Computational Intelligence, Volume 000, Number 000, 0000

NAT Model Based Compression of Bayesian Network CPTs over Multi-Valued Variables [1]

YANG XIANG AND QIAN JIANG

University of Guelph, Canada

Non-Impeding Noisy-AND (NIN-AND) Tree (NAT) models offer a highly expressive approximate representation for significantly reducing the space of Bayesian Networks (BNs). They also improve efficiency of BN inference significantly. To enable these advantages for general BNs, several technical advancements are made in this work to compress target BN Conditional Probability Tables (CPTs) over multi-valued variables into NAT models. We extend the semantics of NAT models beyond graded variables that causal independence models commonly adhered to, and allow NAT modeling in nominal causal variables. We overcome the limitation of well-defined Pairwise Causal Interaction (PCI) bits and present a flexible PCI pattern extraction from target CPTs. We extend parameter estimation for binary NAT models to constrained gradient descent for compressing target CPTs over multi-valued variables. We reveal challenges associated with persistent leaky causes (PLCs) and develop a novel framework for PCI pattern extraction when PLCs exist. The effectiveness of the CPT compression is validated experimentally.

Key words: Knowledge representation; Bayesian networks; causal independence models.

1. INTRODUCTION

When directions of links are (loosely) causally interpreted, a discrete BN quantifies causal strength between each effect and its n causes by a CPT whose number of parameters is exponential on n. Common Causal Independence Models (CIMs), e.g., noisy-OR (Pearl (1986)), reduce the number to being linear on n, but are limited in expressiveness. As members of CIM family, NAT models (Xiang (2012b,a); Xiang and Truong (2014); Xiang and Liu (2014)) 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 approximate representation for significantly reducing the space of BNs.

CIMs are not directly operable by common BN inference algorithms, e.g., the cluster tree method (Jensen et al. (1990)). A number of techniques have been proposed to overcome the difficulty (Zhang and Poole (1996); Madsen and D'Ambrosio (2000); Takikawa and D'Ambrosio (1999); Savicky and Vomlel (2007)). By multiplicatively factorizing NAT models and compiling NAT modeled BNs for lazy propagation (Madsen and Jensen (1999)), it has been shown that efficiency of exact inference with BNs can also be improved significantly (Xiang (2012a); Xiang and Jin (2016)).

The above efficiency gain is applicable to NAT modeled BNs (each CPT is a NAT model), or BNs over binary variables where each CPT must be compressed first into a NAT model (Xiang and Liu (2014)). It is not yet applicable to general BNs over multi-valued variables. The goal of this research is to achieve the efficiency gain for inference with general BNs by compressing their CPTs into multi-valued NAT models (Xiang (2012b)). Advancing compression from binary to multi-valued NAT models encounters several challenges. In this work, we investigate the following.

[1] **This article significantly extends Xiang and Jiang (2016).**

First, a general discrete BN can contain both ordinal variables, often referred to as *graded* variables (Diez (1993)), and nominal variables. CIMs are commonly limited to ordinal variables, e.g., (Zagorecki and Druzdzel (2013)). In this work, we generalize the semantics of NAT models to allow NAT models over nominal causal variables as well. This advancement, coupled with that on PLCs (below), enables NAT modeling on any discrete variables in a general BN.

Second, to gain efficiency with both space and inference time through NAT modeling, each target BN CPT 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 (Xiang et al. (2009)). 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 second contribution is a scheme that meets this challenge.

Third, once candidate NATs are obtained, probability parameters of the corresponding NAT models must be assessed. The third 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 contribution investigates specific constraints for compressing multi-valued CPTs.

Fourth, CIMs allow both explicit causes and implicit causes, termed leaky causes. Leaky causes may be persistent (PLCs) or non-persistent. We show that existence of PLCs raises another challenge. The fourth contribution is a framework for PCI pattern extraction with PLCs.

The remainder is organized as follows. Section 2 introduces the terminology and extends the semantics for NAT models beyond graded variables. How to extract PCI patterns from general target CPTs is presented in Section 3. Section 4 deals with constrained gradient descent for parameter estimation. PCI pattern extraction under the condition of PLCs is developed in Section 5. Experimental validations of the proposed techniques are reported in Section 6. We draw conclusions and discuss future work in Section 7.

2. EXTENDED SEMANTICS OF NAT MODELS

In this section, we extend the semantics for NAT models over multi-valued variables (Xiang (2012b)) beyond the commonly required graded variables, as well as introduce the necessary terminology.

The most commonly used CIMs is the noisy-OR, pioneered by Good (1961) and popularized by Pearl (1986). Henrion (1989) generalized noisy-OR to multi-valued variables and Diez (1993) explicitly defined these CIMs, known as noisy-MAX, to be over *graded* (essentially ordinal) variables. The rationales to require graded variables include (1) the interpretation of multiple values as presence of an entity with degrees of intensity, and (2) the allowance of expression of probability $P(e \leqslant e^j|...)$, where e is an effect variable and e^j is a degree of its intensity. The requirement of graded variables limits application of noisy-OR, noisy-MAX, and related CIMs to ordinal variables, and exclude nominal variables that also exist in general BNs. Below, we extend semantics of NAT models in Xiang (2012b) to relax the requirement for graded variables. Since NAT models generalize noisy-MAX (Xiang and Jin (2016)), the extended semantics also applies to noisy-OR, and noisy-MAX.

NAT models deal with uncertain causes. A cause is *uncertain* if it can render the effect but does not always do so. Smoking is a uncertain cause of lung cancer. We represent the effect and causes by *causal variables* defined below.

Definition 1 (Causal variable): A variable x that can be either *inactive* or be *active* possibly in multiple ways, and is involved in a causal relation, is a *causal variable* if when all causes of the effect are inactive, the effect is inactive with certainty.

Note that x may be either a cause or the effect in the relation. Whether x qualifies as a causal variable cannot be determined individually. According to Def. 1, x is deemed a causal variable when all variables in the relation are causal variables. For example, consider the cause of owning a pet (op) and the effect of fewer doctor visits (fv) (a well-known health benefit to pet owners), where

$$op \in \{none, dog, cat, rodent, fish, bird, horse, other\},$$

$$fv \in \{none, 1 \sim 4\%, 5 \sim 8\%, 9 \sim 12\%, 13\%+\}.$$

For both variables, $none$ is the inactive value. When $op = none$ and all other causes of fv are also inactive, we have $fv = none$ with certainty. A causal variable can be either ordinal, e.g., fv, or nominal, e.g., op. Note that op is not a graded variable.

We index the inactive value of a causal variable e as e^0, and its active values arbitrarily. For variable op, we can index its values $none$, dog, ..., $other$ as $op^0, op^1, ..., op^7$, respectively, and will use this indexing below. In practice, some orders of indexing on active values are preferred over others. However, the semantics of NAT models does not impose constraints on such orders. For variable fv, it's preferable to index its values $none, 1 \sim 4\%, 5 \sim 8\%, 9 \sim 12\%, 13\%+$ as $fv^0, fv^1, fv^2, fv^3, fv^4$, respectively, and we will use this indexing below. But indexing them as $fv^0, fv^4, fv^3, fv^2, fv^1$ is just as valid.

In general, we denote an effect by e and the set of all causes of e by $C = \{c_1, ..., c_n\}$, where e and all c_i are causal variables. The domain of e is $D_e = \{e^0, ..., e^\eta\}$ ($\eta > 0$) and 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 a certain range of values, 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.

A *simple single-causal success* is an event that a cause c_i with value c_i^j ($j > 0$) caused the effect e to occur at a value e^k ($k > 0$), when every other cause is inactive. We denote the probability of the event as $P(e^k \leftarrow c_i^j) = P(e^k | c_i^j, c_z^0 : \forall z \neq i)$ ($j > 0$). For example, $P(fv^3 \leftarrow op^2)$ is the probability of $9 \sim 12\%$ fewer doctor visits given that the only health inducing activity of the person is owning a cat.

A multi-causal success involves a set $X = \{c_1, ..., c_q\}$ ($q > 1$) of active causes, where each $c_i \in X$ has a value c_i^j ($j > 0$), when every other cause $c_m \in C \setminus X$ is inactive. A *congregate multi-causal success* is an event that causes in X collectively caused the effect to occur at a value e^k ($k > 0$) or of a higher index, when every other cause is inactive. We denote the probability of the event as

$$P(e \geqslant e^k \leftarrow c_1^{j_1}, ..., c_q^{j_q}) = P(e \geqslant e^k | c_1^{j_1}, ..., c_q^{j_q}, c_z^0 : c_z \in C \setminus X) \;\; (j > 0),$$

where $X = \{c_1, ..., c_q\}$ ($q > 1$). It is also denoted by $P(e \geqslant e^k \leftarrow \underline{x}^+)$.

A *congregate single-causal failure* refers to an event where $e < e^k$ ($k > 0$) when a cause c_i has a value c_i^j ($j > 0$) and every other cause is inactive. It is a failure in the sense that c_i fails to produce the effect with a value e^k or of a higher index. We denote the probability of the event as

$$P(e < e^k \leftarrow c_i^j) = P(e < e^k | c_i^j, c_z^0 : \forall z \neq i) \;\; (j > 0).$$

For example, $P(fv < fv^3 \leftarrow op^1)$ is the probability of less than 9% fewer doctor visits given that the only health inducing activity of the person is owning a dog.

Note that both success and failure events are based on value indexing of causal variables.

They generally do not have implications on intensity, except that inactive effect is impossible in a congregate success and is always possible in a congregate failure.

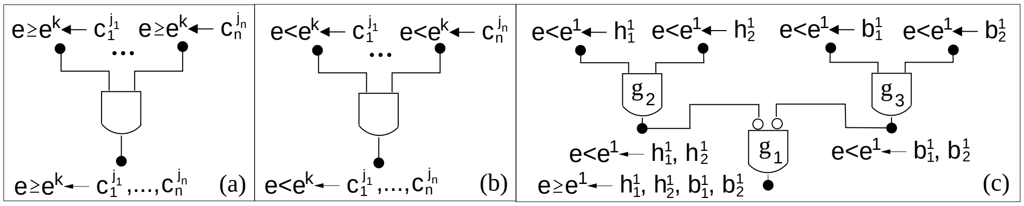
$e \geq e^k \leftarrow c_1^{j_1}$
$e \geq e^k \leftarrow c_n^{j_n}$
$e \geq e^k \leftarrow c_1^{j_1},...,c_n^{j_n}$
(a)
$e < e^k \leftarrow c_1^{j_1}$
$e < e^k \leftarrow c_n^{j_n}$
$e < e^k \leftarrow c_1^{j_1},...,c_n^{j_n}$
(b)
$e < e^1 \leftarrow h_1^1$
$e < e^1 \leftarrow h_2^1$
$e < e^1 \leftarrow b_1^1$
$e < e^1 \leftarrow b_2^1$
g_2
g_3
g_1
$e < e^1 \leftarrow h_1^1, h_2^1$
$e < e^1 \leftarrow b_1^1, b_2^1$
$e \geq e^1 \leftarrow h_1^1, h_2^1, b_1^1, b_2^1$
(c)

FIGURE 1. (a) A direct NIN-AND gate. (b) A dual NIN-AND gate. (c) A NAT.

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 (Fig. 1 (a)) is $e \geqslant e^k \leftarrow \underline{w}_i^+$ and the output event is $e \geqslant e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_q^+$. An input of a *dual* gate (Fig. 1 (b)) is $e < e^k \leftarrow \underline{w}_i^+$ and the output event is $e < e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_q^+$. The 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 as defined below.

Definition 2 (Reinforcing & undermining): Let e^k be an active effect value, $R = \{W_1, ...\}$ 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 $\forall R' \; P(e \geqslant e^k \leftarrow \underline{y}^+) \leqslant P(e \geqslant e^k \leftarrow \underline{x}^+)$. They *undermine* each other relative to e^k, iff $\forall R' \; P(e \geqslant e^k \leftarrow \underline{y}^+) > P(e \geqslant e^k \leftarrow \underline{x}^+)$.

Intuitively, when causes are reinforcing, more active causes render active effects more likely. When causes are undermining, more active causes render active effects less likely. Def. 2 defines causal interactions among both causes (when each W_i is a singleton) and groups of causes (when each W_i is a general set). It captures situations where causes within a group are reinforcing, but the groups undermine each other.

A direct gate models undermining and a dual gate models reinforcing. A NAT organizes multiple gates into a tree and expresses mixtures of reinforcing and undermining recursively. As an example, consider surface enhancer sprays. Acidic enhancers h_1 and h_2 are more effective when both are applied. Basic enhancers b_1 and b_2 work similarly. However, when enhancers from both groups are combined, the effectiveness is reduced. The NAT in Fig. 1 (c) expresses their causal interactions, where $C = \{h_1, h_2, b_1, b_2\}$ and the small ovals negate incoming events.

A NAT specifies the interaction between each pair of c_i and c_j, denoted by the *PCI bit* $pci(c_i, c_j) \in \{u, r\}$, where u stands for undermining and r for reinforcing. The collection of PCI bits is the *PCI pattern* of the NAT. The PCI pattern for the NAT in Fig. 1 (c) is

$$\{pci(h_1, h_2) = r, pci(h_1, b_1) = u, pci(h_1, b_2) = u,$$

$$pci(h_2, b_1) = u, pci(h_2, b_2) = u, pci(b_1, b_2) = r\}.$$

A NAT can be uniquely identified by its PCI pattern (Xiang and Truong (2014)).

Given the NAT in Fig. 1 (c) and probabilities of its input events, called *single-causals*, $P(e \geqslant e^1 \leftarrow h_1^1, h_2^1, b_1^1, b_2^1)$ can be obtained. From the single-causals and all derivable NATs (Xiang (2010)), the CPT $P(e|h_1, h_2, b_1, b_2)$ is uniquely defined (Xiang (2012b)).

3. EXTRACTING PCI PATTERNS FROM GENERAL CPTS

In this work, we extend the framework for compressing CPTs over binary variables (Xiang and Liu (2014)) to target CPTs over multi-valued causal variables. The compression consists of the following steps.

(1) Extract one or more PCI patterns from the target CPT.
(2) Retrieve NAT structures that are compatible with the PCI patterns.
(3) Search for numerical parameters for each NAT structure.
(4) Return the NAT structure and its parameters that best approximate the target CPT.

This section focuses on step (1) and the next section on step (3). Details on step (2) can be found in the above reference.

To compress a target CPT over e and C into a NAT model, we need to determine candidate NATs over C. This can be achieved by searching for a PCI pattern relative to each active e^k and determine the NAT by the best pattern over all $k > 0$. By Def. 2, 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.

$$pci(c_i, c_j) = \\ \left\{ \begin{array}{lllll} u & : & P(e \geqslant e^k \leftarrow c_i^+, c_j^+) & < & min(P(e \geqslant e^k \leftarrow c_i^+), P(e \geqslant e^k \leftarrow c_j^+)), \\ r & : & P(e \geqslant e^k \leftarrow c_i^+, c_j^+) & \geqslant & max(P(e \geqslant e^k \leftarrow c_i^+), P(e \geqslant e^k \leftarrow c_j^+)). \end{array} \right. \quad (1)$$

As shown experimentally (Section 6.1), 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 many candidate NATs, making the subsequent parameter search costly. Below, we develop a scheme to overcome the difficulty where too few bits are well defined in PCI patterns for all k.

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 $\{u, r, null\}$ with $null$ for unclassified.

For a well-defined bit, one condition in Eqn. (1) must hold for all active cause value pairs. Consider the interaction between one value pair first. To indicate the e^k value, we denote the interaction as $pci(e^k, c_i^+, c_j^+) \in \{u, r, null\}$, and refer to it as a *value-pair interaction* relative to e^k. To simplify the notation, we denote

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

as p, q, and t, respectively. The rule below extracts a well-defined value-pair interaction.

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

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

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 $null$ 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 $|p - q| \leqslant \tau_0$, where $\tau_0 \in (0, 1)$ is a given threshold, then $pci(e^k, c_i^+, c_j^+) = null$.

The rational of Rule 2 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 the value $null$ entails.

We refer to the 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 may be 0.4. Its rational follows from the above analysis.

Rule 3 (Sided loose enclosure): If $t \in [min(p,q), max(p,q)]$ and $|p-q| > \tau_0$, then

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

where $\tau_0 \in (0,1)$ and $\tau_1 \in (0, 0.5)$ are given thresholds.

Proposition 1 shows that the above rules are complete for deciding the interaction between a value pair.

Proposition 1 (Completeness): For any value combination of $e^k, c_i^+, c_j^+, \tau_0, \tau_1$ where $k > 0$, exactly one of Rules 1, 2 and 3 applies, which assigns $pci(e^k, c_i^+, c_j^+)$ uniquely.

Proof. The preconditions of Rules 1, 2 and 3 are mutually exclusive and exhaustive. Hence, for any value combination, exactly one rule fires.

The two cases of Rule 1 are mutually exclusive and exhaustive given its precondition. Hence, if Rule 1 fires, u or r is assigned to $pci(e^k, c_i^+, c_j^+)$. If Rule 2 fires, $null$ is assigned to $pci(e^k, c_i^+, c_j^+)$. The three cases of Rule 3 are mutually exclusive and exhaustive given its precondition. Hence, if Rule 3 fires, $pci(e^k, c_i^+, c_j^+)$ is assigned u, r, or $null$. □

Proposition 2 shows that the above rules select u or r whenever one type of causal interaction is more likely than the other.

Proposition 2 (Soundness): Let τ_0 and τ_1 be reduced continuously. In the limit, $pci(e^k, c_i^+, c_j^+)$ can only be assigned $null$, if one of the following holds.

(1) $p = q = t$
(2) $p \neq q$ and $t = 0.5(p+q)$

Proof. Assigning $null$ to $pci(e^k, c_i^+, c_j^+)$ can only occur if Rule 2 or 3 is fired. If τ_0 is reduced continuously, the only situation where Rule 2 can fire is when $p = q = t$.

Suppose that Rule 3 is fired. Then $p \neq q$ and $t \in [min(p,q), max(p,q)]$. If τ_1 is reduced continuously, the only situation where $pci(e^k, c_i^+, c_j^+)$ is assigned $null$ is when $t = 0.5(p+q)$ and $R = 0$. □

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. Its lower bound of τ_2 ensures that the first two cases cannot both be true. 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 $\tau_2 \in (0.5, 1)$,

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

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 one with the most bits as the PCI pattern.

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 $pci(e^k, c_i, c_j) \neq null$.

Then select $pci(e^x)$ as the partial PCI pattern if $N_x = max_k \, N_k$.

The value of N_x is between 0 and $C(n, 2)$ in general. If N_x is too close to $C(n, 2)$, there are very few candidate NATs, which reduces the space for subsequent parameter search, and can ultimately reduce the accuracy of compression. If N_x is too far from $C(n, 2)$, there are many candidate NATs, which renders the subsequent parameter search costly. The value of N_x for a given target CPT depends on the setting of τ_0 through τ_2. If the thresholds are too relaxed, fewer $null$ PCI bits will be assigned and N_x will be closer to $C(n, 2)$. To avoid such situations, initial threshold values should be tight. If the resultant N_x is too small, relax the thresholds to increase the N_x value.

To implement the above dynamic control, we uses an additional threshold $\tau_3 \in (0, 1)$ and tight initial values for τ_0 through τ_2. If $N_x > \tau_3 \, C(n, 2)$, the PCI pattern from Rule 5 is accepted. Otherwise, τ_0 through τ_2 are relaxed and search is repeated, until $N_x > \tau_3 \, C(n, 2)$. The effectiveness of the procedure is experimentally validated in Section 6.

4. PARAMETER ESTIMATION WITH CONSTRAINED GRADIENT DESCENT

Once a partial PCI pattern is extracted, the set of candidate NATs compatible with the pattern can be determined (Xiang and Truong (2014)). For each candidate NAT, single-causals can be estimated from target CPT through a 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 (Xiang and Liu (2014)). In this section, we extend the gradient descent for compression of multi-valued CPTs.

Our objective is to approximate a target CPT P_T with a NAT model M. M consists of a NAT and a set of single-causals, which defines a CPT P_M. P_T consists of a set of conditional probability distributions (CPDs), $P(e|\underline{c})$, where $\underline{c}$ is an instantiation of C. We index the CPDs as $P_T(0), ..., P_T(Z-1)$, where $P_T(0)$ has the condition $\underline{c} = (c_1^0, ..., c_n^0)$, and Z counts the CPDs. **A CPT over an effect and ten binary causes has 1024 CPDs.**

We measure the similarity of P_M from P_T by the average Kullback −Leibler divergence,

$$KL(P_T, P_M) = \frac{1}{Z} \sum_{i=0}^{Z-1} \sum_{j} P_T(i,j) log \frac{P_T(i,j)}{P_M(i,j)}, \tag{2}$$

where i indexes CPDs in P_T and P_M, and j indexes probabilities in each CPD. Gradient descent estimates the set of single-causals of M such that $KL(P_T, P_M)$ is minimized. In the experimental study (Section 6), the average Euclidean distance

$$ED(P_T, P_M) = \sqrt{\frac{1}{K}\sum_{i=0}^{Z-1}\sum_{j}(P_T(i,j) - P_M(i,j))^2}$$

is also obtained, where K counts probabilities 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 parameters $P(e^1 \leftarrow c_i^+), ..., P(e^\eta \leftarrow c_i^+)$ are not 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 as presented below.

At the start of each round of descent, each group of η single-causals under the same constraint are initialized together as follows. Let δ and γ be small constants close to 0. Generate $\eta + 1$ random numbers in $(0, 1)$, normalize them, scale each by $1 - \gamma$, and replace those $< \delta$ by δ. If the sum > 1 due to replacement, repeat the above until no replacement occurs after the scaling. Drop one arbitrarily and assign the remaining as initial single-causals. Lemma 1 summarizes properties of the initialization.

LEMMA 1 (Initialization): 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, $P(e^j \leftarrow c_i^+) \geqslant \delta$, $j = 1, ..., \eta$.
(2) For the subset of parameters, $\sum_{k=1}^{\eta} P(e^k \leftarrow c_i^+) \leqslant 1 - \gamma$.

Proof. The condition (1) holds due to the replacement. After normalization and scaling, if no replacement occurs, the $\eta + 1$ numbers sum to $1 - \gamma$. Hence, the condition (2) holds. □

Each step of gradient descent updates the ηmn parameters in sequence. To ensure that both conditions of Lemma 1 continue to hold, we constrain the descent as follows. For each $P(e^j \leftarrow c_i^+)$, after it is updated, check whether $P(e^j \leftarrow c_i^+) < \delta$. If so, set $P(e^j \leftarrow c_i^+) = \delta$ and stop it from further descent. Otherwise, check if $S = \sum_{k=1}^{\eta} P(e^k \leftarrow c_i^+) > 1 - \gamma$. If so, set $P(e^j \leftarrow c_i^+)$ to $P(e^j \leftarrow c_i^+) - (S - (1 - \gamma))$ and stop it from further descent. If $P(e^j \leftarrow c_i^+)$ passes both tests, commit to the value and continue its descent. Theorem 1 summarizes properties of the method.

THEOREM 1 (Descent): 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, $P(e^j \leftarrow c_i^+) \geqslant \delta$, $j = 1, ..., \eta$.
(2) For the subset of parameters, $\sum_{k=1}^{\eta} P(e^k \leftarrow c_i^+) \leqslant 1 - \gamma$.

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

Proof. Suppose $P(e^j \leftarrow c_i^+)$ is just updated from $P'(e^j \leftarrow c_i^+)$. If $P(e^j \leftarrow c_i^+)$ passes both tests, both conditions holds. If it fails the 1st test, it is modified and the 2nd test is not run (Case 1). If it passes the 1st and fails the 2nd, it is also modified (Case 2). Below, we show that, in either case, both conditions hold.

[Case 1] Failure of the 1st test sets $P(e^j \leftarrow c_i^+)$ to δ and renders the condition (1). By assumption, both conditions hold before $P(e^j \leftarrow c_i^+)$ is updated from $P'(e^j \leftarrow c_i^+)$. If the 1st test fails, we have $P(e^j \leftarrow c_i^+) < \delta \leqslant P'(e^j \leftarrow c_i^+)$. After $P(e^j \leftarrow c_i^+)$ is raised to δ, we have $P(e^j \leftarrow c_i^+) \leqslant P'(e^j \leftarrow c_i^+)$, and the condition (2) continues to hold.

[Case 2] Denote the sum before and after the update by S' and S. By assumption, the condition (2) holds before $P(e^j \leftarrow c_i^+)$ is updated from $P'(e^j \leftarrow c_i^+)$. Hence, the violation is due to the increase $P(e^j \leftarrow c_i^+) > P'(e^j \leftarrow c_i^+)$. We claim that the increase must satisfy $P(e^j \leftarrow c_i^+) - P'(e^j \leftarrow c_i^+) \geqslant S - (1 - \gamma)$.

Assume that it is false and $P(e^j \leftarrow c_i^+) - P'(e^j \leftarrow c_i^+) < S - (1 - \gamma)$. Before the update, we have $S' \leqslant 1 - \gamma$. Combining the two inequalities, we have

$$S = S' - P'(e^j \leftarrow c_i^+) + P(e^j \leftarrow c_i^+) < 1 - \gamma + S - (1 - \gamma) = S,$$

which is a contradiction. Therefore, our claim holds, which implies that the modified value satisfies $P(e^j \leftarrow c_i^+) - (S - (1 - \gamma)) \geqslant P'(e^j \leftarrow c_i^+) \geqslant \delta$. Hence, the condition (1) holds after the modification.

Furthermore, after the modification, we have the new sum

$$S - P(e^j \leftarrow c_i^+) + [P(e^j \leftarrow c_i^+) - (S - (1 - \gamma))] = 1 - \gamma,$$

and the condition (2) also holds. □

By Lemma 1, each round of descent starts with valid single-causals. By Theorem 1, 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.

Once the parameters for each candidate NAT are determined, a candidate NAT model (the NAT and its parameters) is fully specified. The NAT model with the smallest average KL distance is the compressed model of the target CPT.

5. PCI PATTERN EXTRACTION WITH PERSISTENT LEAKY CAUSES

5.1. The Challenge

A leaky cause in a causal model 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 in 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 conditions $(c_0^0, c_1, ..., c_n)$ never hold. 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 takes the form $P'(e|c_1, ..., c_n)$ (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 $k > 0$ as required by Def. 1. Third, the search for parameters cannot be based on the average KL distance as defined in Section 4. Each term of the distance from a target CPT P_T is formed $P_T(i,j)\ log(P_T(i,j)/P_M(i,j))$, where i indexes CPDs and j 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, the corresponding terms for the distance are undefined (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 a value-pair interaction $pci(e^k, c_0^1, c_j^w)$ by Def. 2, we need to compare

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

where $j, k, w > 0$. However, $P(e \geqslant e^k \leftarrow c_j^w)$ is unavailable since c_0 is a PLC. Furthermore, to determine $pci(e^k, c_i^v, c_j^w)$, where $i, j, k, v, w > 0$, we need to compare $P(e \geqslant e^k \leftarrow c_i^v)$, $P(e \geqslant e^k \leftarrow c_j^w)$, and $P(e \geqslant e^k \leftarrow c_i^v, c_j^w)$, but none is available for the same reason.

To overcome the unavailability, it is plausible to compare instead the available

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

We show below that the value-pair interaction $pci(e^k, c_i^v, c_j^w)$ cannot be uniquely determined by the comparison. In particular, we show that the following conditions can coexist.

$$P(e \geqslant e^k \leftarrow c_i^v, c_j^w) \quad > \quad max(P(e \geqslant e^k \leftarrow c_i^v), P(e \geqslant e^k \leftarrow c_j^w)) \tag{3}$$

$$P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) \quad < \quad min(P(e \geqslant e^k \leftarrow c_0^1, c_i^v), P(e \geqslant e^k \leftarrow c_0^1, c_j^w)) \tag{4}$$

Similarly, the following conditions can also coexist.

$$P(e \geqslant e^k \leftarrow c_i^v, c_j^w) \quad < \quad min(P(e \geqslant e^k \leftarrow c_i^v), P(e \geqslant e^k \leftarrow c_j^w)) \tag{5}$$

$$P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) \quad > \quad max(P(e \geqslant e^k \leftarrow c_0^1, c_i^v), P(e \geqslant e^k \leftarrow c_0^1, c_j^w)) \tag{6}$$

Proposition 3 (Reinforce): 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 $P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) > P(e \geqslant e^k \leftarrow c_0^1, c_i^v)$, as well as NAT models where the opposite holds.

Proof. Fig. 2 shows NATs T_a and T_d over c_0, c_i, c_j, where c_i and c_j are reinforcing, and labels of output events are omitted. In T_a, since c_j reinforces c_0 and c_i, we have comparison $P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) > P(e \geqslant e^k \leftarrow c_0^1, c_i^v)$. Note that although Def. 2 allows equality between causal probabilities in the reinforcing case, the equality never occurs to probabilities associated with NIN-AND gates due to product of factors in $(0, 1)$.

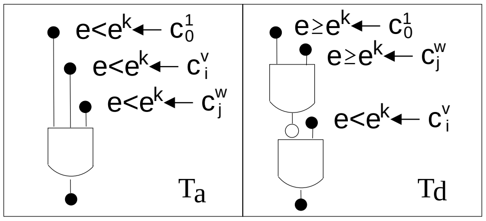

FIGURE 2. (Sub)NATs where c_i and c_j are reinforcing

In T_d, c_0 and c_i are reinforcing, and c_0 is undermined by c_j at the top gate. Hence, $P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) < P(e \geqslant e^k \leftarrow c_0^1, c_i^v)$. □

Proposition 3 shows that, when c_i and c_j are reinforcing ($pci(e^k, c_i^v, c_j^w) = r$) and hence

Eqn (3) holds, there is no guarantee for

$$P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) > \max(P(e \geqslant e^k \leftarrow c_0^1, c_i^v), P(e \geqslant e^k \leftarrow c_0^1, c_j^w)).$$

Proposition 4 (Undermine): 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.

Proof. Fig. 3 shows NATs T_e and T_h over c_0, c_i, c_j, where c_i and c_j are undermining. In T_e, c_j undermines c_0 and c_i, we have $P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) < P(e \geqslant e^k \leftarrow c_0^1, c_i^v)$. That is, $P(e < e^k \leftarrow c_0^1, c_i^v, c_j^w) > P(e < e^k \leftarrow c_0^1, c_i^v)$. In T_h, c_0 and c_i are undermining, and c_0 is reinforced by c_j at the top gate. Hence, $P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) > P(e \geqslant e^k \leftarrow c_0^1, c_i^v)$. That is, $P(e < e^k \leftarrow c_0^1, c_i^v, c_j^w) < P(e < e^k \leftarrow c_0^1, c_i^v)$. □

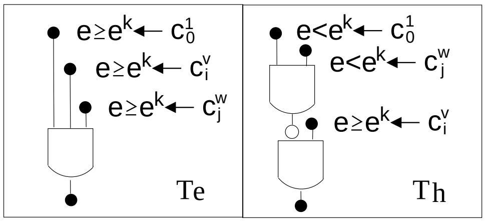

FIGURE 3. (Sub)NATs where c_i and c_j are undermining

Proposition 4 shows that when c_i and c_j are undermining ($pci(e^k, c_i^v, c_j^w) = u$) and hence Eqn (5) holds, there is no guarantee for

$$P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w) < \min(P(e \geqslant e^k \leftarrow c_0^1, c_i^v), P(e \geqslant e^k \leftarrow c_0^1, c_j^w)).$$

From Propositions 3 and 4, it follows that $pci(e^k, c_i^v, c_j^w)$ cannot be determined soly based on comparing

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

Although Propositions 3 and 4 involve c_0, their proofs 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 $pci(e^k, c_i^v, c_j^w)$.

Below, we present a solution to meet the challenge.

5.2. 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:

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

Fig. 4 enumerates (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. Given a target CPT, if we can

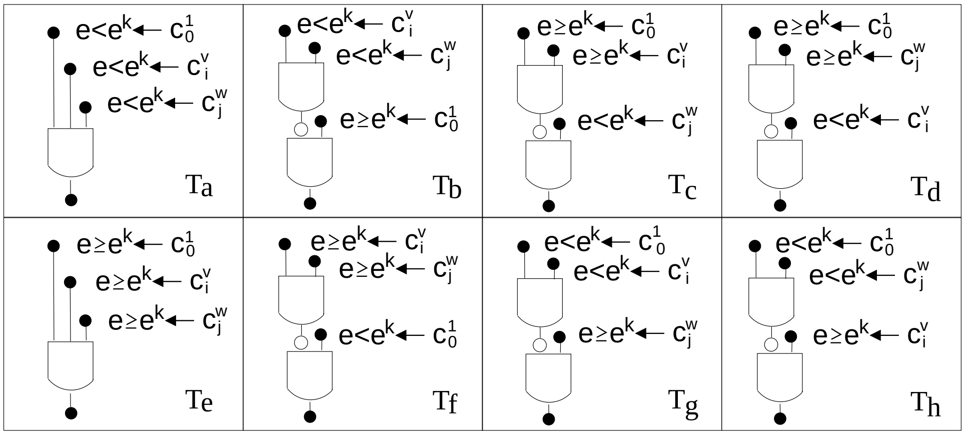

FIGURE 4. (Sub)NATs over c_0, c_i and c_j

TABLE 1. Value-pair interactions of NAT models

	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

identify which NAT in Fig. 4 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 5 and Table 2.

Proposition 5 (Comparison): Let T_a through T_h be NAT models over causes c_0, c_i, and c_j. Pair-wise comparisons among $P(e \geqslant e^k \leftarrow c_0^1)$, $P(e \geqslant e^k \leftarrow c_0^1, c_i^v)$, $P(e \geqslant e^k \leftarrow c_0^1, c_j^w)$, and $P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w)$ hold as Table 2.

TABLE 2. Pairwise causal probability comparison by NAT models

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

Proof. From Fig. 4, c_j reinforces the other two causes in T_a and T_c. In T_b and T_h, c_j reinforces another cause at the top gate. The result is $+$ in row 1 of Table 2 for these NATs. In T_e and T_g, c_j undermines the other two causes. In T_f and T_d, c_j undermines another cause at the top gate. The result is $-$ in row 1 for these NATs. Hence, we have row 1.

Cause c_i reinforces the other two causes in T_a and T_d, and reinforces another cause at the top gate in T_b and T_g. It undermines the other two causes in T_e and T_h, and undermines another cause at the top gate in T_c and T_f. Hence, we have row 2.

Causes c_i and c_j as a group reinforce c_0 in T_a and T_f, but undermine c_0 in T_b and T_e. Hence, we have the corresponding results in row 3 for these NATs. In the other NATs, one of c_i and c_j reinforces c_0 and the other one undermines c_0. The comparison result can go either way, depending on the relative causal strength of c_i and c_j. Hence, we have row 3.

Row 4 compares two double-causal probabilities, with c_i being inactive in one and c_j being inactive in the other. In T_a and T_f, both c_i and c_j reinforces c_0. Which double-causal probability is larger depends on their relative causal strength. In T_b and T_e, both c_i and c_j undermines c_0. The similar applies.

In T_d and T_g, c_i reinforces c_0 and c_j undermines c_0. Hence, the comparison result is $+$. In T_c and T_h, c_i undermines c_0 and c_j reinforces c_0. Hence, the result is $-$.

Row 5 is implied by $pci(e^k, c_0^1, c_i^v)$ in Table 1, and row 6 by $pci(e^k, c_0^1, c_j^w)$. □

From Proposition 5, 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). Below, we explore a novel idea to extend the partial solution into a complete solution.

5.3. 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. 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 $i, j, k > 0$, cannot be identified, which renders the corresponding PCI bit $pci(c_i, c_j)$ unspecified. Since the number of candidate NATs grow exponentially on the number of unspecified PCI bits, presence of many such bits has a significant consequence on the cost of subsequent parameter search.

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

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

If c_0 and c_i are reinforcing, $P(e < e^k \leftarrow c_0^1, c_i^v) = P(e < e^k \leftarrow c_0^1)P(e < e^k \leftarrow c_i^v)$, and we 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 the group $\{T_d, T_g\}$, c_0 and c_i are reinforcing, and c_0 and c_j are undermining. If we can estimate single-causals $P(e \geqslant e^k \leftarrow c_i^v)$ and $P(e \geqslant e^k \leftarrow c_j^w)$ accordingly from the available parameters

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

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

For the 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

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

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

Applying comparisons in rows 1 and 2 of Table 2, it fits the group $\{T_c, T_h\}$ with $(+, -)$. However, since $P(e \geqslant e^k \leftarrow c_0^1, c_j^w) < P(e \geqslant e^k \leftarrow c_0^1)$, c_0 and c_j do not reinforce as T_c and T_h expected. As the result, estimation of $P(e \geqslant e^k \leftarrow c_j^w)$ 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 $P(e \geqslant e^k \leftarrow c_i^v)$ and $P(e \geqslant e^k \leftarrow c_j^w)$ assuming undermining, which will now succeed. We then estimate $P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w)$ 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

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

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

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 are $(+, +)$, making estimation of $P(e \geqslant e^k \leftarrow c_j^w)$ 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

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

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 the above target probabilities 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.

Algorithm 1: *(Input: q, r, s and t)*

```
1  compute sign pattern $pat_1 = (sign(t-r), sign(t-s), sign(t-q))$;
2  if $pat_1$ matches that of $T_a$, $T_b$, $T_e$, or $T_f$, return the matching NAT;
3  compute sign pattern $pat_2 = (sign(r-q), sign(s-q))$;
4  if $pat_2$ matches that of $\{T_d, T_g\}$ or $\{T_c, T_h\}$, do
5     estimate $x \equiv P(e \geqslant e^k \leftarrow c_i^v)$ and $y \equiv P(e \geqslant e^k \leftarrow c_j^w)$ by matching group;
6     for each member NAT $T_\beta$ of the matching group, do
7        compute multi-causal $z \equiv P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w)$ from $q, x, y$ and NAT $T_\beta$;
8        compute difference $|t-z|$;
9     return the NAT with the smaller difference;
10 match $pat_2$ against that of $\{T_a, T_f\}$ or $\{T_b, T_e\}$;
11 estimate $x' \equiv P(e \geqslant e^k \leftarrow c_i^v)$ and $y' \equiv P(e \geqslant e^k \leftarrow c_j^w)$ by the matching group;
12 for each member $T'_\beta$ of the matching group, do
13    compute $z' \equiv P(e \geqslant e^k \leftarrow c_0^1, c_i^v, c_j^w)$ from $q, x', y'$ and $T'_\beta$;
14    compute difference $|t-z'|$;
15 return the NAT with the smaller difference;
```

Theorem 2 establishes the most important property of Algorithm 1.

THEOREM 2 (Soundness): For any target probabilities

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

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

Proof. Assume that no ties are involved in sign computation and difference comparison. If pat_1 matches rows 1, 2, and 3 of Table 2, one of T_a, T_b, T_e, or T_f will be returned by line 2. Otherwise, pat_2 is computed (line 3), with four possible outcomes.

If the outcome is $(+,-)$ or $(-,+)$, lines 4 to 9 will be executed. If $pat_2 = (+,-)$, line 4 matches the group $\{T_d, T_g\}$. Line 5 estimates x, according to reinforcing c_0 and c_i, and estimates y, according to undermining c_0 and c_j. The values z and $|t-z|$ will be computed for both T_d and T_g in lines 6 to 8. One of T_d and T_g will be returned in line 9. If $pat_2 = (-,+)$, the process is similar, but returns one of T_c and T_h.

If $pat_2 = (+,+)$ or $(-,-)$, the group $\{T_a, T_f\}$ or $\{T_b, T_e\}$ will be processed in a similar fashion by lines 10 to 15, and a unique NAT will be returned. □

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, Theorem 2 implies that the causal interaction $pci(e^k, c_i^v, c_j^w)$ for any k, i, j values can be extracted.

However, $pci(e^k, c_i^v, c_j^w)$ from a general target CPT may differ for different k, v, w values (not so for a NAT model). Hence, for a given combination of k, i, j, Rule 4 must be applied to determine $pci(e^k, c_i, c_j)$. Since a null outcome is possible, the resultant PCI pattern for e^k is partial in general. Furthermore, the PCI patterns for different e^k may differ, in which case, we apply all of them to generation of candidate NATs.

6. EXPERIMENTS

To validate the framework for compression of general CPTs into multi-valued NAT models and the techniques presented above, several experiments are conducted. In the following, we report the objective, the setup, and the result for each experiment.

6.1. Necessity of Flexible PCI Extraction

This experiment reveals the difference between general CPTs and NAT CPTs, to justify the need for flexible PCI extraction. Two batches of CPTs are simulated each over $n = 5$ causes with domain sizes of all variables being $k = 4$. The 1st batch consists of 100 random CPTs **(without local structure)** 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.

For the 1st batch, 0 well-defined PCI bit are extracted from 97 CPTs. 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 extraction rate of well-defined PCI bits is $9/3000 = 0.003$. 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 Section 3 is necessary.

6.2. Compression Accuracy Relative to the Optimal

In this experiment, we evaluate the effectiveness of our PIC extraction techniques. The techniques reduce the number of candidate NATs to a small subset in the search space (exponential on n). It is important to assess whether such reduction retains good candidate NATs. To this end, we compare our methods with exhaustively evaluating all NATs (optimal). We refer to our compression method without PLC modeling as NPLC-Comp, and the corresponding optimal method as NPLC-Opt. The methods with explicit PLC modeling are referred to as PLC-Comp and PLC-Opt, respectively. Since the optimal methods are intractable, smaller n values are used. We denote the maximum domain size of variables in each target CPT by k.

To evaluate NPLC-Comp, 100 random CPTs are generated without PLC, where $n = 4$ and $k = 4$. Table 3 shows the experiment result where ED, KL, SR, and RT refer to Euclidean distance, KL distance, space reduction, and runtime (in seconds), respectively.

TABLE 3. Experimental comparison of NPLC-Comp and NPLC-Opt on random CPTs without PLC

	NPLC-Comp		NPLC-Opt	
	Mean	Stdev	Mean	Stdev
ED	0.1928	0.0378	0.1869	0.0356
KL	0.1778	0.0743	0.1636	0.0587
SR	14.67	6.91	14.67	6.91
RT	7.84	6.31	49.96	37.95

NPLC-Comp runs about 6 times faster as NPLC-Opt, and incurred only slightly larger compression errors. We conducted a single-sided t-test based on KL distance with the null hypothesis H_0: NPLC-Comp has the same compression error as NPLC-Opt. The null hypothesis is accepted at the level of significance $\alpha = 0.025$ and is rejected at $\alpha = 0.05$.

To evaluate PLC-Comp, random CPTs are generated with PLCs and $k = 4$. The first 100 CPTs have $n = 3$ and the second 100 CPTs have $n = 4$. Hence, compressed NAT models have $n = 4$ and $n = 5$, respectively. Tables 4 and 5 show the experiment results.

TABLE 4. Experimental comparison of PLC-Comp and PLC-Opt where target CPTs have $n = 3$

	PLC-Comp		PLC-Opt	
	Mean	Stdev	Mean	Stdev
ED	0.1572	0.0428	0.1538	0.0421
KL	0.1021	0.0348	0.0976	0.0329
SR	6.53	2.55	6.53	2.55
RT	11.92	7.50	56.85	40.35

TABLE 5. Experimental comparison of PLC-Comp and PLC-Opt where target CPTs have $n = 4$

	PLC-Comp		PLC-Opt	
	Mean	Stdev	Mean	Stdev
ED	0.1802	0.0385	0.1701	0.0369
KL	0.1495	0.0514	0.1294	0.0365
SR	13.83	6.12	13.83	6.12
RT	23.99	18.47	1026.99	495.50

For target CPTs with $n = 3$, PLC-Comp is about 5 times faster than PLC-Opt. The single-sided t-test accepted the null hypothesis at $\alpha = 0.05$. For target CPTs with $n = 4$, PLC-Comp is about 43 times faster. The KL-distance of PLC-Opt is at 0.1294 while that of PLC-Comp is at 0.1495. As the result, the null hypothesis is rejected at $\alpha = 0.005$.

In summary, the experimental results demonstrate that our PCI extraction techniques (with and without PLCs) reduce NAT search space effectively while retaining good candidate NATs.

6.3. Compressions of Random CPTs

In this experiment, we evaluate both accuracy and efficiency of NAT compression as the number n of causes grows. As a base-line, we compare our methods, NPLC-Comp and PLC-Comp, with the popular noisy-MAX, denoted NMAX below. NMAX uses a fixed structure and hence only parameter search is involved.

For target CPTs without PLC, we generated 100 random CPTs with $n = 4$, another 100 CPTs with $n = 5$, and a third 100 CPTs with $n = 6$, where $k = 4$ for all. Parameters in each CPT are non-extreme, except $P(e|c_1^0, ..., c_n^0)$. The compression results by NPLC-Comp and NMAX are shown in Table 6.

As n grows from 4 to 6, both NAT models and noisy-MAX have space reduction increased from 14.67 to 89.96. When $n = 6$, NMAX runs 22 times faster than NPLC-Comp, as it only parameterizes a single structure. On the other hand, the KL-distance of NAT models is about 37% of noisy-MAX. The Euclidean distance of NAT models is reasonable at about 0.28. We conducted single-sided t-tests based on KL distance with H_0: NPLC-Comp has the same compression error as NMAX. It is rejected at $\alpha = 0.0005$ for all n values.

For target CPTs with PLC, we generated 100 random CPTs for each of $n = 4, 5, 6$,

TABLE 6. Experimental comparison of NPLC-Comp and NMAX on random CPTs without PLC

		NPLC-Comp		NMAX	
n		Mean	Stdev	Mean	Stdev
4	ED	0.1928	0.0378	0.2296	0.0639
	KL	0.1778	0.0743	0.3205	0.2748
	SR	14.67	6.91	14.67	6.91
	RT	7.84	6.31	0.54	0.38
5	ED	0.2353	0.0547	0.3556	0.1042
	KL	0.2940	0.1353	0.8536	0.4082
	SR	36.60	20.73	36.60	20.73
	RT	25.62	25.81	2.12	1.98
6	ED	0.2835	0.0663	0.4425	0.0601
	KL	0.4361	0.1409	1.1683	0.1354
	SR	89.96	53.32	89.96	53.32
	RT	102.81	147.78	4.74	3.77

where $k = 4$. Parameters in each CPT are non-extreme. They are compressed by PLC-Comp, NPLC-Comp and NMAX. When NPLC-Comp is applied to target CPTs with PLCs, the average KL distance in Eqn. (2) is undefined, as explained in Section 5.1. For performance comparison, we apply to NPLC-Comp the *glorified average KL distance*, where CPDs $P_T(0)$ and $P_M(0)$, corresponding to the condition $\underline{c} = (c_1^0, ..., c_n^0)$, are excluded from Eqn. (2) and the CPD count Z is reduced by one. The experimental results are shown in Table 7.

TABLE 7. Comparison of PLC-Comp, NPLC-Comp and NMAX on random CPTs with PLCs

		PLC-Comp		NPLC-Comp		NMAX	
n		Mean	Stdev	Mean	Stdev	Mean	Stdev
3	ED	0.1572	0.0428	0.1980	0.0356	0.2204	0.0407
	KL	0.1021	0.0348	0.1120	0.0374	0.1523	0.0363
	SR	6.53	2.55	6.94	2.77	6.94	2.77
	RT	11.92	7.50	3.22	2.48	0.50	0.26
4	ED	0.1802	0.0385	0.2029	0.0352	0.2402	0.0609
	KL	0.1495	0.0514	0.1687	0.0681	0.3228	0.2771
	SR	13.83	6.12	14.94	6.50	14.94	6.50
	RT	23.99	18.47	8.23	7.64	0.60	0.35
5	ED	0.2166	0.0608	0.2388	0.0540	0.3680	0.0889
	KL	0.2471	0.1165	0.3193	0.1492	0.9319	0.3587
	SR	31.94	15.46	34.38	16.48	34.38	16.48
	RT	56.63	65.62	19.19	19.80	1.37	1.07

As n grows from 3 to 5, space reduction rate grows from about 7 to about 32. NPLC-Comp and NMAX have the same space reduction, while PLC-Comp is slightly less due to encoding of the PLC. The runtime of PLC-Comp is about 3 times as that of NPLC-Comp, as each candidate NAT model is more complex (n value is large). As for compression accuracy, both PLC-Comp and NPLC-Comp are more accurate than NMAX. PLC-Comp has the lowest KL-distance from the target, even though the KL-distance used for NPLC-Comp is *glorified*. We performed single-sided t-tests for each pair of methods. For PLC-Comp vs.

NPLC-Comp, the null hypothesis is rejected for $n = 3$ and $n = 5$ at $\alpha = 0.0005$. For $n = 4$, it is accepted at $\alpha = 0.0005$, but rejected at $\alpha = 0.001$. For the pair PLC-Comp vs. NMAX and NPLC-Comp vs. NMAX, the null hypothesis is rejected at $\alpha = 0.0005$ for all n values.

In summary, compression into NAT models has superior accuracy than noisy-MAX, and explicit PLC modeling significantly further improves accuracy when PLCs exist.

6.4. Compressions of Real BN CPTs

In this experiment, we evaluate the effectiveness of NAT compression in real world CPTs. A total of 11 real world BNs are retrieved from a book website (Nagarajan et al. (2013)), where the maximum domain size of variables $k \geqslant 3$ and the maximum number of parents per node $\geqslant 3$. From these BNs, we selected 362 target CPTs (see Table 8), where the number of parents $n \geqslant 3$ and the majority of parameter values are not extreme (uncertain causes). Among these CPTs, 57 of them involve PLCs and the remaining 305 CPTs do not. The domain sizes of variables range between 2 and 63. Due to the generalization of NAT models beyond graded variables (Def. 1), we are able to conduct the compression without having to ascertain whether each variable is graded.

TABLE 8. Summary of Target CPTs from real world Bayesian networks

BN	# CPTs selected	Max # parents/node
Alarm	3	4
Barley	13	4
Hailfinder	6	4
Heaper2	12	6
Water	5	5
Sachs	1	3
Insurance	5	3
Mildew	9	3
Pathfinder	24	5
Munin	48	3
Link	236	3

The compression results for NPLC-Comp and NMAX on target CPTs without PLC are shown in Table 9. A t-test based on KL-distance rejected the null hypothesis that NPLC-Comp and NMAX have the same compression accuracy ($\alpha = 0.0005$). The commpression error of NPLC-Comp by either distance measure is comparable with that for random CPTs (larger than those for $n = 4$ and 5 in Table 6, and smaller than that for $n = 6$). We identify an extra source of error in the real world target CPTs. A NAT model CPT has non-extreme parameters, except those in $P(e|c_1^0, ..., c_n^0)$. It is also the case for random target CPTs used in Section 6.3, but not so for many real world BN CPTs in the experiment. The extreme parameters in the target CPT cause distribution of probability mass to the remaining parameters that cannot not be perfectly matched by non-extreme parameters in the NAT CPT. This is also true for the following compression.

The compression results for PLC-Comp, NPLC-Comp and NMAX on target CPTs with PLCs are shown in Table 10. Based on the Euclidean distance (the KL-distance for NPLC-Comp is glorified), compressions into NAT models are more accurate than noisy-MAX with PLC-Comp being the most accurate. The accuracy of PLC-Comp is comparable with that for random CPTs (Table 7).

In Zagorecki and Druzdzel (2013), a weighted KL-distance is used, where the KL-distance for each CPD is weighted by the probability of its configuration, before sum-

TABLE 9. Experimental comparison of NPLC-Comp and NMAX on real world CPTs without PLC

	NPLC-Comp		NMAX	
	Mean	Stdev	Mean	Stdev
ED	0.2640	0.0423	0.3117	0.0662
KL	0.3653	0.1576	0.5002	0.9260
SR	7.19	5.39	7.19	5.39
RT	78.57	324.91	1.94	6.98

TABLE 10. Comparison of PLC-Comp, NPLC-Comp and NMAX on real world CPTs with PLCs

	PLC-Com		NPLC-Com		NMAX	
	Mean	Stdev	Mean	Stdev	Mean	Stdev
ED	0.1778	0.0904	0.2056	0.0703	0.2293	0.0797
KL	0.5312	0.4546	0.5243	0.4340	0.6742	0.5243
SR	19.25	311.67	19.95	32.58	19.95	32.58
RT	402.14	1498.11	226.414	751.51	15.63	38.71

ming into the overall KL-distance for the CPT. Hence, the overall KL-distance is conditioned on the particular BN which provides the probability for each parent configuration. The weighting may suppress a large KL-distance for a CPD if its parent configuration has a close-to-zero probability. The average KL-distance used in this work (Section 4) is equivalent to weighted KL-distance with uniform weights, and offers an unbiased distance measure.

7. CONCLUSION

The main contributions of this work are the following. We extended the scope of NAT models from graded variables (ordinal) to causal variables (Def. 1) that can be either ordinal or nominal. We developed a flexible PCI pattern extraction to reduce the NAT search space while retaining good candidate NATs. We presented a constrained gradient descent for parameter search given a NAT structure. We also proposed subNAT based differentiation for PCI pattern extraction when persistent leaky causes exist. The effectiveness the framework for compressing general CPTs into NAT models, coupled with the above techniques, is validated by experimental study with both randomly generated and real world CPTs.

Since NAT-modeled BNs significantly reduce the space and time complexity during inference (Xiang and Jin (2016)), the above contributions are a step forward to significantly improving inference efficiency for BNs. They also provide guiding insight for learning tractable BNs directly from data, which we will pursue as future work. We have measured compression accuracy by comparing the resultant NAT CPT with the target CPT. This is based on the assumption that if the compressed NAT CPTs are reasonably accurate, relative to target CPTs, then inference performed on a so-compressed BN will be reasonably accurate, relative to the original BN. As a future work, the assumption will be verified by assessing accuracy of posteriors resultant from inference with the compressed BNs.

A technique closely related to NAT compression is the rank-one tensor decomposition (Savicky and Vomlel (2007)). It was shown that a CPT over m binary causes and their additive effect of domain size $m+1$ can be decomposed into the sum of $m+1$ rank-one tensors, each of which is the outer product of $m+1$ vectors. Hence, the rank-one tensor decomposition reduces the number of parameters $(m+1)2^m$ of a general CPT to

$3m^2+4m+1$. If the CPT is over m uncertain causes of the above domain sizes for variables, a NAT model requires the specification of m^2 parameters. Hence, NAT compression is more compact than rank-one tensor decomposition. For accuracy, although rank-one tensor decomposition has more parameters, which may lead to better accuracy, a NAT model can select its NAT topology from a super-exponential space. More study is needed to evaluate the relative accuracy between NAT compression and tensor decomposition.

ACKNOWLEDGEMENT

Financial support from the Discovery Grant, NSERC, Canada is acknowledged.

REFERENCES

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

GOOD, I. 1961. A causal calculus (i). British Journal of Philosophy of Science, **11**:305–318.

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

JENSEN, F.V., S.L. LAURITZEN, and K.G. OLESEN. 1990. Bayesian updating in causal probabilistic networks by local computations. *In* Computational Statistics Quarterly, (4), pp. 269–282.

MADSEN, A.L., and B. D'AMBROSIO. 2000. A factorized representation of independence of causal influence and lazy propagation. Inter. J. Uncertainty, Fuzziness and Knowledge-Based Systems, **8**(2):151–166.

MADSEN, A.L., and F.V. JENSEN. 1999. Lazy propagation: A junction tree inference algorithm based on lazy evaluation. Artificial Intelligence, **113**(1-2):203–245.

NAGARAJAN, R., M. SCUTARI, and S. LEBRE. 2013. Bayesian Networks in R with Applications in Systems Biology. Springer.

PEARL, J. 1986. Fusion, propagation, and structuring in belief networks. Artificial Intelligence, **29**(3):241–288.

SAVICKY, P., and J. VOMLEL. 2007. Exploiting tensor rank-one decomposition in probabilistic inference. Kybernetika, **43**(5):747–764.

TAKIKAWA, M., and B. D'AMBROSIO. 1999. Multiplicative factorization of noisy-max. *In* Proc. 15th Conf. Uncertainty in Artificial Intelligence, pp. 622–630.

XIANG, YANG. 2010. Acquisition and computation issues with NIN-AND tree models. *In* Proc. 5th European Workshop on Probabilistic Graphical Models. *Edited by* P. Myllymaki, T. Roos, and T. Jaakkola, Finland, pp. 281–289.

XIANG, YANG. 2012a. Bayesian network inference with NIN-AND tree models. *In* Proc. 6th European Workshop on Probabilistic Graphical Models. *Edited by* A. Cano, M. Gomez-Olmedo, and T. Nielsen, Granada, pp. 363–370.

XIANG, YANG. 2012b. Non-impeding noisy-AND tree causal models over multi-valued variables. International J. Approximate Reasoning, **53**(7):988–1002.

XIANG, YANG, and QIAN JIANG. 2016. Compression of general Bayesian net CPTs. *In* Advances in Artificial Intelligence, LNAI 9673. *Edited by* R. Khoury and C. Drummond. Springer, pp. 285–297.

XIANG, YANG, and YITING JIN. 2016. Multiplicative factorization of multi-valued NIN-AND tree models. *In* Proc. 29th Inter. Florida Artificial Intelligence Research Society Conf.. *Edited by* Z. Markov and I. Russell. AAAI Press, pp. 680–685.

XIANG, YANG, YU LI, and JINGYU ZHU. 2009. Towards effective elicitation of NIN-AND tree causal models. *In* Inter. Conf. on Scalable Uncertainty Management (SUM 2009), LNCS 5785. *Edited by* L. Godo and A. Pugliese. Springer-Verlag Berlin Heidelberg, pp. 282–296.

XIANG, YANG, and QING LIU. 2014. Compression of Bayesian networks with NIN-AND tree modeling. *In* Probabilistic Graphical Models, LNAI 8754. *Edited by* L. vander Gaag and A. Feelders. Springer, pp. 551–566.

XIANG, YANG, and MINH TRUONG. 2014. Acquisition of causal models for local distributions in Bayesian networks. IEEE Trans. Cybernetics, **44**(9):1591–1604.

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

ZHANG, N., and D. POOLE. 1996. Exploiting causal independence in bayesian network inference. J. Artificial Intelligence Research, **5**:301–328.