Inference Complexity in Continuous Time Bayesian Networks

Liessman Sturlaugson and John W. Sheppard
Department of Computer Science
Montana State University
Bozeman, MT 59717
{liessman.sturlaugson, john.sheppard}@cs.montana.edu

Abstract

The continuous time Bayesian network (CTBN) enables temporal reasoning by representing a system as a factored, finite-state Markov process. The CTBN uses a traditional Bayesian network (BN) to specify the initial distribution. Thus, the complexity results of Bayesian networks also apply to CTBNs through this initial distribution. However, the question remains whether propagating the probabilities through time is, by itself, also a hard problem. We show that exact and approximate inference in continuous time Bayesian networks is NP-hard even when the initial states are given.

1 INTRODUCTION

Currently, little theoretical work has been done on the complexity of inference in CTBNs. Most of what is known about the complexity of CTBNs is derived from the fact that a Bayesian network is used to specify the initial distribution. However, despite the similarity in the names, inference in CTBNs is different than inference in Bayesian networks because the CTBN must reason over continuous time. The question becomes whether this problem of calculating evolving probabilities through time is also NP-hard, independent of the initial distribution.

The only known exact inference procedure for CTBNs is exponential in the number of nodes, but this does not necessarily imply that there does not exist an alternative approach for exact inference that performs in polynomial time even for the worst case. Furthermore, what expectations might we have about the various approximate inference algorithms that have been developed for CTBNs? Does the accuracy vs. complexity trade-off of these approximation algorithms avoid the NP-hardness of their discrete-time counterparts? In this work, we prove that exact and approximate inference in CTBNs are both NP-hard, even when the initial states are given. Thus, the complexity is also in reasoning over the factored Markov process, not just in reasoning over the Bayesian network for determining the probabilities for the initial states.

2 BACKGROUND

To place our work in context, we begin by presenting background information on Bayesian networks, dynamic Bayesian networks, and then CTBNs.

2.1 BAYESIAN NETWORKS

Bayesian networks are probabilistic graphical models that use nodes and arcs in a directed acyclic graph to represent a joint probability distribution over a set of variables (Koller & Friedman, 2009). Let \( P(X) \) be a joint probability distribution over \( n \) variables \( X_1, \ldots, X_n \in X \). A Bayesian network \( B \) is a directed, acyclic graph in which each variable \( X_i \) is represented by a node in the graph. Let \( Pa(X_i) \) denote the parents of node \( X_i \) in the graph. The graph representation of \( B \) factors the joint probability distribution as:

\[
P(X) = \prod_{i=1}^{n} P(X_i|Pa(X_i)).
\]

2.2 DYNAMIC BAYESIAN NETWORKS

The traditional Bayesian network is a static model. However, we can introduce the concept of time (or at least sequence) into the network by assigning discrete timesteps to the nodes to create a dynamic Bayesian network.

A dynamic Bayesian network (DBN) is a type of Bayesian network that uses a series of connected timesteps, each of which contains a copy of a Bayesian network \( X_t \) indexed by timestep \( t \). The probability
distribution of a variable at a given timestep can be conditionally dependent on states of that variable or other variables throughout previous timesteps. For first-order DBNs, dependence does not go further than the immediately previous timestep. Therefore, the joint probability distribution for a first-order DBN of \( T \) timesteps factors as:

\[
P(X_0, \ldots, X_{T-1}) = P(X_0) \prod_{t=0}^{T-1} P(X_{t+1}|X_t).
\]

2.3 CONTINUOUS TIME BAYESIAN NETWORKS

As can be seen from the preceding section, the DBN is restricted to discrete timesteps. The CTBN avoids discretizing time by using conditional Markov processes instead of conditional probability tables. We now formally define the CTBN and then survey its inference algorithms and applications.

2.3.1 CTBN Definition

Let \( X \) be a set of Markov processes \( \{X_1, X_2, \ldots, X_n\} \), where each process \( X_i \) has a finite number of discrete states. A continuous time Bayesian network is a tuple \( N' = (B, C) \). The Bayesian network \( B \) has nodes corresponding to \( X \) and is used only for determining \( P(X_0) \), the initial distribution of the process. Evidence at the initial time \( (t = 0) \) is incorporated by setting evidence in \( B \) and performing Bayesian network inference. The continuous-time transition model \( C \) describes the evolution of the process from this initial distribution and is specified as:

- A directed (possibly cyclic) graph \( G \) with nodes \( X_1, X_2, \ldots, X_n \), where \( Pa(X_i) \) denotes the parents of \( X_i \) in \( G \),
- A set of conditional intensity matrices (CIMs) \( A_{X|Pa(X)} \) associated with \( X \) for each possible combination of state instantiations of \( Pa(X) \).

Each conditional intensity matrix \( A_{X|Pa(X)} \) gives the dynamics of node \( X \) when the states of \( Pa(X) \) are fixed. Each entry \( a_{i,j} \geq 0, i \neq j \) gives the transition intensity of the node moving from state \( i \) to state \( j \), and each entry \( a_{i,i} < 0 \) controls the amount of time the node remains in state \( i \). With the diagonal entries constrained to be non-positive, the probability density function for the node remaining in state \( i \) is given by \( |a_{i,i}| \exp(a_{i,i}t) \), with \( t \) being the amount of time spent in state \( i \), making the probability of remaining in a state decrease exponentially with respect to time. The expected sojourn time for state \( i \) is \( 1/|a_{i,i}| \). Each row is constrained to sum to zero, \( \sum_j a_{i,j} = 0 \forall i \), meaning that the transition probabilities from state \( i \) can be calculated as \( a_{i,j}/|a_{i,i}| \forall j, i \neq j \).

2.3.2 CTBN Inference Algorithms

The only exact inference algorithm that exists so far for CTBNs combines all of the conditional intensity matrices into the single full joint intensity matrix, with states as the Cartesian product of all of the node’s states (Fan, Xu, & Shelton, 2010). Thus, the size of this matrix is exponential in the number of nodes and the number of states. Because this method ignores the factored nature of the network, research on CTBN inference has focused exclusively on approximation algorithms.

Expectation propagation (Nodelman, Koller, & Shelton, 2005; Saria, Nodelman, & Koller, 2007) has been developed for CTBNs, in which neighboring nodes employ a message-passing scheme for each interval of evidence. The messages are approximate “marginals,” a projection of a node’s conditional intensity matrix onto a single, approximating unconditional intensity matrix. Messages are continually passed until all of the nodes have a consistent distribution over the interval of evidence.

There have been a number of sample-based inference algorithms developed for CTBNs, including importance sampling (Fan et al., 2010; Fan & Shelton, 2008; Weiss, Natarajan, & Page, 2013) and Gibbs sampling (El-Hay, Friedman, & Kupferman, 2008; Rao & Teh, 2013). Importance sampling answers queries from a set of weighted samples that are generated in conformance to the evidence. The weight of each sample is the likelihood of the sample given the evidence. Gibbs sampling, by contrast, is a sampling procedure that takes a Markov Chain Monte Carlo (MCMC) approach. For each variable over each interval of evidence, the states in the Markov blanket (that is, the node’s parents, children, and children’s parents) are held constant and a random walk is performed on the state of the node. After sufficient sampling, the distribution of the random walk will converge to the true distribution for that interval of evidence.

Methods using variational techniques, such as mean-field approximation (Cohn, 2009; Cohn, El-Hay, Friedman, & Kupferman, 2009) and belief propagation (El-Hay, Cohn, Friedman, & Kupferman, 2010) have also been developed. These methods propagate the products of inhomogeneous Markov processes to approximate the distribution using systems of ordinary differential equations.
2.3.3 CTBN Applications

CTBNs have found use in a wide variety of temporal applications. For example, CTBNs have been used for inferring users’ presence, activity, and availability over time (Nodelman & Horvitz, 2003); robot monitoring (Ng, Pfeffer, & Dearden, 2005); modeling server farm failures (Herbrich, Graepel, & Murphy, 2007); modeling social network dynamics (Fan & Shelton, 2009; Fan, 2009); modeling sensor networks (Shi, Tang, & You, 2010); building intrusion detection systems (Xu & Shelton, 2010; Xu, 2010; Xu & Shelton, 2008); predicting the trajectory of moving objects (Qiao et al., 2010); and diagnosing cardiogenic heart failure and anticipating its likely evolution (Gatti, Luciani, & Stella, 2011; Gatti, 2011).

2.4 PREVIOUS COMPLEXITY RESULTS

As mentioned, most of the complexity theory surrounding CTBNs is derived from the Bayesian network for the initial distribution. However, one complexity result specific to CTBNs arises from the difference between BN and CTBN structure learning. In structure learning, it is common to assign a scoring function to arcs in the network that quantifies how well the network topology matches the training data. Nodelman (2007) gives a polynomial-time algorithm for finding the highest-scoring set of k parents for a CTBN node. The corresponding problem in a Bayesian network has been shown to be NP-hard, even for k = 2, due to the acyclic constraint of Bayesian networks (Chickering, 1996). Essentially, because cycles are allowed in CTBNs, each node can maximize its score independently.

Because the CTBN is relatively new, much of the complexity theory surrounding CTBNs has yet to be fully explored. This work intends to expand the complexity theory of CTBN inference. The work builds on the complexity results of BNs, which we now review.

Theorem 2.1. (Cooper, 1990) Exact inference in Bayesian networks is NP-hard.

Proof. Cooper proved the NP-hardness of Bayesian network inference via a reduction from 3SAT. The 3SAT problem consists of a set of m clauses \( C = \{ c_1, c_2, \ldots, c_m \} \) made up of a finite set \( V \) of n Boolean variables. Each clause contains a disjunction of three literals over \( V \), for example, \( c_3 = (v_2 \land \neg v_3 \land v_4) \). The 3SAT problem is determining whether there exists a truth assignment for \( V \) such that all the clauses in \( C \) are satisfied.

The 3SAT problem can be reduced to a Bayesian network decision problem of whether, for a \( \text{True}(T)/\text{False}(F) \) node \( X \) in the network, \( P(X = T) > 0 \) or \( P(X = T) = 0 \). We can represent any 3SAT instance by a Bayesian network as follows. For each Boolean variable \( v_i \) in \( V \), we add a corresponding \( \text{True}(T)/\text{False}(F) \) node \( V_i \) to the network such that \( P(V_i = T) = \frac{1}{2} \) and \( P(V_i = F) = \frac{1}{2} \). For each clause \( C_j \), we add a corresponding \( \text{True}(T)/\text{False}(F) \) node \( C_j \) to the network as a child of the three nodes corresponding to its three Boolean variables. Let \( w_j \) be the clause corresponding to the state of the three parents of \( C_j \), and let \( \text{eval}(w_j) \) be the truth function for this clause. The conditional probabilities of the node are

\[
P(C_j = T|w_j) = \begin{cases} 
1 & \text{if } \text{eval}(w_j) = T \\
0 & \text{if } \text{eval}(w_j) = F 
\end{cases}
\]

Finally, for each clause \( C_k \), we add a \( \text{True}(T)/\text{False}(F) \) node \( D_k \). Each \( D_k \) is conditionally dependent on \( C_k \) and on \( D_{k-1} \) (except for \( D_1 \)). The conditional probabilities for \( D_1 \) are

\[
P(D_1 = T|C_1) = \begin{cases} 
1 & \text{if } C_1 = T \\
0 & \text{otherwise}
\end{cases}
\]

Similarly, the conditional probabilities for \( D_k \) (\( k > 1 \)) are

\[
P(D_k = T|C_k, D_{k-1}) = \begin{cases} 
1 & \text{if } C_k = T \land D_{k-1} = T \\
0 & \text{otherwise}
\end{cases}
\]

Figure 1 shows the BN topology and conditional probability tables for determining the satisfiability of the clause \( (v_1 \lor v_2 \lor v_3) \land (\neg v_1 \lor \neg v_2 \lor v_3) \land (v_2 \lor \neg v_3 \lor v_4) \).

Importantly, the construction of this Bayesian network is polynomial in the length of the Boolean expression. For a 3SAT instance of \( |V| \) variables and \( |C| \) clauses, the corresponding Bayesian network has \( |V| + 2|C| \) nodes. Furthermore, each node of the Bayesian network has no more than three parents, constraining the largest conditional probability table to have no more than 16 entries, for a maximum of \( 2|V| + 16|C| + 8(|C| - 1) + 4 \) entries for the entire network.

The probabilities of the \( V \) nodes allow for every combination of truth assignments to the Boolean variables. From there, the \( C \) and \( D \) nodes enforce the logical relations of the clauses using the Bayesian network’s conditional probability tables. As such, the 3SAT instance is satisfiable if and only if \( P(D_m = T) > 0 \), that is, if and only if there is a non-zero probability that some instantiation of the \( V \) nodes to \( T \) and \( F \) will cause all of the clauses to be satisfied. Thus, if an algorithm exists that is able to efficiently compute the exact probabilities in arbitrary Bayesian networks, the algorithm
can efficiently decide whether $P(D_m = T) > 0$ for the specially constructed networks that can represent arbitrary instances of 3SAT.

Furthermore, it is known that even absolute and relative approximations in BNs is NP-hard (Dagum & Luby, 1993). These approximations are defined formally as follows. Suppose we have a real value $\epsilon$ between 0 and 1, a BN with binary-valued nodes $V$, and two nodes $X$ and $Y$ in $V$ instantiated to $x$ and $y$, respectively.

**Definition 2.1.** A relative approximation is an estimate $0 \leq Z \leq 1$ such that

$$P(X = x|Y = y) \leq Z \leq P(X = x|Y = y)(1 + \epsilon).$$

**Definition 2.2.** An absolute approximation is an estimate $0 \leq Z \leq 1$ such that

$$P(X = x|Y = y) - \epsilon \leq Z \leq P(X = x|Y = y) + \epsilon.$$

The proof of NP-hardness for relative approximation follows directly from the proof for exact inference. Satisfiability of the clause is determined whether $Z = 0$ or $Z > 0$, which is not influenced by the choice of $\epsilon$.

**Theorem 2.2.** (Dagum & Luby, 1993) Absolute approximate inference in Bayesian networks is NP-hard.

The proof of NP-hardness for absolute approximation starts with the reduction for exact inference as above, representing the variables and clauses with the same network and parameters. This time, one by one a truth assignment is set for each Boolean variable $v_i$, and the corresponding node $V_i$ is removed from the network. The truth assignment for $v_i$ is determined by the higher probability of $P(V_i = T|D_m = T)$ and $P(V_i = F|D_m = T)$. However, if there exists an efficient approximate BN inference algorithm that can guarantee to be within $\epsilon = \frac{1}{2}$ of the exact probability on arbitrary Bayesian networks, this algorithm can be used to efficiently determine satisfying truth assignments to all Boolean variables of an arbitrary instance of 3SAT. Furthermore, any approximation with $\epsilon \geq \frac{1}{2}$ for a two-state node (the simplest case) is no better than random guessing.

These proofs are for Bayesian networks, which apply to the initial distribution of a CTBN. While the CTBN and DBN are formulated differently, Cohn, El-Hay, Friedman, and Kupferman (2010) prove that a DBN becomes asymptotically equivalent to a CTBN as the interval of time between timesteps approaches zero. One might be tempted to argue that the Bayesian network complexity proofs therefore apply to the CTBN. However, it is not always clear that moving from a discrete space to a continuous space preserves the complexity results. For example, take the difference between linear programming and integer linear programming, the former being solvable in polynomial time with the latter being NP-hard. Thus, we prove the complexity results for CTBNs explicitly.

### 3 EXACT INFERENCE IN CTBNs

We show that exact inference in CTBNs is NP-hard, even when given the exact initial states, following a similar reduction as the proof for BNs but using the conditional intensity matrices of the CTBN instead of the conditional probability tables. Figure 2 shows the
CTBN topology and conditional intensity matrices for determining the satisfiability of the clause \((v_1 \lor v_2 \lor v_3) \land (\neg v_1 \lor \neg v_2 \lor v_3) \land (v_2 \lor \neg v_3 \lor v_4)\).

**Theorem 3.1.** Exact inference in Continuous Time Bayesian Networks is NP-hard even when given the initial states.

**Proof.** The CTBN topology matches that of the BN for representing variables and clauses, but the nodes are specified differently. For each Boolean variable \(v_i\) in \(V\), we add a corresponding three-state node \(V_i\) to the network. The three states in order are \(T\), \(F\), and \(S\), which is the initial state for node \(V_i\). We set the unconditional intensity matrix of \(V_i\) to be

\[
A_{V_i} = \begin{pmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
c/2 & c/2 & -c
\end{pmatrix}
\]

for some constant \(c > 0\).

For each clause \(C_j\), we add a corresponding \(True(T)/False(F)\) node \(C_j\) to the network as a child of the three nodes corresponding to its three Boolean variables. As before, let \(w_j\) be the clause corresponding to the state of the three parents of \(C_j\), and let \(eval(w_j)\) be the truth function for this clause. The function \(eval\) is extended to return \(False\) whenever the clause \(w_j\) contains a node in state \(S\). The conditional intensity matrices of \(C_j\) are

\[
A_{C_j|eval(w_j)=T} = \begin{pmatrix}
0 & 0 \\
0 & 0 \\
c & -c
\end{pmatrix}
\]

and

\[
A_{C_j|eval(w_j)=F} = \begin{pmatrix}
0 & 0 \\
0 & 0
\end{pmatrix}.
\]

We set the initial state of each \(C_j\) to be the \(F\) state (the second row of the matrices).

Finally, for each clause \(C_k\), we add a \(True(T)/False(F)\) node \(D_k\). Each \(D_k\) is conditionally dependent on \(C_k\) and on \(D_{k-1}\) (except for \(D_1\)). The conditional intensity matrices for \(D_k\) are

\[
A_{D_k|eval(C_k \land D_{k-1})=T} = \begin{pmatrix}
0 & 0 \\
c & -c
\end{pmatrix}
\]

and

\[
A_{D_k|eval(C_k \land D_{k-1})=F} = \begin{pmatrix}
0 & 0 \\
0 & 0
\end{pmatrix}.
\]

As with the \(C_j\) nodes, we set the initial state of each \(D_k\) to be the \(F\) state.

The conditional intensity matrices of the CTBN enforce the logical constraints of the Boolean expression, replacing the conditional probability tables of the Bayesian network. As before, a 3SAT instance of \(|V|\) variables and \(|C|\) clauses generates \(|V| + 2|C|\) nodes in the corresponding CTBN. Each node still has no more than three parents but now each intensity matrix has 9 or 4 entries, meaning that there is a maximum of 9\(|V| + 108|C| + 16(|C| - 1) + 8\) conditional intensity matrix entries for the entire network.

Let \(D_m(t)\) be the state of \(D_m\) at time \(t\). The 3SAT instance is satisfiable if and only if \(P(D_m(t) = T) > 0\) for any time \(t > 0\). Assume that the Boolean expression is satisfiable by some combination of \(T/F\) state assignments to the variables in \(V\). The \(V_i\) nodes start in the \(S\) state at time \(t = 0\). The time that each variable remains in \(S\) is exponentially distributed, after which the variables transition to either \(T\) or \(F\) with equal probability and remain in that state. Therefore, there is a non-zero probability for each combi-

Figure 2: Network with conditional intensity matrices for example reduction from 3SAT to CTBN inference.
Therefore, it must be that \( \hat{P}_T^i > \hat{P}_F^i \) whenever \( \epsilon < \frac{1}{2} \). We compute both \( \hat{P}_T^i \) and \( \hat{P}_F^i \) and change the initial state of \( V_i \) to \( T \) if \( \hat{P}_T^i > \hat{P}_F^i \) and to \( F \) otherwise. This process continues for \( i = 1, \ldots, |V| \) to determine truth assignments for all variables in the Boolean expression. Therefore, if there exists a polynomial-time approximation algorithm for CTBN inference with \( \epsilon < \frac{1}{4} \) that can condition on evidence, it can be used to solve arbitrary instances of 3SAT in polynomial time as well. \( \square \)

5 EMPIRICAL VALIDATION

We can empirically validate these theoretical results by taking Boolean expressions and performing inference in the corresponding CTBN. Specifically, we demonstrate three Boolean expressions, listed as follows.

\[
\begin{align*}
BE1 & = (v_1 \lor v_2 \lor v_3) \land (\neg v_1 \lor \neg v_2 \lor v_3) \\
BE2 & = (v_1 \lor v_1 \lor v_1) \land (\neg v_2 \lor \neg v_2 \lor \neg v_2) \\
BE3 & = (v_1 \lor v_1 \lor v_1) \land (v_1 \lor v_1 \lor \neg v_2) \\
& \quad \land (\neg v_1 \lor \neg v_1 \lor \neg v_2) \land (\neg v_1 \lor \neg v_1 \lor \neg v_2)
\end{align*}
\]

Note that \( BE1 \) is the Boolean expression given as an example earlier and with the CTBN shown in Figure 2. A total of 10 out of its 16 possible truth assignments satisfy the expression. Note that \( BE2 \) has a single satisfying assignment and that \( BE3 \) is unsatisfiable.

To determine the satisfiability of each of these expressions using the corresponding CTBN, we performed forward sampling with 100,000 samples and \( c = 100 \) over the interval time \([0, 0.2]\). We queried the proportion of samples with which \( D_m(t) = T \) for \( t = 0 \) to \( t = 0.2 \) in increments of 0.01. The results are shown in Figure 3. For the two satisfiable expressions, \( BE1 \) and \( BE2 \), \( P(D_m(t) = T) > 0 \) for \( t > 0.01 \), while for the unsatisfiable query \( BE3 \), \( P(D_m(t) = T) = 0 \) for all \( t \in [0, 0.2] \).

Also note the values to which the probabilities are converging. For \( BE1 \), the probability ended at an estimated 0.622, whereas the proportion of satisfying assignments is 10/16 = 0.625. For \( BE2 \), the probability ended at an estimated 0.127, whereas the proportion of satisfying assignments is 1/8 = 0.125. As the number of samples increases, the probabilities converge to the proportion of satisfying assignments.

Next, we validate the approach through which an approximation of \( P(V_i(t) = T | D_m(t) = T) \) is able to determine a satisfying assignment to each \( V_i \). Because we are conditioning on evidence, we use importance sampling (Fan et al., 2010) and smooth zero entries in the unconditional intensity matrices with \( \pm 10^{-6} \).
ignore samples with infinitesimal weights, as an infinitesimal weight implies that the corresponding sample contains a transition that violates the Boolean expression. The results with 100,000 samples are shown in Table 1.

The table shows that the importance sampling algorithm was correctly able to determine a satisfying truth assignment to each variable or determine that no truth assignments were possible. For $BE_1$, by setting $v_2$, $v_3$, and $v_4$ to $T$, the Boolean expression is satisfied regardless of the value of $v_1$, which is why the estimate was approximately 0.5, that is, either $T$ or $F$ is equally probably for satisfying the expression. For $BE_2$, the importance sampling algorithm determined the single satisfying truth assignment. For $BE_3$, no feasible samples could be generated because it is conditioned on an impossible event $D_m(0.2) = T$, indicating that the expression is unsatisfiable.

While we showed that we are able to solve these instances of $3$SAT by CTBN sampling methods, the complexity is still exponential in the length of the Boolean expression. To demonstrate this, we show the average sample count necessary to determine the satisfiability of the Boolean expression

$$\bigwedge_{i=1,...,n} (v_i \lor v_i \lor v_i)$$

for $n = 2, \ldots, 9$. Each expression has exactly one truth assignment that satisfies it (all variables set to $True$). We count the number of samples generated until we have the first sample for which $D_m(0.2) = T$, making $P(D_m(0.2) = T) > 0$ and thus showing that the $3$SAT instance is satisfiable. For each number of variables, we average the number of samples generated over 100 runs. The average sample counts along with confidence intervals for $\alpha = 0.01$ are plotted in Figure 4. The log$_2$ scale on the y-axis shows that the algorithm is exponential in the length of the expression.

6 CONCLUSION

We have shown that exact and approximate inference in CTBNs is NP-hard, even when given the initial states. Thus, the difficulty of CTBN inference is found not only in Bayesian network inference for calculating the initial distribution, but also in accurately propagating the probabilities forward in time. Given the similar results with Bayesian networks, these results are not surprising. However, as with Bayesian networks, further research may reveal special cases of the CTBN, whether in their structures or their parameters, which admit polynomial-time algorithms for approximate or even exact inference.

References


Cohn, I., El-Hay, T., Friedman, N., & Kupferman, R. (2009). Mean field variational approximation for...


