Compressing Bayes Net CPTs with Persistent Leaky Causes

Yang Xiang YXIANG@UOGUELPH.CA
Qian Jiang JIANGQ@UOGUELPH.CA
School of Computer Science, University of Guelph, Canada

Abstract

Non-Impeding Noisy-AND (NIN-AND) Trees (NATs) offer a highly expressive compressed casual model for significantly reducing space and inference time of Bayesian Nets (BNs). A causal model often includes a leaky cause for all causes not explicitly named. A leaky cause may be persistent or not. A conditional probability table (CPT) in a BN often behaves as if there is a persistent leaky cause (PLC). We discuss limitations for not modeling PLC explicitly during compression. We also reveal challenges if PLC is explicitly modeled. We extend an earlier solution that is limited to binary NAT models and is incomplete, to a solution that is applicable to multi-valued NAT models and is complete. We demonstrate the effectiveness of the solution experimentally for compressing general BN CPTs with PLCs.

Keywords: Knowledge representation; Bayesian networks; causal independence models.

1. Introduction

We consider compression of CPTs in general discrete Bayesian networks (BNs) into multi-valued NAT models (Xiang (2012a)). Once so compressed, space and time complexity of inference with BNs can be significantly reduced (Xiang (2012b); Xiang and Jin (2016)).

A number of space-efficient models exist, including noisy-OR (Pearl (1988)), noisy-MAX (Henrion (1989); Diez (1993)), CSI (Boutilier et al. (1996)), recursive noisy-OR (Lemmer and Gossink (2004)), tensor-decomposition (Vomlel and Tichavsky (2012)), and cancellation model (Woudenberg et al. (2015)). Merits of NAT models include being based on simple causal interactions (reinforcement and undermining), expressiveness (allowing the above interactions as well as their recursive mixtures), and being suited for exact multiplicative factorization (Xiang and Jin (2016)).

A causal model may include a *leaky cause* that represents all causes not explicitly named. A leaky cause may be active or inactive (*non-persistent*), in which case it behaves similarly as any normal cause. As a causal model, when all causes in a NAT model are inactive, the probability that the effect is active is zero. In many real world BN CPTs, when all causes of an effect are inactive, the probability of active effect is still positive. This may be modeled as a PLC that is always active.

To compress general CPTs into NAT models, one may choose to include only non-persistent leaky causes. We discuss limitations of this approach. Alternatively, one may choose to include PLCs. We reveal challenges that will be encountered. The issue of compressing CPTs with PLCs is considered in (Xiang and Jiang (2016)). That result has two limitations. First, only binary NAT models are treated. Second, only a partial solution is devised. In this work, we treat general NAT models (multi-valued variables) and we offer a complete solution to compress BN CPTs with PLCs. We demonstrate the proposed solution experimentally with real-world BN CPTs.

The remainder of the paper is organized as follows. Section 2 reviews the background. The task of this research and its challenges are described in Section 3. A complete solution to compressing

CPTs with PLCs is presented in Sections 4 through 6. Our experimental result is presented in Section 7. We omit all proofs due to space limit.

2. Background

Consider an effect e and the set of all causes $C = \{c_1, ..., c_n\}$ that are multi-valued and graded. The domain of e is $D_e = \{e^0, ..., e^\eta\}$ ($\eta \geq 1$), where e^0 is *inactive*, $e^1, , ..., e^\eta$ are *active*, and a higher index signifies higher intensity. The domain of c_i is $D_i = \{c_i^0, ..., c_i^{m_i}\}$ ($m_i > 0$). An active value may be written as e^+ or c_i^+.

A causal event is *success* or *failure* depending on if e is active at a given intensity, is *single-* or *multi-causal* depending on the number of active causes, and is *simple* or *congregate* depending on the effect value range. $P(e^k \leftarrow c_i^j) = P(e^k | c_i^j, c_z^0 : \forall z \neq i)$ ($j > 0$) is the probability of a *simple single-causal success*. $P(e \geq e^k \leftarrow c_1^{j_1}, ..., c_q^{j_q}) = P(e \geq e^k | c_1^{j_1}, ..., c_q^{j_q}, c_z^0 : c_z \in C \setminus X)$ ($j > 0$) is the probability of a *congregate multi-causal success*, where $X = \{c_1, ..., c_q\}$ ($q > 1$), and may be denoted as $P(e \geq e^k \leftarrow \underline{x}^+)$.

Interactions among causes may be reinforcing or undermining as defined below.

Definition 1 *Let e^k be an active effect value, $R = \{W_1, W_2, ...\}$ be a partition of a set $X \subseteq C$ of causes, $R' \subset R$, and $Y = \cup_{W_i \in R'} W_i$. Sets of causes in R reinforce each other relative to e^k, iff $\forall R'$ $P(e \geq e^k \leftarrow \underline{y}^+) \leq P(e \geq e^k \leftarrow \underline{x}^+)$. They undermine each other iff $\forall R'$ $P(e \geq e^k \leftarrow \underline{y}^+) > P(e \geq e^k \leftarrow \underline{x}^+)$.*

A NAT has multiple NIN-AND gates. A *direct* gate involves disjoint sets of causes $W_1, ..., W_m$. Each input event is $e \geq e^k \leftarrow \underline{w}_i^+$ ($i = 1, ..., m$) (successes) and its output event is $e \geq e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_m^+$. Fig. 1 (a) shows one with each W_i being a singleton. Probability of the output event

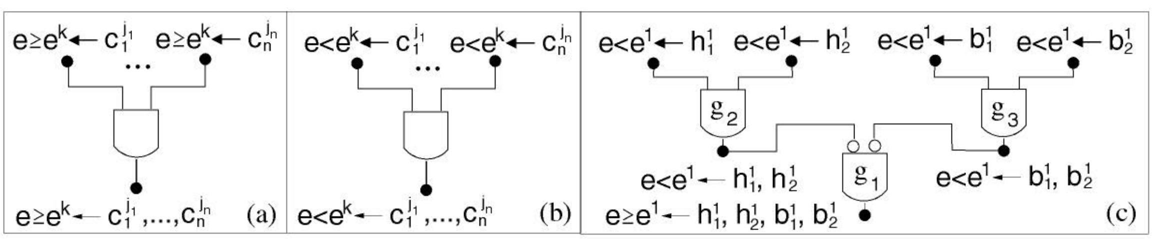

Figure 1: (a) A multi-valued direct NIN-AND gate. (b) A dual NIN-AND gate. (c) A NAT.

is $P(e \geq e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_m^+) = \prod_{i=1}^m P(e \geq e^k \leftarrow \underline{w}_i^+)$. Each input event of a *dual* gate is $e < e^k \leftarrow \underline{w}_i^+$ (failure) and its output event is $e < e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_m^+$, as Fig. 1 (b). Probability of the output is $P(e < e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_m^+) = \prod_{i=1}^m P(e < e^k \leftarrow \underline{w}_i^+)$. Direct gates express undermining among causes and dual gates express reinforcement. Fig. 1 (c) shows a NAT, where causes h_1 and h_2 reinforce each other, so do b_1 and b_2, but the two groups undermine each other.

Each NAT model uniquely determines the pairwise causal interaction (PCI) between each pair of variables c_i and c_j ($i \neq j$), denoted by the PCI bit $pci(c_i, c_j) \in \{u, r\}$ (r for reinforcing) (Xiang and Truong (2014)). The NAT in Fig. 1 (c) has $pci(h_1, h_2) = r$ and $pci(h_1, b_2) = u$. A PCI pattern (the collection of PCI bits over all pairs of variables) uniquely determines a NAT.

To significantly reduce space and inference time of BNs, each (target) CPT can be compressed into a NAT model (Xiang and Jiang (2016)). A partial PCI pattern is first extracted from the target

CPT, and is used to retrieve a small subset of candidate NATs. Constrained gradient descent over each NAT then searches for parameters. The NAT model that has the minimal distance from the target CPT defines the compressed model.

3. The Task and Challenges

To compress a general target CPT into a NAT model, a key step is to extract a partial PCI pattern (with some bits unspecified). Given causes c_i and c_j ($i \neq j$), an atomic interaction is that between a pair of active cause values c_i^v and c_j^w, relative to an active effect value e^k. We refer to it as a *value-pair interaction relative to* e^k, and denote by $pci(e^k, c_i^v, c_j^w) \in \{u, r\}$.

A NAT model has $\forall k_{>0}\ \forall v_{>0}\ \forall w_{>0}\ pci(e^k, c_i^v, c_j^w) = pci(c_i, c_j)$. However, a target CPT, unless already a NAT model, generally has $pci(e^k, c_i^v, c_j^w) \neq pci(e^{k'}, c_i^{v'}, c_j^{w'})$ if $k \neq k'$, $v \neq v'$ or $w \neq w'$. That is, value-pair interactions for the same variable pair may be inconsistent. Hence, to extract $pci(c_i, c_j)$, we first determine $pci(e^k, c_i^v, c_j^w)$ for each combination of k, v, w, and then induce $pci(c_i, c_j)$.

A leaky cause represents all causes that are not explicitly named. We denote the leaky cause by c_0 and other causes by $c_1, ..., c_n$. The c_0 may or may not be persistent. A *non-persistent* c_0 is not always active, and can be modeled the same way as other causes. A target CPT with a non-persistent leaky cause has $P(e|c_0, c_1, ..., c_n)$ fully specified where $P(e^0|c_0^0, c_1^0, ..., c_n^0) = 1$ and $P(e^k|c_0^0, c_1^0, ..., c_n^0) = 0$ for $k > 0$.

A PLC is always active. We model $c_0 \in \{c_0^0, c_0^1\}$, and $c_0 = c_0^1$ always holds. Hence, a target CPT has the form $P(e|c_0^1, c_1, ..., c_n)$. This has two implications. First, parameters $P(e|c_0^0, c_1, ..., c_n)$ are not empirically available, since the condition $(c_0^0, c_1, ..., c_n)$ never holds. Second, since c_0 is a persistent, uncertain cause, we have $0 < P(e|c_0^1, c_1^0, ..., c_n^0) < 1$.

PLC raises an issue to CPT compression. Since $P(e|c_0^0, c_1, ..., c_n)$ is undefined, the target CPT is formed $P'(e|c_1, ..., c_n)$ (only n causes) $= P(e|c_0^1, c_1, ..., c_n)$. One may be misled by the form $P'(e|c_1, ..., c_n)$ and not model c_0 explicitly. This choice, however, suffers from several limitations. First, the resultant NAT model is incapable of expressing causal interactions between c_0 and other causes, and adjusting parameters accordingly. Second, the NAT model M incurs systematic error $P_M(e^k|c_1^0, ..., c_n^0) = 0$ for $k > 0$. Third, the search for parameters cannot be based on KL distance. Each term of KL distance from a target CPT P_T is formed $P_T(i)\ log(P_T(i)/P_M(i))$, where i indexes probabilities. Since $P_M(e^k|c_1^0, ..., c_n^0) = 0$ $(k > 0)$ while $P_T(e^k|c_1^0, ..., c_n^0) > 0$ due to PLC, KL distance is undefined (is infinity).

To avoid these limitations, one may choose to model c_0 explicitly in the compressed NAT model. This, however, encounters the following difficulty. To determine value-pair interaction $pci(e^k, c_i^v, c_j^w)$ where $i, j, k, v, w > 0$, we need to compare $P(e \geq e^k \leftarrow c_i^v)$, $P(e \geq e^k \leftarrow c_j^w)$, and $P(e \geq e^k \leftarrow c_i^v, c_j^w)$, but none of them is available since c_0 is a PLC. To overcome the difficulty, it is plausible to compare instead the available $P(e \geq e^k \leftarrow c_0^1, c_i^v)$, $P(e \geq e^k \leftarrow c_0^1, c_j^w)$, and $P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$. We show below that value-pair interaction $pci(e^k, c_i^v, c_j^w)$ cannot be uniquely determined by the comparison.

Proposition 1 *Let c_0, c_i, c_j be causes where c_i and c_j are reinforcing. There exist NAT models among c_0, c_i, c_j, where $P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w) > P(e \geq e^k \leftarrow c_0^1, c_i^v)$, as well as NAT models where the opposite holds.*

Proposition 1 shows that, when c_i and c_j are reinforcing ($pci(e^k, c_i^v, c_j^w) = r$) with

$$P(e \geq e^k \leftarrow c_i^v, c_j^w) > max(P(e \geq e^k \leftarrow c_i^v), P(e \geq e^k \leftarrow c_j^w)),$$

we cannot guarantee $P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w) > \max(P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w))$.

Proposition 2 *Let c_0, c_i, c_j be causes where c_i and c_j are undermining. There exist NAT models among c_0, c_i, c_j, where $P(e < e^k \leftarrow c_0^1, c_i^v, c_j^w) > P(e < e^k \leftarrow c_0^1, c_i^v)$, as well as NAT models where the opposite holds.*

Proposition 2 shows that when c_i and c_j are undermining ($pci(e^k, c_i^v, c_j^w) = u$) with

$$P(e \geq e^k \leftarrow c_i^v, c_j^w) < min(P(e \geq e^k \leftarrow c_i^v), P(e \geq e^k \leftarrow c_j^w)),$$

we cannot guarantee $P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w) < \min(P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w))$. From Propositions 1 and 2, it follows that $pci(e^k, c_i^v, c_j^w)$ cannot be determined soly based on comparing $P(e \geq e^k \leftarrow c_0^1, c_i^v)$, $P(e \geq e^k \leftarrow c_0^1, c_j^w)$, and $P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$.

Although c_0 is used in Propositions 1 and 2, their proofs (omitted) do not depend on c_0 being a PLC. Hence, both propositions apply to any distinct causes c, c_i and c_j. It then also follows that simple comparison of multi-causal probabilities from NATs with additional causes beyond c_i and c_j cannot help determine $pci(e^k, c_i^v, c_j^w)$.

In the following sections, we present a solution to meet this challenge.

4. Determine PCI Bits by SubNAT Differentiation

We observe that the above extraction of $pci(e^k, c_i^v, c_j^w)$ focuses on c_i and c_j only. It fails since the existence of PLC c_0 deprives us of the necessary target probabilities. To overcome this difficulty, we expand our focus to include c_0. That is, instead of trying to estimate the causal interaction between c_i and c_j, we estimate the causal interactions among c_0, c_i and c_j. The inclusion of PLC c_0 implies that we are now able to conduct the analysis based on the following available target probabilities over only c_0, c_i and c_j:

$$P(e \geq e^k \leftarrow c_0^1), P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w), \text{ and } P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w).$$

We do so by extending the analysis (Xiang and Jiang (2016)) on binary NAT models (where all variables are binary).

Fig. 2 enumerates all possible (sub)NAT models for the value tuple $(c_0^1, c_i^v, c_j^w, e^k)$. For each NAT, value-pair interactions relative to e^k are summarized in Table 1. For a target CPT, if we can identify which NAT in Fig. 2 characterizes the underlying causal interactions, we can obtain the three corresponding value-pair interactions from Table 1.

To this end, we analyze the six pair-wise comparisons of the four available target probabilities. The result is summarized in Proposition 3 and Table 2.

Proposition 3 *Let c_0, c_i, and c_j be multi-valued causes and T_a through T_h be alternative NAT models over them. The pair-wise comparisons among the following as listed in Table 2 hold.*

$$P(e \geq e^k \leftarrow c_0^1), P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w), \textit{ and } P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$$

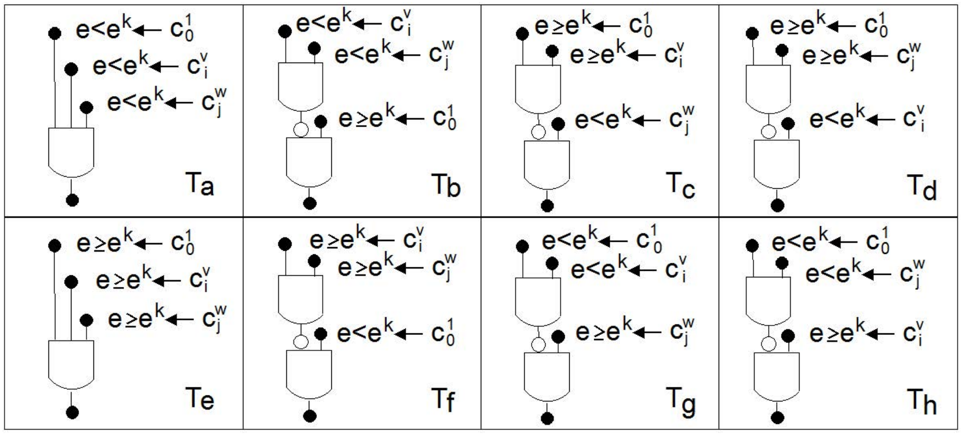

Figure 2: (Sub)NATs over c_0, c_i and c_j

	T_a	T_b	T_c	T_d	T_e	T_f	T_g	T_h
$pci(e^k, c_0^1, c_i^v)$	r	u	u	r	u	r	r	u
$pci(e^k, c_0^1, c_j^w)$	r	u	r	u	u	r	u	r
$pci(e^k, c_i^v, c_j^w)$	r	r	r	r	u	u	u	u

Table 1: Value-pair interactions of NAT models

From Proposition 3, it follows that T_a, T_b, T_e, and T_f can be uniquely identified based on comparisons in the first three rows. T_d and T_g as a group can be identified based on comparisons in the first two rows, and so can T_c and T_h as a group. However, the two members in each group cannot be differentiated by the comparisons (a partial solution). The analysis in (Xiang and Jiang (2016)) is limited to binary NAT models, but otherwise has the similar nature of being a partial solution. The above result is more general and covers multi-valued NAT models. In the next section, we explore a novel idea to extend the partial solution into a complete solution.

5. NAT Group Member Differentiation

The technique described above on average allows unique identification of 50% (4 out of 8) of the NAT models over c_0, c_i and c_j. From Table 1, this means that 50% of value-pair interactions $pci(e^k, c_i^v, c_j^w)$, where $i, j > 0$, can be identified.

From the last two rows of Table 2, both members of group $\{T_d, T_g\}$ have the same value-pair interactions $pci(e^k, c_0^1, c_i^v)$ and $pci(e^k, c_0^1, c_j^w)$. The same is true for the group $\{T_c, T_h\}$. Hence, $pci(e^k, c_0^1, c_i^v)$ and $pci(e^k, c_0^1, c_j^w)$ can always be uniquely identified, even though the underlying NAT cannot be uniquely determined. This means that all $pci(e^k, c_0^1, c_i^v)$ and $pci(e^k, c_0^1, c_j^w)$ can be identified uniquely.

On the other hand, from the last column of Table 1, we observe that $pci(e^k, c_i^v, c_j^w)$ differs between T_d and T_g, and so does between T_c and T_h. This implies that 50% of $pci(e^k, c_i^v, c_j^w)$, where $i, j > 0$, cannot be identified, which causes the corresponding PCI bit $pci(c_i, c_j)$ to be unspecified. Since the number of candidate NATs grow exponentially on the number of unspecified PCI bits, the

Row		T_a	T_b	T_e	T_f	T_d	T_g	T_c	T_h
1	$P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$ $-P(e \geq e^k \leftarrow c_0^1, c_i^v)$	$+$	$+$	$-$	$-$	$-$	$-$	$+$	$+$
2	$P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$ $-P(e \geq e^k \leftarrow c_0^1, c_j^w)$	$+$	$+$	$-$	$-$	$+$	$+$	$-$	$-$
3	$P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$ $-P(e \geq e^k \leftarrow c_0^1)$	$+$	$-$	$-$	$+$	$+/-$	$+/-$	$+/-$	$+/-$
4	$P(e \geq e^k \leftarrow c_0^1, c_i^v)$ $-P(e \geq e^k \leftarrow c_0^1, c_j^w)$	$+/-$	$+/-$	$+/-$	$+/-$	$+$	$+$	$-$	$-$
5	$P(e \geq e^k \leftarrow c_0^1, c_i^v)$ $-P(e \geq e^k \leftarrow c_0^1)$	$+$	$-$	$-$	$+$	$+$	$+$	$-$	$-$
6	$P(e \geq e^k \leftarrow c_0^1, c_j^w)$ $-P(e \geq e^k \leftarrow c_0^1)$	$+$	$-$	$-$	$+$	$-$	$-$	$+$	$+$

Table 2: Pairwise causal probability comparison by NAT models

existence of a large number of such bits will have a significant consequence on the complexity of subsequent search computation.

To resolve the difficulty, we explore a novel idea below. Consider $P(e \geq e^k \leftarrow c_0^1, c_i^v)$. If c_0 and c_i are undermining, we have $P(e \geq e^k \leftarrow c_0^1, c_i^v) = P(e \geq e^k \leftarrow c_0^1)P(e \geq e^k \leftarrow c_i^v)$. The parameter $P(e \geq e^k \leftarrow c_i^v)$ is unavailable, but we can estimate from the available by

$$P(e \geq e^k \leftarrow c_i^v) = P(e \geq e^k \leftarrow c_0^1, c_i^v)/P(e \geq e^k \leftarrow c_0^1).$$

If c_0 and c_i are reinforcing, we have $P(e < e^k \leftarrow c_0^1, c_i^v) = P(e < e^k \leftarrow c_0^1)P(e < e^k \leftarrow c_i^v)$, and can estimate $P(e < e^k \leftarrow c_i^v) = P(e < e^k \leftarrow c_0^1, c_i^v)/P(e < e^k \leftarrow c_0^1)$.

For both members of group $\{T_d, T_g\}$, c_0 and c_i are reinforcing, and c_0 and c_j are undermining. If we can estimate $P(e \geq e^k \leftarrow c_i^v)$ and $P(e \geq e^k \leftarrow c_j^w)$ (called *single-causals*) accordingly from the available parameters

$$P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w), \text{ and } P(e \geq e^k \leftarrow c_0^1),$$

we can then plug in the two single-causals and $P(e \geq e^k \leftarrow c_0^1)$ to T_d and T_g, and obtain the *multi-causals* $P_d(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$ and $P_g(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$. The NAT whose multi-causal is closer to $P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$ from the target CPT will be chosen since it better models interactions among c_0, c_i and c_j.

For group $\{T_c, T_h\}$, c_0 and c_i are undermining in both NATs, and c_0 and c_j are reinforcing. The similar method can be applied to differentiate the two members.

Although the idea seems to have resolved the above difficulty, it is not always applicable. As an example, we observed a target CPT where

$$P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w) = 0.574, P(e \geq e^k \leftarrow c_0^1, c_i^v) = 0.283,$$

$$P(e \geq e^k \leftarrow c_0^1, c_j^w) = 0.651, \text{ and } P(e \geq e^k \leftarrow c_0^1) = 0.845.$$

Applying comparisons in rows 1 and 2 of Table 2, it fits the group $\{T_c, T_h\}$ with $(+, -)$. However, since $P(e \geq e^k \leftarrow c_0^1, c_j^w) < P(e \geq e^k \leftarrow c_0^1)$, c_0 and c_j do not reinforce as T_c and T_h expected. As the result, estimation of $P(e \geq e^k \leftarrow c_j^w)$ by reinforcement is not applicable.

Applying comparisons in rows 5 and 6 of Table 2, the above example has the comparisons $(-, -)$. They do not match those of T_c and T_h, and that is the source of failure to the above attempt. This observation suggests that the above idea works only when comparisons in rows 5 and 6 have the right match. It also suggests that when the comparisons mismatch, comparisons in rows 5 and 6 can be used for identifying NATs.

Following this hint and using comparisons $(-, -)$ in rows 5 and 6, we obtain the new NAT group $\{T_b, T_e\}$ with the matching comparisons. Since comparisons $(+, -)$ in rows 1 and 2 differ from T_b and T_e (each by one comparison), we need to break the tie between T_b and T_e. This can be done by estimating single-causals $P(e \geq e^k \leftarrow c_i^v)$ and $P(e \geq e^k \leftarrow c_j^w)$ assuming undermining, which will now succeed. We then estimate $P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$ for T_b and T_e, and use the multi-causal that is closer to the target CPT to select one.

As another example, we also observed a target CPT where

$$P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w) = 0.960, P(e \geq e^k \leftarrow c_0^1, c_i^v) = 0.970,$$

$$P(e \geq e^k \leftarrow c_0^1, c_j^w) = 0.929, \text{ and } P(e \geq e^k \leftarrow c_0^1) = 0.733.$$

Applying comparisons in rows 1 and 2 of Table 2, it fits the group $\{T_d, T_g\}$ with $(-, +)$. The comparisons in rows 5 and 6 of Table 2 are $(+, +)$, making estimation of $P(e \geq e^k \leftarrow c_j^w)$ by undermining inapplicable. In response, we apply the similar procedure as above to differentiate instead between T_a and T_f.

In summary, from the target probabilities

$$P(e \geq e^k \leftarrow c_0^1), P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w), \text{ and } P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w),$$

we first use comparisons in rows 1, 2 and 3 (breaking ties arbitrarily) to identify the subNAT. If this leads to a group of two subNATs, we estimate the single-causals, compute the implied multi-causals, and differentiate between the group members. If the single-causal estimation is not applicable for the group, we use comparisons in rows 5 and 6 to find an alternative group of two subNATs. We then estimate the single-causals, compute the implied multi-causals, and differentiate between group members. This is elaborated in Algorithm 1, where we denote

$$P(e \geq e^k \leftarrow c_0^1), P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w), \text{ and } P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w),$$

by q, r, s and t, respectively.

In Algorithm 1, ties may occur in sign computation and difference comparison. Such cases rarely occur, and we break ties arbitrarily for simplicity. Proposition 4 establishes the most important property of Algorithm 1. Although we omit the proof, the above analysis serves as an intuitive justification.

Proposition 4 *For any target probabilities*

$$P(e \geq e^k \leftarrow c_0^1), P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w), \text{ and } P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w),$$

Algorithm 1 returns a unique NAT among T_a through T_h, subject to arbitrary tie-breaking.

Algorithm 1 *(Input: q, r, s and t)*
compute sign pattern $pat_1 = (sign(t-r), sign(t-s), sign(t-q))$;
if pat_1 matches that of T_a, T_b, T_e, or T_f, return the matching NAT;
compute sign pattern $pat_2 = (sign(r-q), sign(s-q))$;
if pat_2 matches that of $\{T_d, T_g\}$ or $\{T_c, T_h\}$, do
  estimate single-causals $x \equiv P(e \geq e^k \leftarrow c_i^v)$ and $y \equiv P(e \geq e^k \leftarrow c_j^w)$ by matching group;
  for each member NAT T_β of the matching group, do
    compute multi-causal $z \equiv P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$ from single-causals q, x, y and NAT T_β;
    compute difference $|t-z|$;
  return the NAT with the smaller difference;
match pat_2 against that of $\{T_a, T_f\}$ or $\{T_b, T_e\}$;
estimate $x' \equiv P(e \geq e^k \leftarrow c_i^v)$ and $y' \equiv P(e \geq e^k \leftarrow c_j^w)$ by the matching group;
for each member T'_β of the matching group, do
  compute $z' \equiv P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$ from q, x', y' and T'_β;
  compute difference $|t-z'|$;
return the NAT with the smaller difference;

Given a value-tuple $(e^k, c_0^1, c_i^v, c_j^w)$, once the NAT model is identified, the three value-pair interactions can be found from Table 1. Hence, Proposition 4 implies that PCI bit identification under PLC is solved completely by Algorithm 1.

6. Effect-Dependent PCI Patterns and Partial Distance Measures

Once value-pair interactions $pci(e^k, c_i^v, c_j^w)$ have been identified for a given pair of variables c_i and c_j (over all $v, w > 0$), the *effect-dependent PCI bit* $pci(e^k, c_i, c_j)$ can be determined by a majority vote over all $pci(e^k, c_i^v, c_j^w)$.

Let the number of active values of c_i and c_j be δ_i and δ_j. Given e^k, the total number of distinct $pci(e^k, c_i^v, c_j^w)$ is $\delta_i \times \delta_j$. If both δ_i and δ_j are odd, the number of $pci(e^k, c_i^v, c_j^w)$ votes is odd. Since each vote is binary, a majority vote is guaranteed and $pci(e^k, c_i, c_j)$ is either r or u. If one of δ_i and δ_j is even, then $\delta_i \times \delta_j$ is even, and a tie vote may occur. It signifies that the two alternative causal interactions are equally supported by analysis of $P(e \geq e^k \leftarrow c_0^1), P(e \geq e^k \leftarrow c_0^1, c_i^v), P(e \geq e^k \leftarrow c_0^1, c_j^w)$, and $P(e \geq e^k \leftarrow c_0^1, c_i^v, c_j^w)$. We assign $null$ to $pci(e^k, c_i, c_j)$ in such case. As the result, $pci(e^k, c_i, c_j)$ will be substituted by both r and u in subsequent search for parameters (see below).

With $pci(e^k, c_i, c_j)$ so assigned for each i and j, an *effect-dependent PCI pattern* relative to e^k is well-defined. It is generally a partial PCI pattern as some bits are $null$. There are a total of η effect-dependent PCI patterns. Options exist on how to use them to generate candidate NATs. At the most efficient extreme, one of them is selected to be the PCI pattern (effect independent and made of PCI bits $pci(c_i, c_j)$). The effect-dependent PCI pattern with the least $null$ bits may be selected, and it generates the least number of candidate NATs. At the least efficient extreme, all η PCI patterns are selected and they generate the most number of candidate NATs (this is used in our experiment below). Other options in between exist and the trade-off is between efficiency and accuracy of the final NAT model.

After the candidates NATs are generated, the constrained gradient descent is applied to each candidate NAT to search for single-causals of the NAT model. Note that explicit modeling of PCL ensures that KL distance is well-defined. The descent therefore minimizes KL distance between the target CPT P_T and the NAT model CPT P_M,

$$KL(P_T, P_M) = \sum_i P_T(i) log \frac{P_T(i)}{P_M(i)},$$

where i indexes parameters in P_T and P_M. In our experiment (Sec. 7), we also report average Euclidean distance $ED(P_T, P_M) = \sqrt{\frac{1}{K}\sum_{i=1}^{K}(P_T(i) - P_M(i))^2}$, where K counts parameters in P_T. In computation of KL and ED, note that all parameters of P_T are in the form of $P_T(e|c_0^1, c_1, ..., c_n)$. On the other hand, P_M contains more parameters, including all in the form $P_M(e|c_0, c_1, ..., c_n)$, where c_0 is uninstantiated. Only the parameters $P_M(e|c_0^1, c_1, ..., c_n)$ are used in the KL and ED calculation.

7. Experiments

The effectiveness of PLC-based CPT compression (PLC-Comp) is compared with 3 alternatives. One method (PLC-Opt) evaluates all NAT models exhaustively (no NAT selection by PCI pattern). Another method (NPLC-Comp) does not model PLC (Xiang and Jiang (2016)). The 3rd method (NMAX) is noisy-MAX, which does not model PLC either.

We conducted three groups of experiments using a laptop (Intel i7-3632QM CPU at 2.20 GHz). The 1st group evaluates accuracy of PLC-Comp against the optimal baseline PLC-Opt. The 2nd group compares PLC-Comp, NPLC-Comp, and NMAX on randomly generated CPTs. The 3rd group compares the three methods on real-world BN CPTs.

7.1 Compression Accuracy Relative to the Optimal

PLC-Comp and PLC-Opt are run on 30 randomly generated CPTs, each of $n = 3$ causes with sizes of variable domains bounded at 4. The value $n = 3$ is selected since both methods must evaluate NATs of $n' = n + 1 = 4$ causes (with an extra PLC). For $n' = 4$, the total number of NATs is 52 and, and for $n' = 5$, the number is 472. Since each NAT must be evaluated by PLC-Opt, to keep the computation cost down, we choose $n = 3$ and $n' = 4$.

	PLC-Opt		PLC-Comp	
	Mean	Stdev	Mean	Stdev
ED	0.142	0.032	0.145	0.033
KL	2.791	1.826	2.919	1.918
RT	59.1	39.9	13.8	11.8
SR	5.47	1.39	5.47	1.39

Table 3: Performance summary of PLC-Opt and PLC-Comp

Table 3 summarizes the results with sample means and standard deviations of both distance measures, runtime (RT) in seconds, and space reduction (SR). SR is the ratio of numbers of independent parameters between the target CPT and the NAT model. For instance, if $n = 4$ and $k = 3$ for all variables, the CPT has $3^5 - 1 = 242$ parameters. A NAT model without PLC has $(2 \times 2) \times 4 = 16$

and $SR = 242/16 = 15.13$. With PLC, the number of single-causals is $(2 \times 2) \times 4 + 2 \times 1 = 18$ and hence $SR = 242/18 = 13.44$.

As shown, ED from PLC-Comp is only slightly larger (0.145) than PLC-Opt (0.142), but using only 23% of runtime. In fact, PLC-Comp returned same optimal model in 14 out of the 30 CPTs. It demonstrates that the PCI pattern extraction (Sections 4 and 5) reduces the NAT search space effectively to a good subspace. The 23% scale of time saving is limited by the small n value due to the nature of this experiment, and is expected to be much more significant for larger n.

7.2 Compression of Random CPTs

PLC-Comp, NPLC-Comp, and NMAX are run on 50 randomly generated CPTs, each of $n = 5$ causes with sizes of variable domains bounded at 4. This group compares PLC-Comp using noisy-MAX as the baseline. It also compares explicit PLC modeling versus absence of such modeling.

Strictly speaking, KL distance is well defined only for PLC-Comp, but not so for NPLC-Comp and NMAX, due to the issue discussed in Section 3. It has been suggested to replace 0 probability in P_T by a small constant (Zagorecki and Druzdzel (2013)). The KL values of NPLC-Comp and NMAX in Table 4 are obtained instead by omitting the $\eta + 1$ terms corresponding to target probabilities $P(e|c_1^0, ..., c_n^0)$.

	PLC-Comp		NPLC-Comp		NMAX	
	Mean	Stdev	Mean	Stdev	Mean	Stdev
ED	0.234	0.074	0.256	0.069	0.381	0.094
KL	96.181	88.654	126.470	97.403	305.586	225.428
RT	56.32	60.51	21.97	21.23	1.74	1.35
SR	33.59	16.41	36.73	17.47	36.73	17.47

Table 4: Performance summary of PLC-Comp, NPLC-Comp, and NMAX

As shown in Table 4, both PLC-Comp and NPLC-Comp have smaller compression errors than NMAX, demonstrating the superior expressiveness of NAT modeling over noisy-MAX. PLC-Comp has the smallest error, demonstrating the advantage of explicit PLC modeling. Note that NPLC-Comp and NMAX achieve the same space reduction (36.73). SR by PLC-Comp is slightly less due to the extra parameters associated with modeling PLC.

7.3 Real-world BN CPTs

From the book website (Nagarajan et al. (2013)), we obtained 11 real-world BNs, excluding MUNIN subnets and binary BNs. From them, 61 CPTs are selected where the child variable has at least 3 parents, the CPT is not almost functional (close to extreme probabilities), and PLCs exist. The CPTs are divided into 5 groups by size: Small (20 CPTs, $size \in (0, 200]$), Medium (21 , $(200, 1000]$), Large (5, $(1000, 2000]$), Very Large (11, $(2000, 5000)$, and Massive (4, > 5000). PLC-Comp, NPLC-Comp, and NMAX are run on each CPT. Only EDs (more intuitive) are reported for space.

As shown in Table 5, both PLC-Comp and NPLC-Comp have smaller compression errors than NMAX, and PLC-Comp has the smallest error. Furthermore, EDs by PLC-Comp from all groups are smaller than that of random CPTs (0.234), but the errors do not increase as the CPT sizes become

		PLC-Comp		NPLC-Comp		NMAX	
Size		Mean	Stdev	Mean	Stdev	Mean	Stdev
Small	ED	0.137	0.091	0.185	0.067	0.205	0.074
	RT	12.81	12.26	4.49	8.81	0.85	1.02
	SR	9.58	6.00	10.80	6.32	10.80	6.32
Medium	ED	0.183	0.095	0.210	0.060	0.246	0.071
	RT	21.69	13.82	14.49	10.70	1.56	1.20
	SR	9.38	3.72	10.44	3.95	10.44	3.95
Large	ED	0.217	0.089	0.226	0.087	0.244	0.098
	RT	86.07	78.22	61.53	40.56	8.47	5.26
	SR	17.14	8.89	18.23	9.02	18.23	9.02
Very Large	ED	0.167	0.074	0.175	0.077	0.185	0.081
	RT	367.83	230.65	368.70	434.14	45.12	56.38
	SR	18.78	7.73	19.72	7.76	19.72	7.76
Massive	ED	0.187	0.047	0.190	0.046	0.255	0.058
	RT	4089.62	2945.41	1577.83	788.06	120.58	35.30
	SR	117.12	56.60	121.15	58.71	121.15	58.71

Table 5: Performance of PLC-Comp, NPLC-Comp, and NMAX for real-world BN CPTs

larger (resulting in more than 100 times space reduction). Note that NMAX is much faster as it does not evaluate alternative NAT topologies.

8. Conclusion

Compression of BN CPTs into NAT models can reduce the space and time complexity for BN inference significantly (Xiang and Jin (2016)). Coupled with other supporting techniques, e.g., multiplicative factorization, they will enable BN inference to be deployed in low resource devices. Many real-world BN CPTs behave as if there is a PLC. Whether such a leaky cause should be explicitly modeled during compression is the focus of this work. Either choice is associated with limitations or difficulties.

We opted for explicit modeling of PLC. We presented a novel method to overcome the lack of relevant target probabilities in extracting PCI patterns. It significantly reduces the NAT search space for subsequent model selection. Our method is theoretically well-founded in comparison to not modeling PLC, which requires practical remedies such as replacing zero model probabilities by small constant or omitting these terms in KL distance calculation. Experimentally, the explicit PLC modeling better captured causal interactions and improved compression accuracy. As a future work, sensitivity of posteriors from BN inference to the compression will be evaluated.

Acknowledgments

Financial support from NSERC, Canada is acknowledged. We thank reviewers for their comments. We apologize for being unable to address all of them due to space, but will do so in a sequel.

References

C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proc. 12th Conf. on Uncertainty in Artificial Intelligence*, pages 115–123, 1996.

F. Diez. Parameter adjustment in Bayes networks: The generalized noisy OR-gate. In D. Heckerman and A. Mamdani, editors, *Proc. 9th Conf. on Uncertainty in Artificial Intelligence*, pages 99–105. Morgan Kaufmann, 1993.

M. Henrion. Some practical issues in constructing belief networks. In L. Kanal, T. Levitt, and J. Lemmer, editors, *Uncertainty in Artificial Intelligence 3*, pages 161–173. Elsevier Science Publishers, 1989.

J. Lemmer and D. Gossink. Recursive noisy OR - a rule for estimating complex probabilistic interactions. *IEEE Trans. on System, Man and Cybernetics, Part B*, 34(6):2252–2261, Dec 2004.

R. Nagarajan, M. Scutari, and S. Lebre. *Bayesian Networks in R with Applications in Systems Biology*. Springer, 2013.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

J. Vomlel and P. Tichavsky. An approximate tensor-based inference method applied to the game of Minesweeper. In *Proc. 7th European Workshop on Probabilistic Graphical Models, Springer LNAI 8745*, pages 535–550, 2012.

S. Woudenberg, L. van der Gaag, and C. Rademaker. An intercausal cancellation model for Bayesian-network engineering. *Inter. J. Approximate Reasoning*, 63:3247, 2015.

Y. Xiang. Non-impeding noisy-and tree causal models over multi-valued variables. *International J. Approximate Reasoning*, 53(7):988–1002, Oct 2012a.

Y. Xiang. Bayesian network inference with NIN-AND tree models. In A. Cano, M. Gomez-Olmedo, and T. Nielsen, editors, *Proc. 6th European Workshop on Probabilistic Graphical Models*, pages 363–370, Granada, 2012b.

Y. Xiang and Q. Jiang. Compression of general Bayesian net CPTs. In R. Khoury and C. Drummond, editors, *Advances in Artificial Intelligence, LNAI 9673*, pages 285–297. Springer, 2016.

Y. Xiang and Y. Jin. Multiplicative factorization of multi-valued NIN-AND tree models. In Z. Markov and I. Russell, editors, *Proc. 29th Inter. Florida Artificial Intelligence Research Society Conf.*, pages 680–685. AAAI Press, 2016.

Y. Xiang and M. Truong. Acquisition of causal models for local distributions in Bayesian networks. *IEEE Trans. Cybernetics*, 44(9):1591–1604, 2014.

A. Zagorecki and M. Druzdzel. Knowledge engineering for Bayesian networks: How common are noisy-MAX distributions in practice? *IEEE Trans. Systems, Man, and Cybernetics: Systems*, 43 (1):186–195, 2013.