a large number of dependencies is an issue common among all probabilistic graphical models, and is not limited to the CTBN framework [9–11]. In other frameworks, this issue has been addressed by exploiting features of the model that allow for alternate representations [12]. Within Bayesian networks the notion of disjunctive interaction, also called the Noisy-OR assumption, provides a method for compact parametric representation, reducing the number of required parameters to be linear in the size of the largest parent set rather than exponential. This reduction in complexity is made possible by the assumption that each parent is sufficient to cause an effect in the child. Furthermore, no effect can occur without a parent as a causal mechanism. This assumption is often reasonable for real-world systems that behave according to causal relationships. Unfortunately, the topic of disjunctive interaction has only recently been addressed in the CTBN literature, and the established approach employed in Bayesian networks does not extend to CTBNs due to the introduction of temporal dynamics [13].

Given that CTBNs are by nature more complex and contain a larger number of parameters, the more succinct parameterization afforded by disjunctive interaction could prove especially advantageous in this framework. To this end, we propose a modification to CTBN representation that allows the framework to take advantage of disjunctive interaction, capitalizing on the features of the model in a manner that reduces the complexity associated with its parameterization. This allows for the representation of larger systems within the CTBN framework, which will support its adoption in new domains focusing on novel problems.

In this work, we present a method for describing disjunctive interaction as it applies to CTBNs. Section 2 provides a formal description of the CTBN model, and Section 3 offers an overview of the work most relevant to our approach. In Section 4 we detail our approach to parameterizing a CTBN using generalized disjunctive interaction, and in Section 5 the process for converting between disjunctive and standard parameterization is provided. Section 6 demonstrates how disjunctive interaction works in the special case where all nodes are binary, which allows for assumptions to be made that simplify the computations required to parameterize the model. Section 7 describes how an existing CTBN inference algorithm can be adapted to make use of the disjunctive model representation provided in this work. The approach is tested in Section 8 via a series of experiments. The first set of experiments compares the query results of a traditionally parameterized CTBN with one parameterized using disjunctive interaction. The second set of experiments are intended to investigate the scalability of models that make use of disjunctive interaction. The final section provides a summary of our approach, along with directions for future work in this topic.

2. Background

This section provides the background information necessary to place the contributions of this work in context. To begin, the mathematical foundation upon which CTBNs are based, the continuous time Markov process, is described. Next, the CTBN model itself is described, with a specific focus on the conditional intensity matrix (CIM) representation. For a more thorough description of CTBNs, the reader is referred to the existing literature, especially the work of Nodelman [7].

2.1. Continuous time Markov process

A continuous time Markov process (CTMP) provides a framework for modeling a system of discrete-state variables that evolve in continuous time [14]. By modeling the temporal process of variables, it is possible to determine the probability distribution over the states of the system at any point in time.

Let X be a CTMP. X is comprised of two components: an initial distribution P and an intensity matrix Q. P is a vector that defines the probabilities of the process starting in each state at the start of time $t = 0$. Q is a matrix that defines the transition behavior between the states over time. An entry $q_{i,j}$ in row i, column j of the matrix Q indicates the expected intensity with which the process transitions from state i to state j. P and Q are shown below for a process consisting of n states.

$$P = (p_1, p_2, \cdots, p_n) \qquad \mathbf{Q} = \begin{array}{c c} & \begin{array}{cccc} x_1 & x_2 & \cdots & x_n \end{array} \\ \begin{array}{c} x_1 \\ x_2 \\ \vdots \\ x_n \end{array} & \begin{pmatrix} q_{1,1} & q_{1,2} & \cdots & q_{1,n} \\ q_{2,1} & q_{2,2} & \cdots & q_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ q_{n,1} & q_{n,2} & \cdots & q_{n,n} \end{pmatrix} \end{array}$$

Each $p_i \in P$ represents the probability of starting in state i. The non-diagonal entries in the matrix Q can be any non-negative value, while the diagonals are constrained to equal the negative sum of the remaining entries in the row. More formally, $q_{i,i} = -\sum_{j \neq i} q_{i,j}$, so each row in the matrix sums to zero. The probability that the process transitions from state i to state j is defined by an exponential distribution with a rate of $q_{i,j}$. The probability density function for this distribution is as follows. Note that this is a positive distribution, requiring that the value for time t be non-negative.

$$f(t; q_{i,j}) = q_{i,j} \exp(-q_{i,j} t)$$

A CTMP can be thought of as a generative model, capable of producing a sample of how the system might behave over time. A sample, also commonly referred to as a trajectory, defines the state of the process for the entire interval of time (t_s, t_e), where t_s is the start time, and t_e is the end of a period of interest. A trajectory σ can be generated by first sampling from the initial distribution P to determine the state of the system at time t_s. The next state of the system can be obtained using the intensities in the row corresponding to the current state. The transition times to each of the other states can be sampled from exponential distributions with the rates in the corresponding columns. The soonest transition time can then be taken, and the process can be repeated, with state and transition times recorded for the entire time interval (t_s, t_e).

Sampling from individual exponential distributions for each state and taking the minimum is referred to as the method of “racing exponentials.” Alternatively, a single transition time can be drawn from an exponential distribution with a rate equal to the sum of each of the intensities in the off-diagonal entries in the row of an intensity matrix. Given the restriction that the diagonal entry be equal to the negative sum of the remaining row, this rate can be obtained by taking the absolute value of the diagonal entry. The time drawn from this distribution is the expected time to transition to any of the other states. This is due to the property that the minimum of a set of exponential random variables is defined by an exponential distribution with a rate equal to the sum of the rates for each of the variables in the set. After obtaining the transition time in this fashion, determining the location of the transition can then be done by drawing from a multinomial distribution defined by the relative difference in rates between the states.

Although the CTMP is a highly expressive model, it has a major drawback that makes it impractical for many scenarios. The problem arises when modeling multiple variables. In that case, each state x_i in the CTMP X represents a specific state instantiation to all of the variables being considered. Let $\mathbf{V}$ be the set of variables in the process, and $|V|$ be the number of states associated with variable $V \in \mathbf{V}$. Then the number of states $|X|$ that are needed to represent all possible state instantiations is as follows:

$$|X| = \prod_{V \in \mathbf{V}} |V|. \tag{1}$$

In other words, the number of states in a CTMP is exponential in the number of variables being modeled, meaning that the size of the initial distribution P and intensity matrix Q are also exponential in size. For this reason, a CTMP is not a scalable solution. If multiple variables are to be modeled, a more compact representation is required.

2.2. Continuous time Bayesian network

The continuous time Bayesian network (CTBN) is a model closely related to the CTMP that attempts to address the issue of scalability by providing a more compact representation. This improvement is made possible by exploiting conditional independencies between variables in the system. Unlike a CTMP where each variable is assumed to depend on every other variable, a CTBN focuses on identifying *independencies* and making them explicit, thus reducing the representational size in the process.

CTBNs use a directed, potentially cyclic graph G to encode direct variable dependencies. The nodes correspond to variables $\mathbf{V}$, while the edges describe dependence relationships. The transition behavior of a node depends directly on the current state of that node's parents in the graph. An example of a CTBN graph structure is shown in Fig. 1. This model is intended to describe the impact of a drug on the body through time.

A CTBN is parameterized by specifying an initial distribution and a conditional intensity matrix (CIM) for each node in the graph G [1]. The initial distribution is a vector of probabilities much like the one used to describe a CTMP, except the initial distributions that are assigned to each node define a probability distribution over the states of the associated variable only. A CIM is used to define the transition behavior for a node over time. This CIM is similar to an intensity matrix, except that transition behavior may change as a function of the states of a node's parents in G. Using this parameterization, each node in the network can be thought of as a conditional Markov process. Nodelman et al. provide a more formal description of a conditional Markov process in [7].

“A conditional Markov process X is an inhomogeneous Markov process whose intensity matrix varies with time, but only as a function of the current values of a set of discrete conditioning variables $\mathbf{U}$. Its intensity matrix, called a conditional intensity matrix (CIM), is written $\mathbf{Q}_{X|\mathbf{U}}$ and can be viewed as a set of homogeneous intensity matrices $\mathbf{Q}_{X|\mathbf{u}}$ – one for each instantiation of values $\mathbf{u}$ to $\mathbf{U}$.” [1]

By defining a conditional Markov process for each variable in the system, the size of the parameterized model can be reduced greatly. The size of the initial distributions for each node corresponds to the number of states for that individual

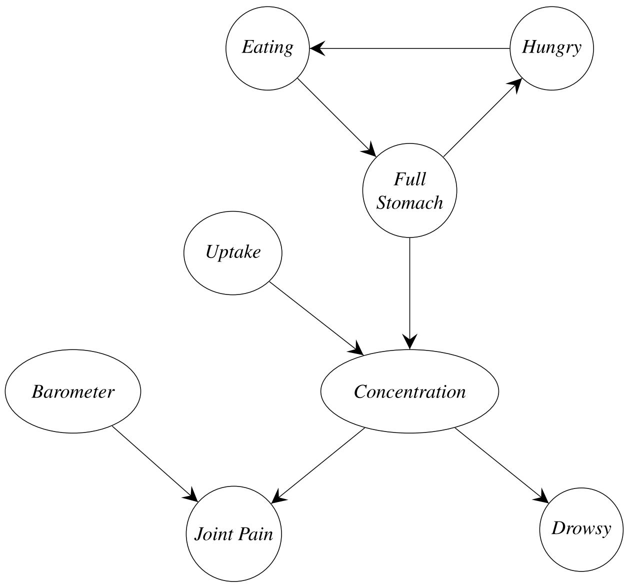

Fig. 1. CTBN corresponding to the Drug Effect Network [1].

variable. As a result, the total number of probabilities required to specify the initial distribution of the process is equal to the sum of the states for each variable; or, $\sum_{V \in \mathbf{V}} |V|$. Compare this summation to the product of state counts shown in Equation (1). While the space complexity for a CTMP initial distribution is exponential in the number of variables, it is linear for CTBNs.

A similar reduction is possible when specifying the rates of the intensity matrix. The number of entries in an intensity matrix for a CTMP is

$$n_{ctmp} = \left(\prod_{V \in \mathbf{V}} |V|\right)^2 .$$

Alternatively, the total number of entries in the CIMs of a CTBN over the same set of variables is

$$n_{ctbn} = \sum_{V \in \mathbf{V}} \left(\left(\prod_{U \in \mathbf{U}} |U|\right) \times |V|^2\right), \tag{2}$$

where $\mathbf{U}$ is the set of parents for node V in the graph. If these parent sets are smaller than the total number of nodes in the graph, then $n_{ctbn} < n_{ctmp}$, and often $n_{ctbn} \ll n_{ctmp}$. In other words, so long as the behavior of a variable only depends on a subset of the other variables in the system, a CTBN is able to model the process more compactly.

Referring back to the drug effect network, a conditional Markov process is specified for each of the eight nodes in the network. As an example, consider the *Joint Pain* node, which has parents *Barometer* and *Concentration*. For the sake of conciseness, these nodes can be referred to using their first initials J, B and C. Node J can be parameterized with an initial distribution of size $|J|$, and a CIM where the number of rows/columns for each intensity matrix is also $|J|$. In accordance with the definition for a conditional Markov process, the CIM can be viewed as a set of homogeneous intensity matrices, one for each unique state instantiation of the parent nodes. The CIM for node J will therefore consist of $|B| \times |C|$ intensity matrices. If it is assumed that $|B| = 3$ and $|C| = 2$, then there are $3 \times 2 = 6$ homogeneous intensity matrices that make up the CIM for node J as follows.

$$\mathbf{Q}_{J|\{B,C\}} = \{\mathbf{Q}_{J|\{b0,c0\}}, \mathbf{Q}_{J|\{b0,c1\}}, \mathbf{Q}_{J|\{b1,c0\}}, \mathbf{Q}_{J|\{b1,c1\}}, \mathbf{Q}_{J|\{b2,c0\}}, \mathbf{Q}_{J|\{b2,c1\}}\} \tag{3}$$

Although the CTBN representation may be substantially more compact than a CTMP over the same set of variables, in many situations the model may still be unmanageably large. The product in Equation (2) accounts for the requirement that all parent instantiations must be enumerated. If a node in a CTBN has a large number of parents, then storing and using a CIM for the node may be difficult due to the exponential number of homogeneous intensity matrices. In the worst case, all nodes may depend on every other node. In this case, no independencies can be exploited, and a CTBN provides no benefits over a CTMP representation. For CTBNs with large parent sets, a more concise representation may be required. In this work, disjunctive interaction is introduced as a partial solution to this problem.

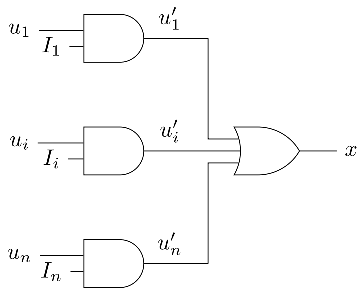

Fig. 2. Line failure model for the Noisy-OR model in Bayesian networks.

3. Related work

The vast majority of work to date on disjunctive interaction in probabilistic graphical models has focused on its use in Bayesian networks. As originally proposed by Pearl, the binary Noisy-OR model for Bayesian networks can be applied to any system that meets the assumptions of *accountability* and *exception independence* [12]. In this context, state zero is treated as *False*, while state one is *True*. Accountability refers to the notion that at least one parent must be *True* if its child is observed to be *True*, which holds because the model can be interpreted as causal. Exception independence means that the factors that inhibit causation are independent of one another, ensuring that each exception to the system's normal behavior can be interpreted as an independent variable. Another way to interpret this is as a series of AND gates for each parent, where the inputs include the cause and a negated inhibitor mechanism. The output of these AND gates becomes the input to a Boolean OR gate, which produces the binary Noisy-OR model. This representation is shown in Fig. 2, where each value I_i is drawn from a binary probability distribution defined by the inhibitor probability, and the function F is defined to be the standard OR gate. These inhibitor probabilities are what gives the Noisy-OR model its name, in that they introduce noise to an otherwise deterministic function. Additionally, Pearl shows how parameters in the conditional probability tables can be calculated on-the-fly using this model, thereby reducing the number of probabilities that need to be stored to describe system behavior.

The Noisy-OR model has been applied successfully to a variety of real-world problems that require modeling disjunctive interaction in Bayesian networks. Oniśko et al. use disjunctive interaction to reduce the number of parameters that must be specified for use in small datasets [15]. The authors learn the reduced set of parameters in the Bayesian network by using a small set of patient records for the purpose of diagnosing liver disorders. Murphy and Mian review methods for learning dynamic Bayesian networks for the purpose of modeling gene expression data and describe how disjunctive interaction allows for compact parameterization of the conditional probability tables in the networks [16]. Bobbio et al. show how to convert fault trees into Bayesian networks [17]. These fault trees naturally exhibit disjunctive interaction, and the authors take advantage of this property to produce a more condensed representation.

Perhaps the most successful application of Noisy-OR occurs in the medical domain, where the faults in the network are considered to be a disease, while the tests are referred to as findings. Noisy-OR played a major role in the development of the Quick Medical Reference, Decision-Theoretic (QMR-DT) diagnostic model [18]. This is a bipartite Bayesian network consisting of a set of diseases connected to a set of findings and constructed using expert knowledge in the medical domain. Heckerman demonstrated the effectiveness of the Noisy-OR parameterization in Bayesian networks by applying it to the QMR-DT model [19]. It was here that Heckerman first introduced an inference algorithm called *quickscore*, which was specifically designed to exploit the structure of binary Noisy-OR. The algorithm is specifically designed to query for the probability of an effect given a set of positive findings or negative findings that act as causal mechanisms for the effect. The algorithm is exponential in the number of positive findings, and only works in the case of binary variables using an OR gate.

There have been several extensions to the original formulation of disjunctive interaction in Bayesian networks. Henrion extends the original model by relaxing the assumption that a variable may only enter the *True* state if a parent is in the *True* state [20]. This is achieved by introducing another synthetically produced parent called a "leak" node that is always on. This leak node represents the potential for unmodeled events to cause a *True* child state. Srinivas further generalizes disjunctive interaction by allowing for multi-state variables, as opposed to the binary restriction that was originally imposed [21]. Srinivas also relaxed the original model to allow the OR gate in Fig. 2 to be any discrete function mapping the inputs $u'_1, \ldots, u'_n$ to the output x. Finally, Srinivas discussed several specific discrete functions that allow for more tractable inference. Heckerman et al. discuss disjunctive interaction within the context of a more general framework called causal independence. This describes the notion of any independence that can be used to simplify the exponential parameterization required for the conditional probability tables in a Bayesian network [22–24]. Finally, the work presented here extends work by Perreault et al. [13], where in this paper, a more formal treatment of disjunctive interaction in CTBNs is provided. Furthermore, this work provides a description of how disjunctive interaction can be used to compactly parameterize CTBNs, even in the case where the variables are non-binary.

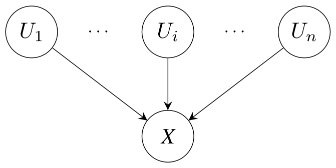

Fig. 3. Generic parent-child relationship.

4. Disjunctive interaction

Although the CTBN is representationally powerful, it is incapable of scaling to models that contain nodes with a large number of parents. The space complexity of the system primarily depends on the node with the largest number of parents, with all other nodes contributing potentially far less to the space requirements of the model. With this in mind, a generic network containing a single child node and n parent nodes is introduced in this section for demonstrative purposes. This network can be thought of as a standalone CTBN, or as a subnetwork in a larger model. The graph structure for this network is shown in Fig. 3.

The number of homogeneous intensity matrices required to describe a CIM for node X is as follows:

$$|\mathbf{Q}_{X|\mathbf{U}}| = \prod_{i=0}^{n} |U_i|. \tag{4}$$

It is this exponential number of intensity matrices that prevents model scalability. CTBNs provides a more compact representation of a CTMP by using conditional independence between variables. Similarly, it is possible to exploit special characteristics of the data in a CTBN to provide an even more compact representation. This provides a method for avoiding the exponential space complexity for cases where this data characteristic applies.

Disjunctive interaction between the variables in a CTBN can be used to simplify model representation. Let X be a node in a CTBN, and let $\mathbf{U}$ be the set of parents for that node. We say that there is a disjunctive interaction between X and its parents if the assumptions of *accountability* and *exception independence* hold. We describe these two assumptions as they exist in a temporal context with an arbitrary number of variable states, which varies slightly from the traditional definitions provided by Pearl [12].

The accountability assumption refers to the notion that a transition between the states of X may occur only as a result of the state of one of the parents in $\mathbf{U}$. In other words, if each of the parents in $\mathbf{U}$ are expected to cause a zero transition rate from a state x_i to x_j, then the final transition rate should be zero as well, since there has been no sufficient cause to explain the transition. If it is desired for X to have some rate of transitioning despite the absence of a cause, an additional synthetic node can be added to the parent set of X that represents the notion of unmodeled causes. This synthetic node is referred to as a “leak” node, in that it captures all other possibilities and explicitly represents otherwise implicit behavior.

As an example, consider the *Joint Pain* variable from Fig. 1, which depends on its parents *Barometer* and *Concentration*. Accountability holds if and only if barometric pressure and drug concentration are the only factors that will produce a change in the state of joint pain. Given that there are likely other factors that contribute to the a person’s joint pain, a leak node *Joint Pain Leak* would need to be added as a parent of *Joint Pain* to account for these other factors while still maintaining the accountability assumption.

The assumption of exception independence stipulates that if a transition event from x_i to x_j occurs, the mechanisms that cause such a transition are independent from one another. This means that any parent $U \in \mathbf{U}$ is sufficient to produce transition rates between the states of a variable X on its own, without the assistance of any other parent. Furthermore, if multiple parents of X are expected to cause a transition from x_i to x_j, then when considered simultaneously, they will cause the transition to occur at a higher rate than considering the parents individually.

Again, the *Joint Pain* variable from Fig. 1 is used to demonstrate exception independence. In this case, the assumption requires that the causal mechanisms *Barometer* and *Concentration* influence joint pain independently. If this assumption holds, the barometric pressure will contribute to a person’s joint pain regardless of the concentration of the drug, and similarly the influence that the drug concentration has on joint pain does not depend on barometric pressure. This behavior must hold for all parents, including any leak nodes that are added to allow for the accountability assumption.

If both of these assumptions hold, then we can say that there is a disjunctive interaction for the parents of X. When this is the case, the node X can be parameterized more concisely than a typical node in a CTBN. The insight that allows for the reduction in parameters is the notion that X depends on each of its parents independently, rather than simultaneously. To formalize this idea, we introduce the notion of a *disjunctive conditional Markov process*.

A disjunctive conditional Markov process X is an inhomogeneous Markov process whose intensity matrix varies with time, but only as a function of the current value of a discrete conditioning variable U drawn from a set of disjunctive variables $\mathbf{U}$. This intensity matrix, called a disjunctive conditional intensity matrix (DCIM), is written $\hat{\mathbf{Q}}_{X|\mathbf{U}}$ and can be viewed as a set of homogeneous intensity matrices $\mathbf{Q}_{X|u}$ – one for each instantiation u to U, for each variable U in $\mathbf{U}$.

Without loss of generality, let X be a node in a CTBN with n parents, and assume that disjunctive interaction holds. A DCIM can be defined in terms of the more traditional CIM data structure by using a conceptual model. For example,

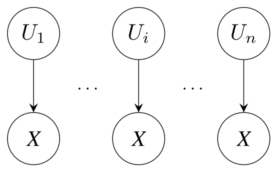

Fig. 4. Conceptual disjunctive network structure.

Fig. 4 shows n separate instances of X, one for each parent $U \in \mathbf{U}$. Although the actual CTBN structure contains only a single instance of X, this figure serves as an aid for conceptualizing DCIMs. Intuitively this structure conforms to the idea of disjunctive interaction in that each of the parents is now able to influence the behavior of X independently without the interference of another parent. Using traditional CIM parameterizations for these disjoint instances of X, each $\mathbf{Q}_{X|U}$ requires $|U|$ homogeneous intensity matrices corresponding to the states of the parent variable U. The DCIM for a node X with parents $\mathbf{U}$ can be thought of as the union of the CIMs required for each X in the conceptual model.

$$\hat{\mathbf{Q}}_{X|\mathbf{U}} = \bigcup_{U \in \mathbf{U}} \mathbf{Q}_{X|\{U\}}. \tag{5}$$

For a more concrete example, consider the drug effect network example provided in Fig. 1. The *Joint Pain* variable was modeled as a conditional Markov process, dependent on both *Barometer* and *Concentration*. By assuming disjunctive interaction between the causes of the *Joint Pain* variable, we would assert that the barometric pressure will affect joint pain without the assistance of drug concentration, and vice versa. In other words, a specific combination of barometric pressure and drug concentration will not have a unique influence on joint pain other than the existing contribution made by *Barometer* and *Concentration* individually. If this assumption holds, then *Joint Pain* can be modeled using a disjunctive conditional Markov process. Recall from our example that *Barometer* consists of three states while *Concentration* has two. Then the set of homogeneous intensity matrices that make up the DCIM for *Joint Pain* is as follows:

$$\hat{\mathbf{Q}}_{J|\{B,C\}} = \{\mathbf{Q}_{J|b0}, \mathbf{Q}_{J|b1}, \mathbf{Q}_{J|b2}, \mathbf{Q}_{J|c0}, \mathbf{Q}_{J|c1}\}. \tag{6}$$

Note that there are five intensity matrices in this set, compared to the six intensity matrices necessary for the CIM shown in Equation (3). This is because a DCIM considers the state of each parent on an individual basis, rather than using state assignments to all parents simultaneously. As a result, the number of homogeneous matrices required to specify a DCIM is the summation of the number of states for each parent variable.

$$|\hat{\mathbf{Q}}_{X|\mathbf{U}}| = \sum_{i=0}^{n} |U_i| \tag{7}$$

This follows directly from Equation (5), which constructs the set of intensity matrices using a union operation. A viable interpretation of disjunctive interaction is that each of the parents acts as an independent cause of a transition in the child. This means that identifying the next transition in state X reduces to the task of taking the minimum of a set of independent random variables that are exponentially distributed with rates defined for each of the parents. Each of these random variables is a transition event, and the earliest of these events will be exponentially distributed with a rate equal to the sum of the parent contributions. This is analogous to how a diagonal entry in a row of an intensity matrix defines the earliest transition time for all other states by summing the remaining columns.

Comparing Equations (4) and (7), we see that the number of intensity matrices required to specify a DCIM is a summation, while a CIM requires a product over the same terms. This change from a product operation to a summation operation is the cause for the reduction in the number of parameters when using DCIMs. As we consider models with a larger number of parents, or parents with a larger number of states, the savings provided by the DCIM representation becomes increasingly evident.

5. Converting DCIMs to CIMs

Although DCIMs are capable of concisely representing disjunctive interaction, existing algorithms designed for CTBN learning and inference are dependent on the CIM framework. To ensure DCIMs are compatible with these algorithms, we require a method for dynamically converting a DCIM to a CIM. The concern however, is that if a DCIM is simply replaced with a CIM, the representational advantages afforded by disjunctive interactions are lost. Instead, we take advantage of the fact that algorithms typically only utilize a small subset of the parameters in a CIM at one time. To facilitate access to specific parts of a CIM, DCIMs can be used to derive only the information that is necessary. This allows an algorithm to access a CIM implicitly, despite the fact that no explicit representation exists.

To derive a CIM from a DCIM, we make use of the properties of the underlying disjunctive interaction. As a reminder, each parent U of a variable X is considered sufficient to explain the behavior of X if the disjunctive assumption holds. As a result, each parent U in some state u will cause X to transition from state x_i to x_j with a rate of $\mathbf{Q}_{X|u}[i, j]$. When considering every parent $U \in \mathbf{U}$, we are left with $n = |\mathbf{U}|$ transition events, each of which are independent exponentially distributed random variables. The rate at which we expect X to transition from x_i to x_j given a full state instantiation $\mathbf{u}$ to its parents $\mathbf{U}$ is therefore the minimum of the transition times obtained from each parent individually. Given that the events are independent and exponentially distributed, the rate of the minimum transition time for a full instantiation of the parents is as follows.

$$\mathbf{Q}_{X|\mathbf{u}}[i,j] = \sum_{u \in \mathbf{u}} \mathbf{Q}_{X|u}[i,j] \tag{8}$$

The property that allows the individual transition rates to be summed is the same property that ensures that the diagonal entry in the row of an intensity matrix encodes the sojourn time for that state. Recall that the diagonal entry in the row of an intensity matrix is the negative sum of the remaining entries for the row, and that the absolute value of the diagonal represents the rate at which the variable will transition to any of the other states. This is directly analogous to how a combination of parents cause a transition event in a child with a rate equal to the sum of their individual effects. Each parent can be thought of as contributing to the overall transition rate, and due to the accountability assumption discussed in the previous section, the entirety of a transition rate must be accounted for by these parent contributions.

Using Equation (8), each rate $q_{i,j}$ for a homogeneous intensity matrix in a CIM can be computed using the rates in a DCIM. This applies to any of the entries in an intensity matrix for a CIM, including the negative entries along the diagonal. Given the summation applies to each of the individual entries of a matrix, entire intensity matrices in a CIM can be computed at once using matrix addition. Specifically, $\mathbf{Q}_{X|\mathbf{u}} = \sum_{u \in \mathbf{u}} \mathbf{Q}_{X|u}$, where $\mathbf{Q}_{X|\mathbf{u}}$ is a homogeneous intensity matrix in the CIM $\mathbf{Q}_{X|\mathbf{U}}$, and $\mathbf{Q}_{X|u}$ is a homogeneous intensity matrix from the DCIM $\hat{\mathbf{Q}}_{X|\mathbf{U}}$. Note that while summing the matrices as a whole may be advantageous in some cases, it will likely be computationally more efficient to sum the individual rates when only a subset of an intensity matrix is required to perform inference.

To better conceptualize how CIM rates are generated dynamically from rates in a DCIM, we can define a CIM in terms of functions rather than statically stored transition rates. Consider an arbitrary homogeneous intensity matrix $\mathbf{Q}_{X|\mathbf{u}}$ from the CIM $\mathbf{Q}_{X|\mathbf{U}}$. Each of the entries in the matrix can be replaced with a call to a function taken directly from Equation (8), and returns an intensity corresponding to the indices provided as input.

$$f(X, \mathbf{u}, i, j) = \sum_{u \in \mathbf{u}} \mathbf{Q}_{X|u}[i,j] \tag{9}$$

The key concept is that an intensity for a position in the matrix does not exist until it is computed by the function. Rather than storing the rates directly, a reference to a function that enables the rate to be computed is provided. After a rate is computed, the value can be discarded and recomputed at a later time if necessary. The CIM with function entries is shown below. This representation provides a means of representing a larger set of parameters with a smaller set of parameters and a mapping function that exploits the disjunctive interaction in the model.

$$\mathbf{Q}_{X|\mathbf{u}} = \begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & \cdots & x_n \end{array} \\ \begin{pmatrix} f(X,\mathbf{u},1,1) & f(X,\mathbf{u},1,2) & \cdots & f(X,\mathbf{u},1,n) \\ f(X,\mathbf{u},2,1) & f(X,\mathbf{u},2,2) & \cdots & f(X,\mathbf{u},2,n) \\ \vdots & \vdots & \ddots & \vdots \\ f(X,\mathbf{u},n,1) & f(X,\mathbf{u},n,2) & \cdots & f(X,\mathbf{u},n,n) \end{pmatrix} \end{array}$$

6. Binary as a special case

The notion of disjunctive interaction as it was first proposed for Bayesian networks assumed that all nodes in the network were binary, where each state had an explicit meaning [12]. Specifically, state one is considered to be the presence of an event, while state zero may be thought of as an absence of an event. The relationship between a parent and its child may then be interpreted as causal, where an event's occurrence in a parent will cause an event to occur with some probability in a child node. This interaction acts as an OR gate in that one or more parents are needed to cause an event in the child node. Furthermore, if no events have occurred in the parent nodes, then no event will occur in the child. The model is referred to as the Noisy-OR gate because there is a non-zero probability that an event fails to occur in the child node, despite an event occurring in a parent.

This section explores disjunctive interaction in CTBNs for the special case where a child node and all of its parents have exactly two states. First, note that the CIM for node X consists of 2^n homogeneous intensity matrices, where n is the number of parents for node X. Each of these homogeneous intensity matrices have four entries, equating to two unique rates. An example of such an intensity matrix is shown below.

$$\mathbf{Q}_{X|\mathbf{u}} = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{cc} 0 & 1 \end{array} \\ \begin{pmatrix} -\lambda_{\mathbf{u}} & \lambda_{\mathbf{u}} \\ \mu_{\mathbf{u}} & -\mu_{\mathbf{u}} \end{pmatrix} \end{array}$$

The parameters in these matrices describe the rate at which the variable is expected to transition from one state to another. Using this, the probability density function f and the cumulative distribution function F can be defined for transitioning from state x_0 to state x_1.

$$f_{\lambda_{\mathbf{u}}}(t) = \lambda_{\mathbf{u}} \exp(-\lambda_{\mathbf{u}} t)$$
$$F_{\lambda_{\mathbf{u}}}(t) = 1 - \exp(-\lambda_{\mathbf{u}} t)$$

The cumulative distribution function $F_{\lambda_{\mathbf{u}}}(t)$ defines the probability of transitioning from a state of x_0 to a state of x_1 by time t. This means that the probability that the process has remained in state x_0 until time t is $\exp(-\lambda_{\mathbf{u}} t)$.

To replicate the event causation interpretation from the Bayesian network literature, restrictions can be applied to the rates of the intensity matrices for X. Specifically, it is assumed that $\lambda_{\mathbf{u}} = 0.0$ when each $u \in \mathbf{u}$ is in state u_0. This ensures that if no event has occurred in the parent nodes, then there is no cause to explain an event in X, and the probability that $X = x_1$ is 0.0. This requirement stems from the accountability assumption discussed earlier, which states that the occurrence of an event may only occur as a result of a cause.

For purposes of analysis, additional constraints are imposed to allow for closed-form analysis of the state probabilities through time. First, the initial distribution for X is defined as $P = (1.0, 0.0)$, which forces X to deterministically start in state x_0 at the starting time. In addition, it is assumed that once an event occurs, the process cannot revert back to an absence of an event. This requirement can be handled by forcing each rate $\mu_{\mathbf{u}}$ to equal 0.0. This guarantees that the event state x_1 is an absorbing state, meaning that an event will only occur once and stay in that state for the remainder of the process. By making these assumptions, it is possible to derive the probability distribution over the states of X over time.

The DCIM model may now be used for the described binary CTBN. Rather than requiring 2^n intensity matrices, a DCIM needs only $2n$. This corresponds to the two separate intensity matrices required for each state of a parent node, which is necessary for all n parents. Using the available $2n$ intensity matrices in the DCIM, it is possible to compute any intensity matrix in the CIM according to the parent state assignment $\mathbf{u}$. The probability that multiple causes fail to produce an event in X is the product of the probabilities that each cause fails to produce an event independently. For more information, refer to Pearl's work on Noisy-OR [12]. Using this property, along with the cumulative distribution function, an equation may be derived that defines the rates of a CIM in terms of the rates in a DCIM.

$$\begin{aligned}
P(X(t) = x_1) &= 1 - P(X(t) = x_0) && \text{(10a)}\\
1 - \exp(-\lambda_{\mathbf{u}} t) &= 1 - \prod_{u \in \mathbf{u}} \exp(-\lambda_u t) && \text{(10b)}\\
\exp(-\lambda_{\mathbf{u}} t) &= \prod_{u \in \mathbf{u}} \exp(-\lambda_u t) && \text{(10c)}\\
-\lambda_{\mathbf{u}} t &= \sum_{u \in \mathbf{u}} -\lambda_u t && \text{(10d)}\\
\lambda_{\mathbf{u}} t &= t \sum_{u \in \mathbf{u}} \lambda_u && \text{(10e)}\\
\lambda_{\mathbf{u}} &= \sum_{u \in \mathbf{u}} \lambda_u && \text{(10f)}
\end{aligned}$$

The derivation starts on Line (10a) with a basic equality showing that the probability of X being in each of the two states sums to 1.0. Next, Line (10b) shows the cumulative distribution function using a rate $\lambda_{\mathbf{u}}$ for a CIM is shown on the left. An alternative representation is shown on the right side of the equation, where the probability of remaining in state x_1 is replaced with the product of probabilities for remaining in state x_1 given each parent individually. This produces an equality that relates the rate in a CIM to the rates in a DCIM. Line (10c) subtracts 1 from both sides and multiplies the resulting terms by -1. Line (10d) takes the log of both sides, which also has the effect of changing the product to a sum. In Line (10e), each side is multiplied by -1, and t is pulled out of the summation. Finally, note that the t value on the left side of the equation is equal to the t value on the right, indicating that this equality holds for all times t. As a result, the t on each side of the equation cancels, resulting in the equality shown in Line (10f).

This derivation indicates that a rate in a CIM can be computed by summing the rates contributed by individual parents. The rate summation matches the general description of disjunctive interaction described in Section 4. This validates the formulation presented in this work by demonstrating its equivalence to previous disjunctive interaction as it was originally proposed for Bayesian networks. Furthermore, if the rate assumptions made in this binary special case hold, then the computation of CIM rates can be simplified. First, it is immediately known that the rate of transitioning from x_1 to x_0 is always 0.0, meaning that no summation needs to be performed for these entries in the CIM. Next, it is assumed that if a parent is in state u_0, then it will not cause X to transition to state x_1. With the knowledge that these rates are equal to 0.0, the summation shown in Equation (10f) may be simplified. Rather than summing over the entire set of parent instantiations $\mathbf{u}$, the sum can instead consider only those parents where $u = u_1$. This excludes the cases where $u = u_0$, which have a known contributing rate of 0.0. The binary rate may be computed more efficiently that a general CIM rate due to the potential reduction in the size of the summation.

Algorithm 1 Retrieve intensity.

```
Input: A state $x \in X$ and an instantiation $\mathbf{u}$ to parents $\mathbf{U}$
Output: Intensity $q_{x|\mathbf{u}}$ for transitioning to another state
1: procedure GetRate($x, \mathbf{u}$)
2:     $q_{x|\mathbf{u}} \leftarrow 0$
3:     if $X$ uses DCIM then
4:         for $u \in \mathbf{u}$ do
5:             $q_{x|\mathbf{u}} \leftarrow q_{x|\mathbf{u}} + \mathbf{Q}_{X|u}[x,x]$
6:     else
7:         $q_{x|\mathbf{u}} \leftarrow \mathbf{Q}_{X|\mathbf{u}}[x,x]$
8:     return $q_{x|\mathbf{u}}$
```

7. Approximate inference with disjunctive interaction

Importance sampling can be used to achieve inference over a CTBN that contains nodes that are parameterized using DCIMs. This works by sampling trajectories from a proposal distribution P' that is guaranteed to conform to evidence. The difference between the proposal distribution P' and the true distribution P is accounted for by weighting each sample by its likelihood.

Trajectories σ can be broken into segments $\sigma[i]$ that are partitioned based on when variables transition between states. The likelihood of a trajectory is decomposable by time, and is therefore calculated by multiplying the likelihood contributions for each segment of the trajectory. The likelihood of sampling a trajectory expressed in terms of the individual contribution functions for each variable is as follows:

$$P_{\mathcal{N}}(\sigma) = \prod_{X \in \mathbf{X}} L_X(T[X|\mathbf{U}], M[X|\mathbf{U}])$$

where $T[X|\mathbf{U}]$ is the sufficient statistics indicating the amount of time X spends in each of its states as a function of its parent variables $\mathbf{U}$, while $M[X|\mathbf{U}]$ contains information about state transition counts between the states in X. The corresponding likelihood contribution for node X is shown below:

$$L_X(T[X|\mathbf{U}], M[X|\mathbf{U}]) = \prod_{\mathbf{u}} \prod_{x} \left(q_{x|\mathbf{u}}^{M[x|\mathbf{u}]} \exp(-q_{x|\mathbf{u}} T[x|\mathbf{u}]) \prod_{x' \neq x} \theta_{xx'|\mathbf{u}}^{M[x,x'|\mathbf{u}]} \right) \tag{11}$$

where $T[x|\mathbf{u}]$ is the amount of time X spends in state x while its parents are in states $\mathbf{u}$, $M[x,x'|\mathbf{u}]$ is the number of transitions from x to x', and $M[x|\mathbf{u}]$ is the total number of transitions from x to any other state, obtained as input to the likelihood contribution function. Importance sampling works by sampling from a proposal distribution P' that guarantees the resulting trajectories conform to evidence, rather than the target distribution $P_{\mathcal{N}}$. The sampled trajectories are then weighted to account for their being drawn from the proposal distribution. The calculation of the weight for a sample is shown below, where $\mathbf{e}$ is the set of evidence indicating known states for specified time durations.

$$w(\sigma) = \frac{P_{\mathcal{N}}(\sigma, \mathbf{e})}{P'(\sigma)}$$

In order to compute likelihoods in the presence of disjunctive interaction, the sampling process must be adjusted slightly. With the introduction of DCIMs, the necessary rate parameters from Equation (11) are no longer readily available, and must be computed as they are required. The importance sampling algorithm can be extended to allow for disjunctive interaction by calling a function to obtain rates for transition times, as well as the multinomial distributions that define the relative likelihood of entering each state in the event of a transition. These rates for transition times and multinomial distributions may be computed using Algorithms 1 and 2 respectively, which either compute the intensities in the event that the node uses a DCIM representation, or draw the intensities directly from the corresponding intensity matrix if the node is using a standard CIM representation.

The pseudocode for importance sampling was originally presented by Fan et al., which demonstrates how the likelihood and weight equations are built into the process of continuous time sampling [25,26]. A modified version of this procedure is shown in Algorithm 3, where asterisks to the right of lines indicate changes that were made to the original code. These changes include calls to either the `GetRate` procedure or the `GetMultinomial` procedure, which compute the necessary rates on the fly using the summation from Equation (8). This differs from the original importance sampling algorithm which obtained these rates directly from the intensity matrices that were available. Algorithm 4 provides the updated pseudocode for the `Update-Weight` procedure, which is a subroutine of the importance sampling algorithm.

Other inference algorithms can be modified in a similar fashion, where the `GetRate` and `GetMultinomial` procedures are called as rates from CIMs are needed. Disjunctive interactions allow for a more compact representation of CTBNs, regardless of the choice of inference algorithm. Furthermore, it is quite likely that large portions of a CIM will never need to be computed during the inference process. Consider sample-based inference, which essentially performs a random walk through state transitions. As the number of homogeneous intensity matrices in a CIM increases, it becomes increasingly

Algorithm 2 Retrieve multinomial distribution.

```
Input: A state $x \in X$ and an instantiation $\mathbf{u}$ to parents $\mathbf{U}$
Output: A multinomial distribution $\theta_{x|\mathbf{u}}$ for transitioning to each other state
 1: procedure GET-MULTINOMIAL($x, \mathbf{u}$)
 2:     $\theta_{x|\mathbf{u}} \leftarrow \{0, \cdots, 0\}$                                ▷ Vector of $|X|$ zeros
 3:     if $X$ uses DCIM then
 4:         for $s \in X$, where $s \neq x$ do
 5:             for $u \in \mathbf{u}$ do
 6:                 $\theta_{x|\mathbf{u}}[s] \leftarrow \theta_{x|\mathbf{u}}[s] + (\mathbf{Q}_{X|u}[x,s]/\mathbf{Q}_{X|u}[x,x])$
 7:     else
 8:         for $s \in X$, where $s \neq x$ do
 9:             $\theta_{x|\mathbf{u}}[s] \leftarrow \theta_{x|\mathbf{u}}[s] + (\mathbf{Q}_{X|\mathbf{u}}[x,s]/\mathbf{Q}_{X|\mathbf{u}}[x,x])$
10:     return $\theta_{x|\mathbf{u}}$
```

Algorithm 3 Importance sampling with disjunctive interaction.

```
Input: An end time $t_{end}$ and a set of evidence $\mathbf{e}$
Output: A sample-weight pair $(\sigma, w)$
 1: procedure CTBN-IMPORTANCE-SAMPLE($t_{end}$, $\mathbf{e}$)
 2:     $t \leftarrow 0$, $\sigma \leftarrow \emptyset$, $w \leftarrow 1$

 3:     for each variable $X \in \mathbf{X}$ do
 4:         if $\mathbf{e}_X^{val}(0)$ defined then
 5:             set $x(0) \leftarrow \mathbf{e}_X^{val}(0)$, and then set $w \leftarrow w \cdot \theta^{\mathcal{B}}_{x(0)|\mathbf{pa}_{\mathcal{B}}(0)}$
 6:         else choose state $x(0)$ according to $\theta^{\mathcal{B}}_{X|\mathbf{pa}_{\mathcal{B}}(X)}$
 7:     Loop:
 8:     for each $X \in \mathbf{X}$ such that $Time(X)$ is undefined do
 9:         if $\mathbf{e}_X^{val}(t)$ is defined then set $\Delta t \leftarrow \mathbf{e}_X^{end}(t) - t$
10:         else if $\mathbf{e}_X^{val}(t_e)$ is defined where $t_e = \mathbf{e}_X^{time}(t)$, $x(t) \neq \mathbf{e}_X^{val}(t_e)$ then
11:             choose $\Delta t$ from an exponential distribution with
12:             parameter GetRate($x(t), \mathbf{u}_X(t)$) given $\Delta t < (t_e - t)$                *
13:         else choose $\Delta t$ from an exponential with
14:             parameter GetRate($x(t), \mathbf{u}_X(t)$)                                  *
15:         Define $Time(X) \leftarrow t + \Delta t$
16:     Let $X \leftarrow \arg\min_{X \in \mathbf{X}}[Time(X)]$

17:     if $Time(X) \geq t_{end}$ then
18:         $w \leftarrow Update\text{-}Weight(X, w, t, t_{end})$
19:         return $(\sigma, w)$
20:     else
21:         $w \leftarrow Update\text{-}Weight(X, w, t, Time(X))$

22:     Update $t \leftarrow Time(X)$

23:     if $\mathbf{e}_X^{end}(t) \neq t$ or $\mathbf{e}_X^{val}(t)$ is defined then
24:         if $\mathbf{e}_X^{val}(t)$ is defined then set $x(t) \leftarrow \mathbf{e}_X^{val}(t)$
25:         else choose $x(t)$, the next value of $X$, from a multinomial with
26:             parameters GetMultinomial($x(t), \mathbf{u}_X(t)$)                          *
27:         Add $\langle X \leftarrow x(t), t \rangle$ to $\sigma$
28:         Undefine $Time(X)$ and $Time(Y)$ for all variables $Y$ for which $X \in \mathbf{U}_Y$
29:     else
30:         Undefine $Time(X)$
```

Algorithm 4 Update weight.

```
Input: A variable $Y$, current weight $w$, and sample times $t_1$ and $t_2$
Output: An updated weight $w$
 1: procedure UPDATE-WEIGHT WITH DISJUNCTIVE INTERACTION($Y, w, t_1, t_2$)
 2:     for each $X \in \mathbf{X}$ such that $\mathbf{e}_X^{val}(t)$ is defined for $t \in [t_1, t_2)$ do
 3:         $w \leftarrow w \cdot \tilde{L}_X(x[t_1 : t_2])$

 4:     for each $X \in \mathbf{X}$ such that $\mathbf{e}_X^{val}(t_e)$ is defined, where
 5:      $t_e = \mathbf{e}_X^{time}(t_1)$, and $x(t_1) \neq \mathbf{e}_X^{val}(t_e)$ do
 6:         if $X = Y$ then
 7:             $w \leftarrow w \cdot (1 - \exp(-\text{GetRate}(x(t_1), \mathbf{u}_X(t_1)) \cdot (t_e - t_1)))$          *
 8:         else $w \leftarrow w \cdot \frac{1-\exp(-\text{GetRate}(x(t_1),\mathbf{u}_X(t_1))\cdot(t_e-t_1))}{1-\exp(-\text{GetRate}(x(t_1),\mathbf{u}_X(t_1))\cdot(t_e-t_2))}$          *

 9:     return $w$
```

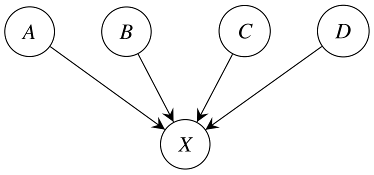

Fig. 5. The network structure for the CTBN used in the expectation comparison experiments. X is parameterized using DCIMs.

less likely that a sample path will use every rate. This relates to the curse of dimensionality in that samples will not be sufficient to cover the entirety of a CIM. For this reason, learning and storing every possible transition rate for a CIM is often unnecessary, and may not be possible when considering memory limitations. This further emphasizes the importance of the DCIM representation and the role it plays during inference.

Note that the expected computational complexity of the algorithm has not been altered, but rather has been adapted to work with DCIMs using the same architecture designed for CIMs. The intent of the importance sampling algorithm presented here is to demonstrate how existing algorithms are able to work with the more compact DCIM representation without the need for significant alterations. Despite the fact that the theoretical bounds for computational complexity are not improved, the algorithm adapted to work with disjunctive interaction tends to result in smaller computation times when the number of parents is large, which is discussed in greater detail in Section 8.2. Future work will focus on developing inference algorithms that use the DCIM representation directly, rather than first translating the parameters to a CIM compatible form.

Finally, it is worth noting that this process is analogous to inference algorithms in Bayesian networks, except that rather than summing rates, Bayesian network algorithms multiply disjunctive conditional probabilities. The reason for this equivalence is the CTBN's reliance on the exponential distribution, where summing transition intensities is the temporal equivalent to multiplying fixed probabilities. Excluding algorithms such as *quickscore* which make use of specific network structures, inference in Bayesian networks in the presence of disjunctive interaction works in much the same way as described here, where the probabilities in a conditional probability table are computed at runtime as necessary.

8. Experiments

8.1. Expectation comparisons

In this section, two experiments are presented that are intended to validate and explore the behavior of the proposed DCIM parameterization. The first experiment restricts the number of states for both the child and parent nodes to be binary, while the second experiment relaxes this restriction. These experiments compare the query results of a network parameterized using DCIMs with the expected transition probabilities. These experiments are designed to justify further the formulation of disjunctive interaction in CTBNs by supporting the theoretical results already provided. In all cases, the modified importance sampling algorithm described in Section 7 was used, which performs on-the-fly calculations of the necessary rates using information encoded in DCIMs.

8.1.1. Binary state nodes

For the first experiment, we constructed a network with four parents A, B, C and D, and a single child node X. The network structure is shown in Fig. 5. All nodes in the network are assumed to have two states, 0 and 1. Each of the parent nodes are parameterized with an initial distribution of $(0.5, 0.5)$ and a single intensity matrix containing a rate of 1.0 for transitioning to either state 0 or 1. The child node X is parameterized with an initial distribution of $(1.0, 0.0)$, and uses the disjunctive interaction model. This means that rather than using a CIM which would require a total of 16 intensity matrices, a DCIM is used to define transition behavior, consisting of only eight. The rate λ_1 of X transitioning to state 1 given an instantiation of its parents is a random number drawn uniformly from the range $[0.0, 10.0]$. These rates are generated using a random number generator with a fixed seed to ensure that parameterization is consistent over numerous experiment runs. For each intensity matrix in the DCIM, the rate of transitioning from state 1 to state 0 is 0.0. The initial distribution for X ensures that the process deterministically starts in state 0, and the zero valued transition rates from state 1 to 0 produce an absorbing state in state 1. This configuration ensures only a single transition from state 0 to state 1 will occur, at which point the process will remain in state 1 indefinitely. An example of the initial distribution for X and an intensity matrix in the DCIM for X is shown below.

$$P_X = [1.0, 0.0] \qquad \mathbf{Q}_{X|\mathbf{u}} = \begin{pmatrix} -\lambda_1 & \lambda_1 \\ 0.0 & 0.0 \end{pmatrix}$$

Node X has four parents with two states each, meaning that there are $2^4 = 16$ possible unique instantiations to these parents. For each instantiation of the parents, evidence was assigned to the model to guarantee that the parents conform to their corresponding states for the interval of time $t = [0.0, 2.0)$. The probability distribution over the states of X was then queried at 100 evenly spaced timesteps over this time interval. Next, the target probabilities for the states of X were

Table 1
KL divergences of probabilities queried from a binary state network from target probabilities. Here, the node is parameterized using a DCIM.

	KL divergence
$P(X\|a_1, b_0, c_0, d_1)$	8.6858e−09
$P(X\|a_0, b_0, c_1, d_1)$	5.5589e−07
$P(X\|a_0, b_1, c_0, d_0)$	7.8173e−08
$P(X\|a_0, b_1, c_1, d_1)$	3.4743e−08
$P(X\|a_0, b_0, c_0, d_0)$	3.4744e−08
$P(X\|a_1, b_1, c_0, d_1)$	5.5599e−07
$P(X\|a_1, b_0, c_0, d_0)$	7.8173e−08
$P(X\|a_1, b_1, c_1, d_1)$	3.4744e−08
$P(X\|a_1, b_0, c_1, d_0)$	8.6859e−07
$P(X\|a_0, b_0, c_0, d_1)$	3.1269e−07
$P(X\|a_1, b_0, c_1, d_1)$	8.6859e−07
$P(X\|a_0, b_0, c_1, d_0)$	8.6859e−09
$P(X\|a_0, b_1, c_1, d_0)$	1.4679e−06
$P(X\|a_0, b_1, c_0, d_1)$	5.5590e−07
$P(X\|a_1, b_1, c_1, d_0)$	7.0356e−07
$P(X\|a_1, b_1, c_0, d_0)$	7.8173e−08

obtained by manually retrieving the probability of X having transitioned to state x_1 from an exponential distribution with a rate equal to the sum of the rates contributed by the corresponding intensity matrices in the DCIM.

To ensure that the inference algorithm adapted to work with DCIMs behaves as anticipated, we compare the results to the target probabilities through time. We use discrete KL divergence to measure the error in the queried probability distribution, defined as follows:

$$KLD(P(X)||Q(X)) = \sum_{x_i} P(x_i) \cdot \log(P(x_i)/Q(x_i)) \quad (12)$$

where $P(X)$ is the queried states for X, while $Q(X)$ is the target distribution over the states of X. This measures the amount by which the queried distribution diverges from the target distribution. These KL divergence values are then averaged over each of the $n = 100$ evenly spaced intervals of $t \in [0.0, 2.0)$. This average divergence serves as a metric for capturing how closely the inference algorithm is able to track the target value over time. Table 1 shows the KL divergence for each of the sixteen parent instantiations considered for this experiment.

Note that the observed KL divergence values are generally very small, indicating that inference closely matches the target values throughout the entire time period $[0.0, 2.0)$. A paired equivalence test with a 0.005 region of similarity and a significance level of 0.05 was used to compare the queried result to the target probability. For all cases, it was found that the probability obtained by querying the DCIM parameterized CTBN was statistically equivalent to the corresponding target probability. What error that exists can be explained by our use of an approximate inference algorithm. For exact inference using amalgamation, there is no observed error given that the rate computation is an exact procedure.

8.1.2. General state nodes

The previous experiment investigated the behavior of disjunctive interaction when a network contains exclusively binary state nodes. This restricts the number of intensity matrices required to specify the child node and mandates that each of these intensity matrices is of size 2×2. This binary restriction is now lifted, allowing nodes with any arbitrary number of states. This experiment makes use of the same network structure shown in Fig. 5, but in this case we let $|A| = 2$, $|B| = 5$, $|C| = 3$, and $|D| = 10$. The number of states for the child node is also increased, such that $[X] = 4$.

With these added states, each intensity matrix for X now contains $4 \times 4 = 16$ entries rather than 4. In addition, a CIM would require $2 \times 5 \times 3 \times 10 = 300$ of these intensity matrices. This is drastically reduced by switching to a disjunctive representation, in that a DCIM requires only $2 + 5 + 3 + 10 = 20$ intensity matrices. This is a reduction of 280 intensity matrices, which, when compared to the reduction of eight matrices observed in the binary experiment, demonstrates the increasing utility of the disjunctive representation as the number of states increase.

Just as before, parent nodes are parameterized such that the initial probability for starting in each state is evenly distributed across all options, and all states transition to all other states with a rate of 1.0. The child node X is set to start deterministically in state x_0 at time $t = 0$ and may then transition to some state $x_i \neq x_0$ with a rate λ_i drawn from a uniform distribution over $[0.0, 10.0]$. Here again, random values are drawn using a seeded generator to ensure consistency throughout the experiment. The initial distribution and an intensity matrix in the DCIM for node X is shown below, where Λ is the negated sum of the randomly drawn rates λ_1, λ_2 and λ_3.

$$P_X = [1.0, 0.0, 0.0, 0.0] \qquad \mathbf{Q}_{X|\mathbf{U}} = \begin{pmatrix} -\Lambda & \lambda_1 & \lambda_2 & \lambda_3 \\ 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix}$$

Table 2
KL divergences of probabilities queried from a general state network from expected probabilities. Here, the node is parameterized using a DCIM.

	KL divergence
$P(X\|a_0, b_0, c_1, d_0)$	6.7193e−06
$P(X\|a_1, b_4, c_0, d_6)$	8.0242e−07
$P(X\|a_1, b_2, c_0, d_8)$	5.8897e−06
$P(X\|a_1, b_4, c_0, d_7)$	1.4459e−02
$P(X\|a_1, b_3, c_0, d_5)$	1.2760e−06
$P(X\|a_0, b_1, c_2, d_0)$	1.8200e−04
$P(X\|a_0, b_2, c_1, d_1)$	1.2923e−02
$P(X\|a_0, b_1, c_2, d_7)$	7.8464e−06
$P(X\|a_1, b_2, c_0, d_1)$	2.8161e−02
$P(X\|a_0, b_2, c_1, d_7)$	6.6768e−06
$P(X\|a_0, b_1, c_1, d_2)$	1.4204e−06
$P(X\|a_1, b_3, c_0, d_0)$	2.0227e−02
$P(X\|a_0, b_0, c_0, d_1)$	1.0695e−04
$P(X\|a_0, b_0, c_1, d_4)$	2.8968e−04
$P(X\|a_0, b_1, c_2, d_5)$	5.7312e−06
$P(X\|a_0, b_3, c_2, d_4)$	8.9228e−06

Note that this restricted parameterization is imposed for the purpose of analysis during this experiment. By forcing a single transition to occur, the inference results may be compared directly to the probability density function for a parameterized exponential distribution. This allows for comparison to an expectation, which would otherwise be unattainable via manual computation. These requirements are lifted for the scalability experiment described in the next subsection, which use unrestricted DCIMs.

Unlike the binary case, node X is now able to transition to several different states from its initial state x_0. To compute the expectation for each of these transitions, we again draw from an exponential distribution with a rate obtained by summing the contributing rates in the corresponding intensity matrices in the DCIM. This results in three exponential distributions, each of which can be thought of as "racing" one another to transition X into any one of the states x_1, x_2 or x_3. To account for the possibility that X may instead transition to another state first, the probabilities obtained from each exponential distribution is weighted by its rate as a fraction of the total sum of the transition rates.

Due to the large number of possible instantiations of parent nodes, and to remain consistent with the previous binary experiment, sixteen combinations were considered for this experiment. Each of these instantiations were randomly selected to ensure a representative cross-section, and correspond to evidence that was applied during individual inference runs. The queried probabilities over states x_1, x_2 and x_3 are computed over 100 uniformly spaced time points in the range $[0.0, 2.0)$. The discrete KL divergence was then computed to describe the difference between this queried distribution and the target distribution. These KL divergence values are shown for the sixteen parent instantiations in Table 2.

The errors shown in Table 2 are generally larger than those observed in Table 1, but typically these errors still amount to fractions of a percent. Using the same paired equivalence test as the previous experiment, it was found that there was again no significant difference between the probabilities obtained by inference and the target probability for each state. This points to the use of an approximate inference algorithm as the source of error, and the increase in error from the binary case is likely due to the increased model complexity. This experiment shows that disjunctive interaction behaves as expected, even in the case of non-binary variables.

8.2. Scalability

The previous experiments focused on validating the disjunctive formulation presented in this work. In this section, we present a series of experiments designed to investigate disjunctive interaction further. More specifically, we run inference over increasingly more complex models in an attempt to gauge how well the DCIM approach scales.

For these experiments, a network with a single child node X, and n parent nodes $\mathbf{U}$ is used. All nodes in the network are assumed to have three states, meaning that a CIM for X will require 3^n, intensity matrices containing 9 entries each, while a DCIM will need only $3n$. Each parent is parameterized using an initial distribution and intensity matrix as shown below:

$$P = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right] \qquad \mathbf{Q}_U = \begin{pmatrix} -\Lambda & \lambda_{0,1} & \lambda_{0,2} \\ \lambda_{1,0} & -\Lambda & \lambda_{1,2} \\ \lambda_{2,0} & \lambda_{2,1} & -\Lambda \end{pmatrix}$$

where each $\lambda_{i,j}$ is a rate that was drawn from a uniform distribution over the range $[0.0, 10.0]$. This is very similar to the parameterization of parent nodes in the previous expectation experiments, except that transitions make occur between states at varying rates.

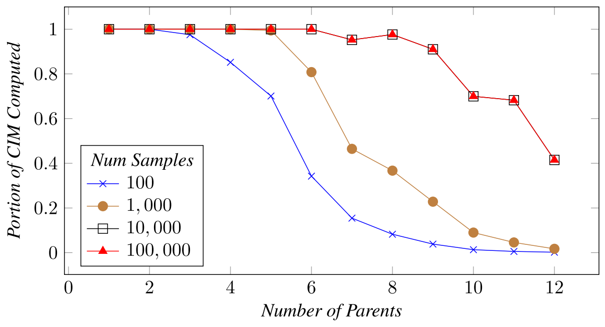

Fig. 6. Plot of the portion of a CIM that is used during inference as a function of the number of parents in the graph. Each series represents inference that uses a different number of samples.

Parameterization of the child node X is also modified from the one presented previously. In the expectation comparisons, it was required that X deterministically started in a single state, and that all other states were absorbing. This ensured that X transitioned only once during the entire process, thereby allowing for the manual computation of state expectations. In this experiment we lift that restriction, and allow X to be fully general. The initial distribution allows for an equal probability of starting in each of the three states, and all rates in the intensity matrices for the DCIM were uniformly drawn from the range $[0.0, 10.0]$. In other words, X is parameterized in exactly the same fashion as the parent nodes, except that rates are drawn to populate all $3n$ intensity matrices, rather than just one.

For all scalability experiments, the number of parents is varied from one to twelve. The baseline run uses the modified inference algorithm presented in Section 7, which generates 100,000 trajectories over the interval of time $[0.0, 2.0)$. Recall that generating these trajectories requires the use of CIM rates, which are not directly available for node X given the DCIM parameterization. Instead, CIM rates are computed dynamically as they are needed by the algorithm. For the purpose of this experiment, a recording is kept of each dynamically generated rate in the CIM for X throughout the inference process. The total number of unique rates that were computed are then compared to the total number of rates in the CIM, which can be obtained directly as 9×3^n. We refer to this ratio between the number of rates computed and the total number of rates possible as the *portion* of the CIM that was actually required

8.2.1. Number of samples

To determine how model complexity affects the portion of the computed CIM, we consider this portion as a function of the number of parent nodes in the graph. To further investigate how the number of samples used in importance sampling impact these results, the number of samples were also varied. The baseline makes use of 100,000 samples, which appeared sufficient for the inference algorithm to converge. For this experiment, three other instances of an inference algorithm were considered that make use of 10,000, 1,000 and 100 samples. Fig. 6 plots the portion of the CIM computed during the four different inference runs as a function of the number of parents in the graph. Note that there is no perceptible change between inference that uses 10,000 samples and one that uses 100,000. As a result, the line for the 100,000 sample is obscured in the figure.

The first notable feature of the plot is that the portion of the CIM that is computed decreases as the number of parents increase, regardless of the number of samples used during inference. This is due to the exponential increase in the number of intensity matrices that comprise the underlying CIM. When there is only one parent, there are three intensity matrices comprising the CIM, one for each state of the parent node. Throughout the sample generation process, it is very likely that all rates in these CIMs will be needed eventually, which is why 100% of the CIM is computed. As the number of parent nodes increase, so does the number of intensity matrices, resulting in a large number of rates for the CIM. At 12 parents, there are 3^{12} intensity matrices in a CIM for X, for a total of approximately 1.6 million rates. This makes it extremely unlikely that every rate be required when sampling during inference, which explains the reduction in the portion of the CIM that is computed.

It should also be noted that the computed portion of the CIM decreases more quickly with fewer samples. This is because fewer samples implies fewer chances to generate unique rates of the CIM. In theory, as the number of samples approaches infinity, the portion of the CIM that is computed will approach 100%. Unfortunately as the number of parents approaches infinity, the portion of the CIM that is computed approaches 0%, creating competing influences. Furthermore, an increase in the number of samples produces only a small increase in the likelihood of computing the remaining rates, especially considering that some state instantiations to the parents may be very unlikely. Conversely, the increased number of parameters is exponential in the number of parents, meaning that the influence imposed by the network structure will dominate the impact of the number of samples used during inference. In practice, as parent sets become large, only a small portion of the CIM will be used during inference, regardless of the number of samples chosen. This can be observed in Fig. 6, where the increase from 10,000 samples to 100,000 samples has no perceivable impact on the portion of the CIM that was computed.

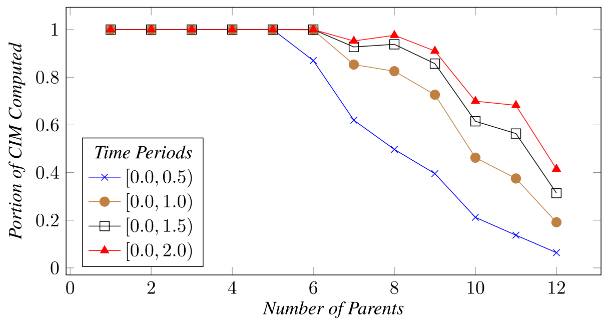

Fig. 7. Plot of the portion of a CIM that is used during inference as a function of the number of parents in the graph. Each series represents a separate run of inference over a varying lengths of time.

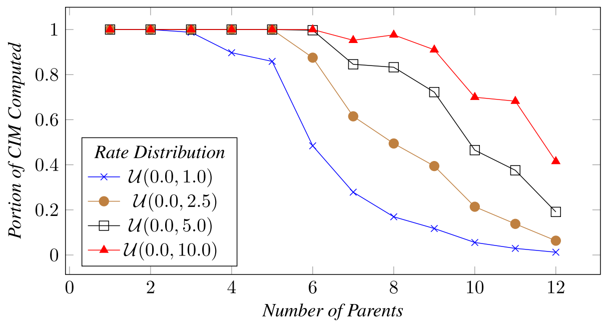

Fig. 8. Plot of the portion of a CIM that is used during inference as a function of the number of parents in the graph. Each series represents a different model parameterized with varying magnitudes of transition intensities.

8.2.2. Time period

The next experiment investigates the impact that time has on the portion of the CIM that is computed. In all previous experiments, inference was computed over a relatively long time interval of $t = [0.0, 2.0)$. To determine the impact this interval has on the portion of the CIM computed, the end time t_e, which was previously fixed at 2.0, is varied. Additional end times of 1.5, 1.0, and 0.5 are considered, each of which are added as separate series in the plot shown in Fig. 7. All instances use a fixed 100,000 samples when performing inference.

First note that the series where inference is run until time $t = 2.0$ matches the series in Fig. 6 with 100,000 samples, since the experimental setup in both cases is the same. This can be thought of as the baseline case. Next note that as the end time t_e decreases, the decrease in the portion of the CIM computed occurs more rapidly. The reason for this is that earlier end times imply shorter time periods in general, which decreases the size of the sampled trajectories. These smaller trajectories contain fewer transitions, meaning fewer rate computations to accommodate these transition events. Just as before, however, the number of parents is ultimately more important than the size of the time period, in that the exponential increase in parameters cannot be matched by a linear increase in trajectory size. For a large number of parents, even very long trajectories will not require computation of the entire CIM.

8.2.3. Magnitude of transition rates

In the final scalability experiment, the magnitudes of transition rates used when parameterizing nodes are altered. In previous experiments, each transition rate λ in an intensity matrix was drawn from a uniform distribution over the range $[0.0, 10.0]$. These rates are relatively large, resulting in many transitions per sample. In this experiment, the range for the distribution is altered such that the maximum magnitude of a rate is limited to values 5.0, 2.5 and 1.0. These three new parameterization ranges, in addition to the baseline case where the maximum is 10.0, are each used to parameterize different models. As before, the portion of the CIM for node X that was computed during inference is recorded as a function of the number of parents, and the four different cases are shown as the four series in Fig. 8. Although the rates are drawn from different distributions for each case, the process is still achieved using a seeded random number generator to guarantee consistency between runs. In this experiment, the number of samples used during inference is fixed at 100,000, and the time period over which inference is computed is $[0.0, 2.0)$.

The baseline series where rates are constrained to a magnitude of 10.0 and below is equivalent to the $[0.0, 2.0)$ series from Fig. 7, and the 100,000 series from Fig. 6. Here again, a decrease in the rates at which transitions occur results in a more rapid decrease in the portion of the CIM computed as the number of parents increase. Smaller rate values result in longer times between transitions. Given that trajectories encode transitions that occur over a fixed length of time, longer

Table 3
Observed inference runtimes in seconds using equivalent DCIM and CIM representations.

$\lvert\text{Pa}(X)\rvert$	DCIM runtimes	CIM runtimes
1	11	11
2	16	12
3	20	14
4	27	19
5	37	25
6	46	38
7	58	54
8	69	132
9	91	611
10	116	2930
11	118	–
12	142	–

times between transition events imply fewer transitions overall. This decreases the probability of covering the entire CIM during the sampling process.

In each of the scalability experiments, it was shown that as the number of parent nodes increase, the portion of the underlying CIM that is actually used decreases. This behavior is exhibited regardless of attributes such as the number of samples used during inference, the time period considered, or the magnitude of the rates used to parameterize intensity matrices. Despite amplifying or lessening the effect, changes to the model or inference algorithm do not impact the overall trend significantly, a CIM cannot be covered when the number of parents becomes very large. The issue is that the number of possible state instantiations to the parent nodes is exponential in size, making it difficult to cover the space with even a large number of samples. This relates to the curse of dimensionality that notes the difficulty in sufficiently covering a space with many dimensions, and therefore extremely large volume.

The scalability experiments indicate that only a subset of the CIM will be accessed when sampling in CTBNs, since it is unlikely to encounter every state instantiation of parents and therefore every intensity matrix contained within the CIM. Given that only a subset of the CIM is in use during the inference process, storing the information that will never be used is wasteful and may limit the size of the models that can be represented. If the disjunctive interaction assumption holds, then a DCIM provides a much more compact representation that allows for the computation of portions of the CIM only as needed. For instance, consider the case where a node has twelve parents, as shown by the right-most points in Fig. 8. For these examples, the portion of the CIM that was actually used during inference runs was, at most, approximately 40%. Using a DCIM formulation, it is unnecessary to directly store all $3^{12} = 531{,}441$ intensity matrices required for a CIM. Instead, $3 \times 12 = 36$ intensity matrices are stored for the DCIM, and the rates required during inference are computed dynamically.

Finally, we consider the observed runtime performances. We acknowledge that CPU time is largely dependent on factors such as implementation and hardware, and therefore provide this information only as a rough guide. A model was built using the bipartite structure from the previous experiments, where the number of parents in the network was varied from one to twelve, and all nodes were constructed to have three states. Inference was run separately over a network with a DCIM parameterization, and an equivalent network using a CIM parameterization, each parameterized with transition rates varying between 0.0 and 10.0. Inference was performed using a machine with modest hardware specifications, and used 100,000 samples and an end time of 2.0. The amount of time in seconds spent running the inference algorithms is shown in Table 3. A dash in this table indicates that inference was unable to finish in the four hours allotted to each run.

As the table shows, the DCIM runtimes are slower by several seconds until the network reaches approximately seven parent nodes. After this point, the amount of time required to perform inference using a CIM representation is substantially slower. The initial increase in runtime for the DCIM representation is due to the small constant overhead associated with performing the rate computations described in Section 5. This deficit is quickly overcome as the number of parents increases. This is due to the rapid increase in the amount of intensity matrices required to specify a CIM, which becomes difficult to store in main memory. When performing inference over a network with ten parent nodes, the DCIM representation decreases the runtime by a factor of 25. For the networks with 11 and 12 parents, memory constraints caused a runtime error that prevented inference using the CIM representation. Improved hardware may help to alleviate these high inference runtimes, but ultimately the exponential increase in model size will make representation and inference infeasible when using CIMs in scenarios with large numbers of parents.

9. Conclusion

In this work, disjunctive interaction was described in the context of CTBNs. The disjunctive conditional intensity matrix was introduced as a compact means of parameterizing nodes with disjunctive interaction. This DCIM parameterization was shown to reduce the number of required intensity matrices when compared to a standard CIM representation. Specifically, the number of intensity matrices required to specify the disjunctive model is linear in the number of parents rather than exponential. In this work a method to compute unspecified rates in a CIM using a DCIM defined over the same set of variables was presented, allowing for dynamic computation of specific regions in a CIM as necessary. Disjunctive interaction was then described in more detail as it works with binary state variables, and the theory presented here was related back to the Bayesian network literature. Finally, a modified version of the importance sampling inference algorithm for CTBNs was presented which made use of the DCIM formulation and the dynamic rate computation process. This demonstrated the ability to use the DCIM formulation with existing algorithms provided in the literature on CTBNs.

A series of experiments was conducted to demonstrate disjunctive interaction in CTBNs. First, expectation comparisons were performed for both binary and general state networks. These experiments performed inference over a CTBN specified using a DCIM, and compared the retrieved probabilities to manually computed expectation values. The results illustrated that the formulation matched the expectation closely, thereby demonstrating the correctness of the disjunctive interaction formulation. Another set of experiments was designed to investigate the benefits obtained by using a DCIM representation, measured in terms of the model's ability to scale to large parent sets. Results showed that regardless of many changing model features, the benefits of the disjunctive representation increasingly evident as the number of parents grew in size.

The DCIM parameterization scheme presented in this work provides a compact way of modeling CTBN nodes that exhibit disjunctive interaction. Although the substantial decrease in the space complexity was found to reduce the observed running times, disjunctive interaction does not impact the theoretical bounds for inference and learning complexity. In future work, we intend to investigate the possibility of developing CTBN algorithms that work with DCIMs directly, rather than requiring dynamic conversion to the traditional CIM format. By working with the disjunctive representation directly, improvements may be achievable to the run times for these algorithms, which are NP-hard, even when using approximation algorithms [8]. As is the case in the Bayesian network literature, it may be that efficient inference algorithms that exploit disjunctive interaction may only exist for special cases. If this is the case, it will be necessary to formally identify when inference can be performed efficiently in terms of network structure, evidence application, and the structure of the disjunctive interaction.

This work also describes the process required to convert a DCIM to a CIM. In the future, we wish to reverse this process and identify a DCIM representation using an existing CIM, even in cases where an approximation is necessary. We also intend to introduce additional compact representations for CTBNs capable of handling scenarios where accountability and exception independence do not hold. We hope to investigate the scalability of these compact representations and disjunctive interaction with respect to the number of parents and the number of states for those parents. These compact representations will allow for CTBNs capable of representing problems that were previously intractable, paving the way for new applications.

Acknowledgements

We would like to thank members of the Numerical Intelligent Systems Laboratory at Montana State University. This research was supported in part under NASA contract NNX14CJ45C.

References

[1] U. Nodelman, C.R. Shelton, D. Koller, Continuous time Bayesian networks, in: Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 2002, pp. 378–387.
[2] L. Sturlaugson, Extensions to Modeling and Inference in Continuous Time Bayesian Networks, Ph.D. thesis, Montana State University, 2014.
[3] U. Nodelman, E. Horvitz, Continuous Time Bayesian Networks for Inferring Users' Presence and Activities with Extensions for Modeling and Evaluation, Microsoft Research Technical Report, MSR-TR-2003-97, 2003.
[4] D. Cao, Novel Models and Algorithms for Systems Reliability Modeling and Optimization, Ph.D. thesis, Wayne State University, 2011.
[5] E. Acerbi, F. Stella, Continuous time Bayesian networks for gene network reconstruction: a comparative study on time course data, in: International Symposium on Bioinformatics Research and Applications, Springer, 2014, pp. 176–187.
[6] L. Perreault, On the Usability of Continuous Time Bayesian Networks: Improving Scalability and Expressiveness, Ph.D. thesis, Montana State University, 2017.
[7] U.D. Nodelman, Continuous Time Bayesian Networks, Ph.D. thesis, Stanford University, 2007.
[8] L. Sturlaugson, J. Sheppard, Inference complexity in continuous time Bayesian networks, in: Uncertainty in Artificial Intelligence, Citeseer, 2014.
[9] G.F. Cooper, The computational complexity of probabilistic inference using Bayesian belief networks, Artif. Intell. 42 (2–3) (1990) 393–405.
[10] C. Wang, N. Komodakis, N. Paragios, Markov random field modeling, inference & learning in computer vision & image understanding: a survey, Comput. Vis. Image Underst. 117 (11) (2013) 1610–1627.
[11] Y. Boykov, O. Veksler, R. Zabih, Markov random fields with efficient approximations, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1998. Proceedings, IEEE, 1998, pp. 648–655.
[12] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, San Mateo, CA, 1988.
[13] L. Perreault, S. Strasser, M. Thornton, J. Sheppard, A noisy-OR model for continuous time Bayesian networks, in: Proceedings of the Twenty-Ninth International Florida Artificial Intelligence Research Society Conference, 2016, pp. 668–673.
[14] C.R. Shelton, G. Ciardo, Tutorial on structured continuous-time Markov processes, J. Artif. Intell. Res. 51 (2014) 725–778.
[15] A. Oniśko, M.J. Druzdzel, H. Wasyluk, Learning Bayesian network parameters from small data sets: application of noisy-or gates, Int. J. Approx. Reason. 27 (2) (2001) 165–182.
[16] K. Murphy, S. Mian, Modelling Gene Expression Data Using Dynamic Bayesian Networks, Tech. Rep., Computer Science Division, University of California, Berkeley, CA, 1999.
[17] A. Bobbio, L. Portinale, M. Minichino, E. Ciancamerla, Improving the analysis of dependable systems by mapping fault trees into Bayesian networks, Reliab. Eng. Syst. Saf. 71 (3) (2001) 249–260.
[18] M.A. Shwe, B. Middleton, D. Heckerman, M. Henrion, E. Horvitz, H. Lehmann, G. Cooper, Probabilistic diagnosis using a reformulation of the internist-1/qmr knowledge base, Methods Inf. Med. 30 (4) (1991) 241–255.
[19] D. Heckerman, A tractable inference algorithm for diagnosing multiple diseases, Uncertainty Artif. Intell. 5 (1) (1990) 163–171.
[20] M. Henrion, Practical issues in constructing a Bayes' belief network, in: Proceedings of the Third Workshop on Uncertainty in Artificial Intelligence, 1987, pp. 132–139.

[21] S. Srinivas, A generalization of the noisy-or model, in: Proceedings of the Ninth International Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 1993, pp. 208–215.
[22] D. Heckerman, Causal independence for knowledge acquisition and inference, in: Proceedings of the Ninth International Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 1993, pp. 122–127.
[23] D. Heckerman, J.S. Breese, A new look at causal independence, in: Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 1994, pp. 286–292.
[24] D. Heckerman, J. Breese, Causal independence for probability assessment and inference using Bayesian networks, IEEE Trans. Syst. Man Cybern., Part A, Syst. Hum. 26 (6) (1996) 826–831.
[25] Y. Fan, C.R. Shelton, Sampling for approximate inference in continuous time Bayesian networks, in: ISAIM, 2008.
[26] Y. Fan, J. Xu, C.R. Shelton, Importance sampling for continuous time Bayesian networks, J. Mach. Learn. Res. 11 (Aug) (2010) 2115–2140.