Vol. 38 No. 4 Transactions of Information Processing Society of Japan Apr. 1997

Regular Paper

On Information of Logical Expression and Knowledge Refinement

NING ZHONG† and SETSUO OHSUGA††

In machine learning, information theory has been recognized as a useful criterion, and several algorithms such as ID3 and Prism have been developed. In these methods, however, information theory is only used for generating inductively knowledge that is represented in decision tree or *if-then* rule, but the issue on knowledge refinement is not considered. Moreover, they are not used for evaluating information of logical expression. This paper discusses a way of calculating quantitatively information of logical expression and its application for refining concept clusters discovered from a database. The calculation is based on the model representation of Multi-Layer Logic (MLL) with the hierarchical structure. Its key feature is the quantitative evaluation for selecting the best representation of the MLL formula by using cooperatively a criterion based on information theory (entropy) and domain knowledge.

1. Introduction

In machine learning, information theory has been recognized as a useful criterion, and several algorithms such as ID3 and Prism have been developed[1),14)]. In these methods, however, information theory is only used for generating inductively knowledge that is represented in decision tree or *if-then* rule, but the issue on knowledge refinement is not considered. Moreover, they are not used for evaluating information of logical expression. On the other hand, the idea of calculating quantitatively information of logical expression has been originally introduced by Ohsuga in his paper[9)]. Although the issues on knowledge representation were discussed from an information theoretic view in that paper, the effective method for an implementation was not described and some theoretical issues related to the implementation were not discussed/proved formally.

This paper concentrates on an effective method of calculating quantitatively information of logical expression, some theoretical issues related to the implementation, and its application for refining concept clusters discovered from a database. The calculation is based on the model representation of Multi-Layer Logic (MLL) with the hierarchical structure[9),10)]. Its key feature is the quantitative evaluation for selecting the best representation of the MLL formula by using cooperatively a criterion based on information theory (entropy) and domain knowledge. The calculation constitutes a theoretical basis of developing Hierarchical Model Learning (HML) that is a sub-system of our GLS discovery system for generating, refining, and managing the knowledge discovered from databases[19),21)].

The remainder of this paper contains a detailed explanation of our method. In Section 2 knowledge representation using MLL is briefly described as a preparation, and information of logical expression including a definition and three theorems is discussed in Section 3. In Section 4, an effective algorithm for calculating the information of the MLL formula is described. How to refine concept clusters discovered from a breast cancer database, as an application of our method, is discussed in Section 5. Lastly, concluding remarks are given in Section 6.

2. Knowledge Representation in MLL

MLL (Multi-Layer Logic) is a predicate logic with a syntax that allows some domain(s) of variable(s) to be the variable(s), which extends for MSL (Many-Sorted Logic) in the syntax[9),10)]. This extension in the syntax of MSL gives a great expressive capability for a predicate logic involving data structure (set, hierarchy, power set, etc.), especially in manipulation of the hierarchical structure.

In MLL, structures can be described as *element-of*, *power-set-of*, *component-of*, and *product-set-of* relations. Other complex operations can be represented as combinations of these primary operations. For example, let a polyhedron as shown in **Fig. 1** be defined as a

† Department of Computer Science and Systems Engineering, Faculty of Engineering, Yamaguchi University
†† Department of Information and Computer Science, School of Science and Engineering, Waseda University

set of surfaces s_1, s_2, s_3, s_4, and let surfaces be defined by a set of the edge lines using the following structure description,

```
/* Description of general concepts */
!make_p *2vertex, *2line, *2surface;
!ins_e *2vertex  line;
!ins_e *2line  surface;
!ins_e *2surface  polyhedron;
/* Description of specific concepts */
!ins_e line  l1, l2, l3, l4, l5, l6;
!ins_e surface  s1, s2, s3, s4;
!ins_e polyhedron  h1;
/* Definition of component sets of a specific
   object with hierarchical structure */
!ins_e h1 : surface  s1, s2, s3, s4;
!ins_e s1 : line  l1, l2, l4;
!ins_e s2 : line  l1, l3, l5;
!ins_e s3 : line  l2, l3, l6;
!ins_e s4 : line  l4, l5, l6;
```

where "!*ins_e* **x** $x_1 \ldots x_n$;" means $x_1 \ldots x_n$ are elements of **x** (i.e., the set-elements relation). "***x**" denotes a power set node whose base set is **x**. The base set of the power set is the one from which the extension of the power set is defined. In other words, a power set is composed of all subsets of the base set. However, MLL does not automatically enumerate all elements of the given power set from the given base set. "!*ins_e* ***x**" defines only parts of members of ***x** (i.e., a subset of x) by the arguments followed by !*ins_e* ***x**. Since ***x** is itself a set, *(***x**) can also be defined in the same way, denoted by ***2x**. In general, ***nx** denotes the power set of $*(\mathbf{n}-\mathbf{1})\mathbf{x}$. "!*ins_e* **x:a**" describes a component set of **x** (i.e., **a** is a discriminator of the component set). In addition, "!*make_p*" is used for declaring and making power set nodes.

In general, the structure description can be divided into three parts as shown above. That is, the description of general concepts, the description of specific concepts and the definition of component sets (or IS-A relations) of a specific object with hierarchical structure. A component set can be regarded as an IS-A relation (i.e., pseudo IS-A). MLL prepares a syntax to discriminate the real IS-A relation and the pseudo IS-A relation. But it is abbreviated here (i.e., both of them are called the IS-A hierarchy in this paper). **Figure 2** is an equivalent graph of the component sets of this specific object shown above.

A MLL formula consists of a matrix, prefix, AND/OR forms, connectors *and* (&), *or* (|) and

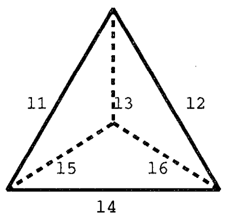


Fig. 1 A polyhedron.

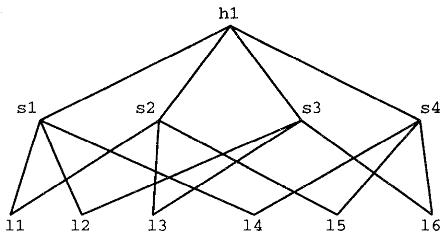


Fig. 2 The hierarchical structure of a polyhedron.

not (~). Similar to MSL, a variable in a MLL formula can have its own domain and can be explicitly included in the prefix. For example, by means of the IS-A hierarchy defined above, we can represent the knowledge "There is some surface in a polyhedron h_1 of which the length of all edge lines is 3" in a MLL formula as follows,

$$[\exists\, S/h_1 : \mathit{surface}][\forall\, L/S : \mathit{line}]\ \mathit{length}(L, 3).$$

The part inside the brackets [ ] in the head of a logic formula is called the prefix in the MLL formula. Here, the domain of variable L is a variable S, the domain of S is h_1 for representing a specific polyhedron. Furthermore, when the domain set of a variable is finite, the MLL formula can be expanded according to the following equivalent expressions,

$$[\forall\, X/\mathbf{x}]p(X) \cap \mathbf{x} = \{x_1, x_2, \ldots, x_n\} \longleftrightarrow p(x_1) \cap p(x_2) \cap \ldots \cap p(x_n),$$
$$[\exists\, X/\mathbf{x}]p(X) \cap \mathbf{x} = \{x_1, x_2, \ldots, x_n\} \longleftrightarrow p(x_1) \cup p(x_2) \cup \ldots \cup p(x_n).$$

It is called *expansion function of MLL*. This function is used for extracting from a set the elements that possess specified properties. It is syntactically defined by appending "#" after the variable to be expanded in the prefix of the MLL formula. For example, let surfaces be $\{(l_1, l_2, l_4), (l_1, l_3, l_5), (l_2, l_3, l_6), (l_4, l_5, l_6)\}$, then the formula

$[\exists\, S\#/h_1 : surface][\forall\, L\#/S : line]$
$length(L, 3)$.

can be expanded into

$(length(l_1, 3) \cap length(l_2, 3) \cap length(l_4, 3)) \cup$
$(length(l_1, 3) \cap length(l_3, 3) \cap length(l_5, 3)) \cup$
$(length(l_2, 3) \cap length(l_3, 3) \cap length(l_6, 3)) \cup$
$(length(l_4, 3) \cap length(l_5, 3) \cap length(l_6, 3))$.

3. Information of Logical Expression

Let us consider a predicate f and let d be a finite base set. For simplicity, we assume that f is a single place predicate $f(x)$. It gives a description on an object in d. Or, in other words, $f(x)$ classifies all elements in the set d into two classes: those that satisfy $f(x)$ and those that do not. In the following, $f(x)$ and $\overline{f}(x)$ mean "$f(x)$: True" and "$\overline{f}(x)$: False" respectively for $x \in d$. Let us define a concept "the state of d before and after the formula". It is assumed that in the prior state, whether $f(x)$ or $\overline{f}(x)$ is not clear for any x in d, while, in the posterior state, either $f(x)$ or $\overline{f}(x)$ is made clear for some or all elements in d. Based on the preparation, we first define the information of MLL.

Definition 1. Information of MLL.
Let $d = \{a_1, a_2, \ldots, a_N\}$ be a finite base set. The state of d is defined as the conjunctions of either $f(a_i)$ or $\overline{f}(a_i)$ for every element a_i in d. Before the formula is given, the state of d includes all possibilities of combinations of $f(a_i)$ and $\overline{f}(a_i)$, $i = 1, 2, \ldots, N$ such that $S_1 : \overline{f}(a_1) \wedge \overline{f}(a_2) \wedge \ldots \wedge \overline{f}(a_N)$ through $S_{2^N} : f(a_1) \wedge f(a_2) \wedge \ldots \wedge f(a_N)$. Let the set S_f^1 be defined as the collection of all possible prior states, and the set S_f^2 be defined as the collection of all possible posterior states. Thus $S_f^1 = \{S_1, \ldots, S_{2^N}\}$. When the formula f is given, the states of some of elements are fixed. Then S_f^2 becomes a subset of S_f^1 as shown in **Fig. 3**. Furthermore, let their cardinalities be $|S_f^1|$ and $|S_f^2|$, and their entropies be defined as $I_{S_{f1}} = \log |S_f^1|$ and $I_{S_{f2}} = \log |S_f^2|$ respectively. Thus, the amount of information of the MLL formula f can be defined by

$$K = I_{Sf1} - I_{Sf2} = \log |S_f^1| - \log |S_f^2|. \qquad (1)$$

That is, the difference of I_{Sf1} and I_{Sf2} is the amount of information K with respect to the predicate symbol f. □

From Eq. (1), we can see that more information is obtained by decreasing the posterior entropy I_{Sf2} of a MLL formula. Moreover, the above definition can be easily extended to the case of n place predicate with n greater than one[9].

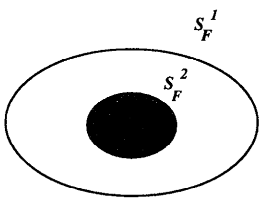


Fig. 3 The sets of the prior and the posterior states of MLL.

Example 1. Assume that we have an IS-A hierarchy shown in Section 2 and a MLL formula is given in the formal quantifiers Q_i,

$[Q_1\, S\#/h_1 : surface][Q_2\, L\#/S : line]$
$length(L, 3)$,

where Q_i $(i = 1, 2)$ denotes either $\forall$ or $\exists$. Then $d = \{l_1, l_2, \ldots, l_6\}$ and the different quantifiers in the prefix of this MLL formula have different amounts of information. Furthermore, let $K_{Q_1Q_2}$ denote the amount of information of a MLL formula with quantifiers Q_1Q_2. Before the formula, the predicate $length(l_i, 3)$ is either true or false for all possible states. Therefore, $|S_f^1| = 2^6$ and $\log |S_f^1| = 6$. Thus,

(1) $[\forall\, S\#/h_1 : surface][\forall\, L\#/S : line]$
$length(L, 3)$,
$K_{\forall\forall} = 6 - \log 1 = 6;$

This formula says that the length of all lines in all surfaces of a polyhedron h_1 is 3. That is, only $length(l_1, 3) \wedge \ldots \wedge length(l_6, 3)$ is allowed as a particular state. Thus $|S_f^2| = 1$.

On the other hand, the formula in case (2) says that the length of some line in some surface of a polyhedron h_1 is 3. That is, this formula allows every element in S_f^1 except $\overline{length}(l_1, 3) \wedge \ldots \wedge \overline{length}(l_N, 3)$. Thus $|S_f^2| = 63$.

(2) $[\exists\, S\#/h_1 : surface][\exists\, L\#/S : line]$
$length(L, 3)$,
$K_{\exists\exists} = 6 - \log 63 = 0.023;$

Furthermore, the following formulae say that the length of some line in all surfaces of a polyhedron h_1 (i.e., case (3)), and the length of all lines in some surface of a polyhedron h_1 (i.e., case (4)), are 3, respectively. That is, the formulae allow a part of elements in S_f^1, and the numbers of possible posterior states become 41 and 23 respectively.

(3) $[\forall\, S\#/h_1 : surface][\exists\, L\#/S : line]$ $length(L, 3)$,
$K_{\forall\exists} = 6 - \log 41 = 0.642;$

(4) $[\exists\, S\#/h_1 : surface][\forall\, L\#/S : line]$ $length(L, 3)$,
$K_{\exists\forall} = 6 - \log 23 = 1.476.$ □

From this example we can see that

- the MLL formulae with different quantifiers may reveal various different information even if the structure is the same. This is one of two aspects of evaluating the information of the MLL formula. Another aspect is that of evaluating a MLL formula with different structures. This aspect will be discussed later.
- the calculation of the posterior states (e.g., cases (3) and (4)), in general, is difficult. Hence, how to develop an effective method for this calculation is a key problem for evaluating the information of the MLL formula. For this purpose, we developed three theorems and the equations based on one of the theorems (i.e., Theorem 1 and Eqs. (9), (10) and (11) to be described later, and the calculation for cases (3) and (4) will be described in Example 3).

In the rest of the section, we would like to discuss the theorems based on Definition 1 for evaluating effectively the information of the MLL formula.

Theorem 1. Complement of the posterior cardinality of MLL.

The posterior cardinalities of the MLL formulae with contrary quantifiers in their prefixes are complementary. That is,

$$|S_f^2|_{\forall\exists} = |S_f^1| - |S_f^2|_{\exists\forall_{\overline{f}}}, \qquad (2)$$

where $|S_f^2|_{\forall\exists}$ is the cardinality for the MLL formula with the form

$[\forall\, Y/*d][\exists\, X/Y]f(X)$,

and $|S_f^2|_{\exists\forall_{\overline{f}}}$ is the cardinality for the MLL formula with the form

$[\exists\, Y/*d][\forall\, X/Y] \sim f(X)$.

Furthermore, if the possibilities of $f(X)$ and $\sim f(X)$ are equiprobable, then

$$|S_f^2|_{\forall\exists} = |S_f^1| - |S_f^2|_{\exists\forall}. \qquad (3)$$

Proof. When the domain sets are finite and without loss of generality, let an IS-A hierarchy be defined as

!ins_e *d $d_1, d_2, \ldots, d_m$;
!ins_e d_1 $a_1, a_2, \ldots, a_l$;
!ins_e d_2 $a_1, a_3, a_5, \ldots, a_j$;
......
!ins_e d_m $a_2, a_3, a_5, \ldots, a_k$;

where elements a_1, a_2, a_3, a_5 in $a_1, a_2, \ldots a_n$ are "tangled" elements (i.e., one a_i can belong to more than one d_j). And let a MLL formula with the prefix $\forall\exists$ be

$[\forall\, Y/*d][\exists\, X/Y]f(X)$.

Let the probability $P([Q_1\, Y/*d][Q_2\, X/Y]\, f(X))$ be $P_{Q_1Q_2}$, and let $f(a_i)$ be simply represented as b_i. Based on Definition 1, the posterior probability of this MLL formula is

$$\begin{aligned} P_{\forall\exists} = P((&b_1 \cup b_2 \cup \ldots \cup b_l) \\ &\cap (b_1 \cup b_3 \cup b_5 \cup \ldots \cup b_j) \\ &\cap \ldots \cap (b_2 \cup b_3 \cup b_5 \ldots \cup b_k)). \end{aligned}$$

And let $P(\sim formula)$ be $\overline{P}(formula)$, then because

$$\begin{aligned} \overline{P}_{\forall\exists} &= P(\overline{(b_1 \cup b_2 \cup \ldots \cup b_l) \cap} \\ &\qquad \overline{(b_1 \cup b_3 \cup b_5 \cup \ldots \cup b_j)} \\ &\qquad \overline{\cap \ldots \cap (b_2 \cup b_3 \cup b_5 \ldots \cup b_k))} \\ &= P(\overline{b_1 \cup b_2 \cup \ldots \cup b_l} \\ &\qquad \cup \overline{b_1 \cup b_3 \cup b_5 \cup \ldots \cup b_j} \\ &\qquad \cup \ldots \cup \overline{b_2 \cup b_3 \cup b_5 \cup \ldots \cup b_k}) \\ &= P((\overline{b}_1 \cap \overline{b}_2 \cap \ldots \cap \overline{b}_l) \\ &\qquad \cup (\overline{b}_1 \cap \overline{b}_3 \cap \overline{b}_5 \cap \ldots \cap \overline{b}_j) \\ &\qquad \cup \ldots \cup (\overline{b}_2 \cap \overline{b}_3 \cap \overline{b}_5 \cap \ldots \cap \overline{b}_k)) \\ &= P_{\exists\forall_{\overline{f}}}, \end{aligned} \qquad (4)$$

$$P_{\forall\exists} + \overline{P}_{\forall\exists} = P_{\forall\exists} + P_{\exists\forall_{\overline{f}}} = 1.$$

Furthermore, if the probabilities of $P(f(X))$ and $P(\sim f(X))$ are equiprobable, then because

$$\begin{aligned} \overline{P}_{\forall\exists} &= P((\overline{b}_1 \cap \overline{b}_2 \cap \ldots \cap \overline{b}_l) \\ &\qquad \cup (\overline{b}_1 \cap \overline{b}_3 \cap \overline{b}_5 \cap \ldots \cap \overline{b}_j) \\ &\qquad \cup \ldots \cup (\overline{b}_2 \cap \overline{b}_3 \cap \overline{b}_5 \cap \ldots \cap \overline{b}_k)) \\ &= P((b_1 \cap b_2 \cap \ldots \cap b_l) \\ &\qquad \cup (b_1 \cap b_3 \cap b_5 \cap \ldots \cap b_j) \\ &\qquad \cup \ldots \cup (b_2 \cap b_3 \cap b_5 \cap \ldots \cap b_k)) \\ &= P_{\exists\forall}, \end{aligned} \qquad (5)$$

$$P_{\forall\exists} + \overline{P}_{\forall\exists} = P_{\forall\exists} + P_{\exists\forall} = 1.$$

Thus, since the posterior cardinalities for the MLL formulae with the prefixes $\forall\exists$ and $\exists\forall$ are

$$|S_f^2|_{\forall\exists} = P_{\forall\exists} \times H \tag{6}$$

and

$$|S_f^2|_{\exists\forall} = P_{\exists\forall} \times H \tag{7}$$

respectively, where H is the maximal number of possible elements and H is also the prior cardinality (i.e., $|S_f^1| = H$),

$$\begin{aligned}|S_f^2|_{\forall\exists} &= P_{\forall\exists} \times H \\ &= (1 - P_{\exists\forall_{\overline{f}}}) \times H \\ &= H - P_{\exists\forall_{\overline{f}}} \times H \\ &= |S_f^1| - |S_f^2|_{\exists\forall_{\overline{f}}},\end{aligned}$$

and if the probabilities of $P(f(X))$ and $P(\sim f(X))$ are equiprobable, then

$$\begin{aligned}|S_f^2|_{\forall\exists} &= P_{\forall\exists} \times H \\ &= (1 - P_{\exists\forall}) \times H \\ &= |S_f^1| - |S_f^2|_{\exists\forall}. \qquad \square\end{aligned}$$

By the same method stated above, we can also prove

$$|S_f^2|_{\exists\exists} = |S_f^1| - |S_f^2|_{\forall\forall}.$$

Furthermore, since

$$\begin{aligned}K &= \log|S_f^1| - \log|S_f^2| \\ &= \log H - \log P \times H \\ &= \log\frac{H}{P \times H} \\ &= \log\frac{1}{P},\end{aligned} \tag{8}$$

and based on Theorem 1, we can calculate the information of the MLL formula by

$$K_{\forall\exists} = \log\frac{1}{1 - P_{\exists\forall_{\overline{f}}}} \tag{9}$$

or

$$K_{\forall\exists} = \log\frac{1}{1 - P_{\exists\forall}} \tag{10}$$

for the formula with quantifiers $\forall\exists$, and

$$K_{\exists\forall} = \log\frac{1}{P_{\exists\forall}} \tag{11}$$

for the formula with quantifiers $\exists\forall$.

In order to calculate $P_{\exists\forall}$, the following Eq. (12) is used if there are "tangled" elements among sub-sets of the base set of an IS-A hierarchy (e.g., the calculation of Eq. (4) or (5)),

$$\begin{aligned}&P(e_1 \cup e_2 \cup \ldots \cup e_{n-1} \cup e_n) \\ &= \sum_{i=1}^{n} P(e_i) - \sum_{1 \le i < j \le n} P(e_i \cap e_j) \\ &\quad + \sum_{1 \le i < j \le n} P(e_i \cap e_j \cap e_k) - \ldots \\ &\quad + (-1)^{n-1} P(e_1 \cap e_2 \cap \ldots \cap e_n),\end{aligned} \tag{12}$$

else Eq. (13) is used.

$$\begin{aligned}&P(e_1 \cup e_2 \cup \ldots \cup e_{n-1} \cup e_n) \\ &= 1 - \overline{P}(e_1 \cup e_2 \cup \ldots \cup e_{n-1} \cup e_n) \\ &= 1 - P(\overline{e}_1 \cap \overline{e}_2 \cap \ldots \cap \overline{e}_{n-1} \cap \overline{e}_n).\end{aligned} \tag{13}$$

From Eqs. (9), (10) and (11), we can further see another aspect of evaluating the information of the MLL formula, that is, a MLL formula with different structures may also reveal various different information.

Theorem 2. The equivalence of information of MLL.

In the case of the IS-A hierarchy, it is possible to refine a hierarchical structure by defining new intermediate nodes. Then the prefix sequence becomes longer. If the same quantifiers appeared in succession in the prefix of a MLL formula such as

(1-1) $[Q\, X^{n-1}/s][\exists\, X^{n-2}/X^{n-1}]\ldots [\exists\, X^2/X^3][\exists\, X^1/X^2]f(X^1)$,

(2-1) $[Q\, X^{n-1}/s][\forall\, X^{n-2}/X^{n-1}]\ldots [\forall\, X^2/X^3][\forall\, X^1/X^2]f(X^1)$,

then they can be respectively regarded as

(1-2) $[Q\, Y/s][\exists\, X/Y]f(X)$,

(2-2) $[Q\, Y/s][\forall\, X/Y]f(X)$,

when calculating their information.

Proof. We would like to prove Theorem 2, only prove the information of the formulae (1-1), (1-2) and (2-1), (2-2),

$$K_{(1-1)} = K_{(1-2)} \quad \text{and} \quad K_{(2-1)} = K_{(2-2)}.$$

Furthermore, based on Eq. (8), only prove their probabilities

$$P_{Q\exists\ldots\exists} = P_{Q\exists} \quad \text{and} \quad P_{Q\forall\ldots\forall} = P_{Q\forall}.$$

Without loss of generality, let the prefix of a MLL formula involves n same quantifiers appeared in succession, and let the base set d involves M elements. It is clear that

$$\begin{aligned}P_{\exists_1\exists_2\ldots\exists_n} &= P_{\exists_1\exists_2\ldots\exists_{n-1}} = \ldots = P_{\exists\exists} = P_{\exists} \\ &= \frac{2^M - 1}{2^M}, \\ P_{\forall_1\forall_2\ldots\forall_n} &= P_{\forall_1\forall_2\ldots\forall_{n-1}} = \ldots = P_{\forall\forall} = P_{\forall} \\ &= \frac{1}{2^M}.\end{aligned}$$

Based on this, the following equalities also hold,

$$\begin{aligned}P_{\forall\exists_1\exists_2\ldots\exists_n} &= P_{\forall\exists_1\exists_2\ldots\exists_{n-1}} \\ &= \ldots = P_{\forall\exists\exists} = P_{\forall\exists},\end{aligned}$$

$$P_{\exists\forall_1\forall_2\ldots\forall_n} = P_{\exists\forall_1\forall_2\ldots\forall_{n-1}} = \ldots = P_{\exists\forall\forall} = P_{\exists\forall}. \qquad \square$$

Theorem 2 shows the equivalence of information of MLL with the same quantifiers appeared in succession in the prefix of a formula. It can be used for convenience in the calculation.

Theorem 3. Information of Tautology of MLL.

Let K_A and K_B denote respectively the amounts of information of the MLL formulae A and B with different quantifiers, and let $A \Rightarrow B$ denote that if A is true then B is true, we have

IF $A \Rightarrow B$, THEN $K_A \geq K_B$.

Proof. Assume this is not true. Then there must be some B such that $A \Rightarrow B$ and $\sim (K_A \geq K_B)$ or $A \Rightarrow B$ and $K_A < K_B$. By Definition 1, we have $K_A = I_{S_{A1}} - I_{S_{A2}}$. Without loss of generality, let $I_{S_{A1}} = I_{S_{B1}}$. Then $K_A < K_B$ is equivalent with $I_{S_{A2}} > I_{S_{B2}}$ or $S_A^2 > S_B^2$. This means that there is some posterior state which is included in S_A^2 but S_B^2, for which A is true but B is not true. This is contradiction. $\square$

It is possible to evaluate the amounts of information of the MLL formulae with the different sequence of quantifiers in the prefix. For example, let the quantifiers set of the MLL formulae $\mathbf{Q} = \{Q_1, Q_2, Q_3\}$, and Q_i (i=1,2,3) denotes either $\forall$ or $\exists$. Based on this, we define a state graph of quantifiers of the MLL formulae as shown in **Fig. 4**, and let d be a finite base set for presenting an IS-A hierarchy and let f be a predicate. Thus, Fig. 4 denotes all different combination of quantifiers of the MLL formula,

$$[Q_1\ Z/d^3][Q_2\ Y/Z][Q_3\ X/Y]f(X).$$

And a partially ordered set of $\mathbf{Q}$ is $\langle Power(\mathbf{Q}), \subseteq\rangle$, i.e.,

$$Power(\mathbf{Q}) = \{\forall\forall\forall, \forall\forall\exists, \forall\exists\forall, \exists\forall\forall, \forall\exists\exists, \exists\forall\exists, \exists\exists\forall, \exists\exists\exists\}.$$

Thus, we can see that Fig. 4 is just its Hasse diagram presented a lattice structure. Therefore, the cover relation of $Power(\mathbf{Q})$ is

$$covPower(\mathbf{Q}) = \{\langle\exists\exists\exists, \exists\exists\forall\rangle, \langle\exists\exists\exists, \exists\forall\exists\rangle, \langle\exists\exists\exists, \forall\exists\exists\rangle, \langle\exists\exists\forall, \exists\forall\forall\rangle, \langle\exists\exists\forall, \forall\exists\forall\rangle, \langle\exists\forall\exists, \exists\forall\forall\rangle, \langle\exists\forall\exists, \forall\forall\exists\rangle, \langle\forall\exists\exists, \forall\exists\forall\rangle, \langle\forall\exists\exists, \forall\forall\exists\rangle, \langle\exists\forall\forall, \forall\forall\forall\rangle, \langle\forall\exists\forall, \forall\forall\forall\rangle, \langle\forall\forall\exists, \forall\forall\forall\rangle\}.$$

Based on this, let $\langle B, A\rangle$ denote any cover relation in $covPower(\mathbf{Q})$. Thus, based on Theorem 3, we have $K_B \leq K_A$ in which the the formulae B, A satisfy $\langle B, A\rangle$. Furthermore, according to Eq. (8), then $K_B \leq K_A$ is equivalent with $P_B \geq P_A$, i.e.,

$$P_{\exists\exists\exists} \geq P_{\exists\exists\forall}$$
$$P_{\exists\exists\exists} \geq P_{\exists\forall\exists}$$
$$P_{\exists\exists\exists} \geq P_{\forall\exists\exists}$$
......
$$P_{\exists\forall\forall} \geq P_{\forall\forall\forall}$$
$$P_{\forall\exists\forall} \geq P_{\forall\forall\forall}$$
$$P_{\forall\forall\exists} \geq P_{\forall\forall\forall}.$$

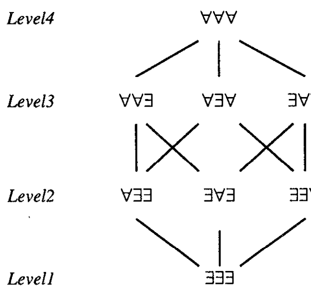


Fig. 4 A state space of quantifiers of MLL.

Example 2. According to Theorem 3, because

$[\forall\ S\#/poly : surface][\forall\ L\#/S : line]$
$length(L, 3) \supset$
$[\forall\ S\#/poly : surface][\exists\ L\#/S : line]$
$length(L, 3)$,

$K_{\forall\forall} > K_{\forall\exists}$. $\square$

In general, we can create a state graph of quantifiers of the MLL formula for learning its quantifiers. The state graph as shown in Fig. 4 is the one in which the number of quantifiers is equal to 3. This state graph is divided into four levels. The uppermost level is with the most information.

4. An Algorithm

Based on the definition and theorems stated above, we developed an effective algorithm for calculating the information of the MLL formula. At first, let us only consider how to calculate the amounts of information of the MLL formulae with the prefixes $\forall\forall$, $\forall\exists$, $\exists\forall$ or $\exists\exists$. That is,

- If we would like to calculate the amounts of information of the MLL formulae with the prefixes $\forall\forall$, $\forall\exists$, $\exists\forall$ and $\exists\exists$, only calculate $\forall$ and $\exists\forall$ according to Theorem 1 and Theorem 2. And the calculation for $\exists\forall$ is in Eq. (11).
- If the quantifiers in the MLL prefix either

$\forall\forall$, $\forall\exists$, $\exists\forall$ or $\exists\exists$ can be used, select $\forall\forall$ according to Theorem 3.

Furthermore, Eq. (12) or (13) can be easily used for calculating the probability of the sum of "tangled" or dependent elements. That is, if we calculate the amount of information of the MLL formula with the prefix $\forall\exists$, then

<u>*step* 1</u> : Calculate the probability of the MLL formula with the prefix $\exists\forall$, $P_{\exists\forall_{\overline{f}}}$ or $P_{\exists\forall}$, in Eq. (12) or Eq. (13) according to whether there are "tangled" elements among sub-sets of the base set of an IS-A hierarchy.

<u>*step* 2</u> : Calculate the amount of information K in Eq. (9) or (10).

Example 3. We would like to use cases (3) and (4) described in Example 1 as an example of using this algorithm and let $length(l_i, 3)$ be simply represented as b_i. Thus, if we would like to calculate the amounts of information of the MLL formulae with the prefixes $\forall\exists$ (i.e., case (3)) and $\exists\forall$ (i.e., case (4)), then first, calculate the probability of the MLL formula with the prefix $\exists\forall$ in Eq. (12). That is,

$$P_{\exists\forall} = P((b_1 \cap b_2 \cap b_4) \cup (b_1 \cap b_3 \cap b_5) \cup (b_2 \cap b_3 \cap b_6) \cup (b_4 \cap b_5 \cap b_6)) = \frac{23}{2^6}.$$

Next, calculate the amounts of information of the MLL formulae with the prefixes $\forall\exists$ and $\exists\forall$ in Eqs. (10) and (11), respectively. That is,

$$K_{\forall\exists} = \log \frac{1}{1 - P_{\exists\forall}} = 0.642.$$
$$K_{\exists\forall} = \log \frac{1}{P_{\exists\forall}} = 1.476. \qquad \square$$

Although most applications only need to consider formulae with 2 alternating quantifiers or the same quantifiers appeared in succession [19], we now discuss how to extend the above algorithm for processing more than 2 alternating quantifiers like $\exists\forall\exists$, $\forall\exists\forall$, etc. This extension can be easily done because we can expand an expression of probability corresponding to a complex sequence of quantifiers of a MLL formula into a simpler form (i.e., the same form as $\exists\forall$), so that Eq. (12) can be used for calculating its probability. We describe it by the following example.

Example 4. If we would like to calculate the amount of information of the following MLL formula with the prefix $\forall\exists\forall$ and the IS-A hierarchy shown in **Fig. 5**,

$$[\forall\, Z/*2d][\exists\, Y/Z][\forall\, X/Y]f(X),$$

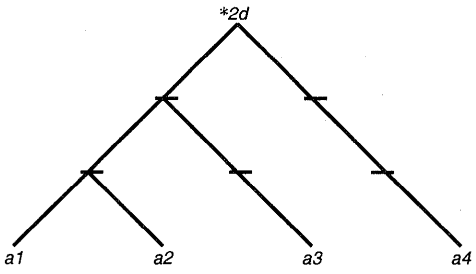


Fig. 5 A sample IS-A hierarchy to describe the more complex MLL prefix.

then first, write out the expression of probability corresponding to the MLL formula as follows,

$$P_{\forall\exists\forall} = P((a_1 \cap a_2 \cup a_3) \cap a_4).$$

Then expand this expression into a simpler form as follows,

$$P_{\forall\exists\forall} = P((a_1 \cap a_2 \cup a_3) \cap a_4) = P((a_1 \cap a_2 \cap a_4) \cup (a_3 \cap a_4)).$$

That is, this is the same form as $\exists\forall$. Thus, Eq. (12) can be used for calculating its probability. A calculated result is

$$P_{\forall\exists\forall} = P((a_1 \cap a_2 \cap a_4) \cup (a_3 \cap a_4)) = \frac{5}{2^4}$$

and

$$K_{\forall\exists\forall} = \log \frac{1}{P_{\forall\exists\forall}} = 1.678. \qquad \square$$

5. Knowledge Refinement

There are many possible applications for this method described in Sections 3 and 4. For example, this method can be applied for refining concept clusters discovered from a database such as the breast cancer database [4),19)]. We argue that the refinement for the discovered knowledge is an important function in knowledge discovery and data mining systems [19]. Here *refinement* means acquiring a more accurate knowledge (hypothesis) from a coarse knowledge (hypothesis) according to data change and/or the domain knowledge.

Although many systems for knowledge discovery and data mining such as INLEN, Forty-Niner, EXPLORA and KDW have been proposed [3),6),13),23)], few of them have addressed the capability of refining the discovered knowledge. In the traditional machine learning community, information theory, as stated in Section 1, is only used for generating inductively knowledge that is represented in decision tree or *if-then* rule, but the issue on knowledge refinement is not considered. On the other hand, the systems based on Bayes' rule such as COBWEB perform incremental conceptual clustering[2]. COBWEB constructs the clusters by examining training examples one at a time. Therefore, it can update its partial definitions of the boundaries of concept clusters when new instances become available, without being able to review the old data from which those partial definitions were formed. Although COBWEB as an *incremental* method has the capability of knowledge refinement, it is not used for evaluating information of logical expression, it can only represent the concept hierarchies with disjunctive relationship, and it cannot represent the concept hierarchies with "tangled" elements (or multi-categorization). Hence, in the rest of the section, we would like to discuss a knowledge refinement method, in which the issues and disadvantages of COBWEB stated above can be solved/overcame systematically, as an application of the method described in Sections 3 and 4.

The process of knowledge refinement in our method can be divided into two main stages. The *first* stage is that of representing the concept clusters as the MLL formulae with hierarchical models. For example, the concept cluster, which is discovered from a breast cancer database☆ and describes the conditions of benign cancer,

clump-t: 1, 4, 2, 6.
u-cell-size: 1, 3, 2, 9.
u-cell-shape: 2, 1, 4.
s-e-cell-size: 1.
bare-nuclei: 0, 3, 8.
bland-chromatin: 1, 3, 4.
mitoses: 2,

can be represented by the hierarchical model as shown in **Fig. 6** and the following MLL formula:

/* The rule for diagnosing breast cancer */
[∀ Y#/benign:symptom][∃ X#/Y]
p-breast-cancer(Y, X).

☆ Each tuple in this database corresponds to one patient and values of 10 attributes are given for each patient. The domain of every attribute is given by the sets of 9 quantized values that are classified as a case of benign or malignant cancer, resulting from clinical examinations related to this disease. Although many clustering methods can be used for our purpose, we would like to use the clustering method described in Ref. 20). This method can automatically analyze and delete the irrelevant data for clustering, which is a kind of noisy data and cannot be used to differentiate different clusters.

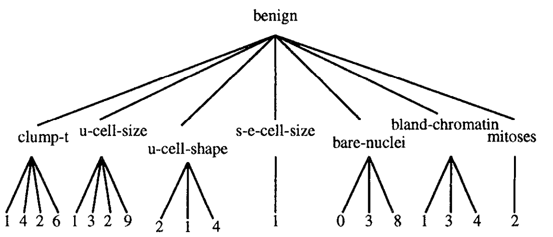


Fig. 6 The hierarchical model of group 1.

The rule reads *"if the symptoms recorded in the set-elements relations about benign are satisfied, then the breast cancer is benign."*.

Thus, there are two important jobs in this stage. The first is *hierarchical modeling*. Where, the process of representing a concept cluster by an IS-A hierarchy (i.e., the hierarchical model represented by the set-elements relation in MLL, and Fig. 6 is an example of its equivalent graph) is called *hierarchical* modeling. Another job is that of selecting quantifiers in the MLL prefix. Since both of the quantifiers ∀ and ∃ can be used for the conditions belonging to different attributes, both of the MLL prefixes,

(1) $[\exists\, Y\#/\text{benign:symptom}][\exists\, X\#/Y]$

and

(2) $[\forall\, Y\#/\text{benign:symptom}][\exists\, X\#/Y]$,

can be used for representing the acquired concept clusters. Where, we can see that if the MLL formula with the prefix (2) is true then the one with the prefix (1) is true, that is,

[∀ Y#/benign:symptom][∃ X#/Y]
p-breast-cancer(Y X) ⇒
[∃ Y#/benign:symptom][∃ X#/Y]
p-breast-cancer(Y X),

from Theorem 3 stated in Section 3.1, we know that the amount of information of the MLL formula with the prefix (2) is larger than the one with the prefix (1). Therefore, the prefix (2) is selected for representing the concept clusters.

The *second* stage is that of refining the hierarchical models by using domain knowledge and/or when new concept clusters are discovered along with data change in a database. This is to select automatically a best (or more refined) hierarchical model from more hierarchical models belonging to a family.

For example, when another concept cluster

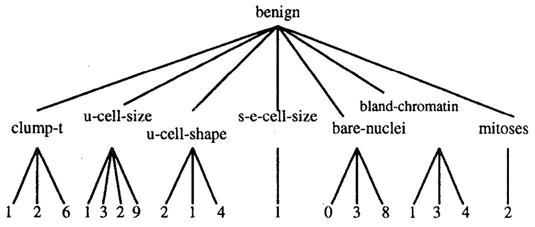


Fig. 7 A HM in which group 2 of data was added.

as shown in **Fig. 7** is discovered along with the adding of data in the breast cancer database, we can calculate the amounts of their information based on the MLL prefix,

$$[\forall\, Y\#/\text{benign:symptom}][\exists\, X\#/Y],$$

for selecting the better one from two hierarchical models shown in Figs. 6 and 7 by the algorithm stated in Section 4. Since the result of the calculation is

$$K_{g2} = 2.864 > K_{g1} = 2.764,$$

the hierarchical model shown in Fig. 7 is selected. That is, learning is to select a hierarchical model with more information.

Furthermore, since experts can bring domain knowledge to bear while refinement, the hierarchical model can be also refined by using cooperatively domain knowledge and informative evaluation. For example, if the following domain knowledge,

!ins_e uniformity u-cell-size, u-cell-shape;
!ins_e cell u-cell-size, u-cell-shape, s-e-cell-size;
!ins_e nuclei bare-nuclei, normal-nucleoli, mitoses;
!ins_e other clump-t, bland-chromatin;

is used, then a more refined hierarchical model as shown in **Fig. 8** can be acquired. That is, domain knowledge is used for conceptual abstraction (generalization). Here, the lowest leaves of the hierarchical model are only *observable values* that are collected in a database. The other values are called *abstract values*, and the second lowest leaves of the hierarchical model are called *the lowest level of abstract values*. However, *abstract values* can be "tangled" (i.e., a value can belong to more than one *abstract value* in a higher level as shown in Fig. 8).

The prefix of the MLL formula with the hierarchical model as shown in Fig. 8 can be represented into

$$[\forall\, Z\#/\text{benign:symptom}][\forall\, Y\#/Z][\exists\, X\#/Y].$$

Since there are the same quantifiers appeared

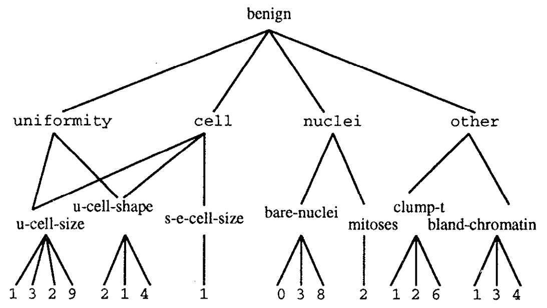


Fig. 8 A hierarchical model used domain knowledge.

in succession in the prefix of this formula, the amount of information of this formula with the hierarchical model shown in Fig. 8 is the same with the one shown in Fig. 7 (according to Theorem 2). Therefore, the hierarchical model shown in Fig. 8, in which *group* 2 of data was added and conceptual abstraction was done, is selected as a more refined one. Here, learning is to select the best hierarchical model by using cooperatively domain knowledge and informative evaluation.

6. Concluding Remarks

We presented a way of calculating quantitatively information of logical expression and its application for knowledge refinement. It can be considered as a typical method in which information theory is used as a criterion for learning knowledge. It is based on the model representation of MLL.

The method described in this paper constitutes a theoretical basis of developing HML that is a sub-system of our GLS discovery system for generating, refining and managing the knowledge discovered from databases[19),21)]. Furthermore, we would like to emphasize that the method described in this paper is only used as a learning phase in multiple learning phases of our GLS discovery system, the results of the knowledge refinement are not the final ones in the discovery process, and can be further used in the next learning phase for acquiring more high-level knowledge. How to acquire more high-level knowledge from the discovered concept clusters, which are represented as the MLL formulae with hierarchical models, is a further research subject. This involves a new sub-system of GLS that is being developed by us[22)].

Acknowledgements

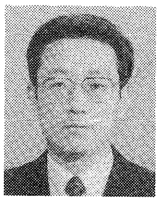
The authors would like to thank Professor Hori and Professor Takasu for useful discussions in OHlab seminar and SIG-KDD meeting. The authors also thank Mr. Yamauchi for his help in the use of KAUS. Further we would like to thank the anonymous reviewers for their valuable comments on the first submitted version of this paper.

This paper is a revised version of a paper presented at the 5th IEEE International Conference on Tools with Artificial Intelligence (TAI'93), Boston, USA (November 1993). We received financial support for the successful international conference from the International Information Science Foundation of Japan.

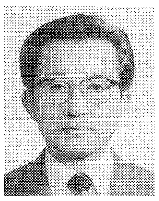
References

1) Cendrowska, J.: PRISM: An Algorithm for Inducing Modular Rules, *Int. J. of Man-Machine Studies*, Vol.27, pp.349–370 (1987).
2) Fisher, D.H.: Knowledge Acquisition via Incremental Conceptual Clustering, *Machine Learning*, Vol.2, pp.139–172 (1987).
3) Klosgen, W.: Problems for Knowledge Discovery in Databases and their Treatment in the Statistics Interpreter Explora, *Int. J. of Intell. Sys.*, Vol.7, No.7, pp.649–673 (1992).
4) Mangasarian, O.L. and Wolberg, W.H.: Cancer Diagnosis via Linear Programming, *SIAM News*, Vol.23, No.5, pp.1–18 (1990).
5) Matheus, C.J., Chan, P.K. and Piatetsky-Shapiro, G.: Systems for Knowledge Discovery in Databases, *IEEE Trans. Knowl. Data Eng.*, Vol.5, No.6, pp.904–913 (1993).
6) Michalski, R.S., Kerschberg, L., Kaufman, K.A. and Ribeiro, J.S.: Mining for Knowledge in Databases: The INLEN Architecture, Initial Implementation and First Results, *J. of Intell. Infor. Sys.*, Vol.1, No.1, pp.85–113 (1992).
7) Michalski, R.S., Carbonell, J.G. and Mitchell, T.M.: *Machine Learning – An Artificial Intelligence Approach*, Vols.1/2/3, Morgan Kaufmann (1983/1986/1990).
8) Nunez, M.: The Use of Background Knowledge in Decision Tree Induction, *Machine Learning*, Vol.6, pp.231–250 (1991).
9) Ohsuga. S.: A Consideration to Knowledge Representation – An Information Theoretic View, *Bulletin of Informatics and Cybernetics*, Vol.21, No.1-2, pp.121–135 (1984).
10) Ohsuga, S. and Yamauchi, H.: Multi-layer Logic – A Predicate Logic Including Data Structure as Knowledge Representation Language, *New Generation Computing*, Vol.3, No.4, pp.403–439 (1985).
11) Ohsuga, S.: Framework of Knowledge Based Systems – Multiple Meta-level Architecture for Representing Problems and Problem Solving Processes, *Knowledge Based Systems*, Vol.3, No.4, pp.204–214 (1990).
12) Piatetsky-Shapiro, G. and Frawley, W.J. (Eds.): *Knowledge Discovery in Databases*, AAAI Press/The MIT Press (1991).
13) Piatetsky-Shapiro, G. and Matheus, C.J.: Knowledge Discovery Workbench for Exploring Business Databases, *Inter. J. of Intell. Sys.*, Vol.7, No.7, pp.675–686 (1992).
14) Quinlan, J.R.: Induction of Decision Trees, *Machine Learning*, Vol.1, pp.81–106 (1986).
15) Quinlan, J.R.: Generating Production Rules from Examples, *Proc. 10th Int. Joint Conf. on Artif. Intell.*, pp.304–307 (1986).
16) Shavlik, J.W. and Dietterich, T.G. (Eds.): *Readings in Machine Learning*, Moran Kaufmann (1990).
17) Smyth, P. and Goodman, R.M.: An Information Theoretic Approach to Rule Induction from Databases, *IEEE Trans. Knowl. Data Eng.*, Vol.4, No.4, pp.301–316 (1992).
18) Watanabe, S.: *Knowing and Guessing – A Quantitative Study of Inference and Information*, John Wiley and Sons (1969).
19) Zhong, N. and Ohsuga, S.: HML – An Approach for Refining/Managing Knowledge Discovered from Databases, *Proc. 5th IEEE Int. Conf. on Tools with Artif. Intell. (TAI'93)*, pp.418–426, IEEE Computer Society Press (1993).
20) Zhong, N. and Ohsuga, S.: Discovering Concept Clusters by Decomposing Databases, *Data & Knowledge Engineering*, Vol.12, No.2, pp.223–244, Elsevier Science Publishers (1994).
21) Zhong, N. and Ohsuga, S.: The GLS Discovery System: Its Goal, Architecture and Current Results, *Methodologies for Intelligent Systems (Proc. 8th Inter. Symp., ISMIS'94)*, Ras, Z.W. and Zemankova, M. (Eds.) Lecture Notes in AI, Vol.869, pp.233–244, Springer-Verlag (1994).
22) Zhong, N. and Ohsuga, S.: From Conceptual Hierarchical Models to More High-level Knowledge (draft).
23) Zytkow, J.M. and Zembowicz, R.: Database Exploration in Search of Regularities, *Journal of Intelligent Information Systems*, Vol.2, No.1, pp.39–81 (1993).

(Received May 27, 1996)
(Accepted January 10, 1997)



Ning Zhong is currently an associate professor in the Department of Computer Science and Systems Engineering, Yamaguchi University, Japan. He is also a cooperative research fellow in Research Center for Advanced Science Technology (RCAST) at the University of Tokyo. He graduated at the Beijing Polytechnic University in 1982 and has been a lecturer in the Dept. of Computer Science, Beijing Polytechnic University. He received the Ph.D. degree in the Interdisciplinary Course on Advanced Science and Technology from the University of Tokyo. His research interests include knowledge discovery in databases, machine learning, and intelligent information systems. He is a member of IPSJ, JSAI, IEEE, ACM, AAAI, AFOS, and IRSS.



Setsuo Ohsuga is currently a professor in the Department of Information Science and Computer Science, Waseda University, Japan. He has been professor and director of Research Center for Advanced Science and Technology (RCAST) at the University of Tokyo. He has also been president of the Japanese Society for Artificial Intelligence (JSAI). He graduated at the University of Tokyo in 1957. From 1957 to 1961 he worked in Fuji Precision Machinery (the present Nissan Motors). In 1961 he moved to the University of Tokyo and received his Ph.D. in 1966. He became associate professor in 1967 and professor in 1981. His research interests include artificial intelligence, knowledge information processing, databases, and CAD. He is a member of the editorial boards of 9 scientific journals.