Compression of General Bayesian Net CPTs

Yang Xiang and Qian Jiang

University of Guelph, Canada

Abstract. Non-Impeding Noisy-AND (NIN-AND) Tree (NAT) models offer a highly expressive approximate representation for significantly reducing the space of Bayesian Nets (BNs). They can also significantly improve efficiency of BN inference, as shown for binary NAT models. To enable these advantages for general BNs, advancements on three technical challenges are made in this work. We overcome the limitation of well-defined Pairwise Causal Interaction (PCI) bits and present a flexible PCI pattern extraction from general target Conditional Probability Tables (CPTs). We extend parameter estimation for binary NAT models to constrained gradient descent for compressing target CPTs into multi-valued NAT models. The effectiveness of the compression is demonstrated experimentally. A novel framework is also developed for PCI pattern extraction when persistent leaky causes exist.

1 Introduction

A discrete BN quantifies causal strength between each effect and its n causes by a CPT whose number of parameters is exponential in n. Common Causal Independence Models (CIMs), e.g., noisy-OR [4], reduce the number to being linear in n, but are limited in expressiveness. As members of CIM family, NAT models [9, 8, 12, 11] express both reinforcing and undermining as well as their recursive mixture using only a linear number of parameters. Thus, NAT models offer a highly expressive approximation for significantly reducing the space of BNs.

CIMs are not directly operable by common BN inference algorithms, e.g., the cluster tree method [1]. Several techniques exist to overcome the difficulty [14, 2, 6, 5]. By applying multiplicative factorization to binary NAT models and compiling NAT modeled BNs for lazy propagation [3], it has been shown that efficiency of exact inference with BNs can also be improved significantly [8].

The above efficiency gain was shown with binary NAT models. However, binary NAT models are not sufficiently general. The ultimate goal of this research is to achieve similar efficiency improvement for inference in general BNs compressed into multi-valued NAT models [9]. Advancing from binary to multi-valued NAT models encounters several challenges. In this work, we investigate the following. To gain efficiency with both space and inference time through NAT modeling, each (target) CPT in BNs is approximated (compressed) into a NAT model. The first step is to find a small set of candidate NAT structures to focus subsequent parameter search. A NAT can be uniquely identified by a function that specifies interactions between each pair of causes, termed a PCI pattern [10]. Therefore, we extract a PCI pattern from the target CPT, which yields the candidate NATs. Since a target CPT is generally not a NAT model, how to extract a PCI pattern that provides good approximation of its causal interaction structure is a challenge. The first contribution of this work is a scheme that meets this challenge.

Once the candidate NATs are obtained, probability parameters of corresponding NAT models must be assessed. The second contribution of this work is to extend the framework for doing so with binary NAT models to multi-valued NAT models. We present a constrained gradient descent as the key component of the extension. Although the general idea of constrained gradient descent already exists, this work investigates specific constraints for compressing multi-valued CPTs.
CIMs allow both explicit causes and implicit causes, termed leaky causes. Leaky causes may be persistent or non-persistent. We analyze implications of both types of leaky causes to NAT compression of CPTs. We show that persistent leaky causes raise another challenge. The third contribution of this work is a framework for PCI pattern extraction with persistent leaky causes.
Sec. 2 briefly introduces the background. Contribution on PCI pattern extraction from general target CPTs is presented in Sec. 3. Constrained gradient descent for compressing multi-valued CPTs is covered in Sec. 4. Their effectiveness is shown through experimental study in Sec. 5. Contribution on PCI extraction with persistent leaky causes is presented in Sec. 6.

2 Background

Consider an effect e and the set of all causes $C = \{c_1, ..., c_n\}$ that are multi-valued and graded. That is, e has domain $D_e = \{e^0, ..., e^\eta\}$ $(\eta \geq 1)$, 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 a *success* or *failure* depending on whether e is rendered active at certain intensity, is *single-causal* or *multi-causal* depending on the number of active causes, and is *simple* or *congregate* depending on the range of effect values. $P(e^k \leftarrow c_i^j) = P(e^k | c_i^j, c_z^0 : \forall z \neq i)$ $(j > 0)$ 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 $X = \{c_1, ..., c_q\}$ $(q > 1)$. It is also denoted $P(e \geq e^k \leftarrow \underline{x}^+)$.
A NAT consists of two types of NIN-AND gates, each over disjoint sets of causes $W_1, ..., W_q$. An input event of a *direct* gate is $e \geq e^k \leftarrow \underline{w_i}^+$ and the output event is $e \geq e^k \leftarrow \underline{w_1}^+, ..., \underline{w_q}^+$. An input of a *dual* gate is $e < e^k \leftarrow \underline{w_i}^+$ and the output event is $e < e^k \leftarrow \underline{w_1}^+, ..., \underline{w_q}^+$. Probability of the output event of a gate is the product of probabilities of its input events. Interactions among causes may be reinforcing or undermining.

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 direct gate models undermining and a dual gate models reinforcing. A NAT organizes multiple gates into a tree and expresses mixture of reinforcing and undermining recursively. A NAT specifies interaction between each pair of c_i and c_j, denoted by *PCI bit* $pci(c_i, c_j) \in \{u, r\}$ with u for undermining. The collection of PCI bits is the *PCI pattern* of the NAT. A NAT can be uniquely identified by PCI

pattern [12]. Given a NAT and probabilities of input events, called *single-causals*, the probability of its output event can be obtained. From the single-causals and all derivable NATs [7], the CPT $P(e|C)$ is uniquely defined [9].

3 Extracting PCI Patterns from General CPTs

To compress a target CPT over e and C into a NAT model, we need to determine a NAT over C. This can be achieved by searching for a PCI pattern relative to each e^k and determine the NAT by the best pattern over all k. By Def. 1, given c_i and c_j, $pci(c_i,c_j)$ is *well defined* relative to e^k when one of the following conditions holds for all active values of c_i and c_j.

$$pci(c_i,c_j) = \begin{cases} u : P(e \geq e^k \leftarrow c_i^+,c_j^+) < min(P(e \geq e^k \leftarrow c_i^+),P(e \geq e^k \leftarrow c_j^+)), \\ r : P(e \geq e^k \leftarrow c_i^+,c_j^+) \geq max(P(e \geq e^k \leftarrow c_i^+),P(e \geq e^k \leftarrow c_j^+)). \end{cases} \quad (1)$$

As shown experimentally (Sec. 5.1), in a general CPT, neither condition may hold for a significant number of cause pairs. For such a CPT, very few PCI bits are well defined, resulting in a *partial* PCI pattern. A partial pattern of a few bits is compatible with a large candidate set of NATs, making subsequent search costly. Hence, a best pattern has the most bits. Below, we develop a scheme to overcome the difficulty where the best PCI pattern has too few well defined bits.
We aim to extract a partial PCI pattern that approximates causal interactions in a target CPT. For a partial pattern, PCI bit of a given cause pair may be u, r, or undefined. For uniformity, we expand the domain of a PCI bit into $\{u,r,nul\}$ with *nul* for unclassified.
For a well-defined bit, one condition in Eqn. (1) must hold for all active cause value pairs. Consider interaction for one value pair first. To indicate the e^k value, we denote the interaction as $pci(e^k,c_i^+,c_j^+) \in \{u,r,nul\}$. To simplify notation, we denote $P(e \geq e^k \leftarrow c_i^+), P(e \geq e^k \leftarrow c_j^+)$, and $P(e \geq e^k \leftarrow c_i^+,c_j^+)$ as p,q, and t, respectively. A well-defined interaction is extracted by the following rule.

Rule 1 (Well-defined) *If $t \notin [min(p,q),max(p,q)]$, then*

$$pci(e^k,c_i^+,c_j^+) = \begin{cases} u & : \quad t < min(p,q), \\ r & : \quad t > max(p,q). \end{cases}$$

A well-defined interaction satisfies $t \notin [min(p,q),max(p,q)]$. Rules below relax this requirement. When $t \in [min(p,q),max(p,q)]$, $pci(e^k,c_i^+,c_j^+)$ is deemed *nul* only if $|p-q|$ is too small, e.g., less than a threshold $\tau_0 = 0.2$.

Rule 2 (Tight enclosure) *If $t \in [min(p,q),max(p,q)]$ and $|p-q| \leq \tau_0$, then $pci(e^k,c_i^+,c_j^+) = nul$, where $\tau_0 \in (0,1)$ is a given threshold.*

Rational of the rule is the following. Under tight enclosure, both u and r may well approximate interaction between c_i and c_j. Hence, NATs compatible with either should be included in the candidate set, which is what value *nul* entails.
We refer to condition $t \in [min(p,q),max(p,q)]$ and $|p-q| > \tau_0$ as *loose enclosure*, where we compute the ratio $R = \frac{t-0.5(p+q)}{|p-q|}$. Ratio $R \in [-0.5,0.5]$ and the

bounds are reached when t equals p or q. When $R < 0$, t is closer to $min(p,q)$. When $R > 0$, t is closer to $max(p,q)$. When $R = 0$, t is equally distant from p and q. We refer to R as *normalized deviation* and specify the interaction as follows, where a possible value for τ_1 is 0.4. Its rational follows from the above analysis.

Rule 3 (Sided loose enclosure) *Given thresholds $\tau_0, \tau_1 \in (0,1)$, if $t \in [min(p,q), max(p,q)]$ and $|p-q| > \tau_0$, then*

$$pci(e^k, c_i^+, c_j^+) = \begin{cases} nul & : \quad |R| \leq \quad \tau_1, \\ u & : \quad R \ < -\tau_1, \\ r & : \quad R \ > \quad \tau_1. \end{cases}$$

Given e^k, the above determines $pci(e^k, c_i^+, c_j^+)$ for a pair c_i^+ and c_j^+. If each cause has $m+1$ values, there are m^2 pairs of active values for c_i and c_j. The next rule determines the PCI bit $pci(e^k, c_i, c_j)$ by majority of value based interactions. A possible value for threshold τ_2 may be 0.51.

Rule 4 (Majority Value Pairs) *Let M be the number of active cause value pairs (c_i^+, c_j^+), M_u be the number of interactions where $pci(e^k, c_i^+, c_j^+) = u$, and M_r be the number of interactions where $pci(e^k, c_i^+, c_j^+) = r$. For a given threshold $\tau_2 \in (0.5, 1)$,*

$$pci(e^k, c_i, c_j) = \begin{cases} u & : \quad M_u > \tau_2 \, M, \\ r & : \quad M_r > \tau_2 \, M, \\ nul & : \quad Otherwise. \end{cases}$$

After PCI bit $pci(e^k, c_i, c_j)$ is extracted for each pair (c_i, c_j), a set $pci(e^k)$ of PCI bits relative to e^k is defined. From η such sets, the next rule selects the best as the PCI pattern, where a possible value for threshold τ_3 may be 0.8.

Rule 5 (Partial PCI pattern) *Let n be the number of causes of e, the set of PCI bits relative to e^k ($k > 0$) be $pci(e^k) = \{pci(e^k, c_i, c_j) \mid \forall_{i,j} \, c_i \neq c_j\}$, and N_k be the number of PCI bits in $pci(e^k)$ such that $pci(e^k, c_i, c_j) \neq nul$. Let $N_x = max_k \, N_k$ and $\tau_3 \in (0.5, 1)$ be a given threshold.*
Then select $pci(e^x)$ as the partial PCI pattern if $N_x > \tau_3 \, C(n,2)$.

If $N_x \leq \tau_3 \, C(n,2)$, the above rule is inconclusive. We require the search procedure to relax thresholds τ_0 through τ_3 until a PCI pattern is selected. Effectiveness of the procedure for reducing NAT space while extracting good NAT candidates is shown in Sec. 5.2.

4 Parameter Estimation with Constrained Descent

Once a partial PCI pattern is extracted, the set of candidate NATs compatible with the pattern can be determined [12]. For each candidate NAT, single-causals can be

estimated from target CPT through gradient descent. From resultant NAT models, the best NAT model can be selected. These steps parallel those for compression of binary CPTs into binary NAT models [11]. In this section, we extend gradient descent to compression of multi-valued CPTs.

A NAT and a set of single-causals define a NAT model M. We measure similarity of a target CPT P_T and the CPT P_M of M by Kullback−Leibler divergence,

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

where i indexes probabilities in P_T and P_M. Gradient descent estimates the set of single-causals of M such that $KL(P_T, P_M)$ is minimized. In experimental study (Sec. 5), average Euclidean distance $ED(P_T, P_M) = \sqrt{\frac{1}{K}\sum_{i=1}^{K}(P_T(i) - P_M(i))^2}$ is also obtained, where K counts parameters in P_T.

During descent, the point descending the multi-dimensional surface is a vector of single-causals. For a binary NAT model with n causes, the vector has n parameters and each can be specified independently. For multi-valued NAT models, where $|D_e| = \eta + 1$ and $|D_i| = m + 1$ for $i = 1, ..., n$, the descent point is a ηmn vector. Each parameter is a $P(e^+ \leftarrow c_i^+)$. Unlike the binary case, the ηmn parameters are not independent. We consider below constraints that they must observe during descent.

First, each parameter $P(e^+ \leftarrow c_i^+) > 0$. That is, each parameter is lower bounded by 0, but cannot reach the bound since otherwise c_i^+ no longer causes e^+.

Second, in the binary case, each parameter is upper bounded by 1, but cannot reach the bound since otherwise c_i is no longer an uncertain cause. In the multi-valued case, this constraint is replaced by a more strict alternative. For each c_i^+, $\sum_{j=1}^{\eta} P(e^j \leftarrow c_i^+) < 1$ must hold. If violated, the resultant parameters $P(e^1 \leftarrow c_i^+), ..., P(e^\eta \leftarrow c_i^+)$ will not be valid single-causals of an uncertain cause. This amounts to mn constraints, each governing η parameters. To satisfy these constraints, we extend gradient descent for binary NAT models below.

At the start of each round of descent, each group of η single-causals under the same constraint are initialized together as follows. Generate $\eta + 1$ random numbers in the range $[\delta, 1 - \delta]$, where $\delta > 0$ is a small real. Let S be their sum and $0 < \gamma < 1$ be a real close to 1. Drop one number arbitrarily, multiply the remaining η numbers by γ/S, and assign results as initial single-causals. Proposition 1 summarizes properties of the initialization, whose proof is omitted due to space.

Proposition 1 *Let $P(e^1 \leftarrow c_i^+), ..., P(e^\eta \leftarrow c_i^+)$ be initial values of parameters with the same active cause value c_i^+. The following hold.*

1. *For each parameter, $P(e^j \leftarrow c_i^+) \geq \delta$, $j = 1, ..., \eta$.*
2. *For the subset of parameters, $\sum_{k=1}^{\eta} P(e^k \leftarrow c_i^+) \leq \gamma$.*

Each step of gradient descent updates the ηmn parameters in sequence. To ensure that both conditions of Proposition 1 continue to hold, we constrain descent as follows. After each $P(e^j \leftarrow c_i^+)$ is updated, check if $P(e^j \leftarrow c_i^+) \geq \delta$. If not, set $P(e^j \leftarrow c_i^+) = \delta$ and stop $P(e^j \leftarrow c_i^+)$ from further descent. Otherwise, check if $S = \sum_{k=1}^{\eta} P(e^k \leftarrow c_i^+) \leq \gamma$ holds. If not, set $P(e^j \leftarrow c_i^+)$ to $P(e^j \leftarrow c_i^+) + \gamma - S$ and stop $P(e^j \leftarrow c_i^+)$ from further descent. If both tests succeed, commit to the updated value of $P(e^j \leftarrow c_i^+)$ and allow it to continue descent. Proposition 2 summarizes properties of the method, whose proof is omitted due to space.

Proposition 2 *Let $P(e^1 \leftarrow c_i^+), ..., P(e^\eta \leftarrow c_i^+)$ be current values of a subset of parameters with the same active cause value c_i^+, such that the following hold.*

1. *For each parameter, $P(e^j \leftarrow c_i^+) \geq \delta$, $j = 1, ..., \eta$.*
2. *For the subset of parameters, $\sum_{k=1}^{\eta} P(e^k \leftarrow c_i^+) \leq \gamma$.*

After each $P(e^j \leftarrow c_i^+)$ is updated during descent, the above conditions still hold.

By Proposition 1, each round of descent starts with valid single-causals. By Proposition 2, for each step of descent, after each parameter is updated, the entire set of single-causals is still valid. Hence, the constrained gradient descent terminates with valid single-causals.

5 Experimental Results

5.1 Necessity of Flexible PCI Extraction

This experiment reveals difference between general and NAT CPTs and need for flexible PCI extraction. Two batches of CPTs are simulated each over $n = 5$ causes with all variable domain sizes being $k = 4$. The 1st batch consists of 100 random CPTs and the 2nd 100 NAT CPTs (of randomly selected NATs and single-causals).

Given a target CPT, for each pair of causes, Eqn. (1) is applied relative to each of e_1, e_2, and e_3. With $n = 5$, there are $C(5,2) = 10$ cause pairs. For each pair, there are $3 * 3 = 9$ active value pairs. For each pair, the PCI bit is well-defined if and only if one condition of Eqn. (1) holds for all 9 value pairs. A target CPT has between 0 and 10 well-defined PCI bits.

In the 1st batch, 97 CPTs have 0 well-defined PCI bit extracted. For each of the 3 remaining CPTs, one well-defined PCI bit is extracted relative to e_1, one relative to e_2, and one relative to e_3. Hence, the rate of well-defined PCI bits is 0.003 for each of e_1, e_2, and e_3. In the 2nd batch, 10 well-defined PCI bits are extracted from each CPT. This shows that general CPTs and NAT CPTs differ significantly and the flexible PCI pattern extraction presented in Sec. 3 is necessary.

5.2 Performance of Flexible PCI Extraction and Descent Search

This experiment examines compression error and efficiency gain from the flexible PCI extraction of Sec. 3, as well as the effectiveness of constrained gradient descent of Sec. 4. A 3rd batch of 100 random CPTs with $n = 4$ and $k \leq 4$ are generated. Each CPT is compressed by flexible PCI extraction and constrained descent, referred to as NAT-Com. It is also compressed by descent search exhaustively (hence optimally) for each of 52 NATs of $n = 4$, referred to as NAT-Opt. The choice $n = 4$ is made as NAT-Opt is much more costly for $n = 5$ with a total of 472 NATs.

Table 1 compares their performance, where ED refers to $ED(P_T, P_M)$, KL refers to $KL(P_T, P_M)$, RT refers to Runtime in seconds, and SR refers to Space Reduction. SR is the ratio of numbers of independent parameters between target CPT and NAT CPT. For instance, if $n = 4$ and $k = 3$ for all variables, the ratio is $(3^5 - 1)/(3 * 3 * 4) = 6.75$. SR and ED of NAT-Opt show that NAT compression by constrained descent is effective with significant space reduction (14.67)

Table 1. Performance summary of NAT-Com and NAT-Opt

	NAT-Com		NAT-Opt			NAT-Com		NAT-Opt	
	Mean	Stdev	Mean	Stdev		Mean	Stdev	Mean	Stdev
ED	0.204	0.043	0.189	0.036	SR	14.670	6.908	14.670	6.908
KL	22.189	32.590	17.287	20.822	RT	4.456	3.849	41.258	29.420

while incurring reasonable error (0.189 ED). NAT-Com has 8% larger ED (0.204) (same space reduction), but is 9 times faster. This shows that flexible PCI extraction trims NAT space significantly while retaining good NAT candidates. This efficiency gain is expected to grow exponentially with n as will be shown below.

5.3 Comparison between NAT and Noisy-MAX Compression

This experiment compares effectiveness of NAT compression with the well-known noisy-MAX as a baseline [13]. A 4th batch of 100 random CPTs with $n = 5$ and $k \leq 4$ and a 5th batch of 100 random CPTs with $n = 6$ and $k \leq 4$ are generated and are processed together with the 3rd batch ($n = 4$ and $k \leq 4$). Each CPT is compressed by NAT-Com, as well as by NMAX-Com where each target CPT is compressed into a noisy-MAX model.
Table 2 compares their performance. From the SR row, as n grows, space reduction by both method grows significantly (from 14.67 to 89.96). Since NAT-Com searches through multiple NATs while NMAX-Com processes a single causal model, NMAX-Com is about 10 times faster.

Table 2. Performance summary of NAT-Com and NMAX-Com

	NAT-Com ($n=4$)		NMAX-Com ($n=4$)		NAT-Com ($n=5$)		NMAX-Com ($n=5$)		NAT-Com ($n=6$)		NMAX-Com ($n=6$)	
	Mean	Stdev	Mean	Stdev	Mean	Stdev	Mean	Stdev	Mean	Stdev	Mean	Stdev
ED	0.20	0.04	0.23	0.07	0.27	0.08	0.37	0.10	0.32	0.09	0.45	0.06
KL	22.19	32.59	41.96	70.15	126.39	119.45	283.07	248.31	453.82	362.47	866.06	547.53
SR	14.67	6.91	14.67	6.91	36.60	20.73	36.60	20.73	89.96	53.32	89.96	53.32
RT	4.46	3.85	0.47	0.32	14.45	16.54	1.49	1.36	54.12	54.41	4.86	3.85

At the same time, as n grows, ED distance by NMAX-Com increases from 0.23 to 0.45, while ED distance by NAT-Com increases from 0.20 to 0.32. On average, NAT-Com reduces distance to target CPTs by 13%, 27%, and 29%, respectively. Since target CPTs are randomly generated, the experiment is conducted at the most general (worst) condition. It is expected that target CPTs from real BNs display more regularity [13] and compression accuracy by NAT-Com will be further reduced. We leave this to future work.

6 PCI Pattern Extraction with Persistent Leaky Causes

A leaky cause in a causal model integrates all causes that are not explicitly named. In the following, we assume that a leaky cause exists. We denote it by c_0 and denote other causes as $c_1,...,c_n$. A leaky cause may be persistent or non-persistent. A *non-persistent* leaky cause is not always active. Hence, it can be modeled the same way as other causes. A target CPT in the form $P(e|c_0,c_1,...,c_n)$ is fully specified with $P(e^0|c_0^0,c_1^0,...,c_n^0)=1$ and $P(e^+|c_0^0,c_1^0,...,c_n^0)=0$.
On the other hand, a *persistent* leaky cause (PLC) c_0 is always active, and a target CPT has the form $P(e|c_0^+,c_1,...,c_n)$. This has two implications. First, for all active e^+, we have $P(e^+|c_0^+,c_1^0,...,c_n^0)>0$. Second, parameters corresponding to $P(e|c_0^0,c_1,...,c_n)$ are unavailable. This raises an issue when we compress the target CPT into a NAT model. Since $P(e|c_0^0,c_1,...,c_n)$ is undefined, the target CPT appears as $P'(e|c_1,...,c_n)=P(e|c_0^+,c_1,...,c_n)$.

Example 1 *A target CPT $P(e|c_0,c_1)$ over binary e, c_0 and c_1, where c_0 is a PLC, is shown below (left). Since it is only partially defined and c_0 is an implicit cause, it may be viewed as $P'(e|c_1)$ (right) which is fully defined.*

c_0	c_1	e	$P(e\|c_0,c_1)$
c_0^0	c_1^0	e^0	*undefined*
c_0^0	c_1^0	e^1	*undefined*
c_0^0	c_1^1	e^0	*undefined*
c_0^0	c_1^1	e^1	*undefined*

c_0	c_1	e	$P(e\|c_0,c_1)$
c_0^1	c_1^0	e^0	*0.85*
c_0^1	c_1^0	e^1	*0.15*
c_0^1	c_1^1	e^0	*0.32*
c_0^1	c_1^1	e^1	*0.68*

c_1	e	$P'(e\|c_1)$
c_1^0	e^0	*0.85*
c_1^0	e^1	*0.15*
c_1^1	e^0	*0.32*
c_1^1	e^1	*0.68*

Should we define the NAT model over $\{e,c_1,...,c_n\}$ to match $P'(e|c_1,...,c_n)$? We reject this option for two reasons. First, the CPT of a NAT model thus defined has $P_M(e^+|c_1^0,...,c_n^0)=0$. From the first implication above, we have $P'(e^+|c_1^0,...,c_n^0)=P(e^+|c_0^+,c_1^0,...,c_n^0)>0$, which leads to an inherent modeling error.
Second, c_0 can undermine or reinforce another cause. There are 2^n possible causal interactions between c_0 and $c_1,...,c_n$. They do not approximate target CPT equally well. It is impossible to parameterize the NAT model according to the most suitable causal interaction, unless c_0 is explicitly represented. In the remainder, we assume that the NAT model is defined over family $\{e,c_0,c_1,...,c_n\}$.
To compress target CPT $P(e|c_0^+,c_1,...,c_n)$ into a NAT model, we need to extract a PCI pattern. By Eqn. (1), PCI bit $pci(c_i,c_j)$ is defined based on comparison among

$$P(e^+\leftarrow c_i^+), P(e^+\leftarrow c_j^+), \text{ and } P(e^+\leftarrow c_i^+,c_j^+).$$

Two difficulties arise when c_0 is a PLC. First, when $i=0$, the target CPT contains $P(e^+\leftarrow c_0^+)$ and $P(e^+\leftarrow c_0^+,c_j^+)$, but not $P(e^+\leftarrow c_j^+)$. Second, for $i,j>0$, none of $P(e^+\leftarrow c_i^+)$, $P(e^+\leftarrow c_j^+)$, and $P(e^+\leftarrow c_i^+,c_j^+)$ is specified. In summary, when c_0 is a PLC, no PCI bit can be extracted based on Eqn. (1), or based on rules in Sec. 3. One alternative is to extract $pci(c_i,c_j)$ for $i,j>0$ from comparison among

$$P(e^+\leftarrow c_0^+,c_i^+), P(e^+\leftarrow c_0^+,c_j^+), \text{ and } P(e^+\leftarrow c_0^+,c_i^+,c_j^+).$$

Unfortunately, although they are available from target CPT, it can be shown that causal interaction between c_i and c_j does not uniquely correspond comparison

among the three. To meet this challenge, we investigate another alternative. For simplicity in presentation, we assume binary variables, i.e., $D_e = \{e^-, e^+\}$ and $D_i = \{c_i^-, c_i^+\}$.
Given c_0, c_i and c_j where $i, j > 0$, there are 8 causal interaction relations as Fig. 1. We refer to the 8 NATs as T_a through T_h. Their PCI patterns are summarized in

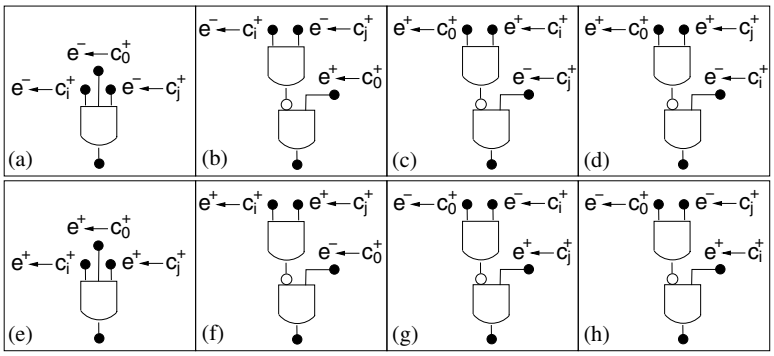

Fig. 1. NATs over c_0, c_i and c_j

Table 3. A target CPT $P(e|c_0^+, c_1, ..., c_n)$ specifies the following parameters that

Table 3. PCI patterns of NATs

	$pci(c_0, c_i)$	$pci(c_0, c_j)$	$pci(c_i, c_j)$		$pci(c_0, c_i)$	$pci(c_0, c_j)$	$pci(c_i, c_j)$
T_a	r	r	r	T_e	u	u	u
T_b	u	u	r	T_f	r	r	u
T_c	u	r	r	T_g	r	u	u
T_d	r	u	r	T_h	u	r	u

involve only c_0, c_i and c_j,

$$P(e^+ \leftarrow c_0^+), P(e^+ \leftarrow c_0^+, c_i^+), P(e^+ \leftarrow c_0^+, c_j^+), \text{ and } P(e^+ \leftarrow c_0^+, c_i^+, c_j^+).$$

We write them compactly as $P(^+0), P(^+0i), P(^+0j)$, and $P(^+0ij)$. If each NAT above can be identified by comparing these parameters, PCI bits in Table 3 will be obtained. We investigate this possibility below.
For T_a, since any disjoint subsets of $\{c_0, c_i, c_j\}$ reinforce each other (Fig. 1 (a)),

$$P(^+0ij) > P(^+0i), P(^+0ij) > P(^+0j), \text{ and } P(^+0ij) > P(^+0).$$

For T_e, since any disjoint subsets of $\{c_0, c_i, c_j\}$ undermine each other (Fig. 1 (e)),

$$P(^+0ij) < P(^+0i), P(^+0ij) < P(^+0j), \text{ and } P(^+0ij) < P(^+0).$$

For T_b (Fig. 1 (b)), $P(^+0i)$ results from interaction between c_0 and c_i. Since $P(^+0ij)$ results from interaction between c_0 and group $\{c_i, c_j\}$ where c_i is reinforced by c_j, it follows that $P(^+0ij) > P(^+0i)$. From symmetry between c_i and c_j, we derive $P(^+0ij) > P(^+0j)$. Since c_0 undermines the group $\{c_i, c_j\}$, it follows that $P(^+0ij) < P(^+0)$. From the dual relation between T_b and T_f (Fig. 1 (f)), we derive

$$P(^+0ij) < P(^+0i), P(^+0ij) < P(^+0j), \text{ and } P(^+0ij) > P(^+0).$$

For T_c (Fig. 1 (c)), since c_j reinforces group $\{c_0, c_i\}$, it follows that $P(^+0ij) > P(^+0i)$. Since $P(^+0ij)$ results from interaction between c_j and group $\{c_0, c_i\}$ where c_0 is undermined by c_i, it follows that $P(^+0ij) < P(^+0j)$. Since T_d (Fig. 1 (d)) is obtained from T_c by switching between c_i and c_j, we derive for T_d the following,

$$P(^+0ij) < P(^+0i) \text{ and } P(^+0ij) > P(^+0j).$$

To compare $P(^+0ij)$ and $P(^+0)$ for T_d, we analyze

$$\begin{aligned} &P(^-0) - P(^-0ij) = [1 - P(^+0)] - [1 - P(^+0)P(^+j)]P(^-i) \\ &= 1 - P(^+0) - P(^-i) + P(^+0)P(^+j)P(^-i) = P(^+i) - P(^+0)[1 - P(^+j)P(^-i)]. \end{aligned}$$

If $P(^+i)$ is close to 1, the sum is about $1 - P(^+0) > 0$. If $P(^+i)$ is close to 0, the sum is about $-P(^+0)P(^-j) < 0$. Hence, comparison of $P(^+0ij)$ and $P(^+0)$ is non-deterministic for T_d. Due to relation between T_d and T_c, the same holds for T_c.

From dual relation between T_d and T_h, for T_h, we have $P(^+0ij) > P(^+0i)$, $P(^+0ij) < P(^+0j)$, and non-deterministic comparison of $P(^+0)$ and $P(^+0ij)$. Since T_g results from switching c_i and c_j in T_h, we derive for T_g $P(^+0ij) < P(^+0i)$, $P(^+0ij) > P(^+0j)$, and non-deterministic comparison of $P(^+0)$ and $P(^+0ij)$.

Table 4. Causal probability comparison

	$P(^+0ij) - P(^+0i)$	$P(^+0ij) - P(^+0j)$	$P(^+0ij) - P(^+0)$	$P(^+0i) - P(^+0)$	$P(^+0j) - P(^+0)$	$P(^+0i) - P(^+0j)$
T_a	> 0	> 0	> 0			
T_b	> 0	> 0	< 0			
T_e	< 0	< 0	< 0			
T_f	< 0	< 0	> 0			
T_d	< 0	> 0	$+/-$	> 0	< 0	> 0
T_g	< 0	> 0	$+/-$	> 0	< 0	> 0
T_c	> 0	< 0	$+/-$	< 0	> 0	< 0
T_h	> 0	< 0	$+/-$	< 0	> 0	< 0

The first 4 columns of Table 4 summarize the above. It can be seen that T_a, T_b, T_e and T_f can be uniquely identified by the comparisons, and hence all three PCI bits in Table 3. The group of T_d and T_g can be identified from two comparisons, and so

can the group of T_c and T_h. This allows specification of $pci(c_0,c_i)$ and $pci(c_0,c_j)$. Since the two NATs in each group cannot be differentiated, $pci(c_i,c_j)$ cannot be specified.

There are $C(4,2)=6$ pairs of comparisons among $P(^+0),P(^+0i),P(^+0j)$, and $P(^+0ij)$, with the remaining shown in the last three columns of Table 4. Comparisons between $P(^+0i)$, $P(^+0j)$ and $P(^+0)$ in col. 5 and 6 are derived from Table 3. Col. 7 compares $P(^+0i)$ and $P(^+0j)$. For T_d and T_g, col. 5 and 6 imply $P(^+0i)>P(^+0)>P(^+0j)$. For T_c and T_h, col. 5 and 6 imply $P(^+0i)<P(^+0)<P(^+0j)$. As can be seen, col. 5, 6 and 7 do not improve differentiation.

We conclude the following from this analysis. If target CPT is an unknown NAT model with a PLC, a partial PCI pattern can be extracted by comparing $P(^+0)$, $P(^+0i),P(^+0j)$, and $P(^+0ij)$ for each pair of $i,j>0$. In particular, $pci(c_0,c_j)$ is extractable for all $j>0$. For $i,j>0$, 50% of bits $pci(c_i,c_j)$ are extractable on average. This result can be extended to general CPTs by applying the technique in Sec. 3 to probability comparison, which we do not elaborate here due to space.

Given $c_0,c_1,...,c_n$, there are $C(n+1,2)=(n+1)n/2$ PCI bits. Among them, $pci(c_i,c_j)$ where $i,j>0$ counts $C(n,2)=(n-1)n/2$ bits and $pci(c_0,c_j)$ counts n bits. Hence, the proposed framework allows extraction of $n(n+3)/4$ PCI bits on average. It follows that, as n grows from 4 (a total of 5 causes) to 12, the expected percentage of extractable PCI bits changes from 70% to 58%. In the next section, we outline future research regarding the remaining PCI bits.

7 Conclusion

The first contribution of this work is a flexible PCI pattern extraction that obtains a partial PCI pattern with a sufficient number of bits from general target CPTs. Experiment in Sec. 5.1 demonstrates the necessity of such a flexible extraction. Experiment in Sec. 5.2 shows that the extraction significantly reduces the number of candidate NATs for subsequent parameter estimation while incurs only minor loss of accuracy. The second contribution extends gradient descent for compression of binary NAT models to constrained descent for compression of multi-valued NAT models. Experiment in Sec. 5.3 shows that compression of random CPTs into NAT models achieves better accuracy than compression into noisy-MAX models. Further research will examine effectiveness of compression in real BN CPTs. The impact of compression errors to BN inference will also be evaluated.

The above compression assumes non-PLCs. If applied to CPTs with PLCs, modeling errors occur when all explicit causes are absent. The third contribution is a framework for extracting PCI pattern when PLCs exist, which significantly differs from PCI pattern extraction with non-PLCs. Further research is needed to answer the following open questions. With PLC presence, if a $pci(c_i,c_j)$ is unclassified, does assigning $pci(c_i,c_j)=u$ or r matter to accuracy of the resultant NAT model? If it does, can $pci(c_i,c_j)$ be inferred from higher order conditional probabilities? Answers to these questions will enable development of a compression algorithm for target CPTs with PLCs.

Acknowledgement

Financial support from NSERC Discovery Grant is acknowledged. We thank anonymous reviewers. We apologize for not moving explanations of figures and tables from text to captions, as it does not appear feasible to us.

References

1. F.V. Jensen, S.L. Lauritzen, and K.G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, (4):269–282, 1990.
2. A.L. Madsen and B. D'Ambrosio. A factorized representation of independence of causal influence and lazy propagation. *Inter. J. Uncertainty, Fuzziness and Knowledge-Based Systems*, 8(2):151–166, 2000.
3. A.L. Madsen and F.V. Jensen. Lazy propagation: A junction tree inference algorithm based on lazy evaluation. *Artificial Intelligence*, 113(1-2):203–245, 1999.
4. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
5. P. Savicky and J. Vomlel. Exploiting tensor rank-one decomposition in probabilistic inference. *Kybernetika*, 43(5):747–764, 2007.
6. M. Takikawa and B. D'Ambrosio. Multiplicative factorization of noisy-max. In *Proc. 15th Conf. Uncertainty in Artificial Intelligence*, pages 622–630, 1999.
7. Yang Xiang. Acquisition and computation issues with NIN-AND tree models. In P. Myllymaki, T. Roos, and T. Jaakkola, editors, *Proc. 5th European Workshop on Probabilistic Graphical Models*, pages 281–289, Finland, 2010.
8. Yang Xiang. Bayesian network inference with NIN-AND tree models. In A. Cano, M. Gomez-Olmedo, and T.D. Nielsen, editors, *Proc. 6th European Workshop on Probabilistic Graphical Models*, pages 363–370, Granada, 2012.
9. Yang Xiang. Non-impeding noisy-and tree causal models over multi-valued variables. *International J. Approximate Reasoning*, 53(7):988–1002, Oct 2012.
10. Yang Xiang, Yu Li, and Jingyu Zhu. Towards effective elicitation of NIN-AND tree causal models. In L. Godo and A. Pugliese, editors, *Inter. Conf. on Scalable Uncertainty Management (SUM 2009), LNCS 5785*, pages 282–296. Springer-Verlag Berlin Heidelberg, 2009.
11. Yang Xiang and Qing Liu. Compression of bayesian networks with nin-and tree modeling. In L.C. vander Gaag and A.J. Feelders, editors, *Probabilistic Graphical Models*, pages 551–566. Springer, 2014.
12. Yang Xiang and Minh Truong. Acquisition of causal models for local distributions in Bayesian networks. *IEEE Trans. Cybernetics*, 44(9):1591–1604, 2014.
13. A. Zagorecki and M.J. 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.
14. N. Zhang and D. Poole. Exploiting causal independence in bayesian network inference. *J. Artificial Intelligence Research*, 5:301–328, 1996.