Learning Pseudo-Independent Models: Analytical and Experimental Results

Y. Xiang[**], J. Hu[@], N. Cercone[#] and H. Hamilton[*]

[**]University of Massachusetts
[@]Fulcrum Technologies Inc.
[#]University of Waterloo
[*]University of Regina

Abstract. Most algorithms to learn belief networks use single-link lookahead search to be efficient. It has been shown that such search procedures are problematic when applied to learning pseudo-independent (PI) models. Furthermore, some researchers have questioned whether PI models exist in practice.

We present two non-trivial PI models which derive from a social study dataset. For one of them, the learned PI model reached ultimate prediction accuracy achievable given the data only, while using slightly more inference time than the learned non-PI model. These models provide evidence that PI models are not simply mathematical constructs.

To develop efficient algorithms to learn PI models effectively we benefit from studying and understanding such models in depth. We further analyze how multiple PI submodels may interact in a larger domain model. Using this result, we show that the RML algorithm for learning PI models can learn more complex PI models than previously known.

Keywords: data mining, learning, uncertainty, belief networks.

1 Introduction

Learning belief networks [12] from data, as an alternative or enhancement to elicitation from experts, has been an active research area in recent years, e.g., [7, 10, 4, 14, 6, 5]. As the task is NP-hard [1, 3], a common search method used in heuristic learning is the single-link lookahead, where successive graphical structures adopted differ by a single link. It has been shown that a class of probabilistic models called *pseudo-independent* (PI) models cannot be learned by single-link search [17]. A more sophisticated method (multi-link lookahead) is proposed in [18] and is improved in [8] for learning decomposable Markov networks (DMNs) from data.

DMNs are less expressive than Bayesian networks (BNs). However, DMNs are the runtime representation of some algorithms for inference with BNs [11, 9, 13], and can be the intermediate results for learning BNs. For example, learning PI models needs multi-link lookahead and the search space for DAGs is much larger than that of chordal graphs. Learning DMNs first can then restrict the search for DAGs to a much smaller space, improving the efficiency.

One question often raised is whether PI models arise in practice. It has been shown [16] that parity and modulus addition are special cases of PI models,

although some consider their occurrences to be less than 'real'. In this paper, we present two PI models discovered from 'real' data. The experimental results provide evidence that PI models *do* occur in practice.

A better understanding of PI models facilitates developing algorithms that learn PI models effectively. We provide an analysis of how multiple PI submodels are embedded and interact in a model. Based on this analysis, we show that the algorithm RML [8] can learn more complex PI models than previously known.

We review the background on PI models in Section 2 and analyze the interaction of PI submodels in Section 3. We review the RML algorithm in Section 4 and then presents our new result on its learning power in Section 5. Our experimental discovery of PI models is presented in Section 6.

2 Overview of PI models

Let N be a set of discrete variables $\{X_1, \ldots, X_n\}$ in a problem domain. Each variable is associated with a finite number of possible values which we shall denote by consecutive integers $0, 1, 2, \ldots$. A *configuration* of $N' \subseteq N$ is an assignment of values to every variable in N', e.g., $(X_1 = 0, X_2 = 1, ...)$ which we denote by $(x_{1,0}, x_{2,1}, ...)$.

Let $P(X_i)$ represents the probability function for X_i and $P(x_i)$ denotes the probability value of $P(X_i = x_i)$. The joint probability distribution (jpd) is $P(N) = P(X_1, X_2, \ldots, X_N)$ and $P(x_{1,0}, x_{2,1}, \ldots, x_{n,0})$ denotes the probability of a particular tuple of N. A *probabilistic domain model* (PDM) $\mathcal{M}$ over N determines the probability of every tuple of N' for each $N' \subseteq N$.

For three disjoint subsets A, B and C, A and B are *conditionally independent* given C, denotes as $I(A, C, B)_{\mathcal{M}}$, if $P(A|B, C) = P(A|C)$ whenever $P(B, C) > 0$. When $C = \phi$, A and B are *marginally independent*. If each variable X in A is marginally independent of $A \setminus \{X\}$, then $P(A) = \prod_{X \in A} P(X)$. We shall say that variables in A are marginally independent. Variables in A are *collectively dependent* if for each proper subset $B \subset A$, there exists no proper subset $C \subset A \setminus B$ such that $P(B|A \setminus B) = P(B|C)$. Variables in A are *generally dependent* if for any proper subset B, $P(B|A \setminus B) \neq P(B)$. We introduce the concept of *marginally independent subsets* to be used in definition of PI models below.

Definition 1 (Marginally independent subsets) *Let N be a set of variables. Two disjoint nonempty subsets N_1 and N_2 of N are* `marginally independent subsets` *if for each $X \in N_1$ and $Y \in N_2$, X and Y are marginally independent.*

A domain may be partitioned into marginally independent subsets.

Definition 2 (Marginally independent partition) *Let $\mathcal{M}$ be a PDM over N. A partition $\{N_1, \ldots, N_k\}$ $(k > 1)$ of N is a* `marginally independent partition` *if every two subsets N_i and N_j $(1 \leq i, j \leq k, i \neq j)$ are marginally independent subsets.*

We refer to each N_i as an element of the partition.

Let A, B and C be disjoint subsets of nodes in an undirected graph $G = (N, E)$. C is said to *separate* A from B, denoted as $< A|C|B >_G$, if every path from A to B has a node in C. Given a PDM $\mathcal{M}$ over N and a graph $G = (N, E)$, G is an *I-map* of $\mathcal{M}$ if for all disjoint A, B, C, we have $< A|C|B >_G \Longrightarrow I(A, C, B)_{\mathcal{M}}$ [12]. A *minimal* I-map is one in which no link can be deleted such that it is still an I-map.

A *pseudo-independent* (PI) model is a PDM where proper subsets of a set of collectively dependent variables display marginal independence [18]. PI models can be classified into three types. The most restrictive type is *full* PI models.

Definition 3 (Full PI model) *A PDM over a set N ($|N| \geq 3$) of variables is a* `full` *PI model if (S1) for each $X \in N$, variables in $N \setminus \{X\}$ are marginally independent; and (S2) variables in N are collectively dependent.*

In a full PI model, every proper subset of variables are marginally independent. This is relaxed in the *partial* PI models. In a partial PI model, not every proper subset of variables are marginally independent.

Definition 4 (Partial PI model) *A PDM over a set N ($|N| \geq 3$) of variables is a* `partial` *PI model if (S1') N forms a marginally independent partition $\{N_1, \ldots, N_k\}$ ($k > 1$); and (S2) variables in N are collectively dependent.*

In a PI model, it may be the case that not all variables in the domain are collectively dependent. An embedded PI submodel displays the same dependence pattern of the previous PI models but involves only a proper subset of domain variables.

Definition 5 (Embedded PI submodel) *Let a PDM be over a set N of generally dependent variables. A proper subset $N' \subset N$ ($|N'| \geq 3$) of variables forms an* `embedded` *PI submodel if (S4) N' forms a partial PI model; and (S5) the partition $\{N_1, \ldots, N_k\}$ of N' by S1' extends into N. That is, there is a marginally independent partition $\{A_1, \ldots, A_k\}$ of N such that $N_i \subseteq A_i$, ($i = 1, .., k$).*

In general, a PI model can contain one or more PI submodels, and this *embedding* can occur recursively for any finite number of times. Since variables in a PI submodel are collectively dependent, in a minimal I-map, the variables in the submodel are completely connected. The marginal independence between subsets in the submodel is thus unrepresented. The undirected I-maps can be extended into *colored* I-maps:

Definition 6 *An undirected graph G is a* `colored I-map` *of a PDM M over N if (1) G is a minimal I-map of M, and (2) for each PI submodel m, links between each pair of nodes from distinct marginally independent subsets in m are colored. Other links are referred to as black.*

Conditional independence relations among variables can be read off a colored I-map by treating it as a normal I-map while recognizing marginal independence between variables connected by colored links.

A PI model is shown in Table 1. It contains four PI submodels over

$$N_1 = \{a, c, d\},\ N_2 = \{a, b, c\},\ N_3 = \{b, c, d\},\ N = \{a, b, c, d\}.$$

The entire domain N forms a partial PI model with the other three PI submodels embedded. Figure 1 shows the colored I-map, where colored links are dotted.

Table 1. A partial PI model with embedded PI submodels.

(d, a, b, c)	$P(.)$	(d, a, b, c)	$P(.)$	(d, a, b, c)	$P(.)$	(d, a, b, c)	$P(.)$
$(0, 0, 0, 0)$	0.02	$(0, 1, 0, 0)$	0.1	$(1, 0, 0, 0)$	0.03	$(1, 1, 0, 0)$	0.09
$(0, 0, 0, 1)$	0.02	$(0, 1, 0, 1)$	0.06	$(1, 0, 0, 1)$	0.01	$(1, 1, 0, 1)$	0.07
$(0, 0, 1, 0)$	0.06	$(0, 1, 1, 0)$	0.14	$(1, 0, 1, 0)$	0.01	$(1, 1, 1, 0)$	0.15
$(0, 0, 1, 1)$	0	$(0, 1, 1, 1)$	0.1	$(1, 0, 1, 1)$	0.05	$(1, 1, 1, 1)$	0.09

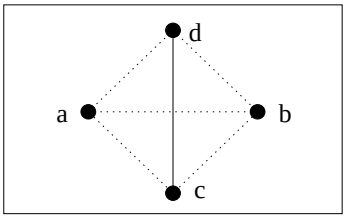

Fig. 1. Colored I-map of the model in Table 1.

3 How PI submodels interact?

A PI model may contain a number of PI submodals. How are these submodels related to each other? We address this question below. An understanding of their interaction will guide us in designing better learning algorithms and evaluating the quality of learning outcomes.

First, we refine the concept of *marginally independent partition*. Given a PDM, it may have multiple marginally independent partitions. We identify the 'finest' partition as follows:

Definition 7 (Minimum partition) *Let $\mathcal{M}$ be a PDM over N. A marginally independent partition $\{N_1, \ldots, N_k\}$ $(k \geq 1)$ of N is* `minimum` *if no N_i $(1 \leq i \leq k)$ can be partitioned into marginally independent subsets.*

For instance, in the PDM of Table 1, $\{\{a\}, \{c, d\}, \{b\}\}$ is a minimum partition. A minimum partition is unique as shown below:

Proposition 8 *For any PDM, it has a unique minimum marginally independent partition.*

Next, we define *covered* and *uncovered* colored links in a PI submodel to describe the relation between PI submodels which share variables. It turns out that this relation has a lot to do with how each submodel can be learned and in what order, as will be seen.

Definition 9 *[Uncovered colored link] Let G be a colored I-map of a PI model $\mathcal{M}$. Let l be a colored link in a PI submodel m which contains k_m colored links. The link l is* `uncovered` *in m if there exists no PI submodel s in $\mathcal{M}$ such that l is also contained (*`covered`*) in s and the number of colored links k_s of s satisfies $k_s < k_m$.*

Figure 2 (a) illustrates PI submodels with both covered and uncovered colored links. Figure 2 (b) illustrates PI submodels which share variables but all colored links are uncovered. Table 1 is a numerical example for the case of Figure 2 (a), where m is over $\{a, b, c\}$ and s is over $\{b, c, d\}$. A numerical example for the case of Figure 2 (b) is given in Table 2, where m' is over $\{a, b, c\}$ and s' is over $\{b, c, d\}$.

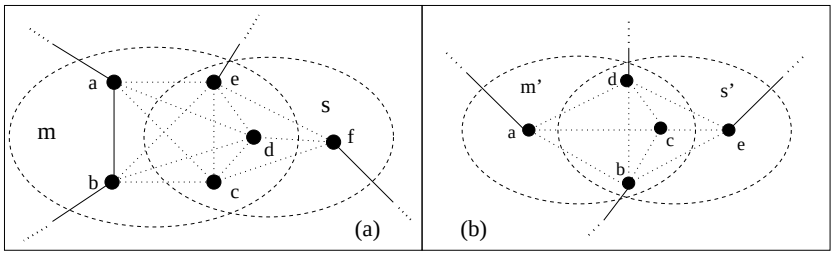

Fig. 2. (a) Two PI submodels m and s (each enclosed in an oval) share variables $\{c, d, e\}$. Assume no other PI submodels share variables with them. Submodel m has nine colored links. Six of them, $(a, c), (a, d), (a, e), (b, c), (b, d), (b, e)$, are uncovered. The remaining are covered. Submodel s has six colored links all of which are uncovered. (b) Two PI submodels m' and s' share variables $\{b, c, d\}$. Submodel m' contains six colored links and all of them are uncovered. Submodel s' has the same number of colored links and all of them are uncovered.

Table 2. A model with two PI submodels

(d, a, b, c)	$P(.)$	(d, a, b, c)	$P(.)$	(d, a, b, c)	$P(.)$	(d, a, b, c)	$P(.)$
$(0, 0, 0, 0)$	0.0024	$(0, 1, 0, 0)$	0.00336	$(1, 0, 0, 0)$	0.00336	$(1, 1, 0, 0)$	0.004704
$(0, 0, 0, 1)$	0.000064	$(0, 1, 0, 1)$	0.000448	$(1, 0, 0, 1)$	0.000448	$(1, 1, 0, 1)$	0.003136
$(0, 0, 1, 0)$	0.002304	$(0, 1, 1, 0)$	0.008064	$(1, 0, 1, 0)$	0.008064	$(1, 1, 1, 0)$	0.028224
$(0, 0, 1, 1)$	0.0024	$(0, 1, 1, 1)$	0.00336	$(1, 0, 1, 1)$	0.00336	$(1, 1, 1, 1)$	0.004704

Next, we analyze several properties of PI models related to the concepts introduced above. Lemma 10 says that the number of colored links in any PI models is lower bounded at 2.

Lemma 10 *Let $\mathcal{M}$ be a PI model and k be the number of colored links in $\mathcal{M}$. Then $k \geq 2$.*

Lemma 11 reveals the features of PI models when the lower bound in Lemma 10 is reached.

Lemma 11 *Let $\mathcal{M}$ be a partial PI model over N with exactly two colored links. Then (1) the minimum marginally independent partition Par of N has two elements and (2) $|N| = 3$.*

Lemma 12 says that the upper bound of uncovered colored links in PI submodels are lower bounded at 2.

Lemma 12 *Let $\mathcal{M}$ be a PI model and each embedded PI submodel in $\mathcal{M}$ contains no more than i uncovered colored links. Then $i \geq 2$.*

Theorem 13 reveals how PI submodels in a PI model are related to each other.

Theorem 13 *Let PDM $\mathcal{M}$ be a PI model and $\{M_1, \ldots, M_j\}$ $(j \geq 1)$ be the PI submodels in $\mathcal{M}$. Let D be a direct graph of j nodes where each node is labeled by a M_i such that M_j is a parent of M_i if M_j has colored links covered by M_i.*
Then D is acyclic.

We shall call D the *PI submodel coverage DAG* of $\mathcal{M}$.

D may be disconnected. If two submodels neither share colored links directly nor share colored links through a chain of intermediate submodels, then the two submodels will be disconnected. In a minimal colored I-map, the two submodels will be connected through black links.

Another case of disconnection is when two submodels share colored links only with each other and have the identical number of colored links. Since none can cover the links of the other, there cannot be a directed link between them in D.

Each PI submodel with all its colored links uncovered is a leaf in D.

For example, in the PDM shown in Table 1, there are four PI submodels M_1, M_2, M_3 and M_4 over N_1, N_2, N_3 and N, respectively. Its PI submodel coverage DAG has a root node M_4 with three child nodes M_1, M_2 and M_3.

4 Overview of RML algorithm

PI models cause difficulty to common algorithms that are based on a single-link lookahead search [17]. The initial attack on learning PI models [18] is based on iterations of *lookahead*(i) as shown in Algorithm 1. It is intended to learn a decomposable Markov network (DMN) from a dataset over a set of N of variables. The K-L cross entropy is used as the score metric. The algorithm consists of a sequence of calls of *lookahead*(i) with i taking the values 1, 2, ..., k for a specified k value.

Algorithm 1 *boolean lookahead(i);*
Parameter i: number of lookahead links.
Input: graph $G = (N, E)$ and threshold δh.

begin
  modified := *false*, $G' := G$;
  repeat
    initialize entropy decrement $dh' := 0$;
    for each set L of links ($|L| = i$, $L \cap E = \phi$), *do*
      if $G^* = (N, E \cup L)$ *is chordal, then*
        compute entropy decrement dh^*;
        if $dh^* > dh'$, *then* $dh' := dh^*$, $G' := G^*$;
    if $dh' > \delta h$, *then*
      $G := G'$, *done* := *false*, *modified* := *true*;
    else done := *true*;
  until done = *true*;
  return modified;
end

The *lookahead*(i) performs a multi-link lookahead search which examines i link(s) at each step. That is, alternative structures that differ from the current structure by i links are evaluated. The i links that decrease the entropy maximally are selected. If the corresponding entropy decrement is significant enough, the i links will be adopted and the search continues until no more links can be learned. We refer to this search as an *i-link-only search*.

The algorithm based on i-link-only search with incrementally larger i values can learn many PI models correctly. However, when a PI model contains recursively embedded PI submodels, the algorithm fails. For example, if the data is populated by the PI model in Table 1, then after learning the black link in the single-link search and submodels M_1 and M_3 in the double-link-only search, the algorithm will halt, missing the colored link $\{a, b\}$.

Algorithm 2 *RML*
Input: data over a set N of variables, a maximum number k of lookahead links.

begin
1 *initialize a graph* $G = (N, E = \phi)$;
2 *for* $j := 1$ *to* k, *do*
3 $i := j$;
4 *while* $i \leq j$, *do*
5 *modified* := *lookahead(i)*;
6 *if* $i > 1$ *AND modified* = *true, then* $i := 1$;
7 *else* $i := i + 1$;
8 *return G.*
end

The algorithm RML was proposed [8] to improve the performance (shown in Algorithm 2).

RML also uses *lookahead*(i). However, whenever links are learned at an i-link-only search, RML backtracks to single-link search as shown in line 6. This allows

RML to learn recursively embedded PI submodels correctly. For the PI model in Table 1, the link $\{a, b\}$ will be learned when backtracking to the single-link search after the double-link-only search. The complexity of RML was analyzed in [8].

5 Models learnable by RML

The PI models that are learnable by RML was analyzed in [8]. It was concluded that if a PI model has no submodel that contains more than k colored links, then RML with the parameter k can learn the model correctly. In the following, we refine that result. The new result shows that RML with the parameter k can actually learn PI submodels with more than k colored links. This expands the known learning power of RML.

We first introduce some background. We define two properties that are satisfied by some PDMs:

Definition 14 *[15] Let A, B, C, V and W be disjoint subset of variables.*
`Composition:` $I(A, B, C)$ & $I(A, B, W) \Longrightarrow I(A, B, C \cup W)$.
`Strong Transitivity:` $I(A, B \cup V, C)$ & $I(B, C \cup V, W) \Longrightarrow I(A, B \cup V, C \cup W)$.

We shall use the following result from [15] which characterizes the learning capacity of a *lookahead*(1)-like single-link lookahead for learning DMNs[1]. For the purpose of analysis, we assume a prefect data (no sampling error) and the threshold δh is then set to zero.

Theorem 15 *[15] Let $\mathcal{M}$ be a PDM that satisfies composition and strong transitivity. Let G be a chordal graph returned by a lookahead(1)-like single-link lookahead search. Then G is an I-map of $\mathcal{M}$.*

Theorem 16 shows the learning capacity of RML.

Theorem 16 *Let PDM $\mathcal{M}$ be a PI model over N. Let $Par = \{N_1, \ldots, N_j\}$ $(j > 1)$ be the minimum marginally independent partition of N. If variables in each N_i satisfy composition and strong transitivity, and each embedded PI submodel contains no more than k $(k \geq 2)$ uncovered colored links, then RML with the parameter k will return an I-map of $\mathcal{M}$.*

Theorem 16 shows that RML with parameter k is *not* limited to learning PI submodels with up to k colored links. Rather, it is limited to learning PI submodels with up to k uncovered color links. Hence, a PI submodel with more than k colored links is learnable by RML as long as it shares colored links with other PI submodels so that it has no more than k uncovered links. Since the time complexity of RML is exponential on k, this result implies that much more complex PI submodels (compared with the previous result [17]) can be learned correctly without increasing the computational complexity.

[1] There are some minor differences between the single-link lookahead used in [15] and *lookahead*(1), e.g., the maximum score improvement at each search step is not required there. However, the difference is irrelevant to the current result.

6 Discovery of PI models from data

Since previous reports on the study of PI models have used constructed models like those in Tables 1 and 2, the practical value of such study has been questioned by some. In the following, we provide two PI models that are discovered from real data in a preliminary experimental study.

The data we used is from the *1993 General Social Survey (GSS) on Personal Risk* conducted by *Statistics Canada* in 1993 [2]. The dataset contains 11960 cases over 469 variables. A preliminary study was performed on a few subjects. The experiment was carried out using *WEBWEAVR-III* toolkit (available for downloading at the first author's homepage) which implements the RML algorithm as one module.

Table 3. Variables in data on *Accident Prevention*

index	*Variable*	*Question*
0	$UseSeatBelt$	*Accident Protection : Use seat belt in vehicle?*
1	$WearHelmet$	*Accident Protection : Wear Helmet riding bicycle?*
2	$MedFrmKid$	*Accident Protection : Store medicines from children?*
3	$SafetyEquip$	*Accident Protection : Use safety equipment?*
4	$SmokeAlarm$	*Do you have a working smoke detector in your home?*
5	$FireExtsher$	*Do you have a working fire extinguisher at home?*
6	$FstAidSuply$	*Do you have first aid supplies at home?*
7	$FstAidTrain$	*You or household members trained in first aid?*

One PI model we discovered is on "Accident Prevention Precautions" (Table 3). The first eight variables (questions) in the data were used in the study. All of them are binary. After deleting cases with missing variables, 4303 cases were used as the learning input. Using $k = 2$, the learning program returned the DMN in Figure 3 (d).

The learning process is shown in Figure 3. The first single-link lookahead search learned a disconnected graph shown in (a). Note that three marginally independent subsets were found. In the following double-link-only search, the three PI submodels below were learned and shown in (b), (c) and (d), respectively.

$M_1 : \{SafetyEquip, FireExtsher, FstAidTrain\}$

$M_2 : \{WearHelmet, MedFrmKid, SafetyEquip\}$

$M_3 : \{SafetyEquip, FireExtsher, FstAidSuply, FstAidTrain\}$

Note that M_1 is recursively embedded in the PI submodel M_3. After the double-link-only search, backtracking occurred without learning additional links.

Another PI model we discovered is on "Harmful Effects of Personal Drinking". The data contains 8 variables (questions) described in Table 4. The first six variable are binary. The last two variables each has the domain {*NoDrinking, 1To2Drinks, EnoughToFeelTheEffects, GettingDrunkIsSometimesOk* }. After deleting cases with missing variables, 8047 cases were selected. The first 7047 cases were used as the learning input, and the other 1000 cases were hold as test set (see below).

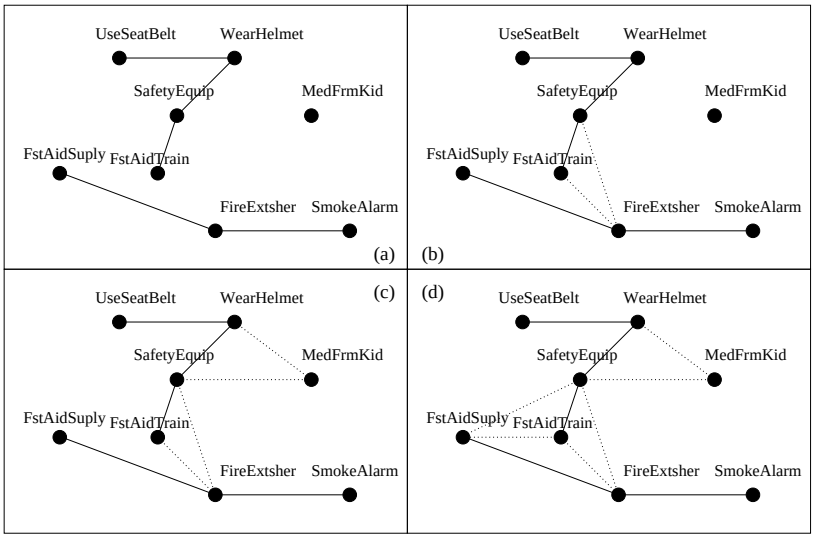

Fig. 3. The process of learning *Accident Prevention* model. Colored links are shown as dotted.

Table 4. Variables in data on *Harmful drinking*

i	$Variable$	$Question$
0	$HarmSocial$	$Did\ alcohol\ harm\ friendships/social\ life?$
1	$HarmHealth$	$Did\ alcohol\ harm\ your\ physical\ health?$
2	$HrmLifOutlk$	$Did\ alcohol\ harm\ your\ outlook\ on\ life?$
3	$HarmLifMrig$	$Did\ alcohol\ harm\ your\ life\ or\ marriage?$
4	$HarmWorkSty$	$Did\ alcohol\ harm\ your\ work,\ studies,\ etc?$
5	$HarmFinance$	$Did\ alcohol\ harm\ your\ financial\ position?$
6	$NumDrivrDrink$	$How\ many\ drinks\ should\ a\ designated\ driver\ have?$
7	$NmNonDrvrDrink$	$How\ many\ drinks\ should\ non-designated\ driver\ have?$

Using $k = 1$, the learning program returned the DMN in Figure 4 (a) with two marginally independent subsets. Using $k = 2$, a partial PI submodel $M = \{HarmHealth, HarmFinance, NmNonDrvrDrink\}$ was detected. The DMN returned is shown in (b) where colored links are shown as broken lines.

A PI model captures more dependence in a data and hence will provide better prediction when used for future decision making. On the other hand, it is also more expensive to learn and to perform inference with. A good model is one that provides sufficiently better prediction without being too much more expensive in inference. To evaluate the overall "goodness" of the learned PI model, we compared the performance of three learned models:

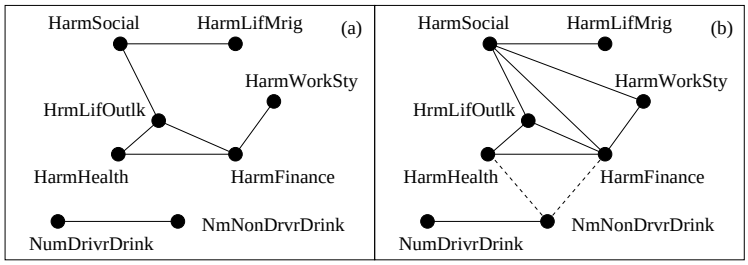

Fig. 4. DMN learned from data on *Harmful drinking*

1. The DMN in figure 4 (a) which we refer to as *Non-PI DMN*. To be able to reason with it in a normal inference engine, we added a dumb link between $HarmFiance$ and $NumDrivrDrink$ to make it connected. However, the link potential table carries no dependence.
2. The DMN in (b) which we refer to as *PI DMN*.
3. A completely connected DMN which we refer to as *jpd DMN*.

Since the Non-PI DMN is a subgraph of the PI DMN which in turn is a subgraph of the jpd DMN, we expect that they provide increasingly better prediction and inferences are increasingly more expensive. We tested the three DMNs using the other 1000 cases which the learning program did not see. For each case, we used the value of the following six variables as observations,

$$HarmSocial,\ HarmHealth,\ HrmLifOutlk,$$
$$HarmLifMrig,\ HarmWorkSty,\ HarmFinance,$$

which were all taken from one marginally independent partition. We then performed inference in each DMN to predict the value of $NmNonDrvrDrink$ in the other marginally independent partition. The results are shown in Table 5.

Table 5. Evaluation summary.

$Learnednet$	$Infer.time\ (s)$	$Hitcount$	$Avg.Euc.$	$Avg.KL$
$NonPI\ DMN$	11.82	315	0.0617	0.01842
$PI\ DMN$	16.24	347	0.0193	0.00786
$JPD\ DMN$	638.57	347	0.0	0.0

Inference for 1000 cases using the Non-PI DMN took 11.82 sec (see the second column). The PI DMN took 37% longer (16.24 sec). However, the jpd DMN took

about 60 times longer. Hence, the PI DMN and Non-PI DMN are comparable in terms of inference efficiency.

Out of 1000 cases, the Non-PI DMN predicted correctly for 315 cases (see the third column), while the jpd DMN predicted correctly for 347 cases, which is 10% better. Note that since the target variable has four possible values, a random guess is expected to hit about 250 cases. Furthermore, since the jpd DMN captured all the probabilistic dependence in the data, we can do no better than its performance given only the training data. Interestingly, the PI DMN predicted just as well as the jpd DMN, while it only used a very small fraction of the inference time of the jpd DMN. The last two columns of the table show the Euclidean and K-L (cross entropy) distances between the posterior distribution by each DMN and that by the jpd DMN, averaged over the 1000 cases. The distances by PI DMN is much smaller than those by the Non-PI DMN.

7 Conclusion

Our experimental discovery of the two PI models suggests that PI models, including recursively embedded PI models, are *not* simply mathematical constructs but are practical reality. In our performance comparison, the learned PI model reached ultimate prediction accuracy with only slight increase in inference complexity compared with the learned Non-PI model. The PI models that we presented were discovered after only a few trials from one data set. The increase in prediction power obtained from the model is far from the potential increase that can be expected according to the theory of PI models. Hinted by the theory, we believe that PI models with more impressive gain in prediction power can be found. We plan to demonstrate that with more search in the future. On the other hand, our performance comparison does show that the concept of PI models is useful in practice when one seeks to discover models with better overall performance.

Given the usefulness of learning PI models, a better understanding of the characteristics of PI models can provide valuable guidance to the design of algorithms that can learn such models effectively. Our analysis of RML algorithm is one more step in that direction. It not only expands the boundary of learnable PI models with given computation resources, but also reenforce our belief that with some controlled increase of complexity, PI models can be learned tractably.

Acknowledgements

This work is supported by Research Grant OGP0155425 from NSERC of Canada, a grant from IRIS in the Networks of Centres of Excellence Program of Canada, a grant from SaskTel and a SaskTel CRD grant from NSERC. Final draft is completed while the first author is on sabbatical from Univ. of Regina.

References

1. R.R. Bouckaert. Properties of Bayesian belief network learning algorithms. In R. Lopez de Mantaras and D. Poole, editors, *Proc. 10th Conf. on Uncertainty in Artificial Intelligence*, pages 102–109, Seattle, Washington, 1994. Morgan Kaufmann.
2. Statistics Canada. The 1993 genaral social survey - cycle 8: Personal risk, 1994.
3. D. Chickering, D. Geiger, and D. Heckerman. Learning Bayesian networks: serach methods and experimental results. In *Proc. of 5th Conf. on Artificial Intelligence and Statistics*, pages 112–128, Ft. Lauderdale, 1995. Society for AI and Statistics.
4. G.F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, (9):309–347, 1992.
5. N. Friedman, K. Murphy, and S. Russell. Learning the structure of dynamic probabilistic networks. In G.F. Cooper and S. Moral, editors, *Proc. 14th Conf. on Uncertainty in Artificial Intelligence*, pages 139–147, Madison, Wisconsin, 1998. Morgan Kaufmann.
6. D. Heckerman, D. Geiger, and D.M. Chickering. Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.
7. E.H. Herskovits and G.F. Cooper. Kutato: an entropy-driven system for construction of probabilistic expert systems from database. In *Proc. 6th Conf. on Uncertainty in Artificial Intelligence*, pages 54–62, Cambridge,, 1990.
8. J. Hu and Y. Xiang. Learning belief networks in domains with recursively embedded pseudo independent submodels. In *Proc. 13th Conf. on Uncertainty in Artificial Intelligence*, pages 258–265, Providence, 1997.
9. 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.
10. W. Lam and F. Bacchus. Learning Bayesian networks: an approach based on the MDL principle. *Computational Intelligence*, 10(3):269–293, 1994.
11. S.L. Lauritzen and D.J. Spiegelhalter. Local computation with probabilities on graphical structures and their application to expert systems. *J. Royal Statistical Society, Series B*, (50):157–244, 1988.
12. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
13. G. Shafer. *Probabilistic Expert Systems*. Society for Industrial and Applied Mathematics, Philadelphia, 1996.
14. S.K.M. Wong and Y. Xiang. Construction of a Markov network from data for probabilistic inference. In *Proc. 3rd Inter. Workshop on Rough Sets and Soft Computing*, pages 562–569, San Jose, 1994.
15. Y. Xiang. A characterization of single-link search in learning belief networks. In P. Compton H. MOtoda, R. Mizoguchi and H. Liu, editors, *Proc. Pacific Rim Knowledge Acquisition Workshop*, pages 218–233, 1998.
16. Y. Xiang. Towards understanding of pseudo-independent domains. In *Poster Proc. 10th Inter. Symposium on Methodologies for Intelligent Systems*, Oct 1997.
17. Y. Xiang, S.K.M. Wong, and N. Cercone. Critical remarks on single link search in learning belief networks. In *Proc. 12th Conf. on Uncertainty in Artificial Intelligence*, pages 564–571, Portland, 1996.

18. Y. Xiang, S.K.M. Wong, and N. Cercone. A 'microscopic' study of minimum entropy search in learning decomposable Markov networks. *Machine Learning*, 26(1):65–92, 1997.