Parallel Distributed Belief Networks That Learn

Wilson X. Wen Andrew Jennings
w.wen@trl.oz.au a.jennings@trl.oz.au

Artificial Intelligence Systems,
Telecom Research Laboratories
770 Blackburn Rd, Clayton,
Victoria 3168, Australia

Abstract

A parallel distributed computational model for reasoning and learning is discussed based on a belief network paradigm. Issues like reasoning and learning for the proposed model are discussed. Comparisons between our method and other methods are also given.

1 Introduction

Comparing with the model of Connectionist Expert Systems (CES) [Gallant, Feb 1988], Belief Networks (BN) [Lauritzen and Spiegelhalter, 1988; Pearl, 1988] has quite a few advantages, eg. (1) it has the probability theory as its theoretical basis and thus guarantees a consistent or correct result; (2) it allows reasoning in all directions, other methods like CES only allow single reasoning direction, eg. bottom-up [Gallant, Feb 1988]; and (3) it is easy to express higher level probabilistic (Non-functional) domain knowledge in BN while it is difficult in CES.

On the other hand, however, CES model also has its attractive features, eg., (1) it is easy to implement parallel distributed CES; and (2) it is possible to automatically generate CES by a set of learning techniques. These make CES more and more popular in recent years. There were discussions about some vital issues for belief networks and some solutions were also proposed.

- Lauritzen and Spiegelhalter [1988] proposed a BN decomposition method to reduce the computational amount when dependencies exist among the variables. This method provides a potential way to make use of the parallelism inherent in BN.
- Pearl [1988] proposed a parallel model for his Bayesian network which is elegant and efficient but difficult to use in multi-connected networks. A method for learning causal tree structure is also discussed in Pearl's book.
- Herskovits and Cooper [1990] used the Maximum Entropy (ME)/Minimum Cross entropy (MCE) methods to learn conditional probabilities to construct BN. These methods assign uniform distributions to the unseen cases in the training set.
- Laskey [1990] proposed a method to construct Bayes networks using Boltzmann machines [Rumelhart *et al.*, 1986]. While this method seems promising, it is not quite clear for this method how to handle the NP-hardness of connectionist learning [Judd, 1987].

Based on [Lauritzen and Spiegelhalter, 1988; Pearl, 1988], we proposed [Wen, Nov 1989] a highly parallel distributed computational model for BN called Boltzmann-Jeffrey Machine Network (BJMNet). In this paper, we will explore further the capabilities of BJMNet and propose a set of efficient methods of BN learning from Statistical Relational Databases (SRDB). A performance analysis and comparison between our method and other methods are also given.

We only consider the case of binary variables in this paper, but our results can be generalised to the general case containing arbitrary variables straightforwardly.

2 Boltzmann-Jeffrey Machine Networks

Suppose that a system X of m binary random variables x_i $(i = 0, ..., m-1)$ has a set of 2^m states $S = \{s_j | 0 \leq j < 2^m\}$ with distribution $p = \{P(s_j)\}$. If we have a prior distribution $p^{(0)}$ that estimates p and we have evidence which changes our estimation of p to $\hat{p}$, according to the Minimum Cross Entropy (MCE) principle [Kullback, 1968], the best estimate $\hat{p}$ of p satisfying the constraints is the one minimising the cross entropy:

$$H(\{s_j\}) = H(\hat{p}, p^{(0)}) = \sum_{j=0}^{2^m-1} \hat{P}(s_j) \log \frac{\hat{P}(s_j)}{P^{(0)}(s_j)}.$$

According to the values of n distinct variables, x_{i_k}, $(0 \leq i_k < m,\ k = 0, ..., n-1,\ n < m)$, in X, we partition the state space S into 2^n exclusive and exhaustive subspaces called events S_l, $l = 0, ..., 2^n - 1$ such that in each of these events the value of each binary number of n bits $l = x_{i_0} ... x_{i_{n-1}}$ is constant. Suppose we know a prior distribution $p^{(0)}$ and some (not necessarily all) of the probabilities of the above events, $p' \subseteq \{\mu_l = P(S_l) \mid l = 0, ..., 2^n - 1\}$. The MCE posterior distribution which satisfies the constraint set p' is [Wen, July 1989]

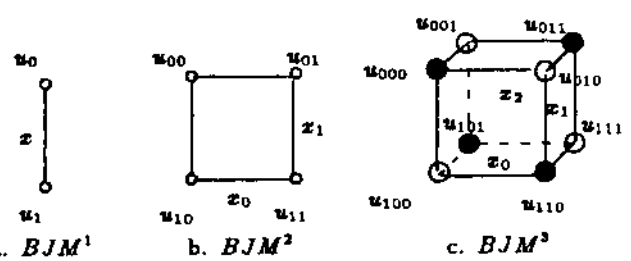
$$\hat{P}(s_j) = \begin{cases} P^{(0)}(s_j) \frac{\mu_l}{P^{(0)}(S_l)} & s_j \in S_l \\ P^{(0)}(s_j) \frac{1 - \sum_{s_l \in p'} \mu_l}{1 - \sum_{s_l \in p'} P^{(0)}(S_l)} & s_j \notin S_l \end{cases} \quad (1)$$

Suppose we know the probabilities of all these events (constraints) and $p^{(0)}$, then the MCE posterior distribution which satisfies the constraint set $\{\mu_l = P(S_l) \mid l = 0, ..., 2^n - 1\}$, is [Lemmer, 1983; Wen, July 1989]

$$\hat{P}(s_j) = P^{(0)}(s_j) \frac{\mu_l}{P^{(0)}(S_l)}, \quad (s_j \in S_l) \qquad (2)$$

This is the well known *Jeffrey's rule* in philosophy and statistics [Jeffrey, 1957], and (1) is called *Partial Normalized Jeffrey's (PNJ) rule*.

Among 2^m equations of Jeffrey's rules (1) and (2) there is no data dependency at all, and the probabilities of all 2^m states can be calculated simultaneously. To fully explore this parallelism inherent in MCE reasoning, we propose a highly parallel computational model, *Boltzmann-Jeffrey Machine (BJM)*. A BJM is a hypercube of 2^m computational units. Each computational unit u_j corresponds to a state s_j in S and needs only three arithmetic operations — addition, multiplication and division — to calculate a state probability of the posterior distribution by (2). Each unit u_j is connected to m neighbors $\{u_{j^*}\}$ $(0 \le j^* < 2^m - 1, |j - j^*| = 2^N, 0 \le N < m-1)$ so that it can receive their probability values to calculate the corresponding prior marginal $P^{(0)}(S_l)$ by accumulating the values $\{P^{(0)}(s_{j^{**}})\}$ together, where $s_{j^{**}} \in \{s_{j^*}\} \setminus S_l$. Three BJMs are shown in Fig. 1.



a. BJM^1 b. BJM^2 c. BJM^3

Figure 1: Boltzmann-Jeffrey Machines

The parallel reasoning procedure in BJM is as follows:

1. Before reasoning, each unit u_j of the BJM is assigned a prior probability $P(u_j) = P^{(0)}(s_j)$.
2. When we observe some evidence which creates a set of constraints $\{P(S_l)\}$, the corresponding prior $\{P^{(0)}(S_l)\}$ is calculated by accumulation of $\{P(s_{j^{**}})\}$, and the results are distributed at each $s_j \in S_l$. For a BJM^m, if we have an n-th order constraint set, then we need $m - n$ addition cycles to complete both accumulation and distribution.
3. Each unit u_j calculates its posterior $\hat{P}(s_j)$ by (2) in parallel using the above results and only one multiplication cycle and one division cycle are needed to complete the computation in this step.

For the BJM^3 in Fig. 1 c, to calculate $P^{(0)}(x_0, x_1)$ in step 2, we need add the corresponding prior of the two planes $\bar{X}_2 : (P(u_{000}), P(u_{100}), P(u_{110}), P(u_{010}))$ and $X_2 : (P(u_{001}), P(u_{101}), P(u_{111}), P(u_{011}))$ to each other in parallel. $3 - 2 = 1$ addition cycle is needed to complete both accumulation and distribution of the results. If we want $P^{(0)}(x_0)$ instead of $P^{(0)}(x_0, x_1)$, then add the corresponding prior of the two planes $\bar{X}_1 : (P(u_{000}), P(u_{001}), P(u_{101}), P(u_{100}))$ and $X_1 : (P(u_{010}), P(u_{011}), P(u_{111}), P(u_{110}))$ to each other, and then add the corresponding results on planes $\bar{X}_2$ and X_2 to each other. Here, we need $3 - 1 = 2$ addition cycles.

Example 1 (XOR Problem): For the exclusive-or (XOR) problem (Table 1), a 3-dimensional BJM (Fig. 1c) is needed. The first step is to store the patterns to be recognised. From Table 1, we can easily obtain the conditional probabilities: $P(x_2|\bar{x}_0, \bar{x}_1) = P(\bar{x}_2|\bar{x}_0, x_1) = P(\bar{x}_2|x_0, \bar{x}_1) = P(x_2|x_0, x_1) = 0$. Thus, $P(\bar{x}_0, \bar{x}_1, x_2) = P(\bar{x}_0, x_1, \bar{x}_2) = P(x_0, \bar{x}_1, \bar{x}_2) = P(x_0, x_1, x_2) = 0$. Using (1) and the above probabilities as constraints to update a flat distribution of $\{x_0, x_1, x_2\}$, we obtain $P(\bar{x}_0, \bar{x}_1, \bar{x}_2) = P(\bar{x}_0, x_1, x_2) = P(x_0, \bar{x}_1, x_2) = P(x_0, x_1, \bar{x}_2) = 0.25$. Thus, we store a probability 0.25 at each of 4 units u_{000}, u_{011}, u_{101}, and u_{110} (black node in Fig. 1c) and store probability 0.0 in each of the other 4 units. Actually, we do not even need these 4 units with 0 probabilities because it is easy to see from (2) that they will always keep the value 0 during any reasoning.

Input			Output	MLP
x_0	x_1	x_2	Output x_2'	Error
0	0	0	0.0095	0.0095
0	1	1	0.9913	0.0087
1	0	1	0.9913	0.0087
1	1	0	0.0109	0.0109

Table 1: The example of exclusive-or

After storing the patterns, the recall is simply performed by (2). The prior probabilities of the input $P(\bar{x}_0, \bar{x}_1) = 0.25$, $P(\bar{x}_0, x_1) = 0.25$, $P(x_0, \bar{x}_1) = 0.25$, $P(x_0, x_1) = 0.25$ can be obtained in only one addition cycle. Suppose we have input $x_0 = 0$ and $x_1 = 0$. The posterior state probabilities can be obtained by Jeffrey's updating in one division and one multiplication cycles. The only non-zero one among them is $\hat{P}(\bar{x}_0, \bar{x}_1, \bar{x}_2) = 0.25 \cdot \frac{1.0}{0.25} = 1.0$, and we have $P(\bar{x}_2) = 1$ after two more addition cycles. That is, the output is $x_2 = 0$. It is easy to test all the other cases using (2).

In summary, we have a time efficient parallel reasoning algorithm for BJM. However, the number of units in a BJM exponentially grows as the number of variables increases although we may save all units which have zero probabilities in BJMs with sparse spaces.

One way to handle the exponential explosion of the number of units in a big BJM is to decompose the BJM into a network of small BJMs (called BJMNet). If we can keep the size of the biggest sub-BJM in the BJMNet fixed, the exponential explosion is avoided.

To keep the consistency of the reasoning results among the small BJMs, it is desirable to organize the BJMNet as an acyclic hypergraph [Beeri *et al.*, July 1983]. Each BJM in the BJMNet corresponds to a hyperedge in the hypergraph and is an autonomous component which has its own prior distribution and always checks its interfaces with its neighbors which affect it. *BJMi* is affected by *BJMj* if *BJMi* is the Running Intersection Predecessor (RIP) or one of the Running Intersection Successors (RIS's, [Beeri *et* al., July 1983]) of *BJMj*. If there

is any belief change in the interfaces it will update its own distribution using (2), and then propagate the belief changes to its other RIP and RIS's. The new changes, in turn, may cause some other BJMs to update their distributions. The network reaches an equilibrium when the gradients of all the original disturbing points are in some small ranges specified beforehand. The network can be reconfigured dynamically using some techniques such as microprogramming to fit it for different applications.

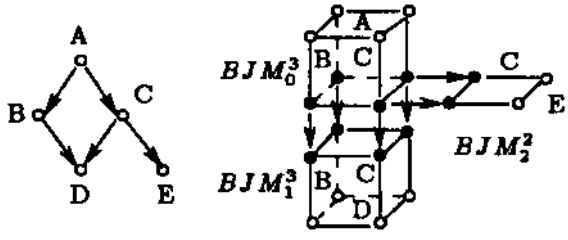
Based on the MCE principle and the theory of Markov fields we have proven [Wen, July 1989]

Theorem 2.1: For an acyclic BJMNet, the following updating procedures are equivalent:

1. Global updating the whole network with single marginal constraint set by (2).
2. Local updating one sub-BJM which contains the constraints and Jeffrey propagation of the result to the whole BJMNet through the running intersections.

Example 2 [Cooper, 1984]: Metastatic cancer (A) is a possible cause of a brain tumor (C) and is also an explanation for increased total serum calcium (B). In turn, either of these could explain a patient falling into a coma (D). Severe headache (E) is also possibly associated with a brain tumor.

The BN and BJMNet for this example are shown in Fig. 2, where the arrows in Fig. 2b show the RIP/RIS relationship between BJMs. The BJM without decomposition has 5 binary variables and thus 32 units. After decomposition using the techniques in [Lauritzen and Spiegelhalter, 1988; Wen, 1990b], the BJM is decomposed into 3 sub-BJMs: {A, B, C}, {B, C, D}, and {C, *E*}. Only $2^3 + 2^3 + 2^2 = 20$ units are needed here. Further saving can be achieved considering that the units in the intersections (the black nodes in Fig. 2b) are never active simultaneously (14 units are needed here).



a. Belief Net b. BJMNet

Figure 2: Belief network and BJMNet For Example 2

Suppose we have an initial distribution in which

$$\begin{array}{llll} P(B|\bar{A}) = 0.2, & P(B|A) = 0.8, \\ P(C|\bar{A}) = 0.05, & P(C|A) = 0.2, \\ P(D|\bar{B},\bar{C}) = 0.05, & P(D|B,\bar{C}) = 0.8, \\ P(D|\bar{B},C) = 0.8, & P(D|B,C) = 0.8 \\ P(E|\bar{C}) = 0.6, & P(E|C) = 0.8, \\ P(A) = 0.2. \end{array}$$

It is easy to calculate the initial marginal distributions for each of the sub-BJMs:

$$\begin{array}{lllll} BJM_0^3\ (P(A,B,C)): & 0.608, & 0.032, & 0.152, & 0.008, \\ & 0.032, & 0.008, & 0.128, & 0.032. \\ BJM_1^3\ (P(B,C,D)): & 0.608, & 0.032, & 0.008, & 0.032, \\ & 0.056, & 0.224, & 0.008, & 0.032. \\ BJM_2^2\ (P(C,E)): & 0.368, & 0.552, & 0.016, & 0.064. \end{array}$$

If we have observed $P(D) = 0$ and $P(E) = 1$, respectively, what is the possibility that a patient suffers from metastatic cancer? The reasoning is as follows:

1. BJM_2^2 uses the constraint $P(E) = 1$ to update its own distribution in parallel and calculates the posterior $P'(C)$, and broadcast it.
2. Only BJM_0^3 is affected by BJM_2^2, and it updates its distribution with $P'(C)$, and calculates the posterior $P'(B,C)$, and broadcast it.
3. BJM_1^3 receives $P'(B,C)$ from BJM_0^3, updates its own distribution in parallel, and calculates posterior $P'(D)$. At this point, we finish the first step of parallel reasoning. Because the gradient $\nabla D = P'(D) - P(D) \neq 0$, the updating continues.
4. BJM_1^3 uses another constraint $P(D) = 0$ to update its distribution, calculates the posterior $P''(B,C)$, and broadcast $P''(B,C)$ to the network.
5. BJM_0^3 and BJM_2^2 receive $P''(B,C)$ and $P''(C)$, respectively, and use them to update their own distributions in parallel. BJM_2^2 calculates $P''(E)$ and finds $\nabla E = P''(E) - P(E) = 0$. The result $P(A) = 0.0973$ is reported and the procedure stops.

3 Learning From Statistical Relational Data

Normally, learning starts from a sample database. Most of the contemporary databases are relational. In [Wen, 1990a], some relationships between Relational Databases (RDB) [Ullman, 1982] and BN are discovered. These include the correspondences between (1) functional dependency ($\rightarrow$) and probabilistic dependency, a special cases of multivalued dependency ($\rightarrow\rightarrow$), in RDB and conditional probability in BN. (2) acyclic database scheme with join dependency in RDB and acyclic decomposition of BN. and (3) three main operations on RDB (projection, selection, and join) and three main operations of BN (belief extracting, updating, and propagation).

Definition 3.1: The *frequency* of an attribute subset X of relation scheme R in a relation r on R is $F_X(R) = \{F_{X=x}(r) = \frac{|\sigma_{X=x}(r)|}{|r|}; \forall x \in D_X\}$ where D_X is the domain of X, $|r|$ is the cardinality of r, and σ is the selection operator in RDB.

Definition 3.2: The *conditional frequency* of X in r given $Y = y$, $Y \subset R$ is

$$F_{X|Y=y} = \{F_{X=x|Y=y}(r) = \frac{|\sigma_{X=x \wedge Y=y}(r)|}{|\sigma_{Y=y}(r)|}; \forall x \in D_x\}.$$

Definition 3.3: Let $X, Y \subset R$, and $Z = R - XY$. r satisfies the *probabilistic dependency* $X \mapsto Y$ if

$$\forall x \in D_X \forall y \in D_Y \forall z \in D_Z, zyz \in r \Longrightarrow$$

$$F_{XYZ=xyz}(r) \cdot F_{X=x}(r) = F_{XY=xy}(r) \cdot F_{XZ=xz}(r).$$

According to the law of large numbers, it is reasonable to assume $\lim_{|r|\to\infty} F_x(r) = P(X = x)$, $\lim_{|r|\to\infty} F_{X=x|Y=y}(r) = P(X = x|Y = y)$, and if $X \mapsto Y$ then $P(Y|XZ) = P(Y|X)$, ie. Y is conditionally independent of Z given X. It is easy to prove

Theorem 3.1 [Wen, 1990a]:

1. $X \to Y \Longrightarrow X \mapsto Y$,
2. $X \mapsto Y \Longrightarrow X \to\to Y$.

(see [Ullman, 1982] for definitions of $\to$ and $\to\to$.)

In Example 2, suppose we have the statistical information in Table 2 from a sample database. Each entry of the table gives the number of occurrences of the records in the database which contain various combinations of the attributes A, B, C, D, and E. This information

	$\bar{D}\bar{E}$	$\bar{D}E$	$D\bar{E}$	DE
$\bar{A}\bar{B}\bar{C}$	23104	34656	1216	1824
$\bar{A}\bar{B}C$	128	512	512	2048
$\bar{A}B\bar{C}$	1216	1824	4864	7296
$\bar{A}BC$	32	128	128	512
$A\bar{B}\bar{C}$	1216	1824	64	96
$A\bar{B}C$	32	128	128	512
$AB\bar{C}$	1024	1536	4096	6144
ABC	128	512	512	2048

Table 2: Statistical information in relation ABCDE

can be obtained by a projection of a universal relation of the database on $\{A, B, C, D, E\}$ without eliminating the duplicates. We can easily check that the probabilistic dependencies $A \mapsto B$, $A \mapsto C$, $B,C \mapsto D$, and $C \mapsto E$ hold. According to Theorem 3.1, we can use the algorithms given in [Ullman, 1982] to decompose relation ABCDE into 4 subrelations AB, AC, BCD, and CE with a lossless join property. All of these relations are in 4th normal form and keep the original dependencies projected to the corresponding relations. Thus, we can readily learn the conditional probabilities for Example 2 given in section 2 from the statistical information (see Table 3) in the decomposed relations using

1. The frequency method: For example, $P(A) = \frac{4000+16000}{4000+16000+64000+16000} = 0.2$ and $P(B|A) = \frac{16000}{4000+16000} = 0.8$, etc.
2. The incremental methods [Lauritzen and Spiegelhalter, 1988]: suppose we have a prior $P(B|A) = \frac{16000}{20000}$ and the next example e comes. The updated value of $P(B|A)$ is

$$P(B|A) = \begin{cases} \frac{16001}{20001}, & e \text{ makes } A \wedge B \text{ true}, \\ \frac{16000}{20001}, & e \text{ makes } A \wedge \bar{B} \text{ true}, \\ \frac{16000}{20000}, & e \text{ makes } \bar{A} \text{ true}. \end{cases}$$

The above learning procedure may be trivial if the database is complete. However, we often cannot obtain a completely specified database in many practical cases. In these cases, we may have to choose one of the following methods to generalize the incomplete statistical information to a complete conditional distribution.

	$\bar{A}$	A
$\bar{B}$	64000	4000
B	16000	16000

(a)

	$\bar{A}$	A
$\bar{C}$	7600	16000
C	4000	4000

(b)

	$\bar{B}\bar{C}$	$\bar{B}C$	$B\bar{C}$	BC
$\bar{D}$	60800	800	5600	800
D	3200	3200	22400	3200

(c)

	$\bar{C}$	C
$\bar{E}$	36800	1600
E	55200	6400

(d)

Table 3: Statistical information in sub-relations

3.1 Floor Method And ME/MCE Method

Floor method assigns a small nonzero probability to the states (tuples) which do not occur in the database. This adds the possibility of Jeffrey updating to these states. However, this method makes no distinction between the cases which are impossible and those which could be encountered in the future.

Similarly, ME/MCE method or Dirichlet distribution [Herskovitz and Cooper, 1990] simply assigns a uniform conditional distribution to the unseen cases. It uses formula $P(X = x|\Pi_X = \pi_X) = \frac{C(X=x,\Pi_X=\pi_X)+1}{C(\Pi_X=\pi_X)+V_X}$ to obtain a conditional distribution, where X is a variable in the underlying BN, x is one of the Vx values can be taken by X_y IIx is the set of parents of X, π_X is a particular instantiation of IIx, and $C(\Phi)$ is the number of tuples in the database that match the instantiated set of variables . For the unseen cases, the above conditional distribution becomes $\frac{1}{V_X}$ and thus is uniform.

3.2 Connectionist Methods

The connectionist learning methods are interesting because of their generalisation ability. It has been proven that the output class probabilities of Multilayer Perceptron (MLP) and Boltsmann machine are good approximations of the conditional probabilities needed by BN [Bourlard and Wellekens, 1990; Laskey, 1990]. In Example 2, using an MLP network with 2 input units, 2 hidden units, and 1 output unit, we can learn *P(D\B, C)* in less than 100 epochs. The results are given in Fig. 4. For example, in Fig. 4a, data for *P(D\B, C)* are missing. After the network has learned the weights from the incomplete data we feed *B* = *C* = 1 to the network and get an output *P(D\B,C)* = 0.82.

3.3 NN/OR Method

According to the Nearest Neighborhood (NN) principle, the conditional probability assigned to an unseen case for the database depends on its neighbor conditional distributions. That is, if the unseen case has many neighbors who have high conditional probabilities then it is assigned a relatively high conditional probability, otherwise, it is assigned a low or even sero conditional probability. This method should be used with the Occam's Rasor (OR) principle when there are two or more nearest neighbors having different conditional probabilities. When there are more than one candidate hypotheses for

a given problem the OR principle chooses the simplest one which is compatible or at least 90% consistent with the observed data. There are many criteria of simplicity, eg. Kolmogorov complexity, Minimum Description Length, Formula complexity, etc. We use formula complexity in the following examples.

Example 3: Suppose we have the training data in Fig. 3a with the functional dependency $x, y, z \rightarrow w$. The

x	y	z	w
0	0	0	0
0	0	1	1
0	1	1	1
1	0	1	0
1	1	0	0
1	1	1	0

a. Data with $x, y, z \rightarrow w$.

$P(w\|x,y,z)$	$\bar{y}\bar{z}$	$\bar{y}z$	$y\bar{z}$	yz
$\bar{x}$	0	1	1	0
x	0	0	0	0

b. The Karnough map

Figure 3: Example 3

information about $P(w|\bar{x}, y, \bar{z})$ and $P(w|x, \bar{y}, \bar{z})$ is missing. ME/MCE methods assign 1/2 to both of these conditional probabilities. This is obviously inconsistent with the assumption of $x, y, z \rightarrow w$. It is easy to determine $P(w|x, \bar{y}, \bar{z}) = 0$ according to the NN principle because all of its nearest neighbors have values 0 (sec Fig. 3b). $P(w|\bar{x}, y, \bar{z})$ is more difficult to determine because its nearest neighbors have different values. Considering Fig. 3b as a Karnough map, and trying to assign 0 and 1 to the entry corresponding to $P(w|\bar{x}, y, \bar{z})$, we have two logic expressions *xz* and *xz* + *xy,* respectively. The OR method always prefers the assignment with the simplest expression and thus assigns $P(w|\bar{x}, y\bar{z}) = 0$.

This method can be generalised to the case of probabilistic dependencies. In Example 2, suppose the statistical information about *P(D\B, C)* (the last column in Table 3c) is missing. According to the NN principle, it is easy to determine *P(D\B,C)* = 0.8 (Fig. 4a). However, if the information about *P(D\B, C)* is missing

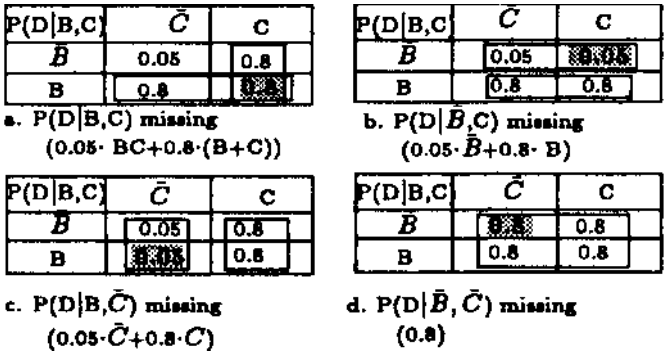


Figure 4: Learning *P(D\B,C)* by OR or MLP

(Fig. 4b), we may have two choices for it: 0.05 and 0.8. Because the former has a probabilistic logic expression 0.05 x $\bar{B}$ + 0.8 x *B* which is simpler than that of the latter $(0.05 \times \bar{B}\bar{C} + 0.8 \times (B + C))$, we prefer the first choice (see Fig. 4b). Similarly, we have the conditional probabilities learned for $P(D|B\bar{C})$ and $P(D|\bar{B}\bar{C})$ when the corresponding information missing (Fig. 4c,d).

4 Performance and Comparisons

A number of typical examples of reasoning have been tested to evaluate the performance of the BJM model. Fig. 4 gives the results of one of our experiments, which uses a 10-dimensional BJM. Fig. 4 shows the results for the reasoning cases with 2,4,8,16 constraints and 1-8 parallel simulated processing units (threads for the Encore Computer). It is easy to see that the speed up is quite satisfactory when the ratio $\frac{No.units}{No.Constraints}$ is small. For greater ratio, the performance suffers from the parallel scheduling overhead a bit.

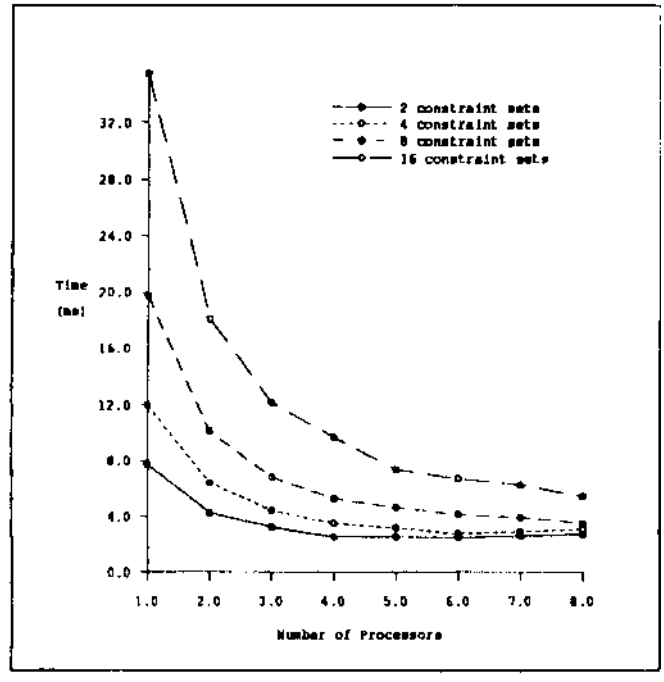


Figure 5: Performance of parallel Jeffrey's updating

Comparing with perceptron models, our model has its obvious advantages. [Minsky and Papert, 1969] showed that no single-layer perceptron can solve XOR problem. For CES/MLP, using back propagation method [Rumelhart *et a*/., 1986], a total sum of squares (tss) 0.0398 is obtained after 298 epochs of training in a feedforward network with 2 input units, 2 hidden units, and 1 output unit. The test result for all the patterns is given in Table 1. From this example, it can be seen that for recall from completely specified distributions of sparse spaces, the BJM model outperforms CES/MLP model in the following aspects: (1) No time-consuming learning procedure is involved. (2) Only addition, multiplication, and division operations but not complicated calculations like sigmoid are needed. (3) The result is accurate. (4) The reasoning direction can be arbitrary. (5) The network structure is much more regular than CES/MLP. The granularity of BJMNet is also good.

For large scale applications, neither MLP nor BJM are Probably Approximately Correctly (PAC) learnable [Judd, 1987]. However, in [Wen, 1990a] we have proven

Theorem 4.1: If the maximum BJM in the BJMNet has a fixed size, the BJMNet is PAC learnable from statistical relational data.

It is possible to decompose an MLP network in a way similar to that of BJMNet. Gallant's connectionist expert systems [Gallant, Feb 1988] may be thought of as one step towards this direction. However, Gallant's method can handle only functional dependencies but not probabilistic dependencies, and does not work properly for problems like Example 2.

5 Conclusions

A parallel distributed model for reasoning/learning is proposed based on BN paradigm. Issues about the structure, learning methods, and the reasoning method for the model are discussed. Comparison between BJMNet and CES shows that the soundness and efficiency of the model can be guaranteed for a wide range of applications:

- The model has probability theory and information theory as its theoretical basis.
- It can be implemented automatically by a set of learning methods from statistical relational data.
- Reasoning in this model can be in any directions, in contrast to the CES models which can only reason in single direction [Gallant, Feb 1988].
- NP-hardness in both reasoning and learning is handled by decomposing the BN into small BJMs.

A prototype of simulator of BJMNet has been developed for the Encore Computer, a shared memory multiprocessor system [Wen, Nov 1989].

Acknowledgement

Thanks to E. A. Sonenberg, B. Marksjo, A. Kowalczyk, H. Liu, and L. Fang for discussions and comments. The permission of the Executive General Manager, TRL, to publish this paper is gratefully acknowledged.

References

[Beeri *et al,* July 1983] Catriel Beeri, Ronald Fagin, David Maier, and Mihalis Yannakakis. On the desirability of acyclic database schemes. *Journal of the ACM,* 30(3):479-513, July 1983.

[Bourlard and Wellekens, 1990] H. Bourlard and C. J. Wellekens. Links between markov models and multilayer perceptrons. *IEEE Trans. Pattern Analysis and Machine Intelligence,* to appear, 1990.

[Cooper, 1984] G. F. Cooper. NESTOR: a computer-based medical diagnostic aid that integrates causal and probabilistic knowledge. *Report HPP-84-48, Stanford University,* 1984.

[Gallant, Feb 1988] S. I. Gallant. Connectionist expert systems. *Communications of the ACM,* 31(2):152-169, Feb. 1988.

[Herskovits and Cooper, 1990] E. Herskovits and G. F. Cooper. Kutato: An entropy-driven system from construction of probabilistic expert systems from databases. In M. Henrion and Bonissone, editors, *Proc. 6th International Conf. on Uncertainty in AI,* Cambridge, MA., July 1990. North Holland.

[Jeffrey, 1957] R. Jeffrey. Contributions to the theory of inductive probability. *Ph. D. thesis, Department of Philosophy, Princeton University,* 1957.

[Judd, 1987] S. Judd. Learning in networks is hard. In *Proc. IEEE First International Conf. on Neural Networks,* volume 2, pages 685-692, San Diego, CA, Jun 1987. IEEE Press.

[Kullback, 1968] S. Kullback. *Information Theory and Statistics.* Dover Publication, Inc., New York, 1968.

[Laskey, 1990] K. B. Laskey. Adapting connectionist learning to bayes networks. *International Journal of Approximate Reasoning,* 4:261-282, 1990.

[Lauritzen and Spiegelhalter, 1988] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *J. R. Statist. Soc. B,* 50(2), 1988.

[Lemmer, 1983] J. F. Lemmer. Generalized bayesian updating of incompletely specified distributions. *Large Scale Systems,* 5, 1983.

[Minsky and Papert, 1969] M. Minsky and S. Papert. *Percepetrons.* MIT Press, 1969.

[Pearl, 1988] J. Pearl. *Probabilistic Reasoning In Intelligeni Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., San Mateo, California, 1988.

[Rumelhart *et al.,* 1986] D. E. Rumelhart, J. L. McClelland, and P. D. P. Group. *Parallel Distributed Processing, Explorations in Micro structure of Cognition.* MIT Press, Cambridge MA & London, 1986.

[Ullman, 1982] J. D. Ullman. *Principles of Database Systems.* Computer Science Press, Rockville, Maryland, 1982.

[Wen, 1990a] W. X. Wen. Learning from statistical relational data. Tech. report, AI Systems, Telecom Research Labs., 770 Blackburn Rd. Clayton, Vic. 3168, Australia., 11 1990.

[Wen, 1990b] W. X. Wen. Optimal decomposition of belief networks. In Max Herion, Piero Bonissone, J. F. Kanal, and J. F. Lemmer, editors, *to appear in UNcertainty in Artificial Intelligence 6.* North Holland, 1990.

[Wen, July 1989] W. X. Wen. Markov and Gibbs Fields and MCE Reasoning in General Belief Networks. *Technical Report 89/13, Computer Science, The University of Melbourne,* July 1989.

[Wen, Nov 1989] W. X. Wen. Parallel MCE Reasoning in Recursive Causal Networks, *in Proc. 1989 IEEE International Conference on Systems, Man, and Cybernetics, Boston, MA, USA,* Nov. 1989.