Mach Learn (2006) 65:199–227
DOI 10.1007/s10994-006-8366-8

Learning decomposable markov networks in pseudo-independent domains with local evaluation

Y. Xiang · J. Lee

Received: 27 October 2003 / Revised: 25 February 2006 / Accepted: 2 March 2006 / Published online: 17 May 2006

Abstract We consider learning probabilistic graphical models in a problem domain of unknown dependence structure. Common learning algorithms rely on single-link lookahead search, which assumes the underlying domain is not *pseudo-independent*. Since the dependence structure of the domain is unknown, such assumption is fallible. We study learning algorithms that make no such assumption and return an approximate dependence structure no matter whether the domain is pseudo-independent or not. The focus of this paper is on learning decomposable Markov networks, which can directly be used for model-based inference or as the intermediate step for further learning of directed graphical models. We identify a small subset of domain variables, termed *crux*, in the graphical models currently being examined during search. We prove that crux is sufficient for computing the incremental change of both model description length as well as data description length given the model. Based on crux, we propose algorithms that reduce evaluation of alternative graphical models to local computation, improve efficiency significantly, and introduce no error to the selection of alternative models.

Keywords Learning graphical models · Knowledge discovery · Decomposable Markov networks · Local computation · Probabilistic reasoning

1. Introduction

We consider learning belief networks, such as Bayesian networks (BNs), in a problem domain of unknown probabilistic dependence structure. Common learning algorithms rely on single-link lookahead search (Lauritzen, 1989; Spirtes & Glymour, 1991; Cooper & Herskovits, 1992; Lam & Bacchus, 1994; Heckerman, Geiger, & Chickering, 1995; Lauritzen, 1996). These algorithms assume that the underlying domain is *non-pseudo-independent*

Editor: Shai Ben-David

Y. Xiang (✉)· J. Lee
Department of Computing and Information Science, University of Guelph, Guelph, Ontario, Canada N1G 2W1
e-mail: yxiang@cis.uoguelph.ca

(Xiang, Wong, & Cercone, 1996). Since the dependence structure of the domain is unknown, such assumption is fallible. Consequently, the dependence structures returned by such algorithms are far from the underlying domain dependence structures when the domain is actually pseudo-independent (PI) (Xiang, Wong, & Cercone, 1996). We study learning algorithms that make no such assumption. That is, the algorithm is "open-minded" about the possibility that the underlying domain may or may not be PI. We require that the algorithm returns an approximate dependence structure no matter whether the domain is PI or not.

Learning belief networks in possibly PI domains requires multi-link lookahead search (Xiang, Wong, and Cercone, 1997). To perform the search effectively, a careful organization of multi-link lookahead is necessary. The search space for directed models (such as BNs) is much larger than for undirected models, and it is even more so for multi-link lookahead. We therefore investigate more efficient learning with two steps. A decomposable Markov network (DMN) is learned first from data, and then a BN is learned constrained by the DMN (Huang & Xiang, 1999). The DMN learned in the first step not only facilitates the second step, but itself can be used directly as the runtime representation of several algorithms for probabilistic inference (Lauritzen & Spiegelhalter, 1988; Jensen, Lauritzen, & Olesen, 1990; Shafer, 1996). This work focuses on the first step: the effective learning of a DMN.

Following the principle of *minimum description length* (MDL) (Rissanen, 1978), for each potential DMN M, we define a score

$$\Gamma(M) = f(\Gamma_1(M), \Gamma_2(\Delta|M)),$$

where $\Gamma_1(M)$ represents the description length of the DMN M and $\Gamma_2(\Delta|M)$ represents the description length of the data set Δ given M. We learn a DMN in a potentially PI domain by multi-link lookahead search, where the successive DMN structures examined differ by a set of links. We present a methodology in which the score evaluation of each new alternative DMN is reduced to the computation over a small set of variables depending on only the set of newly added links, thus improving efficiency significantly. Such local score computation was studied by others, such as (Cooper & Herskovits, 1992; Lam & Bacchus, 1994; Heckerman, Geiger, & Chickering, 1995) in the context of learning BNs with single-link lookahead and (Lauritzen, 1989; Lauritzen, 1996) in the context of learning DMNs with single-link lookahead. Our study is performed in the context of learning DMNs in a possibly PI domain with multi-link lookahead search.

The remaining parts of the paper are organized as follows: Section 2 introduces the background and terminologies. In Section 3, we show why techniques used in single-link lookahead search are insufficient for multi-link lookahead search. The notion of a local structure called *crux* is defined in Section 4 with several useful properties proved. Section 5 presents analysis and algorithm for evaluating the incremental change in DMN model description length, and Section 6 presents analysis and algorithm for evaluating the incremental change in data description length. Section 7 puts this work in the context of related research. Experimental results are reported in Section 8.

2. Background

2.1. Graphs and cluster graphs

We investigate learning a BN by learning first a DMN. Graphical structures of DMNs are called chordal graphs. Chordal graphs can be alternatively represented as a particular type

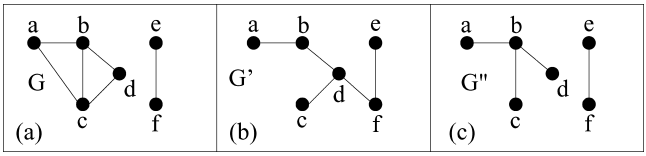

Fig. 1 Examples of graphs

of cluster graphs, called junction forests. A class of algorithms for inference with BNs, e.g., (Lauritzen & Spiegelhalter, 1988; Jensen, Lauritzen, & Olesen, 1990; Shafer, 1996), first converts a BN into a chordal graph and then a junction tree (a connected junction forest). Inference is then performed directly in the junction tree. The primary advantage of junction forest representation over the original BN is that a BN could be multiply connected (more than one path exists between at least one pair of nodes) while a junction forest is singly connected (no more than one path exists between every pair of clusters). This property of junction forests also facilitates learning DMNs, as will be seen. In the following, we introduce key concepts on chordal graphs, cluster graphs and junction forests.

Let $G = (V, E)$ be an undirected graph, where V is a set of nodes and E a set of links. Two subsets X and Y of V are *separated* by a subset Z if every path from a node in X and a node in Y contains a node in Z. A graph is a *forest* if there are no more than one path between each pair of nodes. A forest is a *tree* if it is connected. A set X of nodes is *complete* if elements of X are pairwise adjacent. A maximal set of nodes that is complete is a *clique*. A path or cycle ρ has a *chord* if there is a link between two non-adjacent nodes in ρ. G is *chordal* if every cycle of length ≤ 4 has a chord.

In Fig. 1, G' in (b) is a forest and so is G'' in (c). G' is also a tree but G'' is not. G in (a) is not a forest. Node set $\{a\}$ in (a) is separated from set $\{d, e\}$ by the node set $\{b, c\}$. The node set $\{b, c, d\}$ in (a) is complete and is a clique, but the set $\{a, b, c, d\}$ is not. The link $\langle b, c\rangle$ is a chord in the cycle $\langle a, b, d, c, a\rangle$. All three graphs are chordal. However, if the link $\langle b, c\rangle$ in G is removed, it will no longer be chordal.

A *cluster graph* is a triplet (V, Ω, Ψ). V is a set of elements and is called the *generating* set. Each node in a cluster graph is labeled by a nonempty subset of V. It is called a *cluster* to distinguish it from a node in an undirected graph. Ω is the set of clusters of the cluster graph. Every element in V is contained in at least one cluster. Each link in a cluster graph connects two clusters and is labeled by the intersection of the clusters connected. It is often called a *separator* to distinguish it from a link in an undirected graph. Ψ is the set of separators of the cluster graph. A cluster forest is a *junction forest* (JF) if (nonempty) intersection of every two clusters is contained in every cluster on the unique path between them. Let $G = (V, E)$ be a chordal graph, Ω be the set of cliques of G, and F be a JF (V, Ω, Ψ). We will call F a *corresponding* JF of G. Such a JF exists if and only if G is chordal. A chordal graph G may have multiple corresponding JFs, but all of them have the same set of separators.

Figure 2(a) shows a cluster graph F with the generating set

$$\{a, b, c, d, e, f\}$$

and 4 clusters. The separator between the clusters $\{a, b\}$ and $\{b, e, f\}$ is $\{b\}$ as labeled. It is not a forest. F' in (b) is a forest, but not a junction forest. This is because b is contained in clusters $\{a, b\}$ and $\{b, e, f\}$, but is not contained in the cluster $\{d, f\}$ that is on the path between the two clusters. F'' in (c) is a junction forest. It is not a corresponding JF for the chordal graph G in Fig. 1(a), because its clusters do not correspond to cliques in G. F^* in (d)

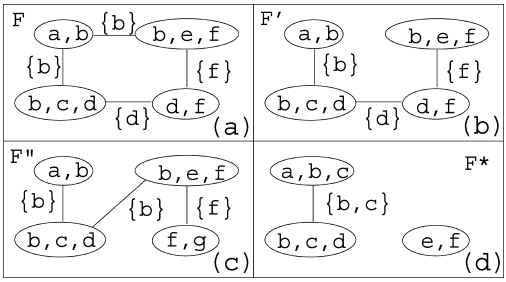

Fig. 2 Examples of cluster graphs

is such a corresponding JF. It has the set of nodes of G as its generating set and its clusters map one to one to cliques of G.

2.2. Probabilistic dependence

For three disjoint subsets A, B and C of variables, A and B are *conditionally independent* given C, if

$$P(A|B, C) = P(A|C) \text{ whenever } P(B, C) > 0.$$

When $C = \phi$, A and B are said to be *marginally independent*. If each variable $x \in A$ is marginally independent of $A\backslash\{x\}$, where $\backslash$ is the operator for set difference, then it can be shown that $P(A) = \prod_{x\in A} P(x)$. 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 $C \subset A/\ B$ such that $P(B|A\backslash B) = P(B|C)$.

Table 1 shows the joint probability distribution (jpd) over a set of binary variables $X = \{x_1, x_2, x_3, x_4\}$. For each variable x_i, we denote its domain by $\{0, 1\}$ or $\{x_{i,0}, x_{i,1}\}$. A configuration of X is denoted as $\mathbf{x} = (x_1, x_2, x_3, x_4)$. Marginal probabilities for these variables are

$$P(x_{1,0}) = 0.7,\ \ P(x_{2,0}) = 0.6,$$
$$P(x_{3,0}) = 0.35,\ \ P(x_{4,0}) = 0.45.$$

Any subset of three variables are marginally independent, e.g.,

$$P(x_{1,1}, x_{2,0}, x_{3,1}) = P(x_{1,1})P(x_{2,0})P(x_{3,1}) = 0.117.$$

The four variables are collectively dependent, e.g.,

$$P(x_{1,1}|x_{2,0}, x_{3,1}, x_{4,0}) = 0.257$$

Table 1 A joint probability distribution over a set X of 4 binary variables

x	P(x)	x	P(x)	x	P(x)	x	P(x)
(0,0,0,0)	0.0586	(0,1,0,0)	0.0517	(1,0,0,0)	0.0359	(1,1,0,0)	0.0113
(0,0,0,1)	0.0884	(0,1,0,1)	0.0463	(1,0,0,1)	0.0271	(1,1,0,1)	0.0307
(0,0,1,0)	0.1304	(0,1,1,0)	0.0743	(1,0,1,0)	0.0451	(1,1,1,0)	0.0427
(0,0,1,1)	0.1426	(0,1,1,1)	0.1077	(1,0,1,1)	0.0719	(1,1,1,1)	0.0353

Table 2 A partial PI model with embedded PI submodels where x = (d, a, b, c)

x	P(x)	x	P(x)	x	P(x)	x	P(x)
(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

and

$$P(x_{1,1}|x_{2,0}, x_{3,1}) = P(x_{1,1}|x_{2,0}, x_{4,0}) = P(x_{1,1}|x_{3,0}, x_{4,0}) = 0.3.$$

A *pseudo-independent* (PI) domain is a problem domain where proper subsets of a set of collectively dependent variables display marginal independence. The most restrictive type is *full* PI domains defined as follows. Table 1 is an example of a full PI domain.

Definition 1. A domain over a set $V(|V| \geq 3)$ of variables is a `full` PI domain if (1) for each $x \in V$, variables in $V \backslash \{x\}$ are marginally independent; and (2) variables in V are collectively dependent.

In a full PI domain, every proper subset of variables are marginally independent. Large problem domains are more likely to be *partial PI* or to contain *embedded PI subdomains*. Table 2 shows such a PI domain from Xiang et al. (2000).

In this domain, the set of variables $X = \{d, a, b, c\}$ is collectively dependent, a is marginally independent of each other variable, and so is b. However, c and d are not marginally independent. Furthermore, this partial PI domain has three embedded PI subdomains:

$$X_1 = \{a, c, d\},\ X_2 = \{a, b, c\},\ X_3 = \{b, c, d\}.$$

X_1 and X_3 are themselves partial PI subdomains and X_2 is a full PI subdomain.

2.3. Probabilistic graphical models

A decomposable Markov network (DMN) is a triplet $M = (V, G, \mathcal{P})$, where V is a set of variables in a problem domain and $G = (V, E)$ is a chordal graph. $\mathcal{P}$ is a set of probability distributions specified as follows: Let F be a corresponding JF of G. Orient links (separators) of F such that each cluster has no more than one parent cluster. For each cluster C with a parent Q and the separator S between them, associate C with $P(C \setminus S|S)$. The jpd defined by M is $P(V) = \prod_C P(C \setminus S|S)$. Note that for a given F, there are multiple ways that its links can be directed, and the resultant $P(V)$'s are equal. In practice, each $P(C \setminus S|S)$ is estimated from available data. Probabilistic conditional independence among variables in V is conveyed by node separation in G, and by separator separation in F. It has been shown (Xiang, 2002) that G and F encode exactly the same dependence relations within V. Hence, we will switch between the two graphical views from time to time. Given a DMN $M = (V, G, \mathcal{P})$ and a corresponding JF F of G, we shall sometimes call F a JF of M.

Figure 3(a) shows a chordal graph G and (b) shows its corresponding JF F. The links (separators) in F are directed such that each cluster has no more than one parent cluster.

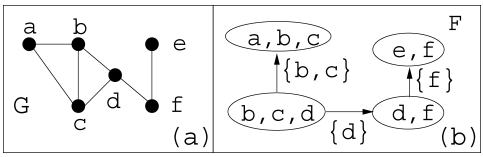

Fig. 3 (a) A chordal graph G. (b) a corresponding JF F of G

Cluster $\{b, c, d\}$ is associated with a probability distribution $P(b, c, d)$. Cluster $\{a, b, c\}$ is associated with $P(a|b, c)$, cluster $\{d, f\}$ is associated with $P(f|d)$, and cluster $\{e, f\}$ is associated with $P(e|f)$. In G, $\{a\}$ is separated from $\{d, e\}$ by $\{b, c\}$. This is also true in F since $\{b, c\}$ is a separator between the cluster containing a and clusters containing d and e. This graphical separation encodes the conditional independence $P(a|b, c, d, e) = P(a|b, c)$. These graphically encoded conditional independence relations imply that

$$P(a, b, c, d, e, f) = P(e|f)P(f|d)P(a|b, c)P(b, c, d).$$

Common algorithms for learning belief networks, e.g., (Cooper & Herskovits, 1992; Spirtes & Glymour, 1991; Lam & Bacchus, 1994), use a single-link lookahead search to discover the graphical dependence structure, where successively examined structures differ by a single link. These algorithms assume that if variables are marginally independent, they cannot be collectively dependent. When the problem domain is PI, however, the assumption is invalid and the algorithms return graphical structures that are far from the underlying domain dependence structures (Xiang, Wong, & Cercone, 1996). Subsequent decisions made using the learned model will be misleading.

Given a problem domain, there is usually no way of knowing whether it is PI prior to the learning. To discover the correct dependence structure in potentially PI domains, multi-link lookahead search is necessary (Xiang, Wong, and Cercone, 1997). A BN uses a directed structure and a DMN uses a undirected structure. Given the set of domain variables, the search space of directed structures is much larger than that of undirected structures. It is more so for multi-link lookahead. To reduce the computational complexity, we adopt a two-step strategy: We learn a DMN first and then use the DMN to constrain the learning of a BN (Huang & Xiang, 1999). In this paper, we focus on the first step: to learn a DMN from a potentially PI domain using multi-link lookahead.

We aim to discover a DMN from data. The search consists of multiple rounds of multi-link lookahead. Algorithm 1 shows the pseudocode which extends that in Hu and Xiang (1997) to take advantage of $\Gamma()$ score defined in this paper. A brief description of the idea is given here and readers are directed to reference for more details. In each round, multiple alternative DMNs are evaluated and the best DMN is selected to enter the next round. Each DMN is obtained by adding a set of links L to the current structure to produce a super-graph, where cardinality $|L| \geq 1$. In the first round, current structure is an empty (chordal) graph (with all the nodes but without links). Successive rounds are organized as single-link lookahead, followed by double-link lookahead, triple-link lookahead, and so on. After any successful search at a higher order, lower order searches are repeated starting at single-link lookahead. Each supergraph is required to be chordal and the set of all endpoints of links in L must be complete. This later requirement is imposed for an efficiency reason. Without it, a sub-structure already evaluated at a lower order multi-link lookahead may be reevaluated unnecessarily during a higher order multi-link lookahead. In all formal results in the paper, L is assumed to satisfy this condition.

Algorithm 1. *RML+*
Input: data over V and a maximum number μ of lookahead links.

initialize graph $G = (V, E = \phi)$;
for $j := 1$ to μ, do
  $i := j$;
  while $i \leq j$, do
    $G' :=$ lookahead(G, i);
    if $i > 1$ AND $G' \neq G$, then $i := 1$;
    else $i := i + 1$;
    $G := G'$;
return G.

Method lookahead(G, i)
Parameter: graph $G = (V, E)$ and number i of lookahead links.

$G' := (V, E' = E)$;
compute score s' of G';
repeat
  $G^* := G'$, $s^* := s'$;
  for each set L of links such that $|L| = i$, $L \cap E' = \phi$, and
    endpoints of L are complete, do
    if $G'' = (V, E' \cup L)$ is chordal, then
      compute score s'' of G'';
      if $s'' < s^*$, then $G^* := G''$, $s^* := s''$;
  if $s^* < s'$, then
    $G' := G^*$, $s' := s^*$, $done := false$;
  else $done := true$;
until $done = true$;
return G';

Given a current structure G and its chordal supergraph G' obtained by adding a set L of links, we refer to G' as a *chordal supergraph* of G induced by L. We denote a corresponding JF of G' by F'. Figure 4 shows the effect of adding two links to a current chordal graph (a) and the resultant chordal supergraph (b). Note that endpoints of two new links are $\{c, e, f\}$ which are contained in a single clique of the supergraph.

For each alternative DMN, a score based on description length is computed. The model with the best score is adopted for the next round of search. We defer computation of description length to later sections. In the following section, we introduce a graphical substructure for efficient computation of description length.

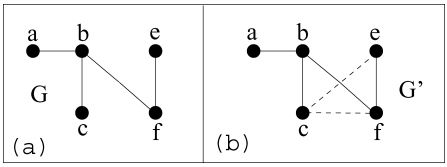

Fig. 4 Adding multiple links to current graphical structure. (a) Current chordal graph. (b) Supergraph with two links (dashed) added

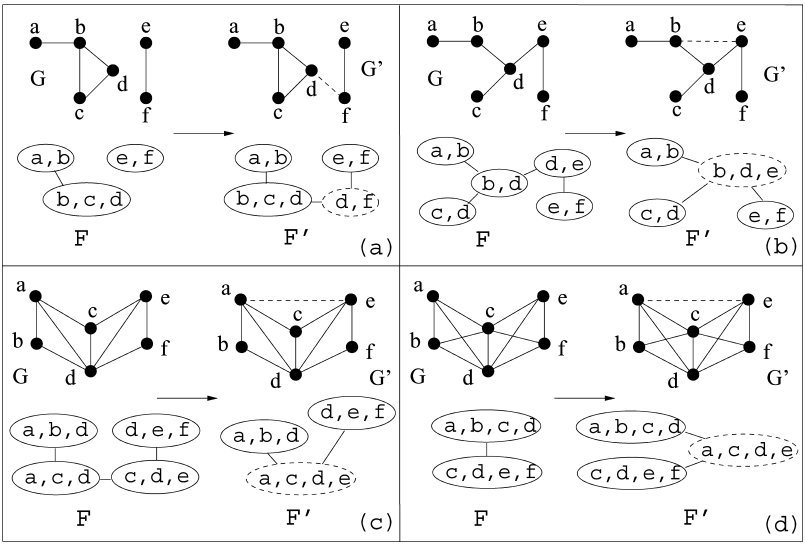

Fig. 5 Adding a single link to current graphical structure

3. Limitation of focusing method in single link lookahead

In order to compute efficiently description length for each alternative model, it is necessary to focus computation to a small subset of variables that are responsible for the change of description length. These variables are easily identified in learning of BNs, since they are variables whose parent sets have been changed. In learning of DMNs with single-link lookahead, the task is a little more involved as illustrated in Fig. 5.

In each box, the upper graphs are chordal graphs G and G', where the added link is dashed. The lower graphs in each box depict corresponding JFs, where the newly formed cluster is dashed. When a single link is added to the current graph, it may connect two components as Fig. 5(a). Change to JF is addition of cluster $\{d, f\}$ and its two separators $\{d\}$ and $\{f\}$. Change to jpd is multiplication of the factor

$$\frac{P(d, f)}{P(d)P(f)}.$$

Alternatively, a single link added to the current graph may connect two nodes in the same component as shown in Fig. 5(b), (c) and (d). In (b), change to JF is replacement of two existing clusters $\{b, d\}$ and $\{d, e\}$ as well as their separator $\{d\}$ by the new cluster $\{b, d, e\}$. Change to jpd is multiplication of the factor

$$\frac{P(d)P(b, d, e)}{P(b, d)P(d, e)}.$$

A similar change is illustrated in (c), where a larger cluster is created. Change to JF is the replacement of existing clusters $\{a, c, d\}$ and $\{c, d, e\}$ as well as their separator $\{c, d\}$ by the new cluster $\{a, c, d, e\}$. Change to jpd is multiplication of

$$\frac{P(c, d)P(a, c, d, e)}{P(a, c, d)P(c, d, e)}.$$

A different change is illustrated in (d), where no existing clusters are replaced. Instead, change to JF is the replacement of a separator $\{c, d\}$ by the new cluster $\{a, c, d, e\}$ and its separators $\{a, c, d\}$ and $\{c, d, e\}$. Change to jpd is multiplication of

$$\frac{P(c, d)P(a, c, d, e)}{P(a, c, d)P(c, d, e)}.$$

In general, let $\langle x, y\rangle$ be the single link added. Changes due to the addition can be summarized as follows:

1. Only a single cluster is created, denoted by $\{x, y\} \cup C$.
2. At most two existing clusters may be affected (e.g., (b) and (c)).
3. At most one existing separator may be affected (e.g., (b), (c) and (d)).
4. At most two new separators may be created (e.g., (a) and (d)).
5. Change to jpd is always multiplication of factor

$$\frac{P(C)P(C, x, y)}{P(C, x)P(C, y)},$$

which is true even when $\langle x, y\rangle$ connects two components, where $C = \emptyset$ and $P(\emptyset) = 1$.

Due to these regularities and especially the last one, focused calculation of change to description length can be performed relatively easily.

In learning of DMNs with multi-link lookahead, however, all regularities listed above may disappear. An example is given in Fig. 6, where three regularities disappear. In the example, three links (dashed) are added. As the result, three existing clusters $\{d\}$, $\{b, f\}$ and $\{e, f\}$ are replaced. Two new clusters $\{b, d, e\}$ and $\{b, e, f\}$ are created. Change to jpd is multiplication of

$$\frac{P(f)P(b, d, e)P(b, e, f)}{P(d)P(e, f)P(b, f)P(b, e)}.$$

Because multi-link lookahead search does not admit regularities above, focused calculation of change to description length becomes much more involved. In the remainder of the paper, we present a methodology that allows such calculation to be performed effectively.

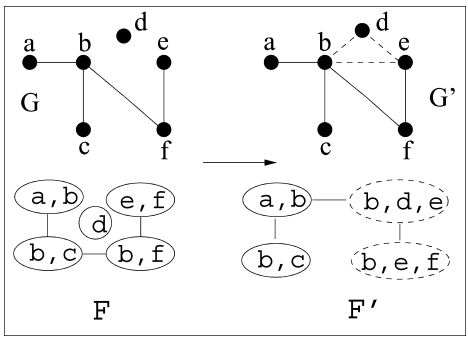

Fig. 6 Adding multiple links

4. Crux and its properties

In this section, we identify a small subset of variables in domain V called *crux* that are defined by structural change due to addition of a set L of links to a chordal graph. We establish several useful properties of crux. In the next two sections, we show that crux is a sufficient subset of variables for evaluating incremental change in description lengths Γ_1 and Γ_2 due to addition of L.

As mentioned above, we require that endpoints ED of L are complete in G'. That is, endpoints ED are contained in at least one clique in G'. It is possible, however, that multiple cliques in G' may enclose ED. The following lemma rejects such possibility. It says that the ED-enclosing clique is unique.

Lemma 2. *Let G be a chordal graph and G' be a chordal supergraph of G induced by a set L of links. Then there exists a unique clique in G' that contains the set of endpoints ED of L.*

Proof: We prove by contradiction. Let F' be the corresponding JF of G'. Suppose that two distinct cliques exist in G' that contain ED. If their corresponding clusters in F' are not adjacent, then every cluster on the path between them contains ED as well. We refer to any pair of adjacent, ED-enclosing clusters (and corresponding cliques in G') as C and Q. It follows that there exist $c \in C$ and $q \in Q$ such that $c \notin Q$ and $q \notin C$. This implies that $\langle c, q\rangle$ is not a link in G'. Otherwise, $ED \cup \{c, q\}$ will be complete and $\{c, q\}$ must be contained in both C and Q. Hence, $\langle c, q\rangle$ is not a link in G' and nor a link in G.

Let $\langle x, y\rangle$ be any link in L. Since x, y and c are all in C, they must be complete in G'. We claim that $\langle x, c\rangle$ and $\langle y, c\rangle$ are not in L, since otherwise c would be contained in ED and hence contained in Q. Therefore, $\langle x, c\rangle$ and $\langle y, c\rangle$ must be links in G. Similarly, $\langle x, q\rangle$ and $\langle y, q\rangle$ must be links in G. We have therefore found a cycle $\langle x, c, y, q, x\rangle$ in G and neither $\langle x, y\rangle$ nor $\langle c, q\rangle$ is a link in G: a chordless cycle. This contradicts that G is chordal. Hence, C and Q cannot co-exist in G' and the lemma holds. □

Based on the uniqueness result of Lemma 2, we define the notion of *crux* below:

Definition 3. Let G be a chordal graph and G' be a chordal supergraph of G induced by a set L of links. Let C be the unique clique in G' that contains endpoints of L. Let Q be any clique of G' such that Q is not a clique in G and $Q \cap C \neq \emptyset$. Denote the collection of all such cliques by Φ.

Then the union of cliques in Φ, $\bigcup_{Q\in\Phi} Q$, is called `crux` induced by G and L, and Φ is called the `generating collection` of crux.

Figure 7 illustrates crux in different cases. In each box, the upper graphs are chordal graphs G and G', where dashed link(s) indicate the set L of links added. The lower graphs in each box depict corresponding JFs, where dashed cluster(s) form the generating collection Φ. For example, in (a) and (b), the generating collection Φ contains only a single cluster which is crux itself. In (c), however, Φ consists of $\{b, c, f\}$ and $\{c, e, f\}$ while crux is $\{b, c, e, f\}$. A similar situation occurs in (d) where L connects two components of G.

Note that crux, to be denoted by R, contains C. Note also that because each pair of cliques in a chordal graph is incomparable, given R, its generating collection Φ is uniquely identifiable.

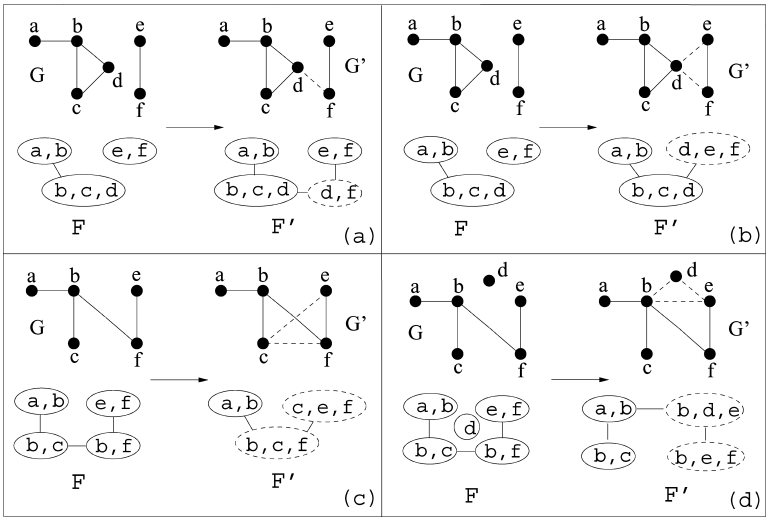

Fig. 7 Illustration of crux

The following proposition says that each clique in Φ contains endpoints of at least one link in L, and Φ is made of all such cliques. When endpoints of a link is contained in clique or cluster Q, we sometime say that the link is contained in Q.

Proposition 4. *Let G be a chordal graph and G' be a chordal supergraph of G induced by a set L of links. Let R be crux induced by G and L, and Φ be its generating collection.*

1. *For each $Q \in \Phi$, there exists a link $\langle x, y\rangle \in L$ such that $\{x, y\} \subset Q$.*
2. *For each clique Q in G', if there exists a link $\langle x, y\rangle \in L$ such that $\{x, y\} \subset Q$, then $Q \in \Phi$.*

Proof: (1) We prove by contradiction. Suppose that there exists $Q \in \Phi$ such that no $\langle x, y\rangle \in L$ is contained in Q. Then Q is not a clique newly created or enlarged by addition of L to G. Hence, Q must be a clique in G: a contradiction to $Q \in \Phi$.

(2) Let C be the unique clique in G' that contains endpoints of L. Let Q be a clique in G' such that the stated condition holds. Then $Q \cap C \neq \emptyset$ and Q is not a clique in G. By Definition 3, the statement holds. □

The following proposition shows that crux is in fact the union of all cliques newly formulated due to addition of L. In the proposition, $\setminus$ is set difference operator.

Proposition 5. *Let G be a chordal graph and G' be a chordal supergraph of G induced by a set L of links. Let Ω be the set of cliques in G, Ω' be the set of cliques in G', R be crux induced by G and L, and Φ be the generating collection of R.*

Then $\Phi = \Omega' \setminus \Omega$.

Proof: Each clique contained in Φ is in $\Omega' \setminus \Omega$ by Definition 3. We only need to show that each $Q \in \Omega' \setminus \Omega$ is also contained in Φ, i.e., $Q \cap C \neq \emptyset$, where C is the unique clique in G' that contains endpoints ED of L. Each clique Q in G' that is created or is modified from cliques of G due to adding L must contain elements of ED, and hence $Q \cap C \neq \emptyset$. □

Propositions 4 and 5 could be presented as alternative definitions of crux to Definition 3. Proposition 4 suggests the most efficient computation of crux. Based on Proposition 5, crux can be obtained by computing $\Omega' \setminus \Omega$. Denote $m = |\Omega'|$. The complexity is $O(m^2)$. On the other hand, based on Proposition 4, only one pass through Ω' is needed to find cliques with non-empty intersections with some link in L. The complexity is thus $O(m)$. The corresponding method for computing crux is outlined in Algorithm 2.

Algorithm 2. *GetCrux*
Input: A set L of links and a chordal supergraph G' induced by L.

compute the set Ω' of cliques in G';
initialize crux $R = \emptyset$;
for each $Q \in \Omega'$, do
  for each $\langle x, y\rangle \in L$, do
    if $\{x, y\} \subseteq Q$, then $R := R \cup Q$;
return R;

The following proposition says that the generating collection of crux forms a subtree in F'.

Proposition 6. *Let G be a chordal graph and G' be a chordal supergraph of G induced by a set L of links. Let R be crux induced by G and L and F' be a corresponding JF of G'. Then the generating collection Φ of R forms a connected subtree in F'.*

Proof: We prove by contradiction. Let C be the unique clique of G that contains endpoints ED of L. Suppose that members of Φ do not form a connected subtree in F'. Then there exists in F' a cluster $Q \in \Phi$ and a cluster $Z \notin \Phi$ such that Z is on the path between C and Q. This implies $Z \supset C \cap Q$. By Proposition 4, there exists $\langle x, y\rangle \in L$ such that $\{x, y\} \subset Q$. By Lemma 2, we also have $\{x, y\} \subset C$. Therefore, we have $\{x, y\} \in Z$. By Proposition 4, this implies that $Z \in \Phi$: a contradiction. Hence, cliques in Φ forms a connected subtree in F'. □

In the next two sections, we use these properties to show how to effectively compute Γ_1 and Γ_2 by means of crux. The following Proposition 7 establishes the upper bound for the cardinality of Φ, which will be used later for complexity analysis.

Proposition 7. *Let G be a chordal graph and G' be a chordal supergraph of G induced by a set L of links. Let R be crux induced by G and L, and Φ be its generating collection. Then $|\Phi| \leq |L|$.*

Proof: Based on Proposition 4 (1), it suffices to show that, for every cluster $Q \in \Phi$, there exists a link $\langle x, y\rangle \in L$ such that $\{x, y\} \subset Q$, and $Q \neq Q'$ implies $\langle x, y\rangle \neq \langle x', y'\rangle$. Note that $\langle x, y\rangle$ is an unordered pair.

We prove by contradiction. Suppose that, for distinct clusters Q and Q' in Φ, only a single link $\langle x, y\rangle$ exists in L such $\{x, y\} \subset Q$ and $\{x, y\} \subset Q'$. Then there exist $u \in Q$ and $v \in Q'$ such that $u \notin Q'$ and $v \notin Q$.

We claim $\langle u, v\rangle \notin G$ for the reason below: Since G' is chordal, a JF exists for G' that contains clusters Q and Q'. Hence, all pathes in G' from $Q \setminus Q'$ to $Q' \setminus Q$ must be through $Q \cap Q'$. As u and v are not in $Q \cap Q'$, $\langle u, v\rangle$ is neither in G' nor in G.

Furthermore, we claim $\langle x, u\rangle \in G$. If this were not the case, $\langle x, u\rangle$ would be in L. But that contradicts the assumption that $\langle x, y\rangle$ is the only link in L such $\{x, y\} \subset Q$ and $\{x, y\} \subset Q'$. Similarly, it must be the case $\langle x, v\rangle \in G$, $\langle y, u\rangle \in G$, and $\langle y, v\rangle \in G$. Since $\langle u, v\rangle \notin G$ and $\langle x, y\rangle \notin G$, we have found a chordless cycle $\langle x, u, y, v, x\rangle$ in G: a contradiction. □

5. Evaluate increment in model description length

In this section, we consider computation of $\Gamma_1(M)$, description length of a DMN M. This computation is dominated by counting the contribution from probability parameters, which is addressed by Lemma 8, Theorem 9 and Corollary 10. Lemma 8 derives such parameter contribution to $\Gamma_1(M)$ from two adjacent clusters in a DMN. Theorem 9 presents parameter contribution to $\Gamma_1(M)$ from a DMN whose dependence structure is connected. Corollary 10 extends Theorem 9 to the case of a general DMN.

To improve efficiency in computing the above parameter contribution, we compute the incremental change in contribution due to multi-link lookahead. Theoretical justification for doing so is addressed in Proposition 11, Proposition 12 and Theorem 13. Proposition 11 shows that a terminal cluster unchanged by link addition is irrelevant to incremental contribution. Proposition 12 further establishes the existence of such clusters. Theorem 13 concludes that after such clusters are removed recursively what remains is crux. Hence, crux is sufficient for computing increment of parameter contribution to model description length. How to perform such local computation is presented in Algorithm 3.

5.1. Description length of DMN

A DMN can be specified by topology of a corresponding JF, elements of each cluster in the JF, and unconstrained parameters needed to specify $\mathcal{P}$.

Consider a DMN over a domain V of n variables with a corresponding JF of m clusters. Links of a JF can be oriented such that each cluster has at most one parent cluster (see Fig. 3 for an example). Hence, topology of the JF can be described by

$$m\ log_2(m)\ bits.$$

For a cluster i with k_i elements, its membership can be specified with $k_i\ log_2(n)$ bits. Hence, memberships of all clusters can be specified with

$$\sum_{i=1}^{m} k_i\ log_2(n)\ bits.$$

Next, we consider contribution to description length from unconstrained probability parameters. We denote the space of a set X of variables by D_X. If each $x \in X$ has at most κ possible values, the cardinality of D_X is $|D_X| = \kappa^{|X|}$. The following lemma derives the contribution to description length from probability parameters in two adjacent clusters of a DMN.

Lemma 8. *Let F be a corresponding JF of a DMN, C and Q be two adjacent clusters in F with separator S. Then the total number of unconstrained parameters required to specify*

$P(C \cup Q)$ is upper-bounded by

$$|D_C| + |D_Q| - |D_S| - 1.$$

Proof: Direct links of F such that (1) each cluster has at most one parent cluster and (2) Q is a root (no parent). Due to conditional independence encoded in F (e.g., $C \setminus S$ and $Q \setminus S$ are conditionally independent given S), $P(C \cup Q)$ can be factorized as $P(Q)P(C \setminus S|S)$ and is not affected by variables in other clusters in F. The number of unconstrained parameters needed to specify $P(Q)$ is at most $|D_Q| - 1$. The number of unconstrained parameters needed to specify $P(C \setminus S|S)$ is at most

$$|D_S| * (|D_{C \setminus S}| - 1) = |D_S| * |D_{C \setminus S}| - |D_S| = |D_C| - |D_S|.$$

Hence, the number of unconstrained parameters to specify $P(C \cup Q)$ is upper-bounded by $|D_C| + |D_Q| - |D_S| - 1$. □

The following theorem describes the contribution to description length from probability parameters of a DMN when the DMN has a corresponding JT (versus JF that is not JT).

Theorem 9. *Let T be a corresponding JT of a DMN over variables V and have m clusters. Then the total number of unconstrained parameters needed to specify $P(V)$ is upper-bounded by*

$$N = \sum_{i=1}^{m} |D_{C_i}| - \sum_{l=1}^{m-1} |D_{S_l}| - 1,$$

where C_i is the ith cluster and S_l is the lth separator.

Proof: We prove by induction on the number m of clusters in T. The statement is trivially true when $m = 1$, and is true when $m = 2$ due to Lemma 8. Assume that it is true when $m = k$ and consider $m = k + 1$.

Direct links of T such that each cluster has at most one parent cluster. Let C_m be a cluster without child and S_{m-1} be its separator. Let T' be a JT obtained by removing C_m from T and $V' = (V \setminus C_m) \cup S_{m-1}$ be the domain of T'. Since T' has k clusters, by inductive assumption, the number of unconstrained parameters to specify $P(V')$ is upper-bounded by

$$N' = \sum_{i=1}^{m-1} |D_{C_i}| - \sum_{l=1}^{m-2} |D_{S_l}| - 1.$$

Since $V' \setminus S_{m-1}$ and $C_m \setminus S_{m-1}$ are conditionally independent given S_{m-1}, $P(V)$ can be factorized as $P(V) = P(V')P(C_m \setminus S_{m-1}|S_{m-1})$. Since the number of unconstrained parameters to specify

$$P(C_m \setminus S_{m-1}|S_{m-1})$$

is upper-bounded by $|D_{C_m}| - |D_{S_{m-1}}|$, the result follows. □

The following corollary extends Theorem 9 to a DMN with a general corresponding JF.

Corollary 10. *Let F be a corresponding JF of a DMN over V, where F consists of j JTs and a total of m clusters. Then the total number of unconstrained parameters needed to specify $P(V)$ is upper-bounded by*

$$N = \sum_{i=1}^{m} |D_{C_i}| - \sum_{l=1}^{m-j} |D_{S_l}| - j, \tag{1}$$

where C_i is the ith cluster and S_l is the lth separator.

Summarizing contributions from JF topology, cluster membership and probability parameters, description length of a DMN M in bits can be computed as

$$\Gamma_1(M) = m\ log_2(m) + \sum_{h=1}^{m} k_h\ log_2(n) + \left(\sum_{i=1}^{m} |D_{C_i}| - \sum_{l=1}^{m-j} |D_{S_l}| - j \right) t, \tag{2}$$

where m is the number of clusters in a corresponding JF F of M, k_h is the cardinality of the hth cluster, j is the number of JTs in F, and t is the number of bits used to store each probability parameter.

Lauritzen (1989) (page 32) contains two rules that bear some resemblance to the results in Lemma 8, Theorem 9 and Corollary 10. They should not be confused. The two rules are concerned with the degrees of freedom in an asymptotic χ^2-distribution defined over a chordal graphical model and are recursive in form. Our results deal with the number of unconstrained probability values of a DMN and Corollary 10 provides direct parameter counting. We apply a different language of analysis (through junction forest representation) which is arguably more intuitive. Most importantly, the issue of increment in the number of unconstrained probability values in a DMN due to multi-link lookahead is never addressed in Lauritzen (1989), which is our focus to be presented below.

5.2. Computation of Γ_1 increment

Complexity of computing $\Gamma_1(M)$ given a DMN M is $O(m\ k)$, where m is the number of clusters in a corresponding JF of M, and $k = \max_i(|C_i|)$ with C_i being the i'th cluster. The factor k accounts for computation to calculate D_{C_i} and D_{S_l} from cardinalities of variable domains. Because $\Gamma_1(M)$ is to be evaluated for each potential DMN during a multi-link lookahead, computation can be expensive. For example, in a domain of 100 variables, there are 4,950 possible links. In a single round of triple-link lookahead, there are more than 20,000,000,000 alternative potential DMNs to be evaluated. This number assumes a brute force search and more efficient search can be devised. Nevertheless, it serves to illustrate that reduction of computational complexity in evaluating $\Gamma_1(M)$ for each potential DMN can have an impact on overall search efficiency.

We investigate local computation of incremental change of Γ_1 value after adding a set L of links to the current DMN. Since complexity of computing $\Gamma_1(M)$ is dominated by computation of parameter contribution, we want to find a small subset of variables sufficient to compute increment of parameter contribution due to link addition. We show below that crux is such a subset.

Proposition 11 shows that a terminal cluster (with no more than one adjacent cluster) unchanged by the addition of L is irrelevant to computing increment of parameter contribution. Proposition 12 goes further to establish the existence of such cluster whenever new corresponding JF shares some clusters with previous JF. Theorem 13 concludes that after such clusters are removed recursively what remains is crux which is sufficient for computing increment of parameter contribution to model description length.

Proposition 11. *Let G be the structure of a DMN M over V and G' be the structure of another DMN M' that is a chordal supergraph of G induced by a set L of links. Let N and N' be the maximum number of unconstrained parameters needed to specify $P(V)$ for M and $P'(V)$ for M', respectively. Let F and F' be corresponding JFs of G and G', respectively. Let Q be a cluster shared by F and F', and is terminal in F'. Let S be the separator (if exists) of Q in F'.*

Then $\delta n = N' - N$ can be computed using reduced DMNs where variables in $Q \setminus S$ are removed.

Proof: Denote $V^* = V \setminus (Q \setminus S)$. Let M'_1 be the DMN obtained from M' by removing variables in $Q \setminus S$ from V and let N'_1 (and N_1) be defined accordingly.

Q may or may not be the only cluster in a JT of F', and may or may not be the only cluster in a JT of F. This yields four combinations. It is impossible that Q is the only cluster in a JT of F' but not so in F. Hence, we only need to consider the other three cases.

If Q is both the only cluster in a JT of F' and the only cluster in a JT of F, then

$$N'_1 = N' - (|D_Q| - 1),$$

and

$$N_1 = N - (|D_Q| - 1).$$

Hence, we have $\delta n = N' - N = N'_1 - N_1$.

Next, consider the case where Q is neither the only cluster in a JT of F' nor in a JT of F, and it is terminal in F' with the separator S. If Q is also terminal in F, then its only separator must be identical to S. This is because Q is shared by F and F'. It cannot contain any link in L and hence its separator in F cannot differ from that in F'. We therefore has

$$N'_1 = N' - (|D_Q| - |D_S|)$$

and

$$N_1 = N - (|D_Q| - |D_S|).$$

If Q is not terminal in F, it must have all its separators identical to S due to the above argument. Let Q' be an adjacent cluster of Q in F. If we connect each other adjacent cluster of Q directly to Q', the new cluster graph is still a corresponding JF of M and Q becomes terminal. Hence, no matter whether Q is terminal in F, we have $\delta n = N' - N = N'_1 - N_1$.

Finally, consider the case where Q is not the only cluster in a JT of F' but is so in a JT of F. The contribution of Q to N is $|D_Q| - 1$. Since $P(Q) = P(Q \setminus S|S)P(S)$, we can express the contribution of $Q \setminus S$ as $|D_Q| - |D_S|$ and that of S as $|D_S| - 1$. After removal of $Q \setminus S$

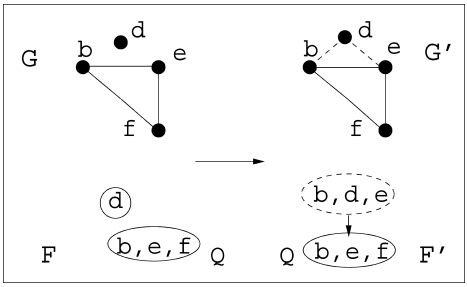

Fig. 8 Computing increment of model description length

from V, the contribution of S to N_1 is still $|D_S| - 1$, but the contribution of $Q \setminus S$ to N, i.e., $|D_Q| - |D_S|$, is no longer present. Hence,

$$N_1 = N - (|D_Q| - |D_S|).$$

The relation between N_1' and N' can be analyzed as follows: In F', we direct the links such that Q is a child cluster. Note that Q now cannot have its own child cluster since it is terminal by assumption. Similar to the argument in Lemma 8, its contribution to N' is $|D_Q| - |D_S|$. This contribution disappears after variables in $Q \setminus S$ are removed, which is equivalent to removing the cluster Q from F'. Therefore, we have

$$N_1' = N' - (|D_Q| - |D_S|).$$

This implies $\delta n = N' - N = N_1' - N_1$. This case can be illustrated using Fig. 8, where $L = \{\{b, d\}, \{d, e\}\}$, $Q = \{b, e, f\}$, $S = \{b, e\}$. Suppose that each variable is ternary. We have

$$N = 28,\ N_1 = 10, N' = 44,\ N_1' = 26.$$

From the above analysis, we conclude that in general $\delta n = N' - N = N_1' - N_1$. □

Proposition 11 promises that if a cluster Q shared by F and F' can be found that is terminal in F', then computation of δn can be performed with the reduced domain. However, it remains open whether such a cluster exists in F'. The following proposition shows that if two corresponding JFs F and F' share some clusters, then there exists at least one such terminal cluster in F'.

Proposition 12. *Let G be a chordal graph and G' be a chordal supergraph of G induced by a set L of links. Let F and F' be corresponding JFs of G and G', respectively. If F' shares clusters with F, then at least one of them is terminal in F'.*

Proof: We prove by contradiction.

Let R be crux induced by G and L, and Φ be the generating collection of R. Suppose that the proposition does not hold. That is, no terminal cluster in F' is shared by F. By Proposition 5, Φ contains all the clusters in F' that are not shared with F. Hence, all terminal clusters

in F' are in Φ. Furthermore, all clusters shared by F and F' are internal in F'. This implies that clusters in Φ do not form a connected subtree in F': a contradiction to Proposition 6. □

Finally, the following theorem establishes that crux is sufficient for computing increment of parameter contribution to model description length.

Theorem 13. *Let G be the structure of a DMN M over V and G' be the structure of another DMN M' that is a chordal supergraph of G induced by a set L of links. Let N and N' be the maximum number of unconstrained parameters needed to specify $P(V)$ for M and $P'(V)$ for M', respectively.*

Then crux R induced by G and L is a sufficient subset of V needed to compute $\delta n = N' - N$.

Proof: Let F be a corresponding JF of G, and F' be that of G'. If F and F' share clusters, by Proposition 12, a terminal cluster Q shared by F and F' can be found. Denote the separator of Q in F' by S and $V^* = V \setminus (Q \setminus S)$. By Proposition 11, δn can be computed using DMNs reduced from M and M' by removing variables in $Q \setminus S$ from V. By recursively applying Propositions 12 and 11, eventually we can remove all clusters shared by F and F'. The remaining clusters form the generating collection Φ of R by Proposition 5. Hence, δn can be computed using reduced DMNs over R. □

5.3. Method and example

Theorem 13 suggests the following local computation to obtain the incremental change of parameter contribution to the model description length: First compute crux R as presented in Section 4. Then compute the subgraphs of G and G' spanned by R. Convert the subgraphs into junction forest representations and compute δn using Corollary 10. The pseudocode is presented in Algorithm 3:

Algorithm 3. *GetModelDLIncrement*
Input: Chordal graphs G and G' and crux R.

compute subgraph SG of G spanned by R;
convert SG into JF SF;
compute $SMDL$ from SF by Eq (2);
repeat the above to compute $SMDL'$ from G' and R;
return $SMDL' - SMDL$;

As an example, consider the case in Fig. 7(d), where a triple-link lookahead has added three links to G. Following Section 4, crux can be computed as $R = \{b, d, e, f\}$. Subgraphs of G and G' spanned by R are shown in Fig. 9 as G_1 and G'_1. Corresponding junction forests are shown as F_1 and F'_1. Suppose that all variables have three possible values. Using F and F' in Fig. 7(d) and Corollary 10, we obtain N from F to be 28, and N' from F' to be 56. Hence, $\delta n = 28$. Using F_1 and F'_1 in Fig. 9 according to Corollary 10, we obtain N_1 from F_1 to be 16, and N'_1 from F'_1 to be 44. The identical value for δn is obtained. Computing the increment of model description length due to topology (the first term in Eq. (2)) and cluster membership (the second term in Eq. (2)) is straightforward and we omit the details from the example.

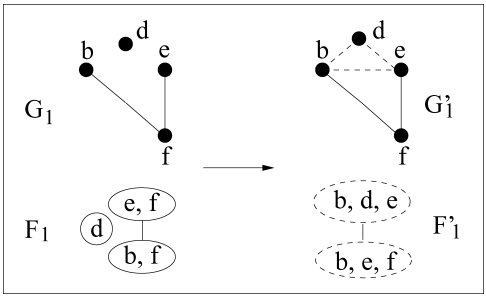

Fig. 9 Computing increment of model description length

Recall from Section 5.2, complexity of computing $\Gamma_1(M)$ without using crux is $O(m\ k)$, where m is the number of clusters in a corresponding JF, $k = \max_i(|C_i|)$, and C_i is the i'th cluster. We now consider the complexity when crux is used. From Section 4, complexity of computing crux is $O(m)$. The complexity of computing δn is $O(q\ k)$, where q is the maximum number of clusters in the generating collection of crux. By Proposition 7, q is upper bounded by μ, the maximum number of lookahead links. Hence, complexity of computing increment of parameter contribution to $\Gamma_1(M)$ is $O(m + \mu\ k)$.

In a large problem domain where the dependence structure is sparse, m is much larger than $\mu\ k$. Hence, $O(m + \mu\ k) \approx O(m)$, which is an improvement over $O(m\ k)$, the complexity without using crux. For example, in a domain of 200 variables, a JF may have $m = 140$ clusters with an average cluster size 5. Suppose four-link lookahead is used. Hence, $m + \mu\ k = 160$ and $m\ k = 700$. Computation using crux will be about 4.4 times faster.

6. Evaluating increment in data description length

In this section, we consider computation of $\Gamma_2(\Delta|M)$, the description length of a data set Δ given a DMN M over V. We establish the relation between $\Gamma_2(\Delta|M)$ and the entropy of M. Proposition 14 shows that the entropy of a DMN can be obtained from entropies over its clusters and separators. Proposition 15 reveals the relation between computable DMN entropy and not directly computable $\Gamma_2(\Delta|M)$, from which Corollary 16 derives data description length in terms of DMN cluster entropies.

To improve efficiency in computing data description length, we explore incremental computation during multi-link lookahead. Proposition 17 says that, after link addition, if a cluster shared by the two corresponding JFs is terminal in the new JF, then its boundary is complete and identical in both chordal graphs. Proposition 18 shows that the increment of Γ_2 can be correctly computed without involving variables in such a shared terminal cluster. Theorem 19 establishes that crux is sufficient for computation of increment of data description length. How to perform such local computation is presented in Algorithm 4.

6.1. Description length of data

A data set Δ consists of K cases, each of which is specified as a configuration of V. The cases are assumed to be independently collected and distributed according to an unknown probability distribution $P^*(V)$. Similar to the approach taken by Lam and Bacchus (1994) in learning of BNs, we describe each case v using a Huffman code (Huffman, 1952). Given a DMN M and its associated jpd $P(V)$, we encode each case v with a code length approximately $log_2(1/P(v))$. A case v occurs, according to $P^*(V)$, approximately $K\ P^*(v)$ times in the

data set. Hence, the description length of the data set given M is

$$\Gamma_2(\Delta|M) = \sum_v K\ P^*(v)\ log_2(1/P(v)).$$

Since $P^*(V)$ is unknown, we need to convert the above into a directly computable form. First, we introduce two existing results:

The entropy over a set X of variables determined by a probability distribution $P(X)$ is

$$H(X) = \sum_x P(x)\ log_2(1/P(x)),$$

where x is a configuration of X. The following proposition, reformatted from Wong and Xiang (1994), says that the entropy of a DMN can be obtained from entropies over its clusters and separators.

Proposition 14 (Wong & Xiang, 1994). *Let $P^*(V)$ be a probability distribution. Let M be a DMN whose associated jpd is $P(V) = \prod_i P^*(C_i \setminus S_i|S_i)$, where C_i is the ith cluster in a corresponding JF F of M and S_i is the separator between C_i and its parent cluster. Then the entropy $H(V)$ determined by $P(V)$ is*

$$H(V) = \sum_i H(C_i) - \sum_j H(S_j),$$

where S_j is the jth separator in F, $H(C_i)$ is the entropy determined by $P(C_i)$ and $H(S_j)$ is the entropy determined by $P(S_j)$.

Note that the set $\{P(C_i), P(S_j)\}$ can be derived from the set $\{P^*(C_i \setminus S_i|S_i)\}$ and vice versa. $P^*(V)$ in Proposition 14 represents a unknown distribution and M represents a DMN learned from a data set Δ that is generated according to $P^*(V)$. Although it is difficult to estimate $P^*(V)$ when the cardinality $|V|$ is large due to limited data (of K cases) and the exponential space (in the order of $\kappa^{|V|}$), we assume that $P^*(C_i)$ can be estimated accurately if $|C_i|$ is small. Hence, $H(V)$ is computable in practice. Note that if the structure of M is also correct relative to the dependence relations in $P^*(V)$, then $H(V) = H^*(V)$, where $H^*(V)$ is the entropy determined by $P^*(V)$.

The following proposition, reformatted from Xiang, Wong, and Cercone (1997), establishes a relation between the computable $H(V)$ and not directly computable $\Gamma_2(\Delta|M)$.

Proposition 15 (Xiang, Wong & Cercone, 1997). *Let $P^*(V)$ be a probability distribution. Let M be a DMN whose associated jpd is $P(V) = \prod_i P^*(C_i \setminus S_i|S_i)$, where C_i is the ith cluster in a corresponding JF F of M and S_i is the separator between C_i and its parent cluster. Then*

$$\sum_v P^*(v)\ log_2(1/P(v)) = H(V),$$

where $H(V)$ is the entropy determined by $P(V)$.

From Proposition 15, we obtain the following Corollary on data description length:

Corollary 16. *Let $P^*(V)$ be a probability distribution and Δ be a data set of K cases generated from $P^*(V)$. Let M be a DMN whose associated jpd is $P(V) = \prod_i P^*(C_i \setminus S_i|S_i)$, where C_i is the ith cluster in a corresponding JF F of M and S_i is the separator between C_i and its parent cluster. Then the data description length given M is*

$$\Gamma_2(\Delta|M) = K\ H(V) = K\left(\sum_i H(C_i) - \sum_j H(S_j)\right).$$

6.2. Computation of Γ_2 increment

Given a DMN M and a corresponding JF F of M, the complexity of computing $\Gamma_2(\Delta|M)$ is

$$O(m\ \kappa^k),$$

where m is the number of clusters in F, κ is the largest number of possible values of a variable, and $k = \max_i(|C_i|)$. We present below how to use crux to reduce the complexity significantly in large problem domains. To this end, we show that crux is sufficient for computing the increment of $H(V)$ after a set L of links is added to the current DMN. The following proposition prepares for this result. It says that if a cluster shared by F and F' is terminal in F', then its boundary is complete and identical in both chordal graphs, G and G'.

Proposition 17. *Let G be a chordal graph and G' be a chordal supergraph of G induced by a set L of links. Let F and F' be corresponding JFs of G and G', respectively. Let Q be a cluster shared by F and F', and is terminal in F' with the separator S. Then S is complete in G and separates $Q \setminus S$ and $V \setminus Q$.*

Proof: Since Q is terminal in F' and S is its separator, S is complete in G' and separates $Q \setminus S$ and $V \setminus Q$ in G'. Since Q is shared by F and F', it does not contain any $\{x, y\} \in L$ by Propositions 4 and 5. Therefore, $Q \setminus S$ must be connected to $V \setminus Q$ in G in the same way as in G'. Hence, S must be complete and separate $Q \setminus S$ and $V \setminus Q$ in G. □

The following proposition shows that the increment of Γ_2 can be correctly computed without involving variables in a shared terminal cluster. Let G be the structure of a DMN M over V and F be a corresponding JF of M. Let Q be a terminal cluster in F with the separator S. If we remove $Q \setminus S$ from G, the resultant graph is still chordal and is a valid structure of a DMN. We call the resultant DMN a *reduced* DMN.

Proposition 18. *Let G be the structure of a DMN M over V and G' be the structure of another DMN M' that is a chordal supergraph of G induced by a set L of links. Let F and F' be corresponding JFs of M and M', respectively. Let $H(V)$ and $H'(V)$ be the entropies determined by M and M', respectively. Let Q be a cluster shared by F and F', and is terminal in F' with the separator S. Then $\delta h = H'(V) - H(V)$ can be computed using the pair of reduced DMNs where variables in $V \setminus (Q \setminus S)$ are removed.*

Proof: Denote $V^* = V \setminus (Q \setminus S)$. Since S is the separator of Q in F', according to M', $V \setminus Q$ is conditionally independent of $Q \setminus S$ given S. By Proposition 14, we have

$$H'(V) = H'(V^*) + H(Q) - H(S)\,,$$

where $H'(V^*)$ is the entropy determined by the reduced DMN obtained through removing variables $Q \setminus S$ from M'.

By Proposition 17, S is complete and separates $Q \setminus S$ and $V \setminus S$ in G. Hence, either Q is terminal in F with the separator S, or F can be modified such that Q is terminal with the separator S. For instance, in Fig. 7(c), $Q = \{b, c\}$ is not a terminal cluster of F. Since $S = \{b\}$ is complete and separates $Q \setminus S$ from $V \setminus Q$ in G, F can be modified such that the cluster $\{a, b\}$ is connected directly to the cluster $\{b, f\}$ and hence Q is adjacent only to $\{b, f\}$. By Proposition 14, we have

$$H(V) = H(V^*) + H(Q) - H(S)\,,$$

where $H(V^*)$ is the entropy determined by the reduced DMN obtained by removing variables $Q \setminus S$ from M. Hence, $\delta h = H'(V^*) - H(V^*)$. □

Lauritzen (1989) (Proposition 4) bears some relation with the above Proposition 18 and our result may be confused with the result in Lauritzen (1989). Result in Lauritzen (1989) concerns the maximum likelihood estimate of a marginal distribution over a single subgraph model with a complete boundary. Our result deals with the increment of entropy between two DMNs which differ by multiple links. Our aim is to derive a local computation scheme (below) for learning PI domains using multi-link lookahead search, an issue that was never considered in Lauritzen (1989).

By recursively applying Proposition 18, the following theorem establishes local computation for the increment of data description length.

Theorem 19. *Let G be the structure of a DMN M over V and G' be the structure of a DMN M' that is a chordal supergraph of G induced by a set L of links. Let $H(V)$ and $H'(V)$ be the entropies of M and M', respectively. Then crux R induced by G and L is a sufficient subset of V needed to compute $\delta h = H'(V) - H(V)$.*

Proof: Let F be a corresponding JF of G, and F' be that of G'. If F and F' share clusters, by Proposition 12, a terminal cluster Q of F' shared with F can be found. Denote the separator of Q in F' by S and $V^* = V \setminus (Q \setminus S)$. By Proposition 18, δh can be computed as $H'(V^*) - H(V^*)$. By recursively applying Propositions 12 and 18, eventually we can remove all clusters shared by F and F'. The remaining clusters is the generating collection Φ of R by Proposition 5, and hence δh can be computed as $\delta h = H'(R) - H(R)$. □

Theorem 19 suggests the following local computation of δh: First, compute crux R as presented in Section 4. Next, compute the subgraphs of G and G' spanned by R. Convert the subgraphs into junction forest representations and compute δh according to Theorem 19. The pseudocode is presented in Algorithm 4.

Algorithm 4. *GetDataDLIncrement*
Input: Data set Δ, chordal graphs G and G', and crux R.

compute subgraph SG of G spanned by R;
convert SG into JF SF;
for each cluster of SF, compute cluster potential from Δ*;*
compute $H(R)$ *from SF;*
repeat the above to compute $H'(R)$ *from* G' *and R;*
return $H'(R) - H(R)$*;*

Recall that the complexity of computing $\Gamma_2(\Delta|M)$ is $O(m\,\kappa^k)$, where m is the number of clusters in a corresponding JF of M, κ is the largest number of possible values of a variable, and k is the cardinality of the largest cluster. By using crux, this complexity is reduced to $O(\mu\,\kappa^{k'})$, where μ is the maximum number of lookahead links that upper bounds the number of clusters in the generating collection of crux and k' is the cardinality of the largest cluster in the collection. In a large domain, m/μ is a large ratio. Furthermore, the largest cluster in the generating collection of crux may be much smaller than the largest cluster in the JF over the whole domain, i.e., k/k' is much larger than 1. Hence, the reduction of computational complexity is in the order

$$\frac{m}{\mu}\,\kappa^{k-k'}$$

and a significant computational saving can be obtained by the local computation.

7. Related work

Chow and Liu (1968) pioneered learning of tree-structured probabilistic networks. Rebane and Pearl (1987), Malvestuto (1991), and Srebro (2003) extended their method. Cooper & Herskovits (1992) developed K2 algorithm to learn a multiply connected BN. A similar algorithm was independently developed by Buntine (1991). Heckerman, Geiger, & Chickering (1995) applied Bayesian method to learning a BN by combining prior knowledge and data. Spirtes & Glymour (1991) developed PC algorithm that learns a BN by deleting links from a complete graph. Lam & Bacchus (1994) applied a minimal description length (MDL) method to learning a BN. Dawid and Lauritzen (1993) studied 'hyper Markov laws' in learning numerical parameters of a DMN with a given decomposable graph. Madigan and Raftery (1994) proposed algorithms for learning a set of acceptable models expressed as BNs or DMNs. A more extensive review of literature for learning probabilistic networks can be found in Buntine (1994), Heckerman (1995) and Dietterich (1997). Details of some learning algorithms are presented in Castillo, Castillo and Gutierrez (1997) and a recent coverage of relevant learning algorithms is available in Neapolitan (2004).

Investigations on chordal graphical models were pioneered by researchers such as (Lauritzen, 1989; Lauritzen, 1996; Whittaker, 1990; Edwards, 1995). However, they did not address learning DMNs in potentially PI domains. The class of PI domains was identified by Xiang, Wong, and Cercone (1997) and their practical relevance was demonstrated in Xiang et al. (2000). The difficulties of learning PI domains by common belief network learning algorithms were analyzed in Xiang, Wong, & Cercone (1996). Frydenberg and Lauritzen (1989) showed that, given two chordal graphs with one being the subgraph of the other, there is an increasing sequence of chordal graphs between them that differ by exactly one link. As shown in Xiang, Wong, & Cercone (1996), DMNs differing by one link are insufficient for learning graphical models in PI domains. Search through DMNs differing by multiple links raise

many new issues. In order to discover graphical models in potentially PI domains effectively, it is crucial that these issues are resolved. The work reported in this paper addresses one of these issues: local computation for effective evaluation of DMNs during multi-link lookahead search.

Researchers on chordal graphical models such as (Lauritzen, 1989; Lauritzen, 1996; Whittaker, 1990; Edwards, 1995) have performed their analysis directly on chordal graphs. Alternatively, in this paper and the related work (Xiang, Wong, and Cercone, 1997), junction forest models have been used. The alternative approach is just as expressive as it has been shown (Xiang, 1998; Xiang, 2002) that junction forest models and chordal graph models are equivalent I-maps (see Pearl, 1988 for definition). The alternative approach has a representational advantage: Because a chordal graph model is usually multiply connected while a corresponding junction forest is singly connected, the junction forest model is simpler and more intuitive to analyze and process.

8. Experimental demonstration of PI model learning

In this section, we demonstrate both improvement in learning efficiency using the proposed methodology and the relevance of learning in PI domains.

For efficiency, we simulated two PI domains $Mid20v$ and $Bigsmas17v$. The two domains contain 20 and 17 discrete variables, respectively. Each variable has between 2 to 9 possible values. $Mid20v$ has one PI subdomain embedded in it and $Bigsma17v$ has two. Up to triple-link lookahead search is performed during learning. The learned models are shown in Figs. 10 and 11, where variables in each PI subdomain are connected by dashed links. Table 3 summarizes the features of the two domains.

From each of the two domains, we generated a dataset of 10000 cases. For each dataset, Algorithm 1 was applied which does not use local computation based on crux. For comparison,

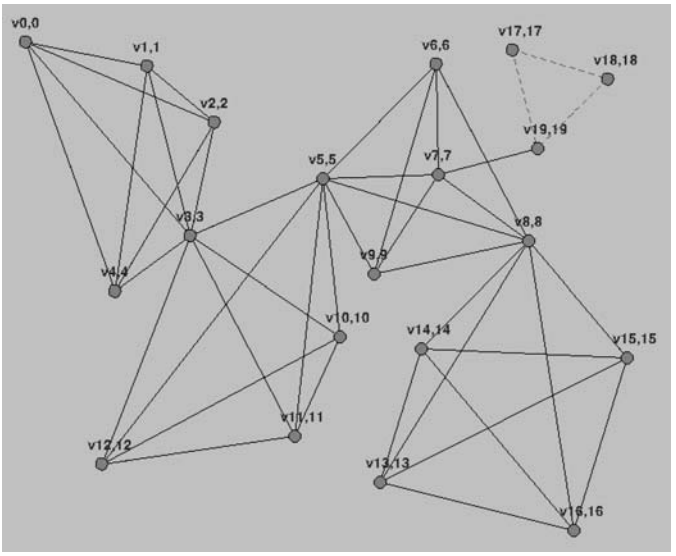

Fig. 10 DMN model learned from $Mid20v$ domain

Table 3 Summary of experiment with domains $Mid20v$ and $Bigsma17v$

Domain	No. Vars	Max var domain	No. PI submodels	Multilink search	Non-crux based	Crux based
mid20v	20	9	1	3	9679 sec	1704 sec
bigsma17v	17	9	2	3	4696 sec	77 sec

Algorithm 1 was enhanced with Algorithms 2, 3 and 4 and then applied to each dataset. Both Algorithm 1 and the enchanced algorithm were implemented using WEBWEAVR (Haddawy, 1999) in Java. For $Mid20v$ dataset, execution time for Algorithm 1 was 9679 seconds. The enhanced algorithm was completed in 1704 seconds, a speedup of about 6 times. For $Bigsma17v$ dataset, execution time for Algorithm 1 was 4696 seconds. The enhanced algorithm was completed in 77 seconds, a speedup of 61 times.

We interpret the difference between the performance enhancement in the two domains as follows: The DMN model from $Mid20v$ domain consists of cliques of about the same sizes (four cliques of five variables each and one clique of 3 variables). As a result, the speed up in each pass during lookahead search is often linear as predicted by m/μ in Section 6.2 On the other hand, the DMN model from $Bigsma17v$ domain consists of cliques whose cardinalities differ more significantly. Two cliques each contains five variables, two cliques each contains 3 variables, and three cliques each contains 2 variables. As a result, the speed up in each pass during lookahead search is often exponential as predicted by $m/\mu\ \kappa^{k-k'}$ in Section 6.2.

For practical relevance of learning in PI domains, we report the experimental result of knowledge discovery performed on a dataset about health care (Washington, 1999). The data set contains 12528 cases over 120 variables. PI models were discovered from a subdomain of 13 variables on physician profile (Table 4). Each variable takes from two to ten possible values. After removing cases with missing values, we acquired 11463 cases. Among them, 80% were used in the training and the rest for testing.

Using single-link lookahead search, with high penalty coefficient on model description length, a disconnected DMN was learned as in Fig. 12(a). It contains four marginally

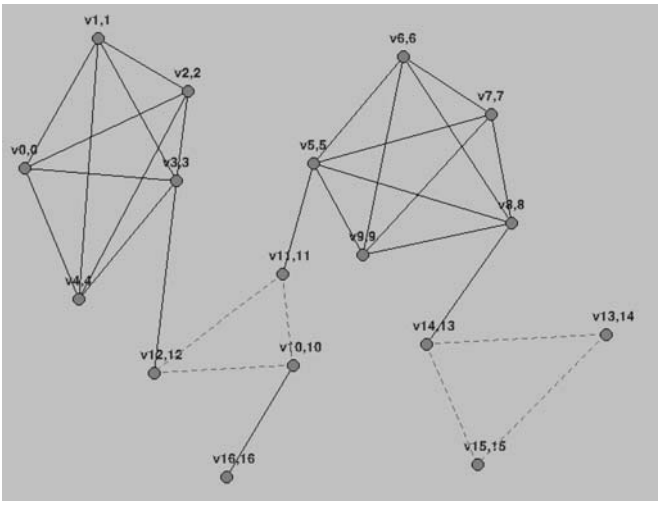

Fig. 11 DMN model learned from $Bigsma17v$ domain

Table 4 Variables on physician profile

Name	Label
IMGUSPR	Foreign med. school graduate
BIRTHX	Year of birth
YRBGNX	Year began practicing medicine
GRADYRX	Year of graduation
PCPFLAG	Primary care provider
SPECX	Specialty
OWNPR	Clinic ownership
PRCTYPE	Practice type
ACQUIRD	Practice acquired in last 2 yrs
NWMCARE	Accept new medicare patients
NWMCAID	Accept new medicaid patients
NWPRIV	Accept new privately insured
CAPAMTC	Capitate managed care revenue

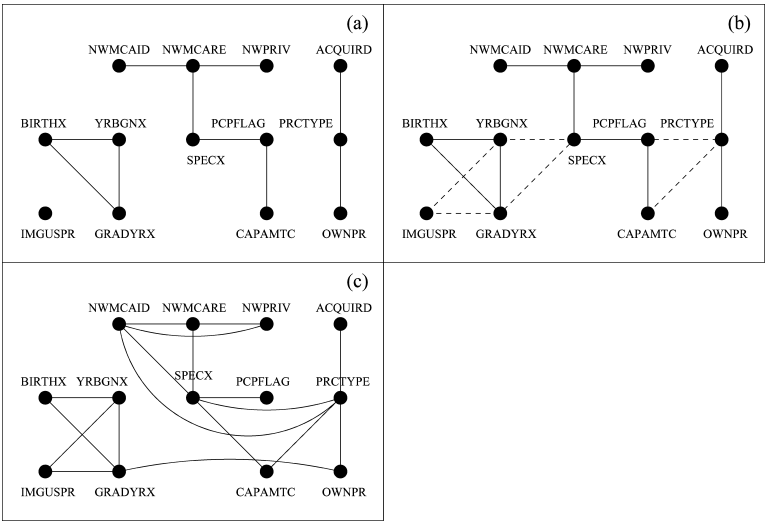

Fig. 12 Learned models as DMNs

independent components. Using double-link search, three PI submodels (b) were discovered:

$$\{IMGUSPR, YRBGNX, GRADYRX\},$$
$$\{YRGNX, GRADYRX, SPECX\},$$
$$\{PCPFLAG, CAPAMTC, PRCTYPE\}.$$

Dashed links represent links learned during double-link lookahead. With low penalty coefficient on model description length, a connected DMN was obtained with single-link lookahead as shown in (c).

To test the learned models, we performed probabilistic inference using the 2313 held-back cases. The value of SPECX was predicted with the observation on YRGNX and GRADYRX.

Table 5 Prediction accuracy

Learned net	Search	Infer. Time	Hits
Net (a)	Single-link	6m 31.97s	579
Net (c)	Single-link	8m 42.21s	579
Net (b)	Double-link	11m 31.18s	697

That is, the posterior probability distribution

$$P(SPECX|YRGNX, GRADYRX)$$

was computed for each held-back case using each learned model. The value of SPECX with the highest probability was regarded as the prediction of the model for the correspond case. A prediction is a *hit* if the value matches that in the held-back case. Otherwise, the prediction is a *miss*. Table 5 shows the total inference time using each learned model and the prediction accuracy. The models learned by single-link lookahead predicted with the same accuracy. Note that since SPECX has seven possible values, blind guess would have about 330 hits. The PI DMN predicted about 20% better than the other two, and it incurred only moderate increase in inference time.

9. Conclusion

Multi-link lookahead search is necessary to discover graphical models in potentially PI domains. We have shown that local computation techniques suitable for learning graphical models based on single-link lookahead are insufficient for multi-link lookahead search. We have proposed a graph theoretic concept called crux in decomposable Markov networks (DMNs) in the context of discovering such models from data. We have shown that crux forms a subset of variables sufficient to compute the incremental change of both model description length (δn) of a DMN and data description length (δh) given the model during multi-link lookahead search over alternative dependence structures. How to compute such changes is presented algorithmically. The overall incremental improvement of description length due to addition of multiple links is a function of δh and δn. Search can terminate when no alternative structures provide further improvement. In a large problem domain, the computation using crux can be much more efficient especially when the underlying model consists of cliques of highly dependent variables with uneven clique sizes. We have shown that the use of crux causes no loss of accuracy. Therefore, this contribution allows knowledge discovery to be performed on even larger domains where pseudo-independence potentially exists, given computational resources.

In analyzing the model description length (Section 5), we have used the upper bound for the number of unconstrained parameters. For instance, in Lemma 8, the upper bound of the number of unconstrained parameters needed to specify $P(Q)$ is given as $|D_Q| - 1$. This upper bound is based on the negation rule of probability theory. Using the dependence structure of a PI domain, tighter upper bounds can be derived, which will result in an improved accuracy of the description length and ultimately the improved learning outcome. Our ongoing research is making progress towards this goal. The expected result will likely change the value of $\Gamma_1(M)$. However, all major results on the effectiveness of crux in reducing the evaluation of description length to local computation still hold.

Acknowledgments The financial support from National Sciences and Engineering Research Council (NSERC) of Canada through Discovery Grant to the first author is acknowledged. We thank anonymous reviewers for insightful comments to the earlier manuscript.

References

Buntine, W. (1991). Classifiers: A theoretical and empirical study. In *Proc. 1991 Inter. Joint Conf. on Artificial Intelligence* (pp. 638–644), Sydney.

Buntine, W. (1994). Operations for learning with graphical models. *J. Artificial Intelligence Research*, *2*, 159–225.

Castillo, E., Gutierrez, J., & Hadi, A. (1997). *Expert systems and probabilistic network models.* Springer.

Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, *9*, 309–347.

Chow, C. K., & Liu, C. N. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Trans. on Information Theory*, (14), 462–467.

Dietterich, T. G. (1997). Machine-learning research. *AI Magazine*, *18*(4), 97–136.

Dawid, A. P., & Lauritzen S. L. (1993). Hyper Markov laws in the statistical analysis of decomposable graphical models. *Annals of Statistics*, *21*(3), 1272–1317.

Edwards D. (1995). *Introduction to graphical modelling*. Springer-Verlag.

Frydenberg M., & Lauritzen S. L. (1989). Decomposition of maximum likelihood in mixed graphical interaction models. *Biometrika*, *76*(3), 539–555.

Center for Studying Health System Change. (1999). Community tracking study physician survey (pp. 96–97), Washington, DC.

Haddawy, P. (1999). An overview of some recent developments in Bayesian problem-solving techniques. *AI Magazine*, *20*(2), 11–19.

Heckerman D. (1995). A tutorial on learning Bayesian networks. Technical report MSR-TR-95-06, Microsoft Research, Mocrisoft.

Heckerman, D., Geiger, D., & Chickering D. M. (1995). Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, *20*, 197–243.

Huffman, D. (1952). A method for the construction of minimum redundancy codes. *IRE Proceedings*, *40*, 1098–1101.

Hu, J., & Xiang, Y. (1997). Learning belief networks in domains with recursively embedded pseudo independent submodels. In: *Proc. 13th Conf. on Uncertainty in Artificial Intelligence* (pp. 258–265), Providence.

Huang, Y., & Xiang, Y. (1999). Learning Bayesian networks by learning decomposable Markov networks first. In: *Proc. IEEE Canadian Conf. on Electrical and Computer Engineering*, 1704–1709 Edmonton.

Jensen, F. V., Lauritzen, S. L., & Olesen K. G. (1990). Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, (4), 269–282.

Lauritzen, S. L. (1989). Lectures on contingency tables. Technical Report R 89-24, Aalborg University, Aalborg, Denmark.

Lauritzen, S. L. (1996). *Graphical models*. Oxford.

Lam, W., & Bacchus, F. (1994). Learning Bayesian networks: an approach based on the MDL principle. *Computational Intelligence*, *10*(3), 269–293.

Lauritzen, S. L., & Spiegelhalter D. J. (1988). Local computation with probabilities on graphical structures and their application to expert systems. *J. Royal Statistical Society, Series B*, (50), 157–244.

Malvestuto F. M. (1991). Approximating discrete probability distributions with decomposable models. *IEEE Tran. on Systems, Man, and Cybernetics*, *21*(5).

Madigan, D., & Raftery A. E. (1994). Model selection and accounting for model uncertainty in graphical models using Occam's window. *J. Amer. Stat. Association*, *89*(428), 1535–1546.

Neapolitan, R. E. (2004). *Learning Bayesian networks*. Prentice Hall.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann.

Rissanen (1978). Modeling by shortest data description. *Automatica*, *14*, 465–471.

Rebane, G., & Pearl, J. (1987). The recovery of causal ploy-trees from statistical data. In L. N. Kanal, J. F. Lemmer, & T. S. Levitt (Eds.), *Proc. of Workshop on Uncertainty in Artificial Intelligence*. (pp. 222–228), Seattle, Elsevier Science, Amsterdam.

Spirtes, P., & Glymour, C. (1991). An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, *9*(1), 62–73.

Shafer, G. (1996). *Probabilistic expert systems*. Society for industrial and applied mathematics, Philadelphia.

Srebro, N. (2003). Maximum likelihood bounded tree-width markov networks. *Artificial Intelligence*, *143*(1), 123–138.

Whittaker, J. (1990). *Graphical models in applied multivariate statistics*. Wiley.

Wong, S. K. M., & Xiang, Y. (1994). Construction of a Markov network from data for probabilistic inference. In: *Proc. 3rd Inter. Workshop on Rough Sets and Soft Computing* (pp. 562–569), San Jose.

Xiang, Y., Hu, J., Cercone, N., & Hamilton, H. (2000). Learning pseudo-independent models: analytical and experimental results. In: H. Hamilton (Eds.), *Advances in Artificial Intelligence.*, (pp. 227–239) Springer.

Xiang, Y. (1998). A characterization of single-link search in learning belief networks. In P. Compton H. Motoda, R. Mizoguchi, & H. Liu (Eds.), *Proc. Pacific Rim Knowledge Acquisition Workshop* (pp. 218–233), Singapore.

Xiang, Y. (2002). *Probabilistic reasoning in multi-agent systems: A graphical models approach*. Cambridge University Press.

Xiang, Y., Wong, S. K. M., & Cercone, N. (1996). Critical remarks on single link search in learning belief networks. In: *Proc. 12th Conf. on Uncertainty in Artificial Intelligence* (pp. 564–571), Portland.

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