A Divide-and-Conquer Method for Large Recursive Models with Incomplete Categorical Data

Seong-Ho Kim and Sung-Ho Kim*

We propose an ML estimation method for a recursive model of categorical variables which is too large to handle as a single model. We first split the whole model into a set of submodels which can be arranged in the form of a tree. Two conditions are suggested as an instrument for estimating the parameters of the whole model yet working within individual submodels. Theorems are proved to the effect that, when missing values are involved, we can generalize and apply the principle of EM to the tree of submodels so that the ML estimation is possible for a recursive model of any size. For illustration, simulation experiments of the ML estimation are carried out for recursive models of up to 158 binary variables, and the proposed method is applied successfully to real data where 28 binary variables are involved.

KEY WORDS: Consistency of distribution; D-split; EM; Family condition; Hyper-EM graph; Removable node; Marginal restructuring; Hyper-EM condition.

1 INTRODUCTION

The use of specialized statistical methods for categorical data has increased dramatically, particularly for applications in the biomedical and social sciences. Also, the statistical concepts of sufficiency and conditional independence have been attracting increasing attention in applied statistics, in particular for representing a relation among a group of variables in a graphical format. Graphical modelling is a body of statistical techniques for fitting graphical models to data. Graphical models of categorical or finitely discrete variables are representable in the form of an undirected graph, a directed acyclic graph, or a mixture of these (Whittaker 1990). The graphical models include graphical log-linear models (Darroch, Lauritzen, and Speed 1980; Fienberg 1980), recursive models for

*Sung-Ho Kim is Associate Professor of Statistics (E-mail: shkim@kaist.ac.kr) and Seong-Ho Kim is a Ph.D. student (E-mail: mathan@kaist.ac.kr) of Statistics, Division of Applied Mathematics, Korea Advanced Institute of Science and Technology, Daejeon, 305-701, South Korea.

contingency tables (Wermuth and Lauritzen 1983), Bayesian networks (Pearl 1988), and influence diagrams (Howard and Matheson 1981; Olmsted 1983; Shachter 1986; Smith 1989). Of these, recursive models, Bayesian networks, and influence diagrams of finitely discrete random variables share a common feature that the joint probability of the variables involved in each of them is expressible as a product of marginal or conditional probabilities. Statistical modelling of such models mostly rests on the methods developed for log-linear modelling (Bishop, Fienberg, and Holland 1975).

The iterative proportional fitting (IPF) algorithm is well known for fitting hierarchical log-linear models along with the Newton-Raphson algorithm. Among the hierarchical log-linear models, graphical log-linear models (Darroch, Lauritzen, and Speed 1980) have received attention since the model structure can be readily read off from a graph, the relation being interpretable in the context of the Markov property. While undirected graphs are used for graphical log-linear models, we use directed acyclic graphs for recursive models (Wermuth and Lauritzen 1983).

The EM (Dempster, Laird, and Rubin 1977) algorithm is a most popular method for estimating parameters of a model which involves latent variables. It consists of two operations, expectation for the missing variables and likelihood-maximization. Literature abounds on the EM concerning the issues of applications, convergence rates, and a variety of improved versions of it (see, for example, Van Dyk and Meng 1997). Birch (1963) considered maximum likelihood estimation for a recursive model of three variables by dealing with its log-linear model as a combination of a log-linear model of a marginal probability and that of a conditional probability, where the whole joint probability is given by the product of the marginal and the conditional probabilities. Goodman (1974a, b) related the latent class models to log-linear models and gave a general algorithm for maximum likelihood estimation for latent class models (also see Haberman 1977, 1979). Lauritzen (1995) derived an EM algorithm for graphical models of contingency tables with missing data by exploiting the computational scheme of Lauritzen and Spiegelhalter (1988), where data may not necessarily be missing for the same set of variables. Thiesson (1995, 1996) explored an acceleration of the EM algorithm for the recursive model by a generalized conjugate gradient algorithm and provided the first and second order derivatives of the log-likelihood and log-posterior distribution to be used for iterative estimation methods under a Bayesian framework. Kim (2000) suggested an experimental form of the method that will be proposed in this paper and proved theorems that constitute a part of the main result of this paper. For methods of parameter estimation for graphical models of mixed variables, some of which are continuous and the others finitely discrete, the readers are referred to Lauritzen (1996) and Edwards and Lauritzen (2001).

Our goal in this paper is to propose a method of maximum likelihood estimation for a recursive model of categorical variables which is too large to handle as a single model. First we split the whole model into a collection of submodels in such a way that the probability model of the whole model may be factorized where each factor corresponds to one and only one submodel and a factor is expressed as a product of conditional probabilities of the variables that are involved in the corresponding submodel. The resulting topology of the submodels is given in the form of a tree where a pair of submodels that share a set of variables are linked by an edge. We then extend the principle of EM (Dempster et al., 1977) and apply it to the tree of submodels for parameter estimation for the whole model. Each submodel may contain part of the unobservable variables of the whole model and similarly for the observed variables. So the E(expectation)-step and the M(likelihood maximization)-step of the EM can only be employed within individual submodels and some additional procedure is required in order to make the resulting estimates for individual submodels contribute toward the ML estimates for the whole model. By this procedure, we can obtain the distribution based on the estimates of the probability model of every individual submodel being consistent throughout the whole model. In this context, we will call the proposed method a *hyper-EM* algorithm.

This paper consists of 8 sections. In section 2 we introduce notations and define the notions of splitting, t-splitting, and d-splitting of a graphical model and in section 3 we present conditions that are to be satisfied so that a model may be handled as a tree of submodels. Several concepts that help our proposed method work for estimating the parameters of a given model are introduced. Section 4 then elaborates on operations that are necessary for making the estimated distribution consistent between submodels. In section 5, we describe what to do in the E-step and M-step of the hyper-EM algorithm for the consistency of the estimated distribution between submodels. We consider three different models in section 6 and apply the proposed estimation method to these models based on simulated data for illustration, and then in section 7 we apply the method to a real data set where 28 binary variables are involved. Section 8 concludes the paper with summarizing remarks.

2 PRELIMINARIES AND D-SPLITTING

A graphical model is a statistical model whose model structure can be represented by a graph, and we will denote the graph of a graphical model by $\mathcal{G} = (V, E)$, where V is the index set of the nodes involved in the model and E a set of edges between the nodes in V. E is given as a set of the ordered pairs (u, v) such that $E \subseteq V \times V$ where (u, v) symbolizes a directed edge or an arrow from node u to node v in graph $\mathcal{G}$. If both (u, v) and (v, u) are included in E, it means that there is an undirected edge between nodes u and v. Thus if $\mathcal{G} = (V, E)$ is the model structure of a recursive model and $(u, v) \in E$, then $(v, u) \notin E$. A node in the graph of a graphical model corresponds to a variable of the model. So we will use the terms node and variable interchangeably.

We will denote by V_i the index set of all the variables involved in submodel i of $\mathcal{G}$, and we will use the lowercase x to denote the cell location of a contingency table and use x_A and x^i for the contingency table of the variables indexed in A and in V_i, respectively. If a pair of sets of nodes share a non-empty set of nodes, then we call the pair *neighboring* sets. If there is an arrow (a, b), then we will say that node a is a parent of node b and node b is a child of node a. We will denote by $pa(v)$ the set of the parents of node v and by $ch(v)$ the set of the children of v and let $fa(v) = \{v\} \cup pa(v)$. For a subset $A \subseteq V$, we will define

$$\begin{aligned}
pa(A) &= \cup_{v \in A} pa(v) \setminus A, \\
ch(A) &= \cup_{v \in A} ch(v) \setminus A, \\
fa(A) &= A \cup pa(A), \quad \text{and} \\
clan(v) &= fa(ch(v) \cup \{v\}), \quad \text{for a node } v \in V.
\end{aligned}$$

The node which does not have any child node will be called a *terminal* node and the node which does not have any parent node a *root* node. If $(a, b) \in E$, we say that node a is *adjacent* to node b or vice versa. A graph is said to be *complete* if all vertices are adjacent each other. A complete subgraph is a subgraph which is complete. A complete subgraph that is maximal in $\mathcal{G}$ is called a *clique* of $\mathcal{G}$. A sequence of adjacent nodes from node a to node b $(a \neq b)$ is called a *chain* from a to b (or from b to a) such as $a = a_1, \cdots, a_r = b$.

If there is a chain from a to b and the arrows in the chain are all heading toward b, we call node b a *descendant* of a. For $A \subseteq V$, an *induced subgraph* of $\mathcal{G}$ confined to A is defined as $\mathcal{G}_A = (A, E_A)$, $E_A = E \cap (A \times A)$ and we will simply write $\mathcal{G}_i = (V_i, E_i)$ for $\mathcal{G}_{V_i} = (V_i, E_{V_i})$, $V_i \subseteq V$. For a set of edges E, we define $sym(E) = \{(b, a) | (a, b) \in E\} \cup E$. A graph $\mathcal{G} = (V, E)$ is *undirected* if $E = sym(E)$. The *associated undirected graph* of graph $\mathcal{G} = (V, E)$ is an undirected version of $\mathcal{G}$ and we will represent it by $\mathcal{G}^{\sim} = (V, sym(E))$.

As in Dawid (1979), we will write $X \perp Y | Z$ to mean that X and Y are conditionally independent given Z. We will denote by $i \wedge j$ the index set of the variables that are involved in both submodels i and j. If V_i and V_j are the index sets of the variables involved in submodels i and j, respectively, then we have $i \wedge j = V_i \cap V_j$.

Definition 2.1. A chain C from α to β in a directed acyclic graph $\mathcal{G}$ is said to be *blocked* by S, if it contains a node $\gamma \in C$ such that either

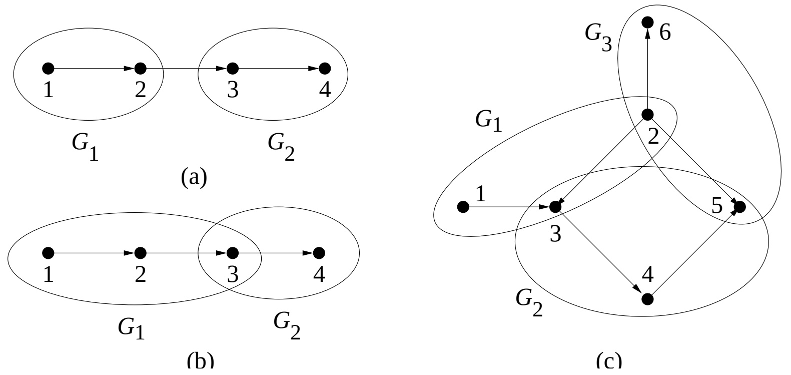


Figure 2.1: Recursive models and splitting. The model in panel (a) is not split into two submodels, but the models in panels (b) and (c) are split into two and three submodels respectively, where bullets symbolize nodes and ovals submodels.

(i) $\gamma \in S$ and arrows of C do not meet head to head at γ, or

(ii) $\gamma \notin S$, nor has γ any descendants in S, and arrows of C do meet head to head at γ.

A chain that is not blocked by S is said to be *active*. Two subsets A and B are said to be *d-separated* by S if all chains from A to B are blocked by S (Pearl 1986).

Theorem 2.1. *If A and B are d-separated by S, $A \perp B|S$.*

Proof: See Lauritzen, Dawid, Larsen and Leimer (1990); Pearl and Verma (1987); Verma (1988). □

Definition 2.2. Consider a recursive model $\mathcal{G} = (V, E)$. If there are k distinct submodels with the corresponding graphs,

$$\mathcal{G}_1 = (V_1, E_1), \mathcal{G}_2 = (V_2, E_2), \cdots, \mathcal{G}_k = (V_k, E_k),$$

for which the following holds :

$$V = \cup_{i=1}^k V_i,\ V_i \nsubseteq V_j \text{ for } i \neq j, \text{ and } E = \cup_{i=1}^k E_i, \tag{2.1}$$

then we will say that $\mathcal{G}$ is *split* into $\{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_k\}$.

Consider recursive models as in panels (a), (b), and (c) in Figure 2.1. The model as in panel (a) is not split into $\{\mathcal{G}_1, \mathcal{G}_2\}$ since edge $(2, 3) \notin E_1 \cup E_2$. The models as in panel (b) and (c) are however split into two and three submodels respectively. Note that for both of these panels, condition (2.1)

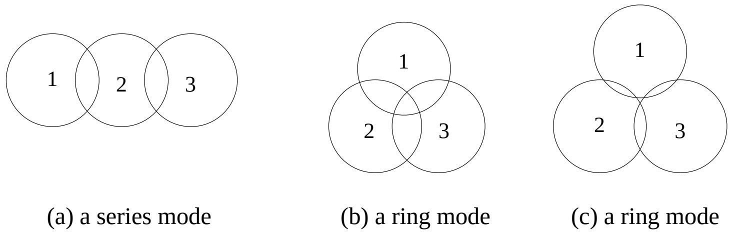


Figure 2.2: Three different modes of submodel arrangement. Circles symbolize submodels

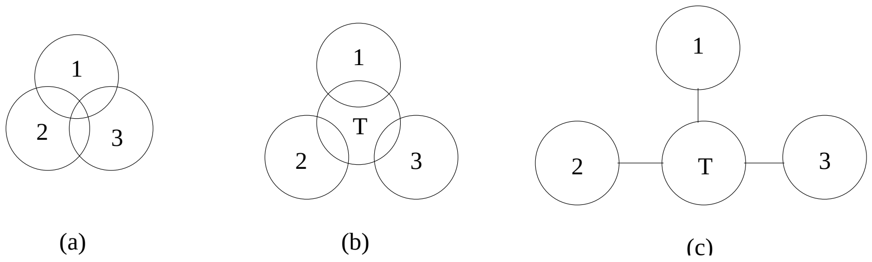


Figure 2.3: A ring mode arrangement and its tree-shape arrangement

is satisfied. When a recursive model is split, every edge of the model is included in at least one of the submodels.

After splitting a model, it can be arranged in the form of submodel-chain. For instance, in Figure 2.2, the graphs of submodels are linked in series in panel (a), while they are linked in a ring in the other panels. In panel (b) all the submodels share a set of variables, while there is no such set in any of the panels (a) and (c). We will call the arrangement as in panel (a) *series-mode arrangement* and the arrangement as in panels (b) and (c) *ring-mode arrangement*. In general, arrangements are a mixture of ring-mode and series-mode and we will call this arrangement a *mixed arrangement*. If a pair of submodels of a model share a non-empty set of variables, we will call the submodels of the pair *neighboring submodels* and call one of them a *neighbor submodel* of the other.

Consider the graphs of submodels $1, 2, \cdots, k$ of a recursive model $\mathcal{G}$ where the k submodels are arranged in a single ring-mode arrangement as in panel (b) or (c) of Figure 2.2 and denote by V_T the index set of the variables that are involved in at least two of the submodels. Then each of the k submodels is separated from the others by T in the associated undirected graph $\mathcal{G}^{\sim}$. We will regard T as a submodel in a ring-mode arrangement. For example, the graph in panel (a) of Figure 2.3 is a ring mode arrangement and the one in panel (b) is another display of the graph in panel (a) except that V_T is presented as a submodel along with the submodels $1, 2, 3$. The graph in panel (b) is expressed in the form of a tree in panel (c). With V_T regarded as a submodel, we arrange the

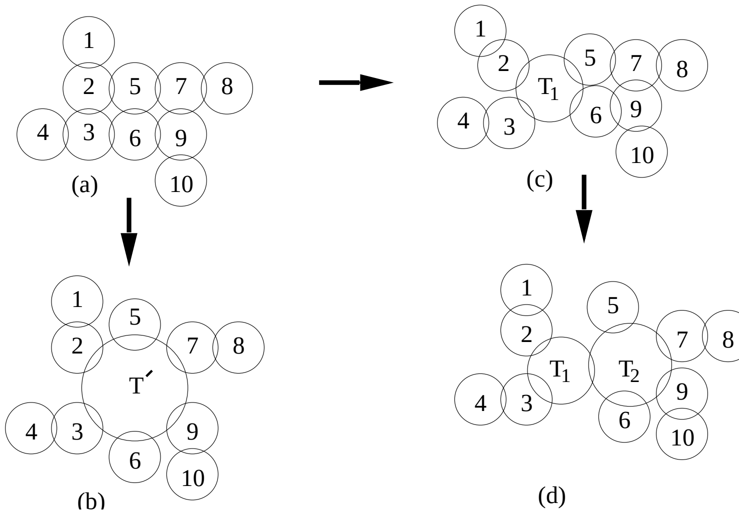


Figure 2.4: Transforming a submodel-arrangement into trees of submodels

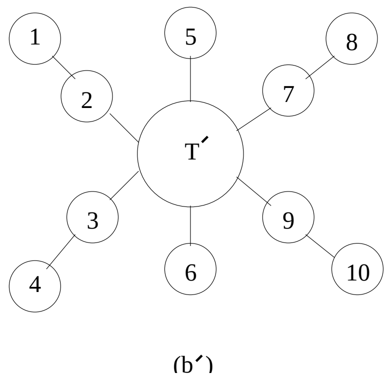
$k+1$ submodels as follows:

$$\mathcal{G}_1 = (V_1, E_1), \mathcal{G}_T = (V_T, E_T), \mathcal{G}_2 = (V_2, E_2), \cdots, \mathcal{G}_k = (V_k, E_k).$$

The "T" is from "Transfusion of estimates." There are as many $\mathcal{G}_T$'s as the rings of submodels in a submodel-arrangement, and in estimation each $\mathcal{G}_T$ submodel will be used as a transfusion box of estimates among the submodels constituting the corresponding ring in a ring-mode arrangement. In this regard, we will call the $\mathcal{G}_T$ submodel a *T-type model*.

In a tree of submodels, a submodel is represented by a node. If the original arrangement from a splitting is in a series mode, its corresponding tree is just a single chain of nodes. If however the original arrangement is in a ring-mode, its corresponding tree is no longer a single chain. In Figure 2.3, submodel T is created, and we will call such a submodel a *branching node* in the corresponding tree. So if a tree of submodels is of multiple branching nodes, it means the original submodel arrangement is of as many rings of submodels. We will call by *t-splitting* the splitting which ends up with a tree of submodels. In a tree of submodels, any two submodels i and j are separated by $V_i \cap V_j$ when their corresponding nodes are adjacent in the tree. Example 2.1 below illustrates how we reexpress a mixed arrangement of submodels in a tree of submodels.

Example 2.1. Suppose a recursive model $\mathcal{G}$ is split into submodels $1, 2, \cdots, 10$ as depicted in Figure 2.4. In panel (a), we may regard submodels, $2, 3, 5, 6, 7, 9$, as arranged in a ring or regard submodels, $2, 3, 5, 6$ and submodels $5, 6, 7, 9$, as arranged in different rings. In the former viewpoint, we form a T-type submodel T' of the variables that are shared by any neighboring submodels of the submodels $2, 3, 5, 6, 7, 9$; in the latter viewpoint, we form two T-type submodels T_1 and T_2 where T_1



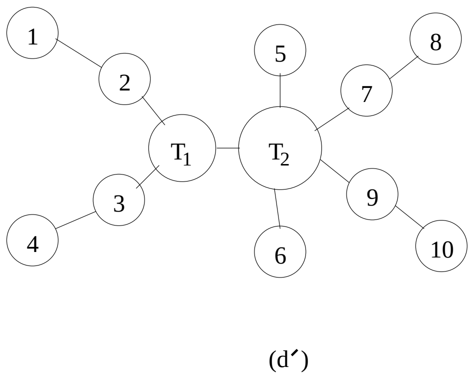


Figure 2.5: Tree-shape arrangements. Panels (b') and (d') are respectively tree-shape arrangements of the arrangements in panels (b) and (d) of Figure 2.4.

consists of the variables that are shared among submodels $2, 3, 5, 6$, and T_2 consists of the variables that are shared among submodels $T_1, 5, 6, 7, 9$. In panel (b) of Figure 2.4, submodel T' separates the remaining submodels into 6 parts; in panels (c) and (d) T' is split into T_1 and T_2. If we let $A = \{2, 3, 5, 6\}$, $B = \{T_1, 5, 6, 7, 9\}$ and $C = \{2, 3, 5, 6, 7, 9\}$, then $V_{T_1} = \cup[V_i \cap V_j \ ; \ i \neq j, i, j \in A]$, $V_{T_2} = \cup[V_i \cap V_j \ ; \ i \neq j, i, j \in B]$, and $V_{T'} = \cup[V_i \cap V_j \ ; \ i \neq j, i, j \in C]$. The arrangements in panels (b) and (d) of Figure 2.4 are reexpressed respectively in a tree-shape in panels (b') and (d') of Figure 2.5. $\square$

Since a series-mode arrangement is a particular form of a tree-shape arrangement, we can say that all the submodels can be arranged in a tree-shape. A chain from submodel i to submodel j is a sequence of neighboring submodels from submodel i to submodel j (or from submodel j to submodel i) such as $i = i_0, i_1, \cdots, i_r = j$ where $i_s \neq i_t$ whenever $s \neq t$. It is important to note in Example 2.1 that as for the submodels arranged through t-splitting, there is a unique chain of submodels between every pair of submodels. We denote the index set of the submodels on the chain from submodel i to submodel j by $cm(i, j)$, and we denote the index set of the neighboring submodel of submodel i on $cm(i, j)$ by $nc(i, cm(i, j))$ and the index set of the neighboring submodel of submodel j on $cm(i, j)$ by $nc(j, cm(i, j))$. Since submodels i and j are located at the end points of the chain $cm(i, j)$, each of submodels i and j has only one neighboring submodel in $cm(i, j)$.

Definition 2.3. We will say that a recursive model with graph $\mathcal{G}$ is *d-split* into a tree of k submodels if $\mathcal{G}$ is t-split into k submodels and, for any neighboring submodels in the tree, say i and j, $V_i \setminus V_j$ is d-separated from $V_j \setminus V_i$ by $V_i \cap V_j$.

For any three sets of variables, A, B, and S, $A \perp B|S$ is equivalent to $(A \setminus S) \perp (B \setminus S)|S$. So

Theorem 2.1 implies that, if neighboring submodels i and j which are obtained through t-splitting is d-separated, then $V_i \perp V_j | V_i \cap V_j$. From this fact follows the theorem below.

Theorem 2.2. *Suppose a recursive model $\mathcal{G} = (V, E)$ is d-split into k, $k \geq 1$, submodels,*

$$\mathcal{G}_1 = (V_1, E_1), \cdots, \mathcal{G}_k = (V_k, E_k).$$

Then, for all $i, j = 1, 2, \cdots, k$,

$$V_i \perp V_j | (V_i \cap V_{nc(i,cm(i,j))}) \text{ and } V_i \perp V_j | (V_j \cap V_{nc(j,cm(i,j))}). \tag{2.2}$$

Proof: Recall that a d-splitting produces a tree-shape arrangement of submodels. So, for any neighboring submodels i and j, $nc(i, cm(i,j)) = j$ and $nc(j, cm(i,j)) = i$, which implies (2.2) by the definition of d-splitting. Suppose submodels i and j are not neighboring submodels, and let $cm(i,j) = \{i = i_0, i_1, \cdots, i_{r+1} = j\}$, $r \geq 1$. Then we have $nc(i, cm(i,j)) = i_1$ and $nc(j, cm(i,j)) = i_r$. If $V_i \not\perp V_j | (V_i \cap V_{i_1})$, then, for some nodes $\alpha \in V_i \setminus V_{i_1}$ and $\beta \in V_j \setminus V_{i_1}$, arrows in a chain from α to β meet head to head at some node, say z, in $V_i \cap V_{i_1}$, and so there exists a node, say w, in $V_{i_1} \setminus V_i$ such that $w \in pa(z)$. Thus arrows in a chain from α to w meet head to head at z, contradicting the d-splitting condition of the theorem. By applying the same argument, we have $V_i \perp V_j | (V_j \cap V_{i_r})$. □

We define an index set of variables as the *boundary* of submodel i, $i \geq 2$, and denote it by $bd(i)$ if

$$V_i \perp \cup_{j=1}^{i-1} V_j \; | bd(i).$$

For a recursive model with graph $\mathcal{G}$, we define a set $E^+(A) \subseteq A \times A$, $A \subseteq V$, as follows. Consider two nodes $\alpha, \beta \in A$, $\alpha < \beta$, each of which is adjacent to some node in $V \setminus A$. If there is a chain from α to β which is blocked by a node in $V \setminus A$ in $\mathcal{G}$, then $(\alpha, \beta) \in E^+(A)$. If $A = V$, it is obvious that $E^+(A) = \emptyset$. We will call $\mathcal{G}^A = (A, E^A)$ with $E^A = E_A \cup E^+(A)$ the *marginalized subgraph* of $\mathcal{G}$ confined to A and we will simply write $\mathcal{G}^i = (V_i, E^i)$ for $\mathcal{G}^{V_i} = (V_i, E^{V_i})$, $V_i \subseteq V$. The marginalized subgraph $\mathcal{G}^A$ is the model structure of the submodel which is obtained by confining a recursive model with graph $\mathcal{G}$ onto A.

Theorem 2.3. *Consider a recursive model with graph $\mathcal{G} = (V, E)$. Suppose $\mathcal{G}$ is d-split into $\mathcal{G}_1$ and $\mathcal{G}_2$. Then $E^+(V_1), E^+(V_2) \subseteq (V_1 \cap V_2) \times (V_1 \cap V_2)$ and $E^+(V_1 \cap V_2) = E^+(V_1) \cup E^+(V_2)$.*

Proof: If $(\alpha, \beta) \in E^+(V_1)$ such that $\alpha, \beta \in V_1$ with $\alpha < \beta$, then, by definition, each of α and β is adjacent with some node in $V_2 \setminus V_1$ in $\mathcal{G}$. By the definition of splitting, $E = E_1 \cup E_2$. So, it must

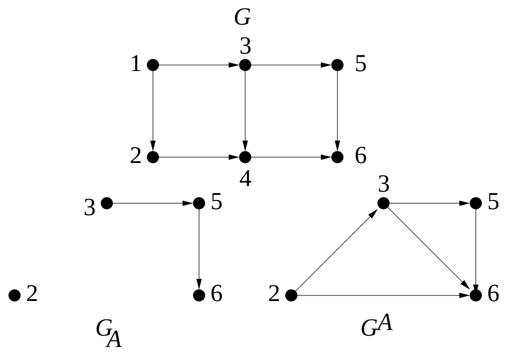


Figure 2.6: An induced subgraph $\mathcal{G}_A$ and a marginalized subgraph $\mathcal{G}^A$ for $A = \{2, 3, 5, 6\}$

be that $\alpha, \beta \in V_1 \cap V_2$. Similarly, $E^+(V_2) \subseteq (V_1 \cap V_2) \times (V_1 \cap V_2)$. Therefore, $E^+(V_1), E^+(V_2) \subseteq (V_1 \cap V_2) \times (V_1 \cap V_2)$.

For $\alpha, \beta \in V_1 \cap V_2$ where $\alpha < \beta$,

$$\begin{aligned}(\alpha, \beta) \in E^+(V_1 \cap V_2) \quad &\iff \quad \text{There exists a chain from } \alpha \text{ to } \beta \text{ which is blocked by some} \\ &\qquad \text{node in } V_2 \setminus V_1 \text{ or } V_1 \setminus V_2 \\ &\iff \quad (\alpha, \beta) \in E^+(V_1) \text{ or } (\alpha, \beta) \in E^+(V_2) \\ &\iff \quad (\alpha, \beta) \in E^+(V_1) \cup E^+(V_2).\end{aligned}$$

This completes the proof. □

Example 2.2. Consider a recursive model $\mathcal{G} = (V, E)$ with $V = \{1, 2, 3, 4, 5, 6\}$ and $E = \{(1, 2), (1, 3), (2, 4), (3, 4), (3, 5), (4, 6), (5, 6)\}$. Let $A = \{2, 3, 5, 6\}$. $\mathcal{G}_A$ is given in Figure 2.6. Then $E_A = \{(3, 5), (5, 6)\}$. The nodes which are adjacent to some node in $V \setminus A (= \{1, 4\})$ are $2, 3, 6$, and chains $2 - 1 - 3$, $2 - 4 - 6$ and $3 - 4 - 6$ are blocked respectively by nodes, $1, 4, 4$, in $V \setminus A$. Thus $(2, 3), (2, 6), (3, 6) \in E^+(A)$ and $E^A = E_A \cup E^+(A) = \{(2, 3), (2, 6), (3, 5), (3, 6), (5, 6)\}$. The marginalized subgraph $\mathcal{G}^A$ is at the bottom right in Figure 2.6. □

It is important to note that the d-splitting always ends up with a tree of submodels and a tree of submodels produces uniqueness of the chains of submodels. Especially, by Theorem 2.2 and the uniqueness of the chains of submodels, we can obtain that for all neighboring submodels i and j,

$$V_i \perp V_j | V_i \cap V_j$$

where a recursive model $\mathcal{G}$ is d-split into k submodels. This conditional independence is helpful for representing the probability model for the whole model in a factorized format in terms of (conditional) probabilities of submodels.

3 TWO CONDITIONS

For a subset A of V, $P_A(x_A)$ (or $P(x_A)$) represents the cell-probability at the cell x_A; $P_i(x^i)$ represents the cell-probability at x^i for submodel i. We let $P_{B|A}(x_B|x_A) = P(X_B = x_B|X_A = x_A)$ for $B \subseteq V$, $P_{i|A}(x^i|x_A) = P(X^i = x^i|X_A = x_A)$ and $P_{i|j}(x^i|x^j) = P(X^i = x^i|X^j = x^j)$. We denote the cell frequency at the cell-entry x^i for submodel i by $n^i(x^i)$ and by $N^i(x^i)$ the corresponding random quantity. Analogously, we denote by $n_A(x_A)$ the cell frequency at the cell-entry x_A for a set A of variables and by n the total frequency. The superscripts of n and x are submodel labels and the subscripts index sets of variables. We will denote the collection of all the cell locations x_A (or x^i), for an index set A (or submodel i), by $\mathcal{X}_A$ (or $\mathcal{X}^i$).

The chain of submodels must satisfy some mild conditions so that a variation of EM for the whole model produces estimates under the model structure of the whole model. The conditions are *Family condition* and *Hyper-EM condition* as will be described in this section.

> **Family Condition:** Consider graphs of submodels $\mathcal{G}_1, \cdots, \mathcal{G}_k$ of a recursive model $\mathcal{G}$. Then for every node $v \in V$, there exists at least one $j \in \{1, 2, \cdots, k\}$ such that $fa(v) \in V_j$.

Theorem 3.1. *Suppose a recursive model with graph $\mathcal{G}$ is d-split. Then the graphs of submodels always satisfy the Family condition.*

Proof: Assume that the model is d-split into k submodels. Suppose there are nodes α and β in $pa(v)$, $v \in V$, such that $\alpha \in V_i \setminus V_j$ and $\beta \in V_j \setminus V_i$ for some $1 \le i < j \le k$. So, $v \in V_i \cap V_j$ since $\mathcal{G}$ is d-split. This means that arrows in a chain from α to β meet head to head in $V_i \cap V_j$. Thus, by the definition of d-separation, a chain from α to β is not blocked by $V_i \cap V_j$. This contradicts the condition of the theorem that $\mathcal{G}$ is d-split. Therefore, the Family condition must hold when $\mathcal{G}$ is d-split. $\square$

This theorem shows that the Family condition is consequential to the d-splitting. As a matter of fact, the Family condition and the d-split are closely related each other.

Theorem 3.2. *A recursive model $\mathcal{G}$ is assumed to be t-split into k submodels. Then the Family condition is satisfied by the k submodels if and only if $\mathcal{G}$ is d-split into k submodels.*

Proof: By Theorem 3.1, we have only to show the necessity of the theorem. Suppose $\mathcal{G}$ is not d-split. Then there must exist submodels i and j such that at least one of chains from $V_i \setminus V_j$ to

$V_j \setminus V_i$ meets head to head in $V_i \cap V_j$. In other words, there are nodes $\alpha \in V_i \setminus V_j$ and $\beta \in V_j \setminus V_i$ such that $\alpha, \beta \in pa(v)$ for some $v \in V_i \cap V_j$, contradicting the Family condition. This completes the proof. $\square$

The *maximum likelihood estimate* (MLE) of $P_V(x)$ is given, under the recursive model $\mathcal{G}$, by

$$\widehat{P}_V(x) = \prod_{v \in V} \widehat{P}_{v|pa(v)}(x_v|x_{pa(v)}) = \prod_{v \in V} \frac{n_{fa(v)}(x_{fa(v)})}{n_{pa(v)}(x_{pa(v)})} \tag{3.1}$$

where $n_\emptyset(x_\emptyset) = n$. It is possible that $pa(v) = fa(v')$ for some nodes v and v'. Thus after crossing out the equal terms in the numerator and the denominator of the right hand side of (3.1), we end up with

$$\widehat{P}_V(x) = \prod_{i=1}^{\rho} \frac{n_{C_i}(x_{C_i})}{n_{S_i}(x_{S_i})} \tag{3.2}$$

where $C_i, S_i \subseteq V$, $1 \le i \le \rho \le |V|$ and $C_i \ne S_j$ for all i, j with $1 \le i, j \le \rho$ and for some i, it is possible that $S_i = \emptyset$. Obviously,

$$\{C_i \mid i = 1, 2, \cdots, \rho\} \subseteq \{fa(v) \mid v \in V\} \text{ and } \{S_i \mid i = 1, 2, \cdots, \rho\} \subseteq \{pa(v) \mid v \in V\}. \tag{3.3}$$

We will call S_i and C_i respectively *f-separator* and *f-clique* (think of factorization for "f") of the recursive model. For convenience's sake, we will denote

$$\begin{aligned} n_{fa(v)}(x_{fa(v)})/n &\quad \text{by} \quad [fa(v)](x_{fa(v)}), \\ n_{pa(v)}(x_{pa(v)})/n &\quad \text{by} \quad [pa(v)](x_{pa(v)}), \\ n_{C_i}(x_{C_i})/n &\quad \text{by} \quad [C_i](x_{C_i}), \\ \text{and } n_{S_i}(x_{S_i})/n &\quad \text{by} \quad [S_i](x_{S_i}). \end{aligned}$$

If confusion is not likely, we will ignore the argument 'x'.

It is worthwhile to look into when $|V| > \rho$ holds in expression (3.2). As a matter of fact, $|V| > \rho$ if and only if there exists a node $v \in V$ such that

$$pa(v) = fa(v') \tag{3.4}$$

for some other node $v' \in V$. If $|V| > \rho$, then the procedure from (3.1) to (3.2) implies that there are nodes v and v' satisfying (3.4). We can think of typical situations concerning $|V|$ and ρ in Example 3.1.

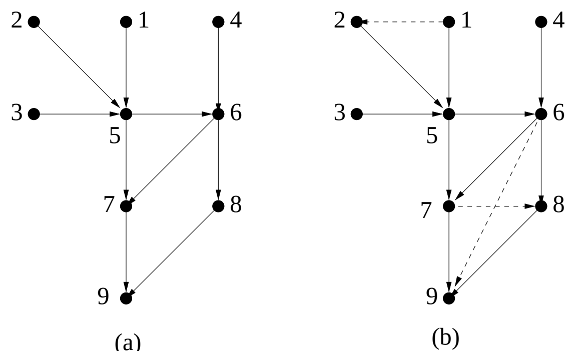


Figure 3.1: Graphs referred to in Example 3.1

Example 3.1. The graph in panel (a) of Figure 3.1 is of a recursive model for which

$$\begin{aligned}\widehat{P}(x_V) &= \frac{[\{1\}](x_{\{1\}})[\{2\}](x_{\{2\}})[\{3\}](x_{\{3\}})[\{4\}](x_{\{4\}})[\{1,2,3,5\}](x_{\{1,2,3,5\}})[\{4,5,6\}](x_{\{4,5,6\}})}{[\{1,2,3\}](x_{\{1,2,3\}})[\{4,5\}](x_{\{4,5\}})} \\ &\quad \times \frac{[\{5,6,7\}](x_{\{5,6,7\}})[\{6,8\}](x_{\{6,8\}})[\{7,8,9\}](x_{\{7,8,9\}})}{[\{5,6\}](x_{\{5,6\}})[\{6\}](x_{\{6\}})[\{7,8\}](x_{\{7,8\}})}\end{aligned}$$

In this expression, there are no nodes for which (3.4) is satisfied. This means all the $fa(v)$'s and $pa(v)$'s are f-cliques and f-separators, respectively.

If, however, we add edges, $(1,2), (6,9)$, and $(7,8)$, as in panel (b), then $pa(2) = fa(1)$ and $pa(9) = fa(8)$; thus yielding the inequality $|V| > \rho$ and the irreducible expression of (3.2) as in

$$\begin{aligned}\widehat{P}(x_V) &= \frac{[\{1,2\}](x_{\{1,2\}})[\{3\}](x_{\{3\}})[\{4\}](x_{\{4\}})[\{1,2,3,5\}](x_{\{1,2,3,5\}})[\{4,5,6\}](x_{\{4,5,6\}})}{[\{1,2,3\}](x_{\{1,2,3\}})[\{4,5\}](x_{\{4,5\}})} \\ &\quad \times \frac{[\{5,6,7\}](x_{\{5,6,7\}})[\{6,7,8,9\}](x_{\{6,7,8,9\}})}{[\{5,6\}](x_{\{5,6\}})[\{6,7\}](x_{\{6,7\}})}\end{aligned}$$

Note in this expression that nodes 8 and 9 appear just once and the other nodes more than once. □

Consider a recursive model with graph $\mathcal{G}$ and let $x_{(v)}$ be the subvector of x with $x_{\{v\}}$ only excluded; analogously for a subset A of V, we will denote $x_{(A)}$ the subvector of x with x_A only excluded.

Definition 3.1. Let a node v be a node in $\mathcal{G}$ and let P_V be a distribution function of a set of discrete random variables indexed in V. If for all $x_{(v)} = x^*_{(v)} \in \mathcal{X}_{V\setminus\{v\}}$,

$$\sum_{x_{\{v\}}} \widehat{P}_V(x) = \widehat{\sum_{x_{\{v\}}} P_V(x)}, \tag{3.5}$$

i.e., the marginal of $\widehat{P}_V$ onto $V \setminus \{v\}$ is the same as the MLE of the marginal of P_V onto $V \setminus \{v\}$, then we will call the node v a *removable* node of $\mathcal{G}$.

If a node v is removable in $\mathcal{G}$, it means that we may obtain the MLE of $P_{V\setminus\{v\}}$ in $\mathcal{G}^{V\setminus\{v\}}$ by marginalizing $\widehat{P}_V$ on $V \setminus \{v\}$ in $\mathcal{G}$. This is illustrated below.

Example 3.2. Consider a recursive model with graph $\mathcal{G} = (V, E)$ where $V = \{1, 2, 3\}$ and $E = \{(1, 2), (1, 3)\}$. Then we have at $x_{(2)} = x^*_{(2)}$

$$\sum_{x_{\{2\}}} \widehat{P}(x_V) = \sum_{x_{\{2\}}} \frac{[\{1,2\}](x_{\{1,2\}})[\{1,3\}](x_{\{1,3\}})}{[\{1\}](x_{\{1\}})} = [\{1,3\}](x_{\{1,3\}})$$

and

$$\widehat{\sum_{x_{\{2\}}} P(x_V)} = [\{1,3\}](x_{\{1,3\}}).$$

So, node 2 is removable from $\mathcal{G}$ and so is node 3 for the same reason. However, at $x_{(1)} = x^*_{(1)}$

$$\begin{aligned}\sum_{x_{\{1\}}} \widehat{P}(x_V) &= \sum_{x_{\{1\}}} \frac{[\{1,2\}](x_{\{1,2\}})[\{1,3\}](x_{\{1,3\}})}{[\{1\}](x_{\{1\}})} \\ &\neq [\{2,3\}](x_{\{2,3\}}) = \widehat{\sum_{x_{\{1\}}} P(x_V)}.\end{aligned}$$

As for a contingency table of X_1, X_2, X_3 as in Table 3.1, we have at $x_{\{2,3\}} = (0, 0)$

$$\begin{aligned}\sum_{x_{\{1\}}} \frac{[\{1,2\}](x_{\{1,2\}} = (x_1, 0))[\{1,3\}](x_{\{1,3\}} = (x_1, 0))}{[\{1\}](x_{\{1\}})} &= \frac{5 \cdot 3}{20 \cdot 8} + \frac{4 \cdot 7}{20 \cdot 12} = \frac{101}{480} \\ &\neq \frac{5}{20} = [\{2,3\}](x_{\{2,3\}} = (0, 0)).\end{aligned}$$

Node 1 is not removable here. □

Definition 3.1 says that the node removability has to do with the invariance property of the MLE. It has an attractive property as in the theorem below.

Theorem 3.3. *Consider a node v^* in V with a recursive model $\mathcal{G} = (V, E)$. Then the following statements are equivalent.*

(i) The node v^* is contained in one and only one f-clique in $\mathcal{G}$.

(ii) $clan(v^*)$ becomes a unique clique that includes the node v^* when $pa(v^*)$ is made into a complete subgraph in $\mathcal{G}$.

(iii) The node v^* is removable from $\mathcal{G}$.

Table 3.1: A contingency table for $V = \{1, 2, 3\}$.

x_1	x_2	x_3	$n(x)$
0	0	0	2
0	0	1	3
0	1	0	1
0	1	1	2
1	0	0	3
1	0	1	1
1	1	0	4
1	1	1	4
	Total		20

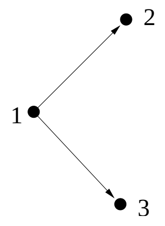


Figure 3.2: A DAG of three nodes where nodes 2 and 3 are removable but node 1 is not.

Proof: See Appendix.

Note that if $ch(v) = \emptyset$, then node v is terminal and so Theorem 3.3 holds for the node. We have shown two equivalent conditions for a node to be removable. We will see two simple examples of removable nodes, the former being simpler than the latter.

Example 3.3. Consider a recursive model $\mathcal{G} = (V, E)$ in Example 3.2. We have that

$$\widehat{P}(x_V) = \frac{[\{1,2\}](x_{\{1,2\}})[\{1,3\}](x_{\{1,3\}})}{[\{1\}](x_{\{1\}})}.$$

In this equation, we can see that nodes 2 and 3 are each contained in one and only one f-clique, but node 1 is contained in both of f-cliques $\{1, 2\}$ and $\{1, 3\}$. Furthermore, nodes 2 and 3 are terminal, and $clan(1) = \{1, 2, 3\}$ cannot be made into a clique in the context of condition (ii) of Theorem 3.3 as is obvious in Figure 3.2. Thus node 1 is not removable while nodes 2 and 3 are. $\square$

Example 3.4. Consider a recursive model with graph $\mathcal{G}$ in panel (a) of Figure 3.3.

$$\widehat{P}(x_V) = \frac{[\{1,2\}](x_{\{1,2\}})[\{1,3\}](x_{\{1,3\}})[\{2,3,4,5\}](x_{\{2,3,4,5\}})[\{2,5,6\}](x_{\{2,5,6\}})}{[\{1\}](x_{\{1\}})[\{2,3\}](x_{\{2,3\}})[\{2,5\}](x_{\{2,5\}})}.$$

Node 4 is contained in the f-clique $\{2, 3, 4, 5\}$ only. Furthermore, if $pa(4) = \{2, 3\}$ is made into a complete subgraph in the graph, node 4 is contained in the unique clique $\{2, 3, 4, 5\}$ in graph $\mathcal{G}$. The

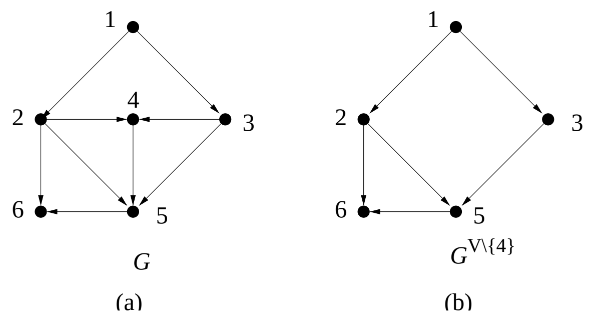


Figure 3.3: A DAG $\mathcal{G}$ of six nodes where nodes 4 and 6 are removable and another DAG where node 4 is removed from $\mathcal{G}$.

marginalized subgraph $\mathcal{G}^{V\setminus\{4\}}$ of $\mathcal{G}$ is in panel (b). Thus we have at $x_{(4)} = x^*_{(4)}$

$$\begin{aligned}
&\sum_{x_{\{4\}}} \widehat{P}(x_V) \\
=\;& \frac{[\{1,2\}](x_{\{1,2\}})[\{1,3\}](x_{\{1,3\}})\left(\sum_{x_{\{4\}}}[\{2,3,4,5\}](x_{\{2,3,4,5\}})\right)[\{2,5,6\}](x_{\{2,5,6\}})}{[\{1\}](x_{\{1\}})[\{2,3\}](x_{\{2,3\}})[\{2,5\}](x_{\{2,5\}})} \\
=\;& \frac{[\{1,2\}](x_{\{1,2\}})[\{1,3\}](x_{\{1,3\}})[\{2,3,5\}](x_{\{2,3,5\}})[\{2,5,6\}](x_{\{2,5,6\}})}{[\{1\}](x_{\{1\}})[\{2,3\}](x_{\{2,3\}})[\{2,5\}](x_{\{2,5\}})} \\
=\;& \widehat{\sum_{x_{\{4\}}} P(x_V)},
\end{aligned}$$

where the last equality is immediate from the marginalized structure $\mathcal{G}^{V\setminus\{4\}}$. We see that the three conditions $(i), (ii), (iii)$ of Theorem 3.3 are satisfied in graph $\mathcal{G}$ in panel (a). Node 4 is thus removable. Since node 6 is terminal, it is also removable. □

The notion of collapsibility is defined in Asmussen and Edwards (1983) and Whittaker (1990). The notion is important for two reasons. One is that it breaks a large domain of model down into relatively small pieces. The second reason is that regression and recursive models are naturally formulated in terms of conditional and marginal distributions. The relationship of the notion of collapsibility to graphical models is described in Asmussen and Edwards (1983). We will denote the moral graph of a recursive model $\mathcal{G} = (V, E)$ by $\mathcal{G}^m = (V, E^m)$.

Definition 3.2. (Whittaker 1990, p. 395) Consider an undirected independent graph $\mathcal{G}^u = (V, E^u)$. We say that $V = (V_1, V_2)$ is *graphically collapsible* over V_2 in $\mathcal{G}^u$ if and only if the boundary (of each connected component) of V_2 is complete in $\mathcal{G}^u$.

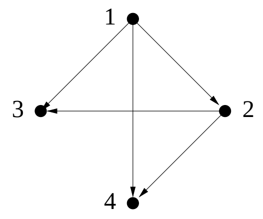


Figure 3.4: A DAG of four nodes

Note that if V is graphically collapsible over V_2 in $\mathcal{G}^u$,

$$\sum_{x_{V_2}} \widehat{P}(x_V) = \widehat{\sum_{x_{V_2}} P(x_V)}$$

in $\mathcal{G}^u$, where the summation goes over $\mathcal{X}_{V_2}$. This equation brings together node-removability and collapsibility on the same footing.

Theorem 3.4. *Let the moral graph of a recursive model $\mathcal{G} = (V, E)$ be $\mathcal{G}^m = (V, E^m)$. If a node v is removable in $\mathcal{G}$, then V is graphically collapsible over v in $\mathcal{G}^m$.*

Proof: If node v is removable, then, by Theorem 3.3, *clan*(v) is the only clique containing node v in $\mathcal{G}$ if *pa*(v) is made into a complete subgraph. Since *clan*(v) becomes a clique in $\mathcal{G}^m$, node v is contained in the clique only. Hence the boundary of a node v is complete in the moral graph, making $\mathcal{G}^m$ graphically collapsible over v. $\square$

Consider a recursive model with graph $\mathcal{G} = (V, E)$. If all the nodes in $A \subseteq V$ can be removed from $\mathcal{G}$ one after another according to some order, we will say that A is *sequentially removable* from $\mathcal{G}$. Therefore, all the nodes which are removable are sequentially removable but not necessarily vice versa. Of course, $\emptyset$ and V are sequentially removable. However, any subset of nodes of a sequentially removable set is not necessarily sequentially removable as we see in the example below. To represent a set of nodes that are sequentially removable, we will use the symbol $\{\cdot\}^{\prec}$. For example, that $\{5, 2, 3\}^{\prec}$ is sequentially removable means that the nodes $2, 3, 5$ are sequentially removable in the order of $5, 2, 3$.

Example 3.5. Consider a recursive model $\mathcal{G} = (V, E)$ with $V = \{1, 2, 3, 4\}$ and $E = \{(1, 2), (1, 3), (1, 4), (2, 3), (2, 4)\}$ as depicted in Figure 3.4. Let $A_1 = \{3, 2\}^{\prec}$ and $A_2 = \{4, 2\}^{\prec}$. Then A_1 and A_2 are sequentially removable, but $A_1 \cap A_2 = \{2\}$ is not. $\square$

We will consider an equivalent condition of sequential removability below.

Theorem 3.5. *Consider a recursive model with graph $\mathcal{G} = (V, E)$. $A \subseteq V$ is sequentially removable* *from $\mathcal{G}$ if and only if*

$$\sum_{x_A} \widehat{P}(x_V) = \widehat{\sum_{x_A} P(x_V)} \tag{3.6}$$

for every possible data set for V in $\mathcal{G}$, where the summation goes over $\mathcal{X}_A$.

Proof: We will prove the "only if" part first. Suppose $A = \{v_1, v_2, \cdots, v_r\}^{\prec}$ is sequentially removable in the order of the indices of v. Then, by Theorem 3.3, we have at $x_{(v_1)} = x^*_{(v_1)}$,

$$\sum_{x_{\{v_1\}}} \widehat{P}(x_V) = \widehat{\sum_{x_{\{v_1\}}} P(x_V)}. \tag{3.7}$$

and proceeding in the same way we have at $x_{\{v_1,v_2\}} = x^*_{\{v_1,v_2\}}$,

$$\begin{aligned}
\sum_{x_{\{v_1,v_2\}}} \widehat{P}(x_V) &= \sum_{x_{\{v_2\}}} \Big(\sum_{x_{\{v_1\}}} \widehat{P}(x_V) \Big) \\
&= \sum_{x_{\{v_2\}}} \Big(\widehat{\sum_{x_{\{v_1\}}} P(x_V)} \Big) \text{ (by (3.7))} \\
&= \widehat{\sum_{x_{\{v_1,v_2\}}} P(x_V)},
\end{aligned}$$

where the last equality follows from the sequential removability. By applying the same reasoning, we can see at $x_{(A)} = x^*_{(A)}$

$$\begin{aligned}
\sum_{x_A} \widehat{P}(x_V) &= \sum_{x_{\{v_r\}}} \cdots \sum_{x_{\{v_1\}}} \widehat{P}(x_V) \\
&= \sum_{x_{\{v_r\}}} \cdots \sum_{x_{\{v_2\}}} \Big(\widehat{\sum_{x_{\{v_1\}}} P(x_V)} \Big) \\
&= \sum_{x_{\{v_r\}}} \cdots \sum_{x_{\{v_3\}}} \Big(\widehat{\sum_{x_{\{v_1,v_2\}}} P(x_V)} \Big) \\
\cdots \\
&= \sum_{x_{\{v_r\}}} \Big(\widehat{\sum_{x_{A\setminus\{v_r\}}} P(x_V)} \Big) \\
&= \widehat{\sum_{x_A} P(x_V)}.
\end{aligned}$$

To prove the "if" part, we let $A = \{v_1, v_2, \cdots, v_r\}^{\prec}$ and assume that equation (3.6) is satisfied for every data set of V. Then there must exist at least one removable node, say v_1. If not, every node in A must be contained in two or more f-cliques in the expression of $\widehat{P_V}(x)$ in (3.2) by Theorem 3.3, making equation (3.6) not guaranteed in general. For a removable node, say v_1, we have at

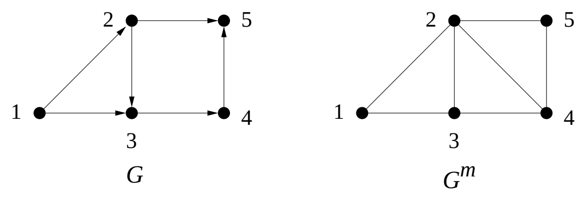


Figure 3.5: A DAG and its moral graph. Nodes 2 and 5 are sequentially removable

$x_{(v_1)} = x^*_{(v_1)}$,

$$\sum_{x_{\{v_1\}}} \widehat{P}(x_V) = \widehat{\sum_{x_{\{v_1\}}} P(x_V)}. \tag{3.8}$$

From equation (3.8), we have at $x_{(A)} = x^*_{(A)}$

$$\begin{aligned} \sum_{x_A} \widehat{P}(x_V) &= \sum_{x_{A\setminus\{v_1\}}} \Big(\sum_{x_{\{v_1\}}} \widehat{P}(x_V) \Big) \\ &= \sum_{x_{A\setminus\{v_1\}}} \Big(\widehat{\sum_{x_{\{v_1\}}} P(x_V)} \Big). \end{aligned} \tag{3.9}$$

Since

$$\widehat{\sum_{x_A} P(x_V)} = \widehat{\sum_{x_{A\setminus\{v_1\}}} \Big(\sum_{x_{\{v_1\}}} P(x_V) \Big)}, \tag{3.10}$$

we have, from (3.9) and (3.10),

$$\sum_{x_{A\setminus\{v_1\}}} \Big(\widehat{\sum_{x_{\{v_1\}}} P(x_V)} \Big) = \widehat{\sum_{x_{A\setminus\{v_1\}}} \Big(\sum_{x_{\{v_1\}}} P(x_V) \Big)}.$$

So, by the same argument as above, $A \setminus \{v_1\}$ has a removable node, say v_2, and by proceeding in the same way, we can remove all nodes in A. This iterative process produces a sequence of removable nodes in A, which complete the proof. □

We will now turn to the relationship between sequential removability and graphical collapsibility.

Example 3.6. Consider a recursive model $\mathcal{G} = (V, E)$ with $V = \{1, 2, 3, 4, 5\}$ and $E = \{(1, 2), (1, 3), (2, 3), (2, 5), (3, 4), (4, 5)\}$ as in Figure 3.5. $A = \{5, 2\}^{\prec}$ is sequentially removable. Since node 5 is terminal, it is removable. Once node 5 is removed, node 2 is removable because it is contained in the clique $\{1, 2, 3\}$ only. The moral graph of $\mathcal{G}$ is $\mathcal{G}^m = (V, E^m)$ with $E^m = sym(E) \cup \{(2, 4), (4, 2)\}$. The boundary of A is $\{1, 3, 4\}$, which is not complete in $\mathcal{G}^m$. Hence V is not graphically collapsible over A in $\mathcal{G}^m$. □

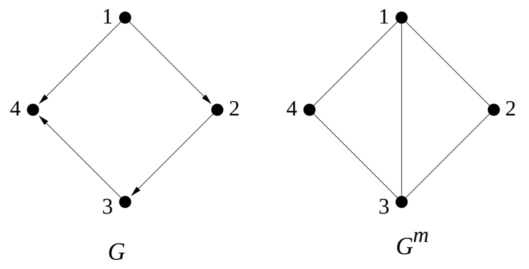


Figure 3.6: A DAG and its moral graph. Nodes 2 and 5 are not sequentially removable

Example 3.7. Consider a recursive model $\mathcal{G} = (V, E)$ with $V = \{1, 2, 3, 4\}$ and $E = \{(1, 2), (2, 3), (1, 4), (3, 4)\}$ as in Figure 3.6. Then $A = \{2, 3\}^{\prec}$ is not sequentially removable since nodes 2 and 3 are not removable. However, $E^m = sym(E) \cup \{(1, 3), (3, 1)\}$ and the boundary of A is complete in $\mathcal{G}^m$. Hence V is graphically collapsible over A in $\mathcal{G}^m$. □

The two examples above show that graphical collapsibility has little to do with sequential removability. That is to say, sequentially removable nodes are not necessarily graphically collapsible nor vice versa.

So far we have considered removability of a set of nodes. This notion and the notion called marginal restructuring constitute an important condition that makes our proposed method work. The marginal restructuring of $\mathcal{G}_i$ as defined below is similar to projecting the model structure $\mathcal{G}$ onto submodel i.

Definition 3.3. Suppose a recursive model $\mathcal{G} = (V, E)$ is d-split into k submodels. The procedure of transforming the induced subgraph $\mathcal{G}_i$ into the marginalized subgraph $\mathcal{G}^i$ is called the *marginal restructuring* for submodel i, $1 \le i \le k$.

For example, consider a recursive model with graph $\mathcal{G} = (V, E)$ in Figure 3.7. If $V = \{1, 2, 3, 4, 5, 6, 7, 8\}$, $E = \{(1, 2), (1, 3), (2, 4), (3, 4), (4, 5), (5, 6), (5, 7), (6, 8), (7, 8)\}$, $V_1 = \{1, 2, 3, 6, 7, 8\}$, and $V_2 = \{2, 3, 4, 5, 6, 7\}$, then, as for V_1, nodes 2 and 6, 2 and 7, 3 and 6, 3 and 7, and 6 and 7 are blocked by node 5, so $E^+(V_1) = \{(2, 6), (2, 7), (3, 6), (3, 7), (6, 7)\}$. As for V_2, since nodes 2 and 3 are blocked by node 1, they must be connected to each other, yielding $E^+(V_2) = \{(2, 3)\}$. Then $E^1 = E_1 \cup E^+(V_1)$, $E^2 = E_2 \cup E^+(V_2)$, $E^{1\wedge 2} = E_{1\wedge 2} \cup E^+(V_1 \cap V_2)$ as is depicted in Figure 3.7. Recall that $E^+(V_1 \cap V_2) = E^+(V_1) \cup E^+(V_2)$ by Theorem 2.3.

Hyper-EM Condition: Let $\mathcal{G} = (V, E)$ be a recursive model which is d-split into k submodels $1, 2, \cdots, k$, and suppose a marginal restructuring is done on every submodel.

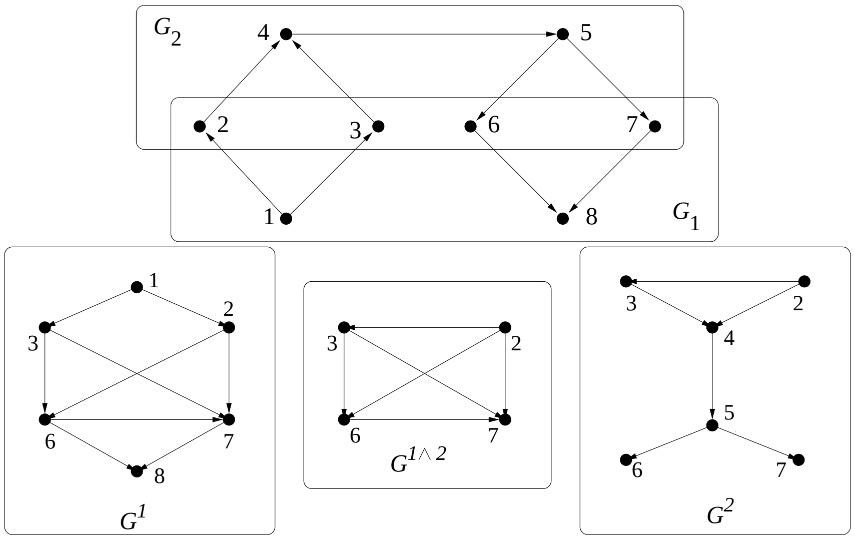


Figure 3.7: An example of the marginal restructuring for $\mathcal{G}_1$ and $\mathcal{G}_2$.

Then for all neighboring submodels i and j with $i < j$, $V_j \setminus V_i$ is sequentially removable from $\mathcal{G}^j$.

Since the Family condition is satisfied for a set of submodels if the submodels are obtained by the d-splitting, the t-splitting under the Family and hyper-EM conditions is the same as the d-splitting under the hyper-EM condition. Therefore, we may well consider d-splitting under the hyper-EM condition only.

4 HYPER-EM GRAPH

Consider two neighboring submodels i and j and assume marginal restructuring is made on them. Let $\widehat{[i]}^{(r_i)}(x^i)$ and $\widehat{[j]}^{(r_j)}(x^j)$ denote the current estimates at the r_ith cycle and r_jth cycle for submodels i and j, respectively, $[i]_j(x^{i\wedge j})$ and $[j]_i(x^{i\wedge j})$ denote the cell-means for $V_i \cap V_j$ that are computed from $[i](x^i)$ and $[j](x^j)$, respectively, and $\widehat{[i]}_j(x^{i\wedge j})$ and $\widehat{[j]}_i(x^{i\wedge j})$ the corresponding estimates. Denote the E-step operation on submodel i by $\epsilon(i)$, the likelihood maximization for submodel i by $\mu(i)$, and the estimate-transfusion(ET) from submodel i into submodel j by $\tau(i,j)$. The formula of the transfusion, $\tau(i,j)$, is given by, assuming that an $\epsilon(i)$(or $\mu(i)$) is done at the r_ith cycle for submodel i,

$$\widehat{[j]}^{(r_j+1)}(x^j) = \widehat{[i]}_j^{(r_i)}(x^{i\wedge j}) \frac{\widehat{[j]}^{(r_j)}(x^j)}{\widehat{[j]}_i^{(r_j)}(x^{i\wedge j})}.$$

We have at $x^{i\wedge j}$,

$$\begin{aligned}\widehat{[i]}_j^{(r_i)}(x^{i\wedge j}) &= \sum_{x_{V_i\setminus V_j}} \widehat{[i]}^{(r_i)}(x) \quad \text{and} \\ \widehat{[j]}_i^{(r_j)}(x^{i\wedge j}) &= \sum_{x_{V_j\setminus V_i}} \widehat{[j]}^{(r_j)}(x).\end{aligned}$$

Let $\widehat{[i_j]}^{(r_i+1)}(x^{i\wedge j})$ and $\widehat{[j_i]}^{(r_j+1)}(x^{i\wedge j})$ denote the estimates for $V_i \cap V_j$ that are obtained as follows.

$$\begin{aligned}\widehat{[i_j]}^{(r_i+1)}(x^{i\wedge j}) &= \prod_{v\in V_i\cap V_j}\{\sum_{x_{V_i\setminus V_j}} \widehat{[i]}^{(r_i)}(x)\}(x_v|x_{pa(v)\cap V_i\cap V_j}) \\ &= \prod_{v\in V_i\cap V_j}\{\widehat{[i]}_j^{(r_i)}(x^{i\wedge j})\}(x_v|x_{pa(v)\cap V_i\cap V_j}) \qquad (4.1)\end{aligned}$$

and

$$\begin{aligned}\widehat{[j_i]}^{(r_j+1)}(x^{i\wedge j}) &= \prod_{v\in V_i\cap V_j}\{\sum_{x_{V_j\setminus V_i}} \widehat{[j]}^{(r_j)}(x)\}(x_v|x_{pa(v)\cap V_i\cap V_j}) \\ &= \prod_{v\in V_i\cap V_j}\{\widehat{[j]}_i^{(r_j)}(x^{i\wedge j})\}(x_v|x_{pa(v)\cap V_i\cap V_j}). \qquad (4.2)\end{aligned}$$

Notice the difference between $\widehat{[i]}_j^{(r_i)}(x^{i\wedge j})$ and $\widehat{[i_j]}^{(r_i)}(x^{i\wedge j})$. The former is a marginal of the estimates onto $V_i \cap V_j$ in $\mathcal{G}^i$, and the latter is an estimate in $\mathcal{G}^{i\wedge j}$ which is obtained by applying likelihood maximization to the marginalization of V_i onto $V_i \cap V_j$.

Theorem 4.1. *Suppose a recursive model with graph $\mathcal{G}$ is d-split and the marginal restructuring is done on all the submodels. If submodels i and j are neighbors, then we have*

$$\widehat{[i]}_j^{(r_i)}(x) = \widehat{[j]}_i^{(r_j+1)}(x) \text{ for } x \in \mathcal{X}_{V_i\cap V_j} \qquad (4.3)$$

if and only if

$$\widehat{[j]}_i^{(r_j+1)}(x) = \widehat{[j_i]}^{(r_j+1)}(x) \text{ for } x \in \mathcal{X}_{V_i\cap V_j}, \qquad (4.4)$$

where $\mu(i)$, $\tau(i,j)$, and $\mu(j)$ are carried out in a row and the resulting estimates are superscribed respectively by (r_i), (r_j), and (r_j+1).

Proof: First, we will prove the necessity of the theorem, i.e., equation (4.4). By the marginal restructuring in the submodels and the transfusion $\tau(i,j)$, we have

$$\widehat{[i]}_j^{(r_i)}(x) = \widehat{[j]}_i^{(r_j)}(x) \text{ for } x \in \mathcal{X}_{V_i\cap V_j}. \qquad (4.5)$$

For $x \in \mathcal{X}_{V_i \cap V_j}$,

$$\begin{aligned}
\widehat{[i]}_j^{(r_i)}(x) &= \prod_{v \in V_i \cap V_j} \{\widehat{[i]}_j^{(r_i)}(x)\}(x_v | x_{pa(v) \cap V_i \cap V_j}) \\
&= \prod_{v \in V_i \cap V_j} \{\widehat{[j]}_i^{(r_j)}(x)\}(x_v | x_{pa(v) \cap V_i \cap V_j}) \quad \text{(by (4.5))} \\
&= \widehat{[j_i]}^{(r_j+1)}(x), \qquad (4.6)
\end{aligned}$$

where the last equation follows by likelihood maximization. So, by (4.3), we have the desired result (4.4). The proof for the other direction is immediate from equations (4.5) and (4.6). □

Consider a simple situation where model $\mathcal{G}$ is d-split into two neighboring submodels 1 and 2 only. The M-step is then to be implemented as follows:

(1) $\mu(1) \to \tau(1,2) \to \mu(2) \to \tau(2,1)$

(2) Check whether the estimates for $1 \wedge 2$ are equal before and after step (1)

(3) If the equality holds in step (2), stop the M-step. Otherwise return to step (1).

However, if equation (4.3) or (4.4) is satisfied with the submodels, the M-step is simplified down to $\mu(1) \to \tau(1,2) \to \mu(2)$. Theorem 4.1 provides a sufficient condition for simplifying an M-step.

Assume that a recursive model with graph $\mathcal{G}$ is d-split and the marginal restructuring is done on all the submodels. Consider neighboring submodels i and j. If

$$\widehat{[j]}_i^{(r_j)}(x) = \widehat{[j_i]}^{(r_j)}(x) \text{ for } x \in \mathcal{X}_{V_i \cap V_j}, \qquad (4.7)$$

then we will say that the *hyper-EM works from submodel i to submodel j* where $\widehat{[j]}^{(r_j)}$ is an estimate of the cell probability for submodel j obtained by likelihood maximization. In other words, if the hyper-EM works, the MLE of the intersection of two neighboring submodels is the same whether marginalization takes place after or before the likelihood-maximization.

At a glance, expression (4.7) seems having much to do with collapsibility or node-removability. But we need to keep in mind that estimates are transfused between neighboring submodels. When $\tau(i,j)$ takes place, it is desirable that (4.3) holds which is possible when and only when the marginalization of $\mathcal{G}_i$ onto $i \wedge j$ is the same as that of $\mathcal{G}_j$ onto $i \wedge j$ and $(i \wedge j)^C$ is sequentially removable from $\mathcal{G}_j$. Collapsibility and node-removability are defined on a submodel while expression (4.7) is to be understood in the context of a tree of submodels.

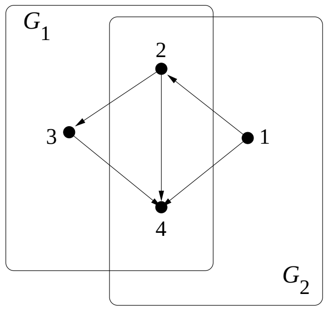


Figure 4.1: A hyper-EM tree.

Suppose $\mathcal{G} = (V, E)$ is d-split into a tree of k submodels. If the hyper-EM works from submodel i to submodel j for all neighboring submodels i and j with $1 \leq i < j \leq k$, then we will call the tree a *right hyper-EM tree* of submodels and if the hyper-EM works from submodel j to submodel i for all neighboring submodels i and j with $1 \leq i < j \leq k$, then we will call the tree a *left hyper-EM tree* of submodels. Furthermore, if the tree is both left hyper-EM and right hyper-EM, we will call the tree a *hyper-EM tree.* Figure 4.1 is an example of d-splitting that leads to a hyper-EM tree of submodels 1 and 2. The following theorem summarizes the relationship between sequential removability and hyper-EM tree.

Theorem 4.2. *Suppose a recursive model with graph $\mathcal{G}$ is d-split and the marginal restructuring is done on all the submodels. Then for neighboring submodels i and j, $V_j \setminus V_i$, $i < j$, is sequentially removable from $\mathcal{G}^j$ if and only if the hyper-EM works from submodel i to submodel j.*

Proof: According to Theorem 3.5, the sequential removability of $V_j \setminus V_i$ from $\mathcal{G}^j$ is that at $x^j_{(V_j \setminus V_i)} = x^{j*}_{(V_j \setminus V_i)}$

$$\sum_{x^j_{V_j \setminus V_i}} \widehat{[j]}(x^j) = \widehat{\sum_{x^j_{V_j \setminus V_i}} [j](x^j)},$$

i.e.,

$$\widehat{[j]}_i^{(r_j)}(x) = \widehat{[j_i]}^{(r_j)}(x) \text{ for } x \in \mathcal{X}_{V_i \cap V_j},$$

which means, by definition, that the hyper-EM works from submodel i to submodel j. This completes the proof □

The corollary below is immediate from Theorem 4.2 by the definition of the hyper-EM condition.

Corollary 4.1. *Assume the condition of Theorem 4.2. Then all the submodels satisfy the hyper-EM condition if and only if the submodels constitute a right hyper-EM tree.*

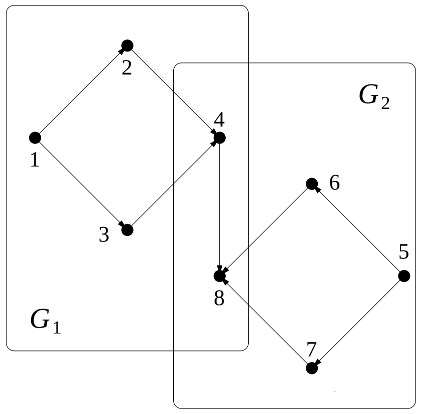


Figure 4.2: A DAG $\mathcal{G}$ for which the hyper-EM does not work.

If a recursive model is d-split but not necessarily under the hyper-EM condition, we can think of an example where Theorem 4.2 does not hold.

Example 4.1. The model considered in Figure 4.2 is d-split into two submodels 1 and 2. Note that $\mathcal{G}_1 = \mathcal{G}^1$ and $\mathcal{G}_2 = \mathcal{G}^2$. Then

$$\widehat{[1]}(x^1) = \frac{[\{1,2\}](x^1_{\{1,2\}})[\{1,3\}](x^1_{\{1,3\}})[\{2,3,4\}](x^1_{\{2,3,4\}})[\{4,8\}](x^1_{\{4,8\}})}{[\{1\}](x^1_{\{1\}})[\{2,3\}](x^1_{\{2,3\}})[\{4\}](x^1_{\{4\}})}$$

and

$$\widehat{[2]}(x^2) = \frac{[\{4\}](x^2_{\{4\}})[\{5,6\}](x^2_{\{5,6\}})[\{5,7\}](x^2_{\{5,7\}})[\{4,6,7,8\}](x^2_{\{4,6,7,8\}})}{[\{5\}](x^2_{\{5\}})[\{4,6,7\}](x^2_{\{4,6,7\}})}.$$

Since each of nodes $1,2,3,5,6,7$ is contained in two or more f-cliques, these nodes do not satisfy the hyper-EM condition. In other words, $V_1 \setminus V_2$ and $V_2 \setminus V_1$ do not satisfy condition (ii) of Theorem 3.3. Thus the graph $\mathcal{G}$ in Figure 4.2 with submodels 1 and 2 does not make a hyper-EM tree. $\square$

To sum up this section, it is desirable that a recursive model is d-split under the hyper-EM condition so that the resulting submodel-arrangement is a right hyper-EM tree. Furthermore, if the submodel-arrangement turns out a hyper-EM tree, the M-step does not need the ET between neighboring submodels, as will be described in detail in next section.

5 ML ESTIMATION FOR A TREE OF SUBMODELS

If a set of neighboring submodels share a set S of variables and the estimates for the variables in S are the same whether they are obtained from one of the neighboring submodels or another, then we say that *consistency of distribution* (CD) holds at S. The notion of CD is similar to the notion of consistency of distributions as described in Dawid and Lauritzen (1993) in the sense that the former notion is of estimates while the latter notion is of distributions. Jensen, Olesen, and Andersen (1990) also introduced the notion of consistency for a graphical model using a belief measure on it.

EM is an algorithm that consists of two steps, expectation (E) step and likelihood-maximization (M) step. When the probability model is multinomial, we compute the conditional mean of the missing values in the E-step conditional on data, and we maximize the likelihood of the given model in the M-step.

When a model contains too many variables to handle at once, an alternative is that we split the model into several submodels of moderate sizes as we have discussed so far and then apply the EM to individual submodels with some additional elaboration that is supplementary to the localized EM on individual submodels.

The additional elaboration is aimed to have the probability distribution consistent throughout the whole model and to have the information from data which is absorbed into the estimates as much as possible. The whole model is split and so is the data set accordingly.

An important issue here is that we need to modify the E- and M-steps so that the information from subsets of data may permeate throughout the whole model without destroying the whole model structure. In the following two subsections, we will discuss further on this issue.

5.1 E-step of the Hyper-EM Algorithm

Suppose k submodels are labelled 1 through k. Then, without loss of generality, we may express the joint probability for the whole model as in

$$P(x) = \prod_{i=1}^{k} P_{i|bd(i)}(x^i|x_{bd(i)}) \tag{5.1}$$

where $bd(1) = \emptyset$.

Note that data are given for individual submodels and so we can apply the IPF method (Bishop, Fienberg, and Holland 1975) to calibrate the estimates to data and the ET method to obtain the CD of estimates. Convergence of the IPF method is well established (Birch 1963; Bishop, Fienberg, and Holland 1975) and the ET into neighboring submodels makes the estimates obtain the CD.

Once an E-step is implemented on submodel i, the resulting estimates are transfused into neighboring submodels and the transfusion continues until the ET takes place into all the submodels

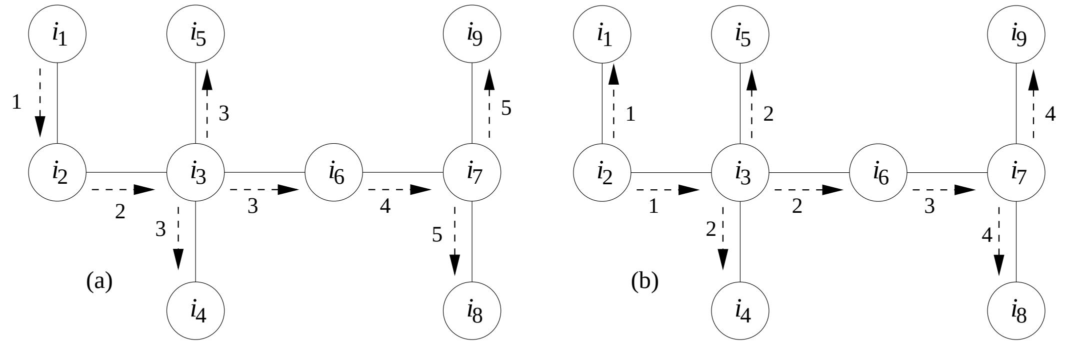


Figure 5.1: A tree of submodels where dotted arrows indicate the ET from (a) submodel i_1 and from (b) submodel i_2. The numbers on the arrows indicate the order of ET. The arrows with the same number on them mean that any ordering among them will do.

except submodel i. For example, if the submodels are arranged as in Figure 5.1 and the E-step is implemented on submodel i_1, then the subsequent ETs into all the submodels proceed as indicated in panel (a) of Figure 5.1. That is,

$$\begin{aligned}&\epsilon(i_1) \to^1 \tau(i_1, i_2) \to^2 \tau(i_2, i_3) \to^3 \tau(i_3, i_4) \to^3 \tau(i_3, i_5) \to^3 \tau(i_3, i_6) \to^4 \tau(i_6, i_7) \\ &\to^5 \tau(i_7, i_8) \to^5 \tau(i_7, i_9), \qquad (5.2)\end{aligned}$$

where the numbers on the arrows have the same meaning as those in Figure 5.1. If the E-step is implemented on submodel i_2, then the subsequent ETs proceed as indicated in panel (b) of Figure 5.1. A general rule is that the ET sweeps through the tree along every possible chain of nodes starting from the submodel where an E-step is implemented. This is to have the estimates of the whole model updated by data. As a matter of fact, this thorough sweeping following an E-step on a submodel can be cut short while making the iterative procedure of E-step-then-ET work for the whole model. This will be described below. For convenience, we will call by SEET (short for Submodel E-step and Estimate-Transfusion) one E-step on a submodel followed by subsequent ETs into all the other submodels.

Let δ be the index set of the observed variables for a given model and denote by δ_i the index set of the observed variables that are involved in submodel i. If a tree of submodels consists of k nodes, then it is possible that $\delta_i = \emptyset$ for some i, $1 \le i \le k$. As for the tree in Figure 5.1, we assume $\delta_i \ne \emptyset$, $i = 1, 2, \cdots, 9$. This means there are 9 subsets of data represented by δ_i, $i = 1, 2, \cdots, 9$. The iterative SEET procedure may be executed in any order of δ_i's as long as all the δ_i's are used in a row. Denote by $\epsilon(i)$ an E-step on submodel i based on δ_i.

A best way of doing the iterative SEET procedure is to find a sequence of $\epsilon(i)$'s so that the ET is implemented as little as possible. A reasonable solution for the tree in Figure 5.1 is given in Figure

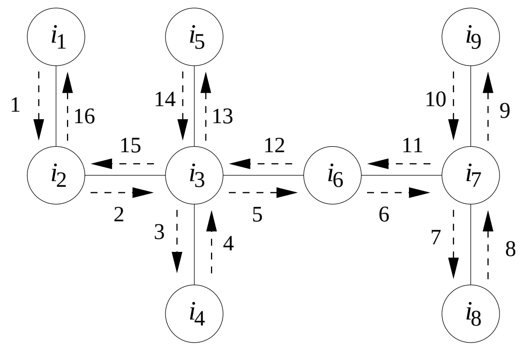


Figure 5.2: A tree of submodels where dotted arrows indicate the flow of the iterative SEET procedure starting from submodel i_1

5.2. It is important to note that once an E-step is done at a submodel, the subsequent ET may have to continue only up to the submodel where next SEET takes place.

The iterative SEET as displayed in Figure 5.2 is optimal in the sense that an ET with a given direction of transfusion is made once and only once between every pair of neighboring submodels. Note however that $\epsilon(i)$ is implemented as many times as the number of the neighboring submodels of submodel i no matter where the whole iterative SEET procedure begins. The iterative SEET as displayed in Figure 5.2 is re-expressed below:

$$\begin{aligned}
&\epsilon(i_1) \to^1 \tau(i_1,i_2) \to^1 \epsilon(i_2) \to^2 \tau(i_2,i_3) \to^2 \epsilon(i_3) \to^3 \tau(i_3,i_4) \to^3 \epsilon(i_4) \\
&\to^4 \tau(i_4,i_3) \to^4 \epsilon(i_3) \to^5 \tau(i_3,i_6) \to^5 \epsilon(i_6) \to^6 \tau(i_6,i_7) \to^6 \epsilon(i_7) \\
&\to^7 \tau(i_7,i_8) \to^7 \epsilon(i_8) \to^8 \tau(i_8,i_7) \to^8 \epsilon(i_7) \to^9 \tau(i_7,i_9) \to^9 \epsilon(i_9) \\
&\to^{10} \tau(i_9,i_7) \to^{10} \epsilon(i_7) \to^{11} \tau(i_7,i_6) \to^{11} \epsilon(i_6) \to^{12} \tau(i_6,i_3) \to^{12} \epsilon(i_3) \\
&\to^{13} \tau(i_3,i_5) \to^{13} \epsilon(i_5) \to^{14} \tau(i_5,i_3) \to^{14} \epsilon(i_3) \to^{15} \tau(i_3,i_2) \to^{15} \epsilon(i_2) \\
&\to^{16} \tau(i_2,i_1). \qquad (5.3)
\end{aligned}$$

We denote by $SEET(i)$ a SEET procedure which begins with $\epsilon(i)$ followed by ETs throughout a given tree of submodels. The estimates of cell means resulting from a $SEET(i)$ can be expressed as

$$n\widehat{P}(x)^{(r+1)} = n(x_{\delta_i})\widehat{P}(x|x_{\delta_i})^{(r)}.$$

As described above, the procedure (5.3) has the some effect as the procedure

$$\begin{aligned}
&SEET(i_1) \to SEET(i_2) \to SEET(i_3) \to SEET(i_4) \to SEET(i_3) \to SEET(i_6) \\
&\to SEET(i_7) \to SEET(i_8) \to SEET(i_7) \to SEET(i_9) \to SEET(i_7) \to SEET(i_6) \\
&\to SEET(i_3) \to SEET(i_5) \to SEET(i_3) \to SEET(i_2).
\end{aligned}$$

By the way, this procedure is an IPF procedure (Bishop, Fienberg, and Holland 1975) which is well known to converge. So the iterative SEET procedure such as in (5.3) converges, which guarantees the CD of the resulting estimates.

Note that since the IPF and the iterative SEET procedures calibrate the estimates to data, we do not consider the structure of a model, and so the hyper-EM condition has nothing to do with the iterative SEET.

5.2 M-step of the Hyper-EM Algorithm

When the iterative SEET procedure converges, we obtain the CD. This however does not necessarily mean unnecessariness of the ET for the M-step. Consider neighboring submodels i and j and suppose that MLEs are obtained for each of them. The point here is whether the MLEs, $[\hat{i}]_j$ and $[\hat{j}]_i$, are the same. If this equality holds for every pair of neighboring submodels, we do not need the ET and the M-step is done simply by the likelihood-maximization on individual submodels. Otherwise, we need the ET. In the theorem below, we will see a sufficient and necessary condition for a successful M-step without the ET.

Theorem 5.1. *Suppose $\mathcal{G} = (V, E)$ is d-split into a tree of k submodels under the hyper-EM condition and the marginal restructuring is done on all the submodels. If the iterative SEET converges and submodels i and j, $i < j$, are neighbors, then*

$$\widehat{[i]}_j^{(r_i)}(x) = \widehat{[j]}_i^{(r_j)}(x) \text{ for } x \in \mathcal{X}_{V_i \cap V_j} \tag{5.4}$$

if and only if

$$\widehat{[i]}_j^{(r_i)}(x) = \widehat{[i_j]}^{(r_i)}(x) \text{ for } x \in \mathcal{X}_{V_i \cap V_j} \tag{5.5}$$

where $\widehat{[i]}^{(r_i)}$ is the estimate for submodel i obtained by likelihood maximization on the estimates resulting from the preceding iterative SEET, and analogously for $\widehat{[j]}^{(r_j)}$.

Proof: We denote by $\widehat{[i]}^{(r_i-1)}$ the estimate for submodel i that are resulting from the preceding iterative SEET and the same for $\widehat{[j]}^{(r_j-1)}$. Since the iterative SEET converges,

$$\widehat{[i]}_j^{(r_i-1)}(x) = \widehat{[j]}_i^{(r_j-1)}(x) \quad \text{for } x \in \mathcal{X}_{V_i \cap V_j}. \tag{5.6}$$

Furthermore, the d-splitting is carried out so that the hyper-EM condition is satisfied. So, by Theorem 4.2, the hyper-EM works from submodel i to submodel j, that is,

$$\widehat{[j]}_i^{(r_j)}(x) = \widehat{[j_i]}^{(r_j)}(x). \tag{5.7}$$

Hence, for $x \in \mathcal{X}_{V_i \cap V_j}$

$$\begin{aligned}
\widehat{[j]}_i^{(r_j)}(x) &= \widehat{[j_i]}^{(r_j)}(x) && (\text{ by } (5.7)\ ) \\
&= \prod_{v \in V_i \cap V_j} \{\widehat{[j]}_i^{(r_j-1)}(x)\}(x_v | x_{pa(v) \cap V_i \cap V_j}) && (\text{ by } (4.2)\ ) \\
&= \prod_{v \in V_i \cap V_j} \{\widehat{[i]}_j^{(r_i-1)}(x)\}(x_v | x_{pa(v) \cap V_i \cap V_j}) && (\text{ by } (5.6)\ ) \\
&= \widehat{[i_j]}^{(r_i)}(x). && (\text{ by } (4.1)\ )
\end{aligned}$$

From this result and (5.4) follows (5.5). As for the other direction of the proof, we can easily see that (5.4) follows from (5.5) and the last equation. □

If expression (5.5) holds for all the pairs of neighboring submodels, the set of the k submodels constitutes a left hyper-EM tree. Hence, if a d-splitting under the hyper-EM condition gives rise to a left hyper-EM tree, then according to Theorem 5.1, we do not need ET in the M-step. Recall that when a d-splitting is made under the hyper-EM condition, the resulting tree of submodels becomes a right hyper-EM tree.

If the set of the k submodels does not constitute a left hyper-EM tree, we need the ET between neighboring submodels in the M-step, and so we must show the convergence of the M-step which consists of the likelihood-maximization on submodels and the ET between neighboring submodels. The ET is made between likelihood-maximizations on submodels in the same way as in the iteration SEET. We will call this M-step an *M-step with ET* and the M-step in the preceding paragraph an *M-step without ET*. Kim (2000) proved theorems (Theorems 5.5 and 5.6 thereof) to the effect that the M-step with ET converges as long as the submodels are d-split under the hyper-EM condition.

6 SIMULATION EXPERIMENT

We will apply the proposed EM (we will call it hyper-EM) algorithm to recursive models of categorical variables whose model structures are presented in three arrangements of submodels, a series-mode arrangement, a tree of submodels with one branching node, and a tree of submodels with multiple branching-nodes. The latter tree is a general situation of submodel-arrangement. So we will first see how the hyper-EM works on two simple arrangements of submodels and then on the latter tree-shape arrangement.

All the variables considered are binary, taking on values 0 or 1, and the d-splitting is done under the hyper-EM condition. In all the graphs in this section, bullets stand for observed variables and

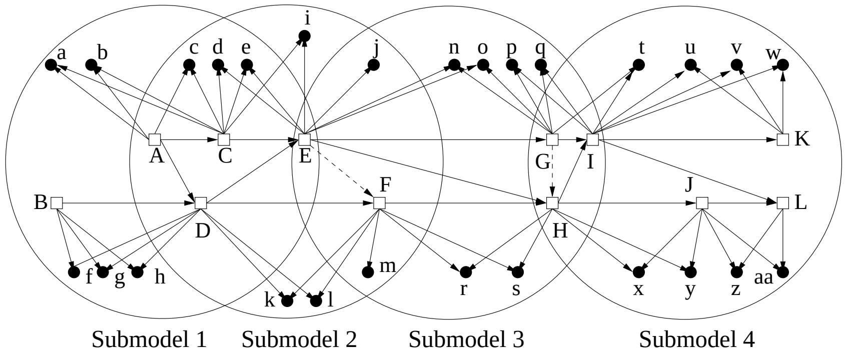


Figure 6.1: A series-mode arrangement where dotted arrows are the ones that are added in the marginal restructuring.

Table 6.1: The goodness-of-fit levels for the series-mode arrangement as in Figure 6.1.

Submodel(d.f.)	1(211)	2(216)	3(214)	4(208)
χ^2	205.67	207.37	206.06	243.15
P-value	0.5906	0.6512	0.6392	0.0476

Sample size = 2,000; Threshold = 0.1.

empty boxes for latent variables. The observed variables are labelled in the lower case and the latent variables in the upper case. If a model involves a large number of variables as in Figure 6.3, the variables are labelled by numbers.

Kim (2002) proposed a method for generating initial values for an EM that are calibrated to data and showed that the initials end up with estimates that are in general better than the initials that are generated in a random manner. We used these calibrated initial values in the simulation experiment.

6.1 Series-mode Arrangement

The first model that we consider for simulation is in a series-mode submodel arrangement as in Figure 6.1. The model is of 39 variables with 12 latent variables and 27 observed variables. The variables are connected by 67 arrows.

The model is d-split into 4 submodels under the hyper-EM condition. Note that $\mathcal{G}_1 = \mathcal{G}^1$ and $\mathcal{G}_2 = \mathcal{G}^2$. However, the marginal restructuring upon submodels 3 and 4 ends up with $E^3 = E_2 \cup \{(E, F)\}$ and $E^4 = E_4 \cup \{(G, H)\}$. The dotted arrows in Figure 6.1 are a required addition from the marginal restructuring. Out of the 39 nodes, the graph has 2 root nodes giving rise to 2

Table 6.2: The estimates of the marginal probabilities of the latent variables in Figure 6.1.

Node(X)	A	B	C	D	E	F
$P(X=1)$	0.85	0.85	0.74	0.65	0.52	0.60
$\widehat{P}(X=1)$	0.70	0.70	0.73	0.63	0.52	0.63

Node(X)	G	H	I	J	K	L
$P(X=1)$	0.52	0.43	0.37	0.45	0.41	0.33
$\widehat{P}(X=1)$	0.53	0.43	0.39	0.43	0.48	0.38

Sample size = 2,000; Threshold = 0.1.

parameters, 7 nodes each of which has 1 parent node giving rise to $2 \times 7 = 14$ parameters, 30 nodes each of which has 2 parent nodes giving rise to $4 \times 30 = 120$ parameters. Thus the total number of parameters is 136. If we consider the whole model as one, it means we have to compute the estimates of the cell means for the $2^{27} = 134,217,728$ (about 134 million) cells. Dealing with this model as one may lead us nowhere.

However, if we d-split the whole model under the hyper-EM condition into four submodels as in Figure 6.1, we can obtain reasonably good estimates for the model. The total numbers of parameters for the four submodels 1, 2, 3, and 4 are 44, 39, 41, and 46, respectively, with the corresponding degrees of freedom, 211, 216, 214, and 208.

The goodness-of-fit levels for the four submodels are given in Table 6.1, which is obtained based on simulated data of size 2,000 for each submodel and the stopping threshold was 0.1 for the estimates of the cell means. The p-values of the goodness-of-fit test are about 0.6 for the first three submodels and it is 0.05 for the last submodel. Since the estimates are affected by the initial values used, trying some more initial values may yield higher p-values for the last submodel.

The estimates for the marginal probabilities of the latent variables are given in Table 6.2. The node labels in the table are as in Figure 6.1. Except the first two estimates, the estimates look very close to the actual values. Although they are not tabulated, the estimates of all the marginal or conditional probabilities of the variables in the submodels look good in general located close to the actual values.

6.2 A Tree of Submodels with A Single Branching Node

The second model that we consider for simulation is given in panel (a) of Figure 6.2. The model is of 34 variables with 10 latent variables and 24 observed variables, and the variables are connected by 54 arrows. We d-split the model into four submodels under the hyper-EM condition that are

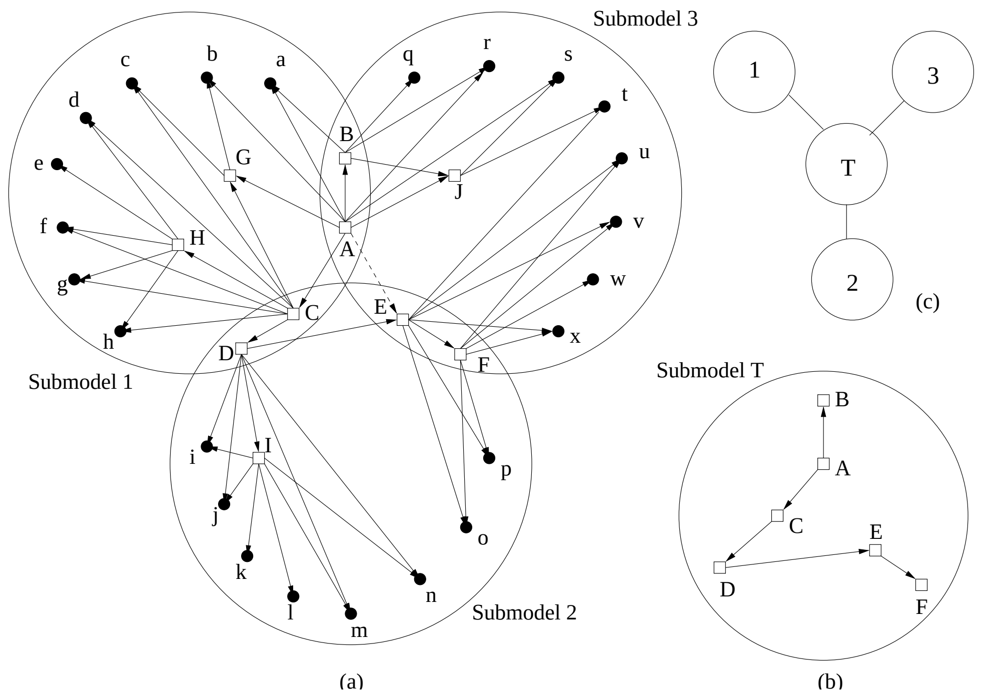


Figure 6.2: A ring-mode arrangement (a) that turns into a tree (c) of submodels with a single branching node where submodel T (b) is taken as a branching node. In panel (a), a dotted arrow is added as a result of the marginal restructuring on submodel 3.

Table 6.3: The goodness-of-fit levels for the tree of submodels with a single branching node.

Submodel(d.f.)	1(213)	2(220)	3(216)
χ^2	197.74	241.73	248.67
P-value	0.7658	0.1503	0.0630

Sample size = 3,000; Threshold = 0.01.

arranged in a tree of submodels with a single branching node as in Figure 6.2. The dotted arrow in the figure is an addition from the marginal restructuring on submodel 3 of the whole model.

Note that $\mathcal{G}_1 = \mathcal{G}^1$, $\mathcal{G}_T = \mathcal{G}^T$, and $\mathcal{G}_2 = \mathcal{G}^2$. However, the marginal restructuring upon submodels 3 ends up with $E^3 = E_3 \cup \{(A, E)\}$. Out of the 34 nodes, the graph has 1 root node giving rise to 1 parameter, 12 nodes each of which has 1 parent node giving rise to $2 \times 12 = 24$ parameters, 21 nodes each of which has 2 parent nodes giving rise to $4 \times 21 = 84$ parameters. Thus the total number of parameters is 109. If we consider the whole model as one, it means we have to compute the estimates of the cell means for the $2^{24} = 16,777,216$ (about 17 million) cells.

The total numbers of parameters for the three submodels 1, 2, and 3 are 42, 35, and 39, respec-

Table 6.4: The estimates of the marginal probabilities of the latent variables in Figure 6.2.

Node(X)	A	B	C	D	E	F	G	H	I	J
$P(X=1)$	0.85	0.75	0.66	0.75	0.66	0.67	0.67	0.62	0.62	0.58
$\widehat{P}(X=1)$	0.88	0.76	0.69	0.70	0.58	0.61	0.61	0.74	0.62	0.75

Sample size = 3,000; Threshold = 0.01.

tively, with the corresponding degrees of freedom, 213, 220, and 216. Submodel T does not contain any observed variables.

The goodness-of-fit levels for the three submodels are given in Table 6.3, which is obtained based on simulated data of size 3,000 for each submodel and the stopping threshold was 0.01 for the estimates of the cell means. The p-values of the goodness-of-fit test are about 0.76 for the first submodel, 0.15 for the second and it is 0.06 for the last submodel.

The estimates for the marginal probabilities of the latent variables are given in Table 6.4. The node labels in the table are as in Figure 6.2. Except for node J, the estimates are within 0.1 of the actual values.

6.3 A Tree of Submodels With Multiple Branching Nodes

The last model that we consider for simulation is of 158 variables with 46 latent variables and 112 observed variables. The variables are connected by 270 arrows. We d-split the model under the hyper-EM condition into 18 submodels among which four are used as branching nodes in the tree of submodels. The whole model along with its submodel arrangement is given in Figure 6.3, and its tree-shape arrangement is given in Figure 6.4. Note in the latter figure that four branching nodes, $T_1, \cdots, T_4$, are involved.

The marginal restructuring upon submodels gives rise to several additional arrows as indicated in Figure 6.3. $E^1 = E_1 \cup \{(1,2)\}$, $E^3 = E_3 \cup \{(7,11)\}$, $E^5 = E_5 \cup \{(13,14)\}$, $E^7 = E_7 \cup \{(21,25)\}$, $E^9 = E_9 \cup \{(23,24)\}$, and $E^{12} = E_{12} \cup \{(27,28)\}$. For all the submodels except these submodels $1,3,5,7,9,12$, we have $\mathcal{G}_i = \mathcal{G}^i$. Out of the 158 nodes, the whole model has 2 root nodes giving rise to 2 parameters, 42 nodes each of which has 1 parent node giving rise to $2 \times 42 = 84$ parameters, 114 nodes each of which has 2 parent nodes giving rise to $4 \times 114 = 456$ parameters. Thus the total number of parameters is 542. If we consider the whole model as one, the total number of the cells is equal to $2^{112} \doteqdot 5.19 \times 10^{33}$ (about 5 million times one trillion times one trillion!). The d-splitting however saves us from this enormous computational burden. The total numbers of parameters for

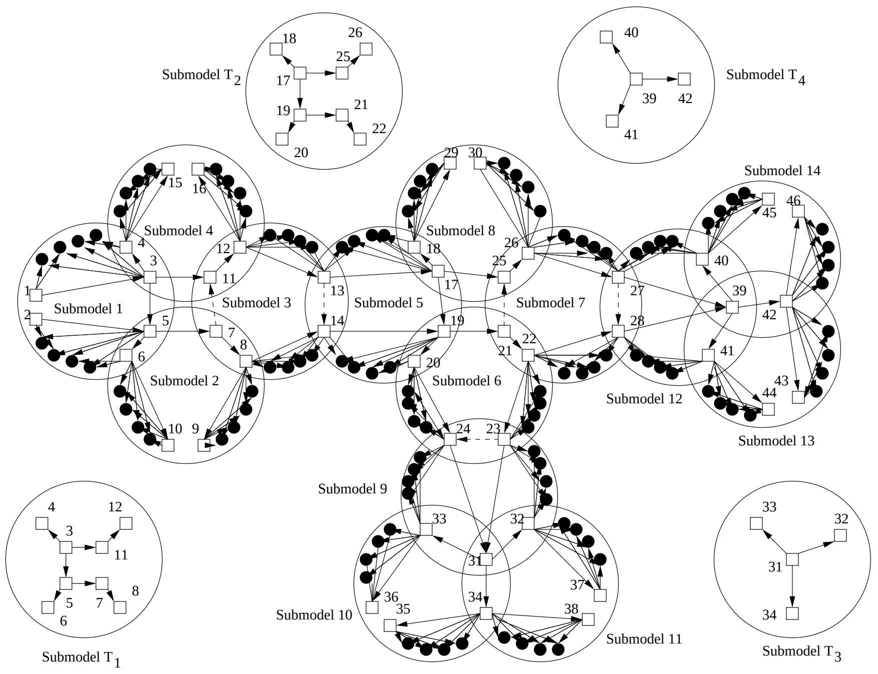


Figure 6.3: A recursive model of 158 variables which is d-split into 18 submodels under the hyper-EM condition. Submodels T_1, T_2, T_3, and T_4 are placed as branching nodes in the tree of the 18 submodels in Figure 6.4.

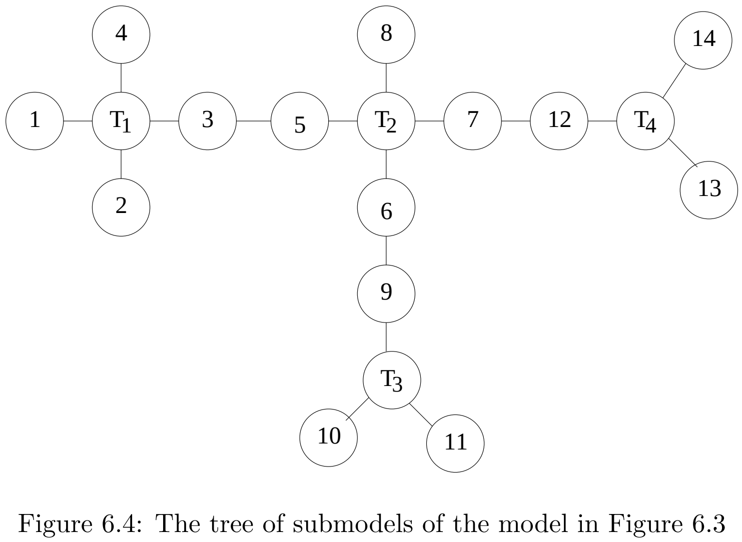


Figure 6.4: The tree of submodels of the model in Figure 6.3

Table 6.5: The goodness-of-fit levels for a tree of submodels with multiple branching nodes.

Submodel(d.f.)	1(213)	2(212)	3(212)	4(212)
χ^2	262.5	191.7	257.3	209.1
P-value	0.012	0.838	0.018	0.543

Submodel(d.f.)	5(212)	6(212)	7(212)	8(212)
χ^2	238.3	250.1	212.5	227.3
P-value	0.104	0.037	0.477	0.224

Submodel(d.f.)	9(212)	10(214)	11(214)	
χ^2	239.8	202.5	200.2	
P-value	0.093	0.704	0.742	

Submodel(d.f.)	12(212)	13(214)	14(214)	
χ^2	240.7	218.6	251.9	
P-value	0.086	0.400	0.039	

Sample size = 100,000; Threshold = 1.0.

Table 6.6: The estimates of the marginal probabilities of the latent variables in Figure 6.3.

Node(X)	1	2	3	4	5	6	7	8	9	10	11	12
$P(X=1)$	0.85	0.85	0.75	0.67	0.59	0.57	0.57	0.55	0.53	0.55	0.67	0.62
$\widehat{P}(X=1)$	0.69	0.62	0.74	0.62	0.58	0.53	0.52	0.54	0.60	0.58	0.67	0.59

Node(X)	13	14	15	16	17	18	19	20	21	22	23	24
$P(X=1)$	0.58	0.53	0.62	0.58	0.56	0.54	0.36	0.41	0.41	0.43	0.45	0.43
$\widehat{P}(X=1)$	0.54	0.57	0.62	0.62	0.50	0.49	0.47	0.42	0.46	0.43	0.43	0.38

Node(X)	25	26	27	28	29	30	31	32	33	34	35	36
$P(X=1)$	0.54	0.53	0.52	0.45	0.53	0.52	0.31	0.37	0.37	0.37	0.41	0.41
$\widehat{P}(X=1)$	0.59	0.50	0.51	0.45	0.47	0.54	0.32	0.35	0.36	0.35	0.41	0.34

Node(X)	37	38	39	40	41	42	43	44	45	46		
$P(X=1)$	0.41	0.41	0.32	0.38	0.38	0.38	0.41	0.41	0.41	0.41		
$\widehat{P}(X=1)$	0.46	0.39	0.32	0.36	0.36	0.35	0.41	0.34	0.47	0.39		

Sample size = 100,000; Threshold = 1.0.

all the submodels are all 43 except submodels 1, 10, 11, 13, and 14. The number of parameters for submodel 1 is 42 and it is 41 for submodels 10, 11, 13, and 14 having 41. The degrees of freedom of all the submodels are 212 except submodel 1 whose degrees of freedom is 213 and submodels 10, 11, 13, and 14 for which the degrees of freedom are all 214.

Since submodels T_1, T_2, T_3 and T_4 do not have any observed variables, we can not think of degrees of freedom for them. The goodness-of-fit levels for the other 14 submodels are given in Table 6.5, which is obtained based on simulated data of size 100,000 for each submodel and the stopping threshold was 1.0 for the estimates of the cell means. The p-values of the goodness-of-fit test are about 0.012 for submodel 1, 0.018 for submodel 3, 0.037 for submodel 6, 0.039 for submodel 14 and it is more than 0.05 for the other submodels. We used a much larger sample size of $100,000$ for this large model than for the preceding two experiments in order to save time until convergence by using a larger stopping threshold of 1.0.

The estimates for the marginal probabilities of the latent variables are given in Table 6.6. The node labels in the table are those in Figure 6.3. Except the first two and the nineteenth estimates, the estimates are within 0.1 of the actual values. Although they are not tabulated, the estimates of all the marginal or conditional probabilities of the variables in the submodels were in general close to the actual values.

7 MODELLING WITH REAL DATA

We analyzed a data set of 20 multiple choice items of Mathematics section of the Korean SAT that was administered in 1999. After investigating the 20 items, we ended up with eight knowledge units or ability factors that are relevant to the 20 test items. This means we have 20 observed binary variables for item scores (0 for incorrect answer and 1 for correct answer) and 8 unobservable binary variables for the states of the knowledge units (0 for a poor state of knowledge and 1 for a good enough state). The data set is available from the web-site, *http://amath.kaist.ac.kr/∼slki/research/data*.

The 28 variables are related as in panel (a) in Figure 7.1, where an arrow from a box to a bullet stands for a causal relation between the corresponding knowledge unit and test item and an arrow from a box to a box mostly stands for a prerequisite relationship between the corresponding pair of knowledge units (Mislevy 1994). If an item can be solved when a test-taker possesses a good knowledge of certain knowledge units, then the item-score variable is said to be causally related with the knowledge units and arrows run from the corresponding knowledge-state variables to the item-score variable. The initial structure of the relationship among the item-score variables and the knowledge-state variables is based on the opinions of a group of experts of the test subject. The knowledge units are listed in Table 7.1.

As a whole, we need to deal with 28 binary variables or $2^{28} = 268,435,456$ cell probabilities for parameter estimation of the model as in Figure 7.1. The parameters are given in the form of

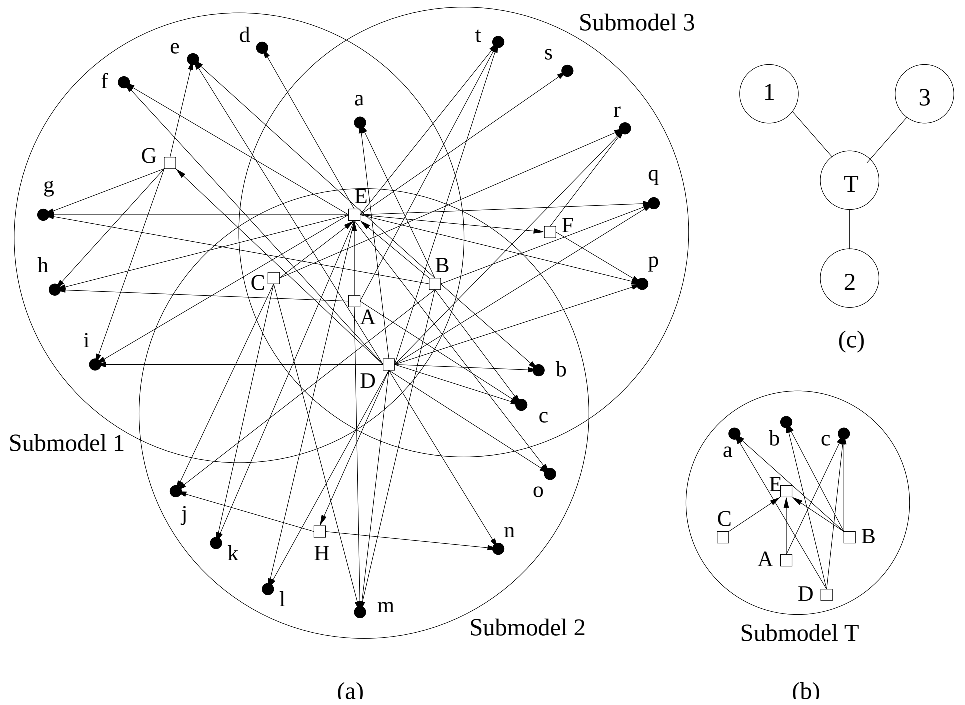


Figure 7.1: A DAG for real data. Bullets are for the item score variables and boxes for the knowledge states.

Table 7.1: The list of the knowledge units involved in the model in Figure 7.1

Code	Contents
A	DK about sets
B	DK about numbers and equations
C	DK about plane geometry
D	PK for inference
E	DK about one-variable functions
F	DK about trigonometrical functions
G	PK for problem recognition
H	PK for problem solving

NOTE: DK is an acronym of “declarative knowledge” and PK of “procedural knowledge.”

Table 7.2: The goodness-of-fit levels for the three submodels, submodel 1, 2, and 3, as in Figure 7.1.

Submodel(d.f.)	1(79)	2(201)	3(203)
χ^2	89.89	220.91	226.43
P-value	0.189	0.160	0.124

Sample size = 1,000; Threshold = 0.05.

Table 7.3: The estimates of the marginal probabilities for the knowledge-state variables.

Node(X)	A	B	C	D	E	F	G	H
$\widehat{P}(X = 1)$	0.71	0.56	0.58	0.58	0.52	0.56	0.76	0.56

Sample size = 1,000; Threshold = 0.05.

marginal or conditional probability. As for the item-score variable g in the figure, knowledge units B, E, and G are relevant. So we need to estimate $P(X_g = 1|X_{B,E,G} = (i, j, k))$ for $i, j, k = 0, 1$. We can expect that the conditional probabilities be all positive since the choice of a correct answer is possible by a lucky guessing on an item which is too much for a test-taker.

Working with the whole model of the 28 variables might take up an overwhelming computing time of several months by a fast-computing workstation with Intel Xeon 1.6GHz/4CPU when we need to use an EM algorithm. However, when we applied our method to this data set, the computing time came down to 24.5 hours. We d-split the whole model into a tree of submodels as in panel (c) of Figure 7.1, where submodels 1, T, 2, and 3 are of 13, 8, 14, and 14 variables, respectively. As aforementioned, submodel T is regarded as a submodel although it is created for estimate-transfusion among the submodels that are neighbors of the T-type submodel. Since the maximum number of the variables involved in a submodel is 14, the computing time when we handle the whole model as a single model would be several months provided the memory capacity is large enough for the 270 million table-cells.

The marginal restructuring on all the submodels yields that $\mathcal{G}_1 = \mathcal{G}^1$, $\mathcal{G}_T = \mathcal{G}^T$, $\mathcal{G}_2 = \mathcal{G}^2$, and $\mathcal{G}_3 = \mathcal{G}^3$. The degrees of freedom for the submodels 1, 2, and 3 are given in Table 7.2. The sample size of the data we used is 1,000 and the stopping threshold for the hyper-EM method is 0.05 for each submodel.

The goodness-of-fit levels for the three submodels are given in Table 7.2. The p-values of the goodness-of-fit test are all larger than 0.12, indicating a good overall fit for individual submodels. The estimates of the marginal probabilities of the knowledge-state variables are given in Table 7.3,

Table 7.4: The proportions correct of the 20 test items as appearing in Figure 7.1

Item	a	b	c	d	e	f	g	h	i	j
Proportion	0.60	0.54	0.52	0.75	0.61	0.84	0.79	0.64	0.60	0.53

Item	k	l	m	n	o	p	q	r	s	t
Proportion	0.59	0.67	0.48	0.59	0.30	0.52	0.54	0.54	0.40	0.66

Sample size = 1,000.

where the node labels are as in Figure 7.1. The marginal probabilities are average levels of the knowledge states of the test-takers and their estimates range from 0.52 to 0.71. If we look at the proportions correct for the 20 test items as in Table 7, we can see that the proportions range from 0.3 to 0.84.

We can see in the estimation result that the knowledge state of the knowledge unit which is causally related to the items whose proportions correct are low is estimated accordingly low. For instance, knowledge-state variable X_F is related to item-score variables X_p and X_r whose proportions correct are 0.52 and 0.54 respectively, and the marginal for X_F is estimated as 0.56; a similar situation is observed regarding X_H. However, when a knowledge-state variable is related to many item-score variables or vice versa, such an accordance in estimates becomes less apparent.

8 CONCLUDING REMARKS

A key idea behind the proposed method called *hyper-EM* for dealing with too large a model is that we partition the whole model into a set of submodels of manageable sizes and use the submodels as separate models while allowing the submodels to share a given data set as much as possible. The data set is marginalized into as many marginal frequency tables as the number of the submodels which contain at least one observed variable.

The hyper-EM is a generalized version of EM. Its E-step is carried out in the form of iterative proportional fitting calibrating the estimates to the marginal frequency tables. The ET between submodels is necessary to have all the estimates of the whole model consistent throughout the model. The marginal distribution on a submodel is, based on the estimates, the same whether it is obtained from the whole model or not. The M-step of the hyper-EM is a generalized version of the M-step of EM. We do the likelihood-maximization for individual submodels and if necessary estimates are transfused between neighboring submodels to obtain the consistency of the distribution throughout the whole model. The ET is not necessary when the estimates of the parameters pertaining to the intersection of a pair of neighboring submodels are the same whether the estimates are obtained from one of the submodels or from the other. If a tree of submodels is a hyper-EM tree, we do not need the estimate transfusion; otherwise, we need it.

When we divide a large recursive model into submodels, it is very important that the two conditions, the Family condition and the hyper-EM condition, are satisfied. The Family condition makes it possible that the probability model for the whole model can be expressed as a product of functions of submodels. The hyper-EM condition is instrumental for the M-step so that the ET works for the consistency of the distribution for the whole model.

After a d-split under the hyper-EM condition, the resulting submodel-arrangement is given in the form of a tree where branching is made at the nodes of the submodels each of which consists of the variables that are contained in more than one submodel in a ring-mode arrangement. This tree-shape arrangement of submodels means that the probability model for the whole model can be expressed as a product of (conditional) probability models of the submodels.

In the simulation experiment, we considered three models which were d-split into a series-mode arrangement, a tree with a single branching node and a tree with multiple branching nodes. The simulation result says that the hyper-EM can be used for a recursive model of any size as long as the model is split under the two conditions as introduced in this paper. As for the tree of multiple branching nodes with 158 binary variables, the p-values of the goodness-of-fit levels were larger than 0.05 except four submodels for which the p-values were 0.012, 0.018, 0.037, and 0.039. The estimates of the marginal probabilities of the latent variables were within 0.1 for 43 of the 46 latent variables and within 0.05 for 38 of them. The estimates of the other variables in the model were close to the actual values. This result would not be possible if we were to handle the whole model as a single model.

Although we considered a real data set for a relatively smaller model compared with the simulation models, the result is additional support for the proposed method as a practical tool for building a recursive model of categorical variables which is too large to handle as a single model.

If we do not split a model, the hyper-EM is the same as a regular EM algorithm. When a model is d-split, the only difference between the hyper-EM and the EM is that we do the ET between neighboring submodels when applying the hyper-EM.

Once a d-split is made on a recursive model, it is desirable that we check if individual submodels are appropriate to their corresponding marginals of data and the hyper-EM is more properly employed when the individual submodels all fit better with the data.

APPENDIX: PROOF OF THEOREM 3.3

Let $clan(v^*) = \{v_1, \cdots, v_m, \cdots, v_\kappa\}$ where v_i's are ordered such that $v_m = v^*$ and $ch(v_m) = \{v_{m+1}, \cdots, v_\kappa\}$. Then we have

$$\begin{aligned}
\widehat{P}_V &= \prod_{v \in V} \frac{[fa(v)]}{[pa(v)]} \\
&= \Big(\prod_{v \in (V \setminus clan(v_m))} \frac{[fa(v)]}{[pa(v)]} \Big) \Big(\prod_{v \in clan(v_m)} \frac{[fa(v)]}{[pa(v)]} \Big) \\
&= \Big(\prod_{v \in (V \setminus clan(v_m))} \frac{[fa(v)]}{[pa(v)]} \Big) \Big(\prod_{i=1}^{m-1} \frac{[fa(v_i)]}{[pa(v_i)]} \Big) \Big(\frac{[fa(v_m)]}{[pa(v_m)]} \Big) \Big(\prod_{i=m+1}^{\kappa} \frac{[fa(v_i)]}{[pa(v_i)]} \Big) \qquad \text{(A.1)}
\end{aligned}$$

Since v_m is contained only in $fa(v_i)$, $m \le i \le \kappa$, and $pa(v_i)$, $m+1 \le i \le \kappa$, we consider only these $fa(v_i)$, $m \le i \le \kappa$, and $pa(v_i)$, $m+1 \le i \le \kappa$. For convenience's sake, let

$$F = \frac{[fa(v_m)]}{[pa(v_m)]} \prod_{i=m+1}^{\kappa} \frac{[fa(v_i)]}{[pa(v_i)]} \qquad \text{(A.2)}$$

and

$$R = \Big(\prod_{v \in (V \setminus clan(v_m))} \frac{[fa(v)]}{[pa(v)]} \Big) \Big(\prod_{i=1}^{m-1} \frac{[fa(v_i)]}{[pa(v_i)]} \Big). \qquad \text{(A.3)}$$

When $ch(v_m) = \emptyset$,

$$F = \frac{[fa(v_m)]}{[pa(v_m)]}.$$

This means that v_m is contained in one and only one f-clique $fa(v_m)$ in expression (A.1), which satisfies condition (i) of the theorem. That $ch(v_m) = \emptyset$ means that v_m is terminal in $\mathcal{G}$. So, if $pa(v_m)$ is made complete, $clan(v_m)$ becomes a clique, satisfying condition (ii) of the theorem. Furthermore, since v_m is terminal, $clan(v_m) = fa(v_m)$ and

$$\begin{aligned}
\widehat{P}_V &= R \cdot F \\
&= \Big(\Big(\prod_{v \in (V \setminus clan(v_m))} \frac{[fa(v)]}{[pa(v)]} \Big) \Big(\prod_{i=1}^{m-1} \frac{[fa(v_i)]}{[pa(v_i)]} \Big) \Big) \Big(\frac{[fa(v_m)]}{[pa(v_m)]} \Big) \\
&= \Big(\widehat{\sum_{x_{\{v_m\}}} P_V} \Big) \Big(\frac{[fa(v_m)]}{[pa(v_m)]} \Big).
\end{aligned}$$

Thus, from $fa(v_m) = pa(v_m) \cup \{v_m\}$, we have at $x_{(v_m)} = x^*_{(v_m)}$

$$\begin{aligned}\sum_{x_{\{v_m\}}} \widehat{P}_V &= \Big(\widehat{\sum_{x_{\{v_m\}}} P_V}\Big)\Big(\sum_{x_{\{v_m\}}} \frac{[fa(v_m)]}{[pa(v_m)]}\Big) \\ &= \widehat{\sum_{x_{\{v_m\}}} P_V}.\end{aligned}$$

In other words, v_m is removable from $\mathcal{G}$. Therefore, nodes which are terminal always satisfy this theorem.

From now on, we will consider only v_m such that $ch(v_m) \neq \emptyset$. We prove the theorem by showing, first, equivalence of condition (i) with condition (ii) and then equivalence of condition (ii) with (iii). We assume condition (ii). From condition (ii) follows that $fa(v_i) = pa(v_{i+1})$, $m \leq i \leq \kappa - 1$. So, expression (A.2) becomes

$$F = \frac{[fa(v_\kappa)]}{[pa(v_m)]}$$

and in expression (A.1), v_m is involved in the term $[fa(v_k)]$ only. In other words, v_m is contained in the f-clique, $fa(v_\kappa)$, only.

Now for the other direction of the equivalence, assume v_m is contained in one f-clique only, say C_{v_m}. Then

$$F = \frac{[C_{v_m}]}{[pa(v_m)]} \tag{A.4}$$

since $v_m \notin pa(v_m)$. Equation (A.4), when multiplied by $[pa(v_m)]$, becomes

$$[fa(v_m)] \prod_{i=m+1}^{\kappa} \frac{[fa(v_i)]}{[pa(v_i)]} = [C_{v_m}]. \tag{A.5}$$

The left-hand side of (A.5) is a function of the X variables indexed in a set which contains $pa(v_m) \cup \{v_m, \cdots, v_\kappa\}$. According to the structure of $clan(v_m)$, $v_{m+1}, \cdots, v_\kappa$ are dependent upon v_m at the very least. This means that $[C_{v_m}]$ must be a function of the X variables indexed in a set which contains $\{v_m, \cdots, v_\kappa\}$. In other words,

$$\{v_m, \cdots, v_\kappa\} \subseteq C_{v_m}. \tag{A.6}$$

However, $v_\kappa \notin \cup_{i=m+1}^{\kappa-1} fa(v_i)$ and $v_\kappa \in fa(v_\kappa)$. From (3.3) and (A.6), it follows that

$$C_{v_m} \in \{fa(v_i) \mid m \leq i \leq \kappa\}.$$

This implies that $fa(v_\kappa) = C_{v_m}$ since $v_\kappa \in C_{v_m}$. Thus we have $[fa(v_\kappa)] = [C_{v_m}]$. That is,

$$[fa(v_m)] \prod_{i=m+1}^{\kappa} \frac{[fa(v_i)]}{[pa(v_i)]} = [fa(v_\kappa)]. \tag{A.7}$$

By the definition of f-clique and (A.7), we can see that for each i, $m+1 \leq i \leq \kappa$, there exist j, $m \leq j \leq \kappa - 1$, such that

$$pa(v_i) = fa(v_j).$$

Thus

$$pa(v_\kappa) \in \{fa(v_i) \mid m \leq i \leq \kappa - 1\}. \tag{A.8}$$

Since $pa(v_\kappa) \subseteq fa(v_\kappa) = C_{v_m}$, we have, from (A.6) and the fact that $pa(v_\kappa) \cup \{v_\kappa\} = fa(v_\kappa)$,

$$\{v_m, \cdots, v_{\kappa-1}\} \subseteq pa(v_\kappa). \tag{A.9}$$

However, $v_{\kappa-1} \notin \cup_{i=m}^{\kappa-2} fa(v_i)$, and $v_{\kappa-1} \in fa(v_{\kappa-1})$. Thus, by (A.8) and (A.9),

$$fa(v_{\kappa-1}) = pa(v_\kappa), \tag{A.10}$$

which means $[fa(v_{\kappa-1})] = [pa(v_\kappa)]$. Proceeding to $v_{\kappa-1}$, we have from (A.8) and (A.10) that

$$pa(v_{\kappa-1}) \in \{fa(v_i) \mid m \leq i \leq \kappa - 2\}. \tag{A.11}$$

Since $pa(v_{\kappa-1}) \subseteq fa(v_{\kappa-1}) = pa(v_\kappa)$, we have, from (A.9) and the fact that $pa(v_{\kappa-1}) \cup \{v_{\kappa-1}\} = fa(v_{\kappa-1})$,

$$\{v_m, \cdots, v_{\kappa-2}\} \subseteq pa(v_{\kappa-1}).$$

However, $v_{\kappa-2} \notin \cup_{i=m}^{\kappa-3} fa(v_i)$, and $v_{\kappa-2} \in fa(v_{\kappa-2})$. Thus, by (A.11),

$$fa(v_{\kappa-2}) = pa(v_{\kappa-1}) \tag{A.12}$$

since $v_{\kappa-2} \in pa(v_{\kappa-1})$. Hence,

$$[fa(v_{\kappa-2})] = [pa(v_{\kappa-1})].$$

As for $v_{\kappa-2}$, we have from (A.8), (A.10), and (A.12),

$$pa(v_{\kappa-2}) \in \{fa(v_i) \mid m \leq i \leq \kappa - 3\}.$$

By applying the same argument as for (A.12), we can have $fa(v_\eta) = pa(v_{\eta+1})$ for $m \leq \eta \leq \kappa - 3$. Since $fa(v_i) = pa(v_{i+1})$ for $m \leq i \leq \kappa - 1$, all the nodes in $clan(v_m)$ becomes a clique when $pa(v_m)$ is made complete and v_m is contained in $clan(v_m)$ only since v_m is surrounded by $pa(v_m)$ and $ch(v_m)$. This completes the proof for the sufficiency of condition (i) for condition (ii).

We will next prove the equivalence of *(ii)* with *(iii)*. First we assume condition *(ii)*. Since $clan(v_m)$ is a clique in $\mathcal{G}$ when $pa(v_m)$ is made into a complete subgraph in $\mathcal{G}$, we know, under the set-up of $clan(v_m)$ at the beginning of the proof, that $fa(v_i) = pa(v_{i+1})$ for $m \le i \le \kappa - 1$ and so that

$$F = \frac{[fa(v_\kappa)]}{[pa(v_m)]}.$$

So (A.1) can be written as

$$\begin{aligned} \widehat{P}_V &= R \cdot F \\ &= \Bigg(\Big(\prod_{v \in (V \setminus clan(v_m))} \frac{[fa(v)]}{[pa(v)]}\Big)\Big(\prod_{i=1}^{m-1} \frac{[fa(v_i)]}{[pa(v_i)]}\Big)\Bigg)\Big(\frac{[fa(v_\kappa)]}{[pa(v_m)]}\Big). \end{aligned} \tag{A.13}$$

In comparing the marginal structure $\mathcal{G}^{V\setminus\{v_m\}}$ with $\mathcal{G}$, we may have to confine ourselves to the part of $clan(v_m)$. The interrelationships among the variables in a recursive model is defined in the form of conditional probability of each variable in the model according to $clan(v_m)$ as in condition *(ii)*, $fa(v_i) = pa(v_{i+1})$, $m \le i \le \kappa - 1$. Therefore, the removal of v_m from $\mathcal{G}$ does not affect any other variables in $\mathcal{G}$ in the context of conditional independence. It is important to note that all the children and parents of v_m, if any, are included in $clan(v_m)$. For convenience, we will call this situation as for $clan(v_m)$ *clan condition*.

Marginalizing $\widehat{P}_V$ onto $V \setminus \{v_m\}$ at $x_{(v_m)} = x^*_{(v_m)}$ yields, from (A.13), the following

$$\begin{aligned} \sum_{x_{\{v_m\}}} \widehat{P}_V &= \sum_{x_{\{v_m\}}} \left(R \cdot \frac{[fa(v_\kappa)]}{[pa(v_m)]}\right) \\ &= R \cdot \Big(\frac{\sum_{x_{\{v_m\}}}[fa(v_\kappa)]}{[pa(v_m)]}\Big) \\ &= R \cdot \Big(\frac{[fa(v_\kappa) \setminus \{v_m\}]}{[pa(v_m)]}\Big) \end{aligned}$$

Since $pa(v_m) = fa(v_m) \setminus \{v_m\} = pa(v_{m+1}) \setminus \{v_m\}$, we have at $x_{(v_m)} = x^*_{(v_m)}$

$$\sum_{x_{\{v_m\}}} \widehat{P}_V = R \cdot \Big(\frac{[fa(v_\kappa) \setminus \{v_m\}]}{[pa(v_{m+1}) \setminus \{v_m\}]}\Big) \tag{A.14}$$

Note that

$$\frac{[fa(v_\kappa) \setminus \{v_m\}]}{[pa(v_{m+1}) \setminus \{v_m\}]}$$

is the MLE of $P_{ch(v_m)|pa(v_m)}$. Thus (A.14) can be expressed as, by condition (ii) and the clan condition,

$$\begin{aligned} & R \cdot \Big(\frac{[fa(v_\kappa) \setminus \{v_m\}]}{[pa(v_{m+1}) \setminus \{v_m\}]} \Big) \\ = \; & R \cdot \Big(\prod_{v \in (clan(v_m) \setminus fa(v_m))} \frac{[fa'(v)]}{[pa'(v)]} \Big), \end{aligned}$$

where, for $v \in (clan(v_m) \setminus fa(v_m))$, $[fa'(v)]$ and $[pa'(v)]$ are computed based on the marginal on $\mathcal{G}^{V \setminus \{v_m\}}$. Since $[fa(v)]$ and $[pa(v)]$ remain the same, for $v \in V \setminus (\{v_m\} \cup ch(v_m))$, between $\mathcal{G}$ and $\mathcal{G}^{V \setminus \{v_m\}}$ by the clan condition, the above expression is the MLE for $\mathcal{G}^{V \setminus \{v_m\}}$, i.e.,

$$\sum_{x_{\{v_m\}}} \widehat{P}_V = \widehat{\sum_{x_{\{v_m\}}} P_V}.$$

This proves that condition (ii) implies condition (iii).

For the proof for the other direction, we begin by supposing that condition (ii) does not hold. Then, without loss of generality, we can think of three possible situations which are displayed in $(a), (b)$, and (c) in Figure A.1. In the figure, the three situations are featured by three types of elementary violations of condition (ii). The violations are no edge between a parent and a child of v_m as depicted in panel (a), no edge between children of v_m as in panel (b), and no edge between v_m and a parent of a child of v_m as depicted in panel (c). A general form of violation may be given in a mixture of three elementary violations. In the proof, we will consider each of these elementary violations and then a general form of violation. $(a'), (b')$, and (c') are respectively the marginalized subgraphs of $\mathcal{G}$ as in $(a), (b)$, and (c) in Figure A.1.

Note that $v^* = v_m$ in Figure A.1. The left hand side of equation (3.5) is expressed at $x_{(v_m)} = x^*(v_m)$ for the three violations as in panels $(a), (b)$, and (c) in Figure A.1 respectively by

$$\sum_{x_{\{v_m\}}} \frac{[\{1, v_m\}][\{v_m, 3\}]}{[\{v_m\}]},$$

$$\sum_{x_{\{v_m\}}} \frac{[\{v_m, 2\}][\{v_m, 3\}]}{[\{v_m\}]},$$

and

$$\sum_{x_{\{v_m\}}} \frac{[\{v_m\}][\{2\}][\{v_m, 2, 3\}]}{[\{v_m, 2\}]}.$$

None of these values are guaranteed to be equal to the marginals corresponding to the graphs in panels $(a'), (b')$, and (c'), which are expressed respectively by

$$[\{1,3\}], \; [\{2,3\}], \; \text{and } [\{2,3\}]. \tag{A.15}$$

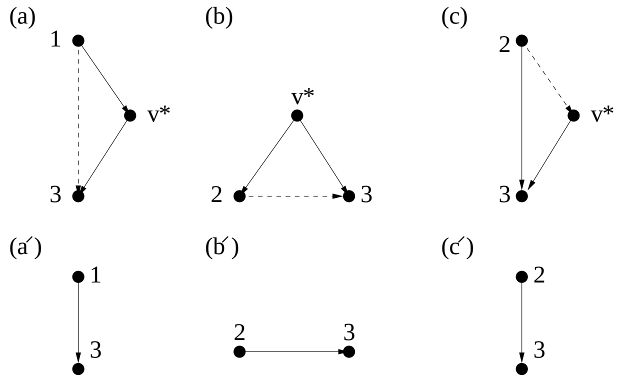


Figure A.1: A display of the three elementary violations of condition (ii) is given in panels $(a), (b)$, and (c). $v^* = v_m$ and the dotted arrows are needed to satisfy condition (ii). $(a'), (b')$, and (c') are marginalized subgraphs of their counterparts in (a), (b), and (c).

Since these marginalized subgraphs are saturated, the values in (A.15) are MLEs for the marginalized subgraphs. This result means negation of condition (iii) as far as the elementary violations are concerned.

We will now turn to a general form of violation of condition (ii) concerning $clan(v_m)$ and see how the elementary violations affect toward negation of condition (iii) regarding the node v_m. We aim to show that the equality

$$\sum_{x_{\{v_m\}}} \widehat{P}_V = \widehat{\sum_{x_{\{v_m\}}} P_V} \tag{A.16}$$

cannot hold when any of the elementary violations takes place in $clan(v_m)$. Since the elementary violations have nothing to do with $v_1, \cdots, v_{m-1}$ among the nodes in $clan(v_m)$, the summation in (A.16) applies only to F as defined in (A.2) among all the factors in expression (A.1). In a formal expression, we have at $x_{(v_m)} = x^*_{(v_m)}$

$$\begin{aligned}&\sum_{x_{\{v_m\}}} \widehat{P}_V \\ =& \Big(\prod_{v\in(V\setminus clan(v_m))} \frac{[fa(v)]}{[pa(v)]}\Big)\Big(\prod_{i=1}^{m-1} \frac{[fa(v_i)]}{[pa(v_i)]}\Big) \sum_{x_{\{v_m\}}} \left(\frac{[fa(v_m)]}{[pa(v_m)]} \prod_{i=m+1}^{\kappa} \frac{[fa(v_i)]}{[pa(v_i)]}\right).\end{aligned}$$

By the clan condition, the removal of v_m from $\mathcal{G}$ does not affect any other variables in $\mathcal{G}$ in the context of conditional independence. Thus we can obtain

$$\widehat{\sum_{x_{\{v_m\}}} P_V} = \Big(\prod_{v\in(V\setminus clan(v_m))} \frac{[fa(v)]}{[pa(v)]}\Big)\Big(\prod_{i=1}^{m-1} \frac{[fa(v_i)]}{[pa(v_i)]}\Big)\Big(\prod_{i=m+1}^{\kappa} \frac{[fa'(v_i)]}{[pa'(v_i)]}\Big),$$

where, for $v \in (clan(v_m) \setminus fa(v_m))$, $[fa'(v)]$ and $[pa'(v)]$ are computed based on the marginal on $\mathcal{G}^{V\setminus\{v_m\}}$.

We will make use of the simple fact that, for a subset A of V,

$$\sum_{x_A}\Big(\sum_{x_{\{v_m\}}}\widehat{P_V}\Big) \neq \sum_{x_A}\Big(\widehat{\sum_{x_{\{v_m\}}} P_V}\Big) \tag{A.17}$$

implies

$$\sum_{x_{\{v_m\}}}\widehat{P_V} \neq \widehat{\sum_{x_{\{v_m\}}} P_V} \tag{A.18}$$

Recall that the children nodes of v_m are ordered as $v_{m+1}, \cdots, v_\kappa$ and it is important to note that the elementary violations are incurred only by the child nodes of v_m. We will show that for any single elementary violation of condition (ii) invalidates the removability of v_m from $\mathcal{G}$. To make our argument as simple as possible, we describe a procedure of detecting a node in $ch(v_m)$ which violates condition (ii) the first time in the order of node-indexes as follows:

Find the node $v_s \in ch(v_m)$ for which

$$s = \min\{i \mid fa(v_{i-1}) \neq pa(v_i), v_i \in ch(v_m)\}. \tag{A.19}$$

Negation of condition (ii) means the existence of the node v_s in $ch(v_m)$. To show (A.18), assuming the existence, we use the simple fact concerning (A.17) and (A.18). For this, we will let $A = \{v_i \in V \mid i > s\}$. The two sides of (A.17) are derived below.

$$\begin{aligned}
&\sum_{x_A}\Big(\sum_{x_{\{v_m\}}}\widehat{P_V}\Big)\\
=\;& \sum_{x_{\{v_m\}}}\Bigg(\sum_{x_A}\Big(\big(\prod_{v\in(V\setminus clan(v_m))}\frac{[fa(v)]}{[pa(v)]}\big)\big(\prod_{i=1}^{m-1}\frac{[fa(v_i)]}{[pa(v_i)]}\big)\big(\frac{[fa(v_m)]}{[pa(v_m)]}\big)\big(\prod_{i=m+1}^{\kappa}\frac{[fa(v_i)]}{[pa(v_i)]}\big)\Big)\Bigg)\\
=\;& \sum_{x_{\{v_m\}}}\Bigg(\big(\prod_{i=1}^{m-1}\frac{[fa(v_i)]}{[pa(v_i)]}\big)\big(\frac{[fa(v_m)]}{[pa(v_m)]}\big)\sum_{x_A}\Big(\big(\prod_{v\in(V\setminus clan(v_m))}\frac{[fa(v)]}{[pa(v)]}\big)\big(\prod_{i=m+1}^{\kappa}\frac{[fa(v_i)]}{[pa(v_i)]}\big)\Big)\Bigg)\\
=\;& \sum_{x_{\{v_m\}}}\Bigg(\Big(\prod_{v\in V\setminus(clan(v_m)\cup A)}\frac{[fa(v)]}{[pa(v)]}\Big)\Big(\prod_{i=1}^{m-1}\frac{[fa(v_i)]}{[pa(v_i)]}\Big)\Big(\frac{[fa(v_m)]}{[pa(v_m)]}\Big)\Big(\prod_{i=m+1}^{s}\frac{[fa(v_i)]}{[pa(v_i)]}\Big)\Bigg)\\
=\;& \Big(\prod_{v\in V\setminus(clan(v_m)\cup A)}\frac{[fa(v)]}{[pa(v)]}\Big)\Big(\prod_{i=1}^{m-1}\frac{[fa(v_i)]}{[pa(v_i)]}\Big)\sum_{x_{\{v_m\}}}\Bigg(\frac{[fa(v_m)]}{[pa(v_m)]}\prod_{i=m+1}^{s}\frac{[fa(v_i)]}{[pa(v_i)]}\Bigg)
\end{aligned} \tag{A.20}$$

and

$$\begin{aligned}
&\sum_{x_A}\Big(\widehat{\sum_{x_{\{v_m\}}} P_V}\Big)\\
=\;& \Big(\prod_{i=1}^{m-1}\frac{[fa(v_i)]}{[pa(v_i)]}\Big)\sum_{x_A}\Bigg(\Big(\prod_{v\in(V\setminus clan(v_m))}\frac{[fa(v)]}{[pa(v)]}\Big)\Big(\prod_{i=m+1}^{\kappa}\frac{[fa'(v_i)]}{[pa'(v_i)]}\Big)\Bigg)\\
=\;& \Big(\prod_{v\in V\setminus(clan(v_m)\cup A)}\frac{[fa(v)]}{[pa(v)]}\Big)\Big(\prod_{i=1}^{m-1}\frac{[fa(v_i)]}{[pa(v_i)]}\Big)\Big(\prod_{i=m+1}^{s}\frac{[fa'(v_i)]}{[pa'(v_i)]}\Big).
\end{aligned} \tag{A.21}$$

In comparing the two sides of (A.17), we have only to compare the following two expressions:

$$\sum_{x_{\{v_m\}}} \left(\frac{[fa(v_m)]}{[pa(v_m)]} \prod_{i=m+1}^{s} \frac{[fa(v_i)]}{[pa(v_i)]} \right) \tag{A.22}$$

and

$$\prod_{i=m+1}^{s} \frac{[fa'(v_i)]}{[pa'(v_i)]}. \tag{A.23}$$

Note in (A.20) and (A.21) that node v_s may be regarded as a terminal node in a recursive model.

According to (A.19), there is no v_i, $i < s$, in $clan(v_m)$ such that $fa(v_i) = pa(v_s)$. So in (A.22), v_m must appear in both of numerator and denominator. As a matter of fact, we can see from $(A.19)$ that at $x_{(v_m)} = x^*_{(v_m)}$

$$\sum_{x_{\{v_m\}}} \left(\frac{[fa(v_m)]}{[pa(v_m)]} \prod_{i=m+1}^{s} \frac{[fa(v_i)]}{[pa(v_i)]} \right) = \sum_{x_{\{v_m\}}} \left(\frac{[fa(v_{s-1})]}{[pa(v_m)]} \frac{[fa(v_s)]}{[pa(v_s)]} \right) \tag{A.24}$$

and

$$\prod_{i=m+1}^{s} \frac{[fa'(v_i)]}{[pa'(v_i)]} = \frac{[fa'(v_{s-1})]}{[pa'(v_{m+1})]} \frac{[fa'(v_s)]}{[pa'(v_s)]} \tag{A.25}$$

for any mixture of elementary violations. Since $pa(v_m) = pa'(v_{m+1})$, we may compare the two values in (A.24) and (A.25) in terms of

$$\sum_{x_{\{v_m\}}} \left([fa(v_{s-1})] \frac{[fa(v_s)]}{[pa(v_s)]} \right) \quad \text{and} \quad [fa'(v_{s-1})] \frac{[fa'(v_s)]}{[pa'(v_s)]}$$

whose equality are not guaranteed in general. This completes the proof of the theorem. □

REFERENCES

Asmussen, S. and Edwards, D. (1983), " Collapsibility and response variables in contingency tables," *Biometrika*, 70, 567-578.

Birch, M. W. (1963), " Maximum likelihood in three way contingency tables," *Journal of The Royal Statistical Society,* Ser. B, 25, 220-233.

Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1975), *Discrete Multivariate Analysis: Theory and Practice*, Cambridge, MA: MIT Press.

Darroch, J. N., Lauritzen, S. L., and Speed, T. P. (1980), " Markov fields and log-linear models for contingency tables," *Annals of Statistics*, 8, 522-539.

Dawid, A. P. (1979), " Conditional independence in statistical theory"(with discussion), *Journal of The Royal Statistical Society,* Ser. B, 41, 1-31

Dawid, A. P. and Lauritzen, S. L. (1993), " Hyper markov laws in the statistical analysis of decomposable graphical models," *Annals of Statistics*, 21, 1272-1317.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), " Maximum likelihood from incomplete data via the EM algorithm"(with discussion), *Journal of The Royal Statistical Society,* Ser. B, 39, 1-38.

Edwards, D. and Lauritzen, S.L. (2001), "The TM algorithm for maximising a conditional likelihood function," *Biometrika*, **88**, 4, 961-972.

Fienberg, S. E. (1980), *The Analysis of Cross-Classified Categorical Data* (2nd ed.), Cambridge, MA: MIT Press.

Goodman, L. A. (1974a), " The analysis of systems of qualitative variables when some of the variables are unobservable. Part I: A modified latent structure approach," *American Journal of Sociology*, 79, 1179-1259.

——— (1974b), " Exploratory latent structure analysis using both identifiable and unidentifiable models," *Biometrika*, 61, 215-231.

Haberman, S. J. (1977), " Product models for frequency tables involving indirect observation," *Annals of Statistics*, 5, 1124-1147.

——— (1979), *Analysis of Qualitative Data* (Vol. 2), New York: Academic Press.

Howard, R. A. and Matheson, J. E. (1981), *Influence diagrams*, in: *Reading in The Principles and Applications of Decision Analysis, Strategic Decisions Group*, eds. R. A. Howard and J. E. Matheson, pp. 719-762.

Jensen, F. V., Olesen, K. G., and Andersen, S. K. (1990), "An algebra of Bayesian belief universes for knowledge-based systems," *Networks*, 20, 637-659.

Kim, S. -H. (2000), " Hyper-EM for large recursive models of categorical variables," *Computational Statistics and Data Analysis*, 33, 401-424.

——— (2002), " Calibrated initials for an EM applied to recursive models of categorical variables," *Computational Statistics and Data Analysis*, 40, 91-110.

Lauritzen, S. L. (1995), “ The EM algorithm for graphical association models with missing data,” *Computational Statistics and Data Analysis*, 19, 191-201.

——— (1996), *Graphical Models*, Oxford: Clarendon Press.

Lauritzen, S. L., Dawid, A. P., Larsen, B. N., and Leimer, H.-G. (1990), “ Independence properties of directed Markov fields,” *Networks*, 20, 491-505.

Lauritzen, S. L. and Spiegelhalter, D. J. (1988), “ Local computations with probabilities on graphical structures and their application to expert systems”(with discussion), *Journal of The Royal Statistical Society*, Ser. B, 50, 157-224.

Mislevy, R. J. (1994), “ Evidence and inference in educational assessment,” *Psychometrika*, 59, 439-483.

Olmsted, S. M. (1983), “ On representing and solving decision problems,” Ph. D. dissertation, Stanford University, Dept. of Engineering-Economic Systems.

Pearl, J. (1986), “ A constraint propagation approach to probabilistic reasoning,” in *Uncertainty in Artificial Intelligence*, eds. L. M. Kanal and J. Lemmer, Amsterdam: North-Holland, pp. 357-370.

——— (1988), *Probabilistic Reasoning in Intelligence Systems*, San Mateo, CA: Morgan Kaufmann.

Pearl, J. and Verma, T. (1987), “The logic of representing dependencies by directed graphs,” in *Proceedings of the 6th conference of American Association of Artificial Intelligence, American Association of Artifical Intelligence*, pp. 374-379.

Shachter, R. D. (1986), “ Intelligent probabilistic inference,” in *Uncertainty in Artificial Intelligence*, eds. L. M. Kanal and J. Lemmer, Amsterdam: North-Holland, pp. 371-382.

Smith, J. Q. (1989), “ Influence diagrams for statistical modelling,” *Annals of Statistics*, 17, 654-672.

Thiesson, B. (1995) “ Accelerated quantification of Bayesian networks with incomplete data,” in *Proceedings of First International Conference on Knowledge Discovery and Data Mining*, AAAI-Press, pp. 306-311.

——— (1996) “Score and information for recursive exponentional models with incomplete data,” Technical Report R-96-2024, Aalborg University, Dept. of Mathematics and Computer Science.

Van Dyk, D. and Meng, X. L. (1997), “ On the ordering and groupings of conditional maximizations within ECM-type algorithms,” *Journal of Computational and Graphical Statistics*, 6, 202-223.

Verma, T. (1988), “ Causal networks: Semantics and expressiveness,” in *Proceedings of the 4th Workshop on Uncertainty in Artificial Intelligence*, MN: Minneapolis, pp. 352-359.

Wermuth, N. and Lauritzen, S. L. (1983), “Graphical and recursive models for contingency tables,” *Biometrika*, 70, 537-552.

Whittaker, J. (1990), *Graphical Models in Applied Multivariate Statistics*, England: John Wiley.