TR-546

Knowledge Acquisition by Inductive Operation Presumption

by

H. Taki & S. Terasaki

April, 1990

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

Knowledge Acquisition by Inductive Operation Presumption

Hirokazu Taki and Satoshi Terasaki

ICOT Research Center, Institute for New Generation Computer Technology, 4-28, Mita 1-Chome, Minato-ku, Tokyo 108 Japan Phone: +81-3-456-3192, E-mail:htaki%icot.jp

Abstract

This paper describes a knowledge acquisition approach which is an integration of an interactive knowledge acquisition and an inductive knowledge generation. In this approach, problem solving knowledge is extracted as a set of operations. There are three knowledge acquisition phases in the knowledge acquisition process. The first is the expert model construction phase, in which problem solving knowledge is extracted as a set of operations and the type of each operation is defined. Operation types are similar to operations of relational algebra. The second phase is the model instantiation phase, in which detailed knowledge is extracted according to the operation types. Sometimes, it is not easy to extract detailed information without proposals of knowledge. Therefore, this approach supports an inductive mechanism which makes operation presumptions in order to stimulate experts to associate detailed knowledge. The third phase is the knowledge refinement phase, in which the knowledge base is evaluated at the operation level to check whether the knowledge is sufficient. This paper also introduces an interactive knowledge acquisition support system, EPSILON/One, which has an interactive knowledge acquisition function and operation presumption algorithms based on induction.

1. Introduction

There is a big problem in the building knowledge base of expert systems, called the

* This paper is a revised version of "Operation Presumption: Knowledge Acquisition by Induction" which was presented in 3rd European Knowledge Acquisition on Knowledge Based Systems Workshop 1989.

knowledge acquisition bottleneck. Knowledge acquisition support systems have been developed to solve this problem. There are three phases in the knowledge acquisition process; (1) the expert model modeling phase, (2) the expert model instantiation phase, and (3) the knowledge refinement phase. In the expert model modeling phase, a problem solving model is extracted. In the expert model instantiation phase, knowledge, which is used in problem solving, is extracted. Knowledge consists of concepts, relations between concepts, and problem solving knowledge using concepts and their relations.

In the knowledge refinement phase, the knowledge acquisition system detects shortage and inconsistency of knowledge, and guides the expert to modify the knowledge. Many knowledge acquisition support systems have assumed problem solving knowledge before knowledge acquisition. A problem solving strategy, which MORE (Kahn, G., 1985) assumes for expert systems, includes the usage of domain models for diagnosis tasks. A problem solving strategy, which ETS (Boose, J., 1986) assumes for expert systems, includes the classification method of items using knowledge in the form of a rating grid. EPSILON/EM (Taki, H., Tsubaki, K. and Iwashita, Y., 1987) and EPSILON/One (Taki, H. and Tsubaki, K., 1990) have no problem solving strategy before knowledge acquisition. They extracts problem solving knowledge first, then acquires other knowledge which is used by problem solving knowledge. These knowledge acquisition support systems use some methods to extract knowledge; ETS uses the personal construct theory (Kelly, G. A., 1955; Ford, K. M. and et al., 1989) to stimulate the expert to associate constructs related to items. It also uses a refinement method which translates a rating grid into an implication graph and cluster trees to stimulate the expert to find some shortage or inconsistency of knowledge. MORE uses a refine method which refines a domain model using eight interview strategies. EPSILON/One also supports a knowledge acquisition method, the pre-post method, to stimulate the expert to remember operations of his expert jobs and to extract detailed knowledge of each operation using operation types.

However, it is difficult for us to prepare a general method to extract knowledge. To associate expert knowledge, we must prepare proposals of knowledge. EPSILON/One asks the expert for detailed information of operations according to the operation type, but sometimes, the expert cannot answer or remember information. Humans can modify knowledge more easy than they can remember it. Therefore, to give EPSILON/One the capacity to extract operation information like this, we have been developing the operation presumption method, which supports making proposals of knowledge to associate expert knowledge by induction.

Some of the major problems of using induction is that it requires a large amount of computing power and creates numerous hypotheses. In order to make hypotheses by induction more efficient, the forms of hypotheses must be limited (Genesereth, M. and Nilsson, N., 1986). EPSILON/One has a knowledge representation, "Expert Model", which consists of operations. The operations are limited to seven types which are derived from analyzing real diagnostic knowledge bases. The system realizes efficient inductive hypothesis generation by using this limited numbers of operation representations.

The following sections introduce knowledge representation for knowledge acquisition, Expert Model; a knowledge acquisition support system, EPSILON/One; and an operation presumption method using inductive inference.

2. Expert Model

This section explains the original idea of the Expert Model, and gives details of the Expert Model structures.

2.1 Basic Idea of Expert Model

The Expert Model is based on two ideas: the simplified expert task model and analysis and grouping of diagnosis expert knowledge written in production rule form.

(1) Simplified Expert Task Model

Expert systems can be distinguished as different task types. According to "Building Expert Systems" (Hayes-Roth, F., Waterman, D. A. and Lenat, D. B., 1983), there are 10 generic categories of knowledge engineering applications: interpretation, prediction, diagnosis, design, planning, monitoring, debugging, repair, instruction and control. We tried to define these categories as simple models, because simple expert task models provide the expert task images for the expert to represent his knowledge. Two samples are introduced later.

Simple Diagnosis Task Model

Fig.1 shows a diagnosis task. Diagnosis tasks are represented in tree structures and their search algorithm in general AI architectures. However, the tree structure does not represent expert task images. A simpler model is a filter model. In Fig.1, G1 denotes the possible results of trouble positions, F1 is a filter to select real results, and G2 denotes the results of a diagnosis task. Fig.2 also shows a diagnosis task, but this model has

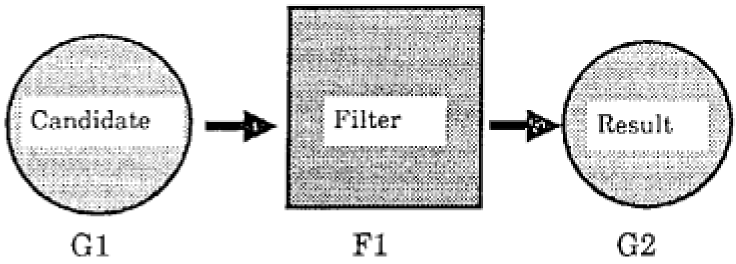

Fig.1 Simple Diagnosis Task Model

some sub-tasks which are the same as these of Fig.1.

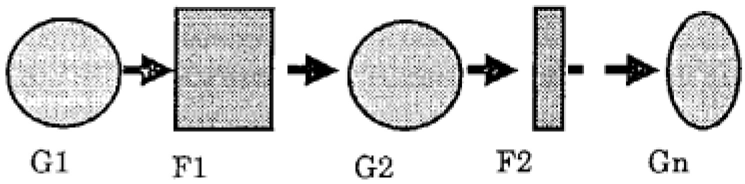

Fig.2 Simple Diagnosis Task Model(No.2)

Simple Design Task Model

Fig.3 shows a very simple design task. Design tasks are very complex; therefore, this model is actually a sub-task of a design task. It consists of a part modification phase, a part assembly phase and a good selection result phase. G4 and G5 are design-part groups, F4 is a modifier, G10 is a group of modified G4 parts, F5 is an assembly or combination operation, G11 is an interim result group, F6 is a selector, and G12 is a group of design task results. (G5 should not be modified.) We have considered simplified expert task models and created <u>a model (operation) which is constructed with a source element-group, evaluators (e.g. select filter) and a destination element-group.</u>

(2) Study from Production Rules of Diagnosis Expert Systems

A production rule is a general knowledge representation for expert systems. Therefore,

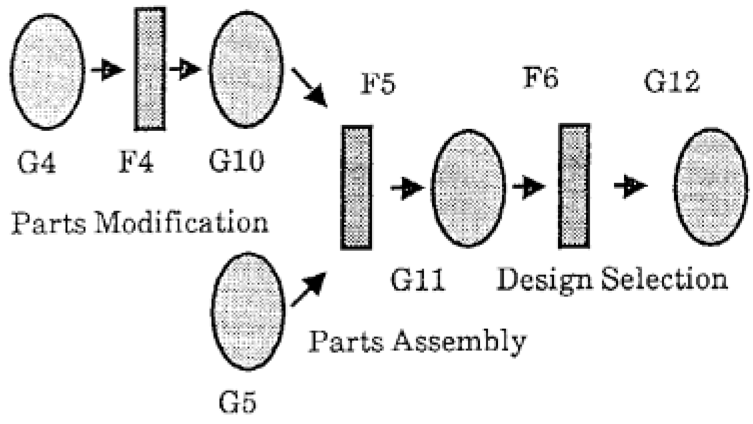

Fig.3 Simple Design Task Model

knowledge engineers must have techniques of rule writing to represent many kinds of knowledge. We assumed that those techniques may appear in the form of rules. Seven operations have been discovered in production rule sets: *selection, classification, ordering, combination, translation, input* and *output*. The combined result of both ideas is the generic operation.

2.2 Operation Types

The Expert Model is a specialized knowledge representation to collect expert task smoothly from experts. It is not necessary for the knowledge acquisition process to provide a general knowledge representation, i.e. , production rule form. This section explains the components of the Expert Model.

2. 2. 1 Generic Operations

Generic operations are kernel knowledge representations in the Expert Model. The generic operations have been derived from simple expert task models and by analyzing production rules of diagnosis expert systems. The benefits of these frameworks are explained as knowledge acquisition methodology in section 3.

There are seven generic operations: *selection, classification, ordering, combination, translation, input* and *output*. Each operation has one or more source element groups, destination element groups and an evaluator.

SELECTION Operation

The SELECTION operation picks up elements from the source group according to the selection conditions of the evaluators. Fig.4 shows an example of the SELECTION operation.

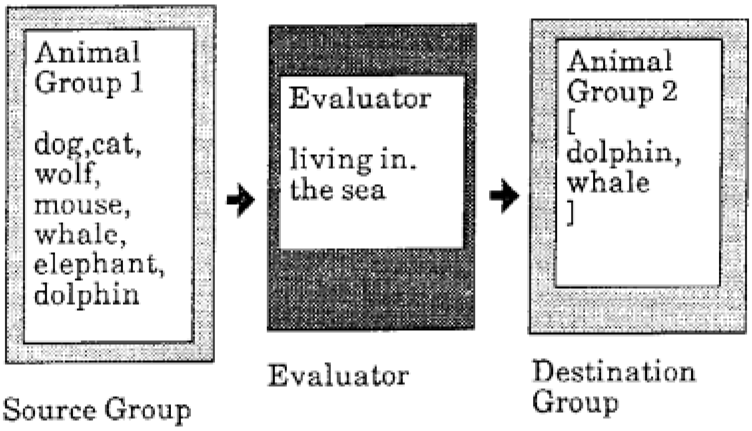

Fig.4 Example of SELECTION operation

CLASSIFICATION Operation

The CLASSIFICATION operation separates source elements into groups. The upper SELECTION operation is an example which is a special case of CLASSIFICATION. Fig.5 shows an example of the CLASSIFICATION operation.

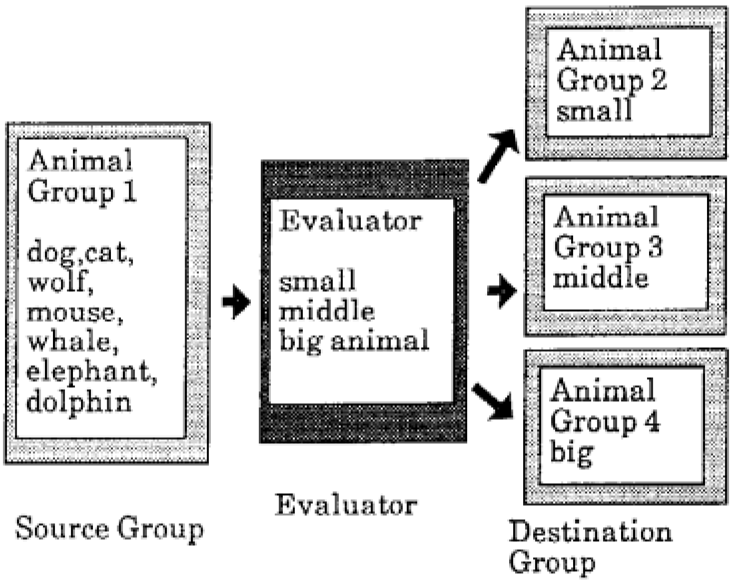

Fig.5.Example of CLASSIFICATION operation

ORDERING Operation

Elements of groups take turns to evaluate. The ORDERING operation sorts elements of the source group and makes a destination group, in which elements take their turns in order of the ordering condition.

COMBINATION Operation

The COMBINATION operation assembles source elements into new elements. Generally, there are multiple sources, as shown in Fig.6.

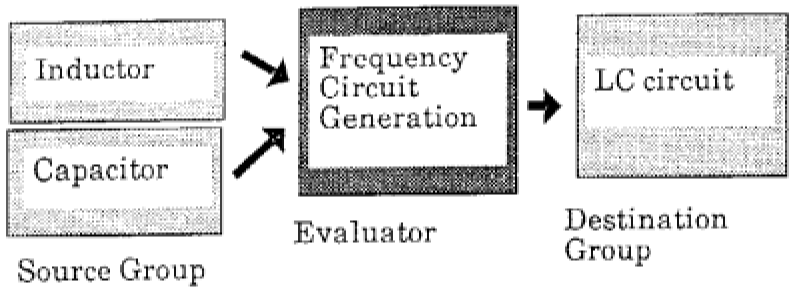

Fig.6.Example of COMBINATION operation

TRANSLATION Operation

The TRANSLATION operation is used for data abstraction and data interpretation. In this operation, elements of the source group are translated and the destination group is generated. New elements must be generated, or attributes of elements must be added or changed.

INPUT Operation and OUTPUT Operation

These are interaction operations between an expert system and a user (or a measuring device). The INPUT operation makes a new group without source groups. The OUTPUT operation does not make a destination group. It gives source group information to the user according to the output process.

These operations can be related to operations of relational algebra (Date, C. J., 1986) which are used to define data manipulation of data bases.(Taki, H., 1989a)

2.2.2 Elements

Elements consist of items and their attributes. Elements are frame-based knowledge representations which have attribute-slots and an inheritance, but they do not have the attached procedures as demons. Each element must belong to one or more element groups. An element is accessed by one or more generic operations.

2.2.3 Evaluator

An evaluators consists of an evaluation criterion and a procedure which realizes a function of an operation type. For example, if an operation is a selection operation, its evaluator consists of a selection criterion and a selection algorithm.

2.2.4 Operation Control: Script

The expert tasks have their scenario for solving problems, therefore, this operation control is called Script in the Expert Model. It is a representation to describe meta-level knowledge. Fig.7 shows its location in the knowledge base.

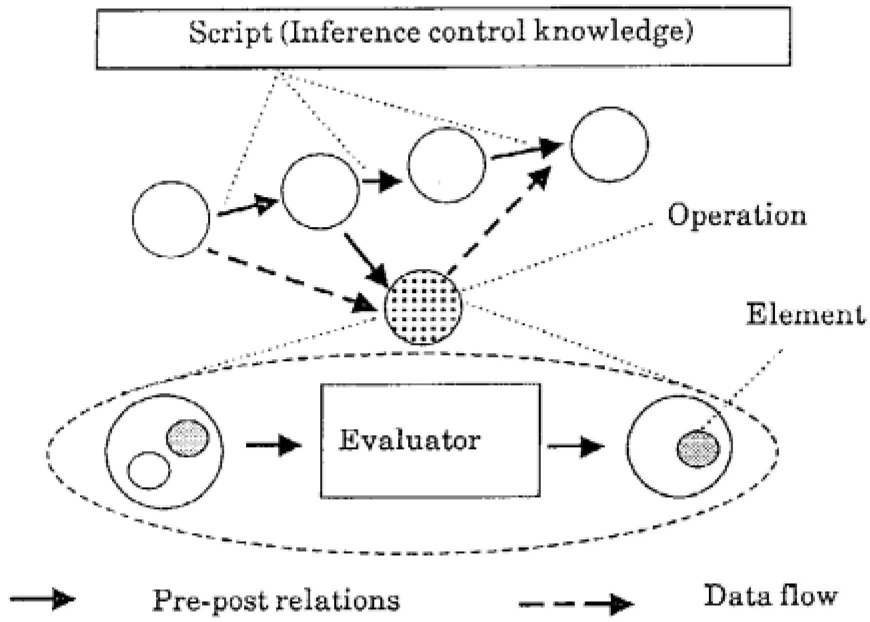

Fig.7 Hierarchy of the Expert Model

The Expert Model is recognized as an object-oriented model, and a generic operation is an object. The communication between a generic operation and other operations is a message passing process, which has two types in the Expert Model: the inheritance and the post operation call.

3. Knowledge Acquisition Method : Pre-Post Method

The pre-post method is a problem solving model construction and model instantiation method. This method has a few model building strategies. The main strategy of this method stimulates the expert to remember pre-post operations which are associated with a focal operation. The expert can easily answer what operations are necessary before or after the operation. For example, when the car does not move, the expert is asked *"What do you do before checking the engine ?"*. He can easily answer, *"I have to check the remaining gas" or "I have to check the battery"*. The following steps are knowledge acquisition steps based on the pre-post method.

Step 1: Collecting several expert operations at random. It is not necessary for each operation to have any relation to the others. These operations are used as the starting points of knowledge acquisition.

Step 2: Making questions to obtain pre and post operations for each collected operation. This step continues until no new operations are extracted from the expert. The same process is repeated for new operations.

Step 3: Many operations have been gathered and they have pre-post information which operations have to execute before or after operations. This step is the pre-post relation check step. One way of checking is to display pre-post relations graphically for the expert.

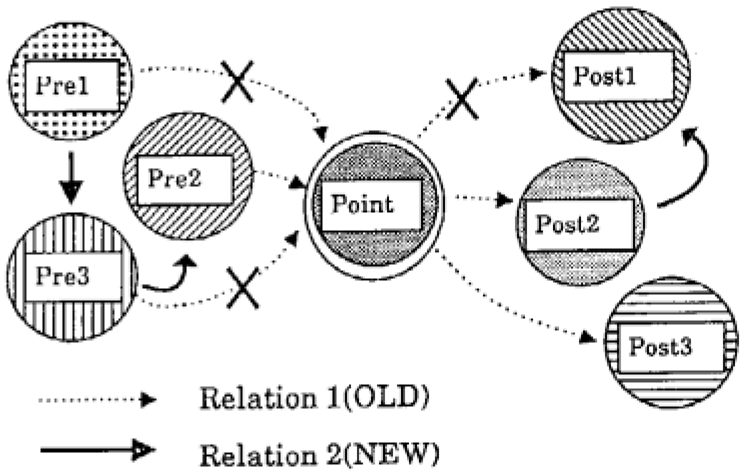

Fig.8 Pre-post relation example

Fig.8 shows pre-post relations. Relation1 shows only that Pre1 to Pre3 are pre-operations of a point operation and Post1 to Post3 are its post-operations. In relation2, there are new orders among pre-post operations defined by the expert. Pre1 is a pre-operation of Pre3. Pre2 is a post-operation of Pre3. Post1 is a post-operation of Post2. The more pre-post information collected, the clearer pre-post relations.

The sequence of some operations may be unclear. We treat the group of these operations as a weak sequence strategy group. When an operation of this group is finished, sometimes the result of the operation affects the other operations of this group. It is important for this group to obtain constraint information.

Step 4: This step is the instantiation step of each operation. The expert must select a generic operation type to define the role of each operation. Sometimes, operations defined by the expert are big tasks which contain several sub-operations. To distribute these big tasks, it is important to ask *"What sub-operations exist in XX (an operation name)? Please answer at random"*. After this question, pre-post operations must be extracted for sub-operations. Then, the Step 3 process is executed again.

Note: From step 1 to step 4, a problem solving model is made. From step 5, the details of operations in this model is extracted.

Step 5: This step merges operations. According to pre-post information and generic operation types, some operations are merged to one generic operation.

Step 6: Each operation has evaluators. This step defines these evaluators. It is not difficult to do this process, because the role of the operation is clear and its function has been identified by its generic operation type.

Step 7: The elements of the source group are defined in this step. Sometimes, it is difficult to decide the elements and group of a focal operation, because they are decided dynamically after its pre-operation has been executed. Therefore, the expert must search for the candidates of the source group from the result (destination group) of its pre-operations.

Step 8: Attributes and attribute values of the elements are defined in this step. Each element must have the attribute evaluated by the evaluators. Generally, elements flow through a lot of operations, and need the attribute which is evaluated by the evaluators in the flow.

Other Steps: There are other steps in pre-post method. They are operation blocks making step to group operations with same inference control, and operation inference control definition step.

4. Operation Presumption

4.1 The Expert Model and Knowledge Acquisition

The knowledge representation of EPSILON/One is the Expert Model(Taki, H. and Tsubaki, K., 1990). EPSILON/One extracts general knowledge from the expert. Therefore, if the expert is not aware of general evaluation criterion of an operation, the knowledge becomes very difficult to extract. However, it is not difficult for the expert to show examples which are input and output elements of the operation. The knowledge acquisition support system must contain a function which presumes general evaluation criterion from input and output element examples of the operation.

There are three famous reasoning methods to make hypotheses; analogy, abduction and induction. Our approach uses induction. However, there are some problems in using induction for generating knowledge. It generates numerous hypothesis candidates and its calculation cost is high. For efficient induction, a language of hypothesis representation must be limited to reduce candidates of knowledge, and domain specific knowledge must be used in the induction process to eliminate hypotheses. We tried to use the special feature of the Expert Model to limit the form of consequence of induction.

4.2 Inductive Knowledge Presumption Algorithm Using Operation Types

Operation presumption algorithms which uses operation types for consequence limitation of induction are introduced in this section. The seven operation types are *selection, classification, ordering, combination, translation, input* and *output*. The translation operation consists of three sub operations; *attribute value replacement operation, mathematical calculation operation* and *element decomposition operation*. To use induction, we must prepare *language which represents a target concept, positive and negative examples related to the target concept, knowledge which defines generalization of the concept, and induction algorithms.*

Language which represents a target concept:

Evaluation criterion for each operation type

A basic criterion is related to one attribute.

Positive and negative example related to the target concept:

Input and output elements of an operation

Knowledge which defines generalization of the concept:

Generalization method for each evaluation criterion

Induction algorithms:

Operation presumption algorithms for each evaluation criterion

EPSILON/One supports an efficient operation presumption function using operation types to limit language of inference consequence, treatment of examples, and methods of criterion generalization(Taki, H. and Fujii, Y., 1989). Here, we introduce three operation presumption algorithms; *selection and classification operation, attribute value replacement operation and ordering operation.* Currently, the algorithm of the combination operation is fixed and doesn't require any criterion. Generating the evaluator of the mathematical calculation operation from examples is equivalent to solving simultaneous equations. Therefore, we don't prepare induction algorithms for the combination operation and the calculation operation. It is not necessary to presume evaluation criterion of "input operation", "output operation", and "element decomposition operation" because criteria of these operations are easily extracted from the expert. We explain the following items for three operation presumption algorithms:

(1) Evaluation algorithms of operations

(2) Evaluation criterion for each evaluation algorithm

(3) Treatment of examples

(4) Evaluation criterion presumption algorithms

4.3 Selection and Classification Operation Presumption

The selection operation selects elements which are suitable for a selection criterion from a given set. The classification operation classifies elements of a given set into sub sets according to classification criteria. This operation can be realized to combine several selection operations.

(1) Selection Algorithm

If an attribute and its value of an element satisfy a selection criterion, then the element is selected. The selection criterion consists of an attribute and its possible value sets or a conjunction of attributes and their possible value sets. If an element satisfies the selection criterion, the value of the attribute of the element is included in a value set of the criterion.

(2) Selection Criterion

A selection criterion related to one attribute is defined as follows:

(attribute-name, a set of possible values)

(3) Treatment of Examples

Examples are given as multiple elements of an input element group and an output element group. Elements selected by a selection operation are positive examples which represent a concept of selection criterion. Elements which aren't selected by the selection operation are negative examples. These examples (elements) are defined by the expert. Elements for examples have some attributes which have one value for each attribute.

Example-1:

Given an input and an output element group as follows.

Input element group = {e1, e2, ... , ei, ei + 1, ... , en}

Output element group = {e1, e2, ... , ei} , ej (j = 1 ~ n) means an element.

then

Positive examples = {e1, e2, ... , ei}

Negative examples = {ei + 1, ... , en}

(4) Selection Criteria Presumption Algorithm

Step1: Making a set of attributes which are included in all examples

This set is called *a total attribute list.*

Step2: Making sets of values of each attribute in the total attribute list from positive examples

If an element doesn't have an attribute,its value is *undefined.*

Step3: Making sets of values of each attribute in the attribute set from negative examples

A matrix of the total attribute list (aj, j = 1˜m) and elements (ek, k = 1˜n) with values vjk (k = 1˜n, j = 1˜m), is made from the results of step2 and step3.

		a1	a2	a3 ...	am
Positive example	e1 :	v11	v21	v31 ...	vm1
	e2 :	v12	v22	v32 ...	vm2
			:		
	ei :	v1i	v2i	v3i ...	vmi
Negative example	ei+1:	v1i+1	v2i+1	v3i+1 ...	vmi+1
	ei+2:	v1i+2	v2i+2	v3i+2 ...	vmi+2
			:		
	en :	v1n	v2n	v3n ...	vmn

Sets of values for an attribute "aa" are defined as P(aa) when made from positive examples and N(aa) when made from negative examples.

P(a1) = {v11, v12, ..., v1i}
P(a2) = {v21, v22, ..., v2i}
:
P(am) = {vm1, vm2, ..., vmi}

N(a1) = {v1i+1, v1i+2, ..., v1n}
N(a2) = {v2i+1, v2i+2, ..., v2n}
:
N(am) = {vmi+1, vmi+2, ..., vmn}

Step4: Checking intersection of P(aa) and N(aa)

if P(aj)$\wedge$N(aj) = Φ (empty set) then select attribute aj
else reject attribute aj , (j = 1˜m)

If selected attributes are "aa", "ab" and "ac", P(aa), P(ab) or P(ac) is a selection criterion.

[**Caution:** As $P(aj) \wedge N(aj) = \Phi$ (empty set), $\neg N(aa) \wedge \neg N(ab) \wedge \neg N(ac)$ is also a selection criterion. $\neg N(aa)$ means that the value of the attribute "aa" isn't included in N(aa).]

Step5: Showing selection criterion

If there are multiple criteria, the system shows them in order.
In this example, the system shows P(aa), P(ab), P(ac), and $\neg N(aa) \wedge \neg N(ab) \wedge \neg N(ac)$ in this order.

(5) Guidance of generalization of selection criterion

The system shows how to make wider selection criterion by a set of attribute value generalization.

Example-2: $\{1, 2, 3, 4, 5\}$ --> $\{>= 1, <= 5\}$ --> Integer

{small-apple, big-apple} --> {apple} --> {fruit}

The system shows existence of elements which don't belong to either positive or negative examples.

Example-3: $P(aa) = \{1, 2, 3\}$, $N(aa) = \{5, 6, 7\}$

An element, where the value of an attribute "aa" is 4, doesn't belong to either P(aa) or N(aa). If a selection criterion is P(aa), this element is recognized as a negative example. If a selection criterion is $\neg N(aa)$, this element is recognized as a positive example. Therefore, $\neg N(aa)$ is a more general selection criterion than P(aa).

If there is no attribute aj which satisfies at $P(aj) \wedge N(aj) = \Phi$, then there is some dependency among attributes. In this case, the system must make selection criteria containing multiple attributes. (cf section 4.6).

4.4 Ordering Operation Presumption

(1) Ordering Algorithm

The ordering operation sorts elements according to ordering criterion related to one attribute. This ordering criterion is a total order. An order of elements means an ordering of processing elements in an operation.

(2) Ordering Criterion

An ordering criterion of one attribute is defined as follows:

(attribute-name, order list of values), (attribute-name, **ascending**),
or (attribute-name, **descending**)

(3) Treatment of examples

Examples are elements of an input element group and an output element group. An order of elements in the output element group is an ordering criterion of positive examples. Negative examples are relations of two items which differ from the input element group and the output element group. The input element group is not always necessary. Because there is no intersection between positive and negative examples, and negative examples aren't useful in limiting a criterion made from positive examples.

Example-4:

Given an input and an output element group as follows.

Input element group = {e1, e2, e3, e4}

Output element group = {e1, e3, e4, e2}

A relation that the expert prefers "a" to "b" is shown as "a >> b".

In this example, positive ordering relations are "e1 >> e3 >> e4 >> e2", negative ordering relations are "e2 >> e3" and "e2 >> e4".

(4) Ordering Criterion Presumption Algorithm

This algorithm deals with only the output element group as positive examples.

Step1: Making a set of attributes called a total attribute list which are included in positive examples

Step2: Selecting one attribute from the total attribute list

Step3: Making a value list of the attribute according to the order of elements

Step4: Checking inconsistent order pairs of two items

An inconsistent order pair means that both "a >> b" and "b >> a" exist.

Step5: If there is no consistent relation, return to step2.

Step6: If elements of the value list are all numbers, then check whether their relation is ascending or descending. Even if they are numbers and the value list has other relations (ex. preference of odd number), the system treats them as literal symbols.

The system shows one ordering criterion . If there are some candidates of criteria related to different attributes, the system shows them in order.

4.5 Attribute Value Replacement Operation Presumption

(1) Attribute Value Replacement Algorithm

Values of a certain attribute of elements are replaced with different values according to a replacement criterion. This criterion is a replacement rule table which contains pairs consisting of a source value and a destination value.

(2) Replacement Criterion of Attribute Values

A replacement criterion of one attribute is defined as follows:

(attribuite-name, a set of (source value, destination value))

(3) Treatment of Examples

Examples are elements of an input and an output element group. Concerning an attribute, positive examples have different input and output values, negative examples have the same input and output values.

(4) Replacement Criterion of Attribute Values Presumption Algorithm

Step1: Searching for an attribute which has different values in elements of input and output

This attribute is called a remarkable attribute.

Step2: Making pairs of source values and destination values of the remarkable attribute in positive examples

Step3: Selecting combinations where the source value is the same but destination values are different

These are uncertain replacement rules.

Step4: Picking up values of the remarkable attribute in negative examples

Step5: Checking whether the pairs created in Step2 have source values included in values made in Step4

If a source value of a pair is included in values made in Step4, the pair is an inconsistent rule.

Step6: Displaying candidate pairs

Step7: Displaying pairs which are uncertain and inconsistent for reference

4.6 Evaluation Criteria Related to Multiple Attributes

We introduced basic presumption algorithms which treat criteria related to one attribute. These criteria are positioned in most general parts of a version spaces(Mitchell, T.M., 1978) shown in Fig.9. This section describes selection criteria

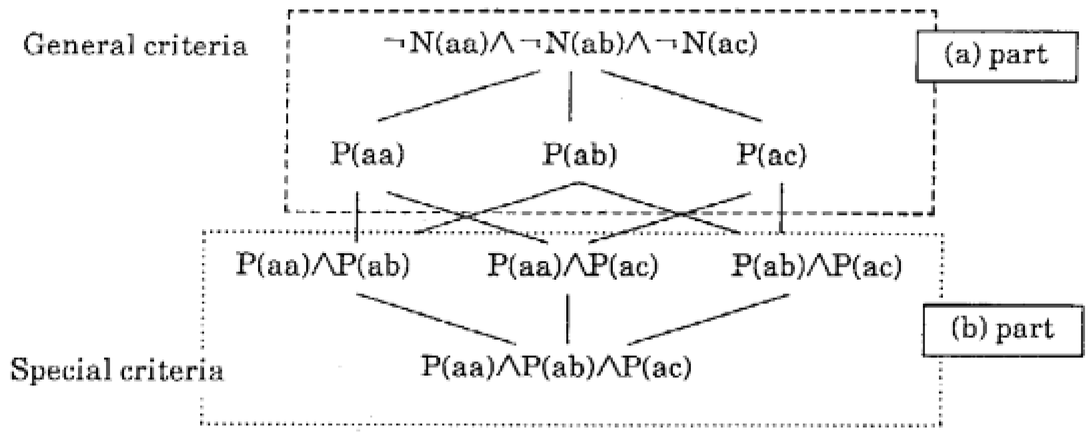

Fig.9 A version space

related to multiple attributes. A method which we introduce in this section presumes complex criteria while it spreads target parts of the version space to include special concepts step by step. This method also uses the algorithm descried in section 4.2. In order to control the spreading the target part of the version space, this system uses learning heuristics from the expert through interaction. Selection criteria related to multiple attributes is represented in conjunction with some selection criteria of one attribute as follows:

(attribute-name-1, a set of values)∩(attribute-name-2, a set of values)∩...

[**Caution:** The interview system of EPSILON/One elicits not only conjunctive form, but also disjunctive form and inequalities.]

The version space (version graph) of the example in section 4.2 is shown in Fig.9. In this figure, the algorithm of section 4.2 processes (a)-part (P(aa), P(ab), P(ac), $\neg N(aa) \wedge \neg N(ab) \wedge \neg N(ac)$) of this figure.

To treat selection criteria related to multiple attributes, the system must deal with (b)-part in this figure. In order to stimulate the expert to remember necessary attributes, the system calculates similarities between positive and negative examples. This information enables the expert to remember which attributes are important for separating positive and negative examples.

Example-5:

Positive example:	at1	at2	at3	at4
e1	1	1	1	0
e2	1	1	0	1
e3	1	1	0	0
Negative example:				
e4	0	1	0	1
e5	1	0	0	0

In this example, positive and negative examples cannot be separated by a selection criterion related to one attribute. Therefore, the system calculates distances of pairs between positive and negative examples for all attributes.

$$distance(e1, e4) = |v11 - v41| + |v12 - v42| + |v13 - v43| + |v14 - v44|$$

If "a" is equal to "b", then $|a - b|$ means 1 else it means 0.

Nearest pairs of a positive and a negative example are (e2, e5) and (e3, e5) which distances are 1. The system shows these pairs to the expert to order attributes according to his preference. The expert can eliminates unnecessary attributes. If the expert selects at2, at1 and at4 in this order, the ordering of hypothesis generation and testing is a sequence of "at2 $\wedge$ at1, at2 $\wedge$ at4, at1 $\wedge$ at4, at2 $\wedge$ at1 $\wedge$ at4". According to conjunctions of attributes, lists of attribute values are made.

The algorithm in section 4.2 processes one attribute as a conjunction of attributes and one value as their value list. For example, in the case related to two attributes, combination information of element "e1" is shown as follows:

pairs of attributes	(at1, at2)	(at1, at4)	(at2, at4)
pairs of values	(1, 1)	(1, 0)	(1, 0)

A candidate of selection criteria is (at1, at2) = { (1, 1) } in this example.

5. Knowledge Acquisition Support System : EPSILON/One

5.1 EPSILON/One System Over View

The knowledge acquisition support system, EPSILON/One consists of four sub-systems. They are a knowledge elicitation system, an expert model inference engine, a knowledge refinement system, and an operation presumption system (shown in Fig.10). The

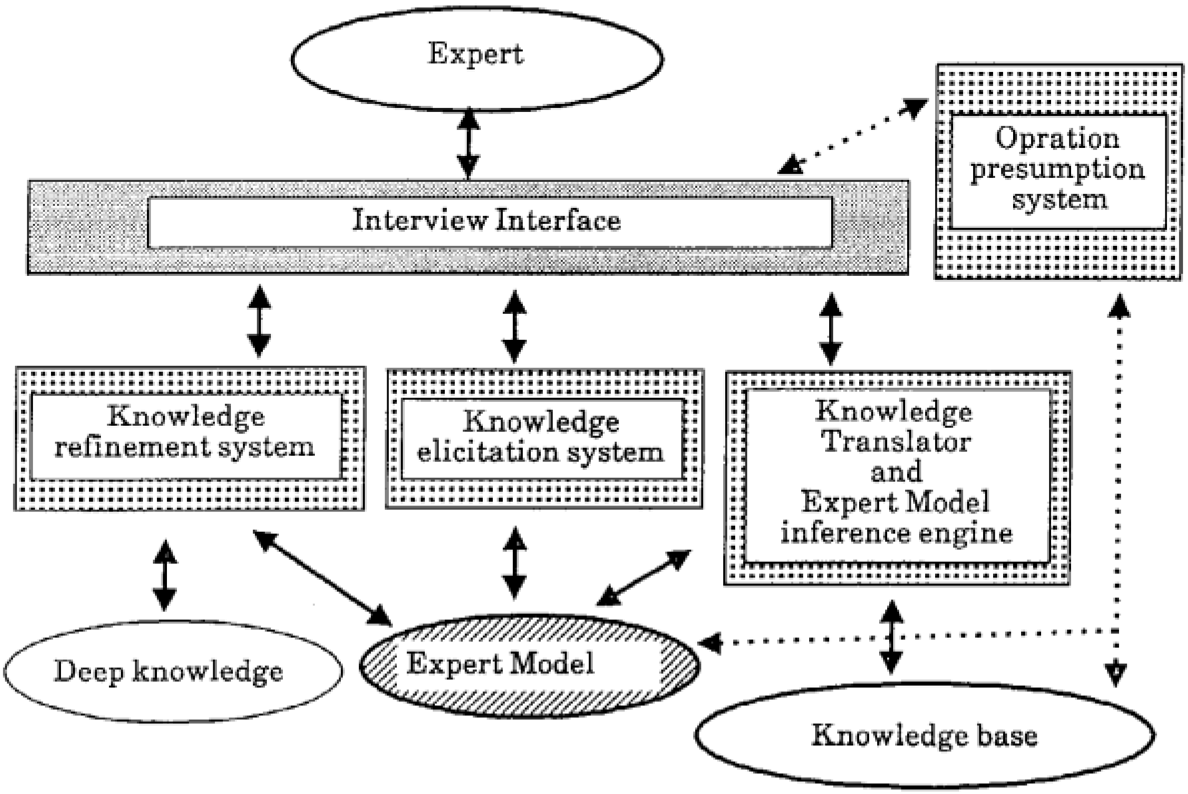

Fig.10 EPSILON/One system structure

knowledge elicitation system interviews the expert directly, and extract knowledge in form of the Expert Model. This sub-system uses pre-post method to acquire knowledge by interview. The knowledge refinement system checks lacks and inconsistency of knowledge base. Extracted the Expert Model is translated into logical knowledge representation and checked its inconsistency. Lacks of it is checked to compare the Expert Model and other knowledge. For example, if the Expert Model is knowledge for diagnosis of a machine, it is compared to the machine design information which represents its parts and the structure. The expert model inference engine evaluates the Expert Model. The expert can check his knowledge to observe behaviors of operations. The operation presumption system generates hypotheses from examples which are input and output information of operations. We call this interview system *a proposal guided knowledge acquisition support system.*

5.2 Knowledge Acquisition by Proposal Generation

The basic frame work of proposal guided knowledge acquisition is to generate hypotheses from given information and to feedback these to the expert to remember new knowledge. Information of this interaction (cf. Fig.11) is shown as follows:

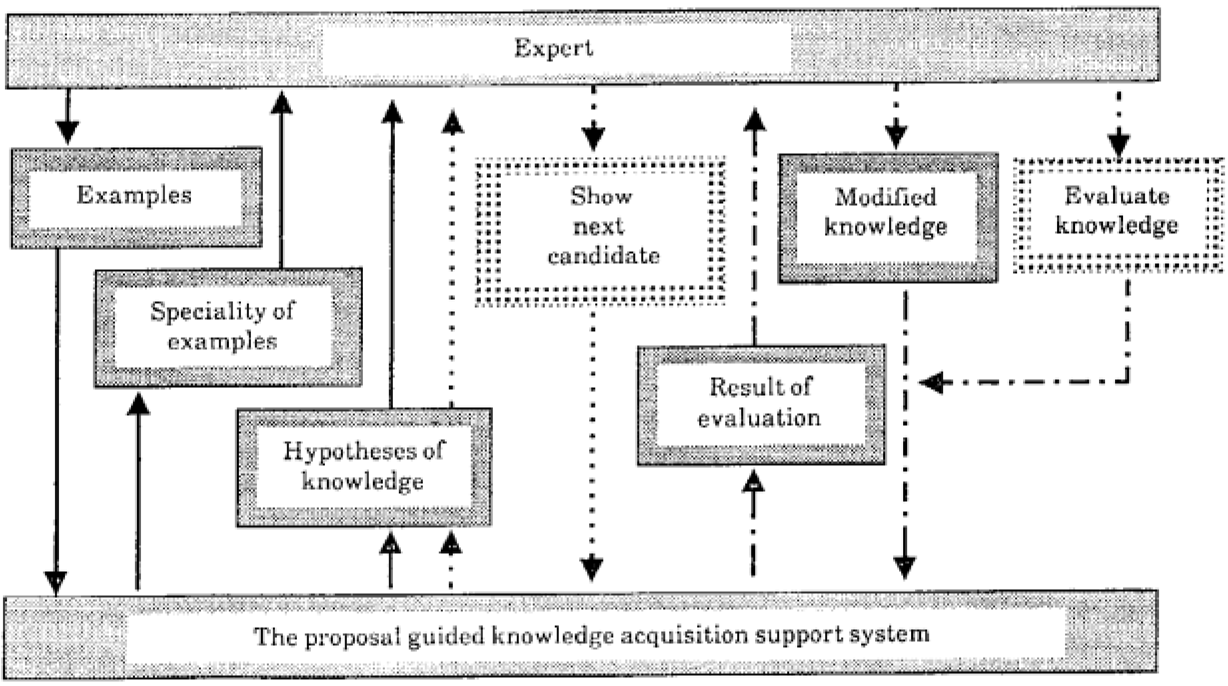

Fig.11 Interaction between the proposal guided knowledge acquisition support system and the expert

Information from the expert to the system:

(1) Examples (elements of input and output for an operation)

The two types of supply examples which are incremental supply and total supply.

(2) Knowledge modified by the expert (The expert can modify knowledge as hypothesis proposed by the system.)

Information generated by the knowledge acquisition support system

(3) Speciality in examples (ex. similarity of examples)

(4) General knowledge generated from examples by induction

(5) Results of evaluation of knowledge modified by the expert

The system evaluates modified knowledge with given input elements and shows output elements of the operation.

Requirements of the expert for the system

(6) Requirement of next hypothesis

(7) Requirement of evaluation of knowledge modified by the expert

The current target problem of the proposal guided knowledge acquisition support system is the diagnosis problem because the system handles seven operation types for diagnosis. The operation presumption system (Taki, H. and Terasaki, S., 1990) is implemented on PIMOS which is a parallel operating system on the PSI-2 (personal sequential machine) in a parallel logic programming language KL-1(Chikayama, K., and et al., 1989).

5.3 An Example

We introduce a simple example of the operation presumption about bearing selection, bearing ordering and their value replacement in a machine tool design problem(Inoue, K., Nagai, Y., and et al., 1988). In this example, the expert selects some bearing types from a total of twenty bearings(Nippon Seiko Corp., 1986), orders the bearings and replaces values of an attribute (cf. Fig.12).

(1)Presumption of Bearing Selection Knowledge

The expert sets twenty bearing types in an input element group and five bearing types

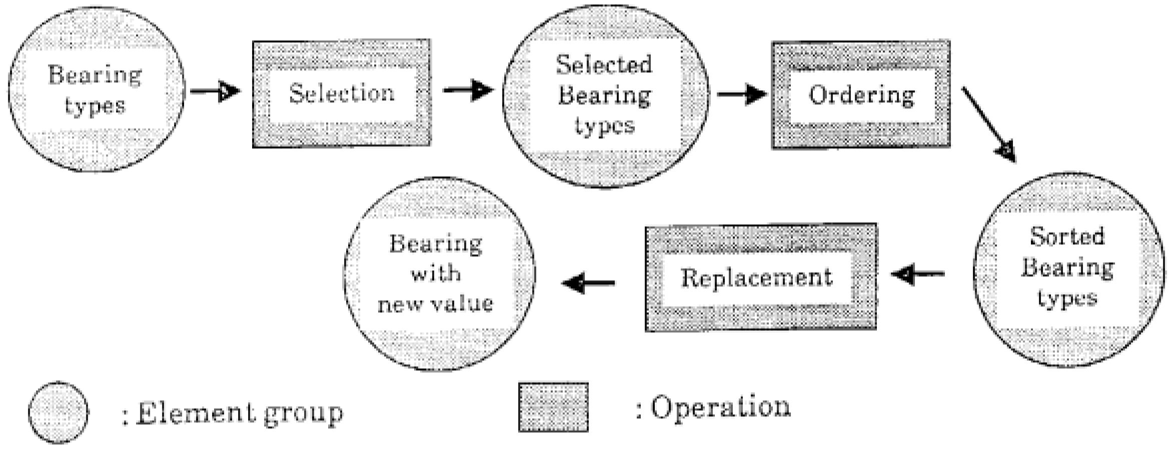

Fig.12 An example

in an output element group. These five bearings are positive examples and the rest of the bearings are negative examples. A bearing type consists of the following:

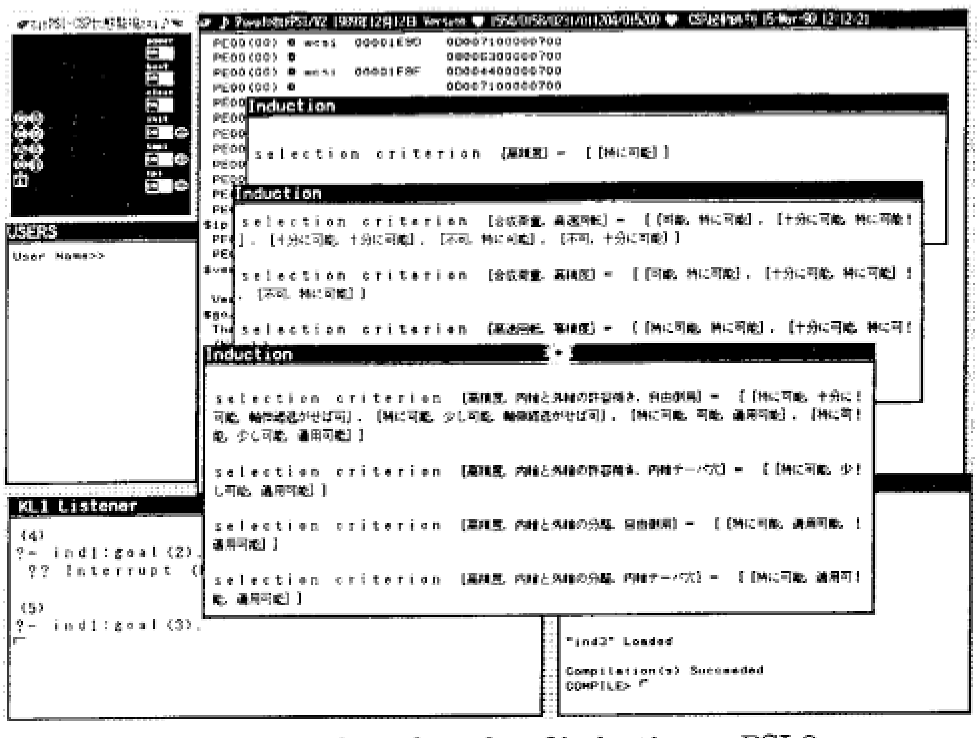

Fig.13 An window of results of induction on PSI-2

[deep-slot-ball-bearing, [force-of-radial, good],
[force-of-axial, good],
[direction-of-axial, bi-direction],
[mixed-force, good],
[high-speed, best],

[high-accuracy, best],
[low-noise, best],
[possible-sloop, better],
[fixed-case, possible],
[free-case, conditional]]

Examples are given as follows:

Input element group = [twenty types of bearings]

Each bearing has an attribute "high-accuracy" which values are

"best", "better", and "undefined".

Output element group = [deep-slot-ball-bearing, angular-ball-bearing, cylinder-roll-bearing, combinatorial-angular-ball-bearing, multiple-cylinder-roll-bearing]

A presumption result is high-accuracy = [[best]](shown in Fig.13 in Japanese). Fig.13 also shows some windows of criteria related to two attributes and three attributes.

(2) Presumption of Bearing Ordering Knowledge

The expert sets an ordering of the five bearings in an output element group. The ordering is as follows:

Input element group = [
[deep-slot-ball-bearing, [high-speed, best], ..],
[angular-ball-bearing, [high-speed, best], ..],
[cylinder-roll-bearing, [[high-speed, best], ..],
[combinatorial-angular-ball-bearing, [[high-speed, better], ..],
[multiple-cylinder-roll-bearing , [[high-speed, better], ..]]

This ordering is related to the attribute "high-speed", so the system shows

"high-speed = [best, better]" as a presumption result.

(3) Presumption of Bearing Attribute Value Replacement Knowledge

The expert sets an output element group. The input element group of this operation is equal to the output element group of the ordering operation. A value of an attribute of each element in the output group is different from one in the input element group.

Examples are shown as follows:

Output element group = [
[deep-slot-ball-bearing, [high-speed, range-4], ..],

[angular-ball-bearing, [high-speed, range-4], ..],
[cylinder-roll-bearing, [[high-speed, range-4], ..],
[combinatorial-angular-ball-bearing, [[high-speed, range-2], ..],
[multiple-cylinder-roll-bearing , [[high-speed, range-2], ..]]

This replacement is related to the attribute "high-speed", so the system shows "high-speed = [[best -> range-4], [better -> range-2]]" as a presumption result.

6. Related Works

AQUINAS(Boose, J.H. and Bradshaw, J.M., 1987) is a knowledge acquisition support system for classification problems. It extracts classification knowledge in the form of a rating grid. It also makes an implication graph which includes many hypotheses of implication relations among traits. The expert can refine these implications. This refinement style is a kind of proposal guided knowledge acquisition. The induction method of AQUINAS is simpler than our method.

KADS methodology (Breuker, J. and Wielinga, B., 1989) supports analysis phase of building knowledge bases. It uses problem solving models (task structures) which consist of concepts prepared by KADS. These task structures are made by knowledge engineers before knowledge acquisition. EPSILON/One also supports the Expert Model instead of task structures. A task configuration of the Expert Model is made by EPSILON/One during interviewing the expert. The reason why EPSILON/One can make task configuration easily is that the number of operation types is limited to seven which is less than the number of concepts of KADS. EPSILON/One supports not only analysis phase but also design phase of knowledge bases.

ID3(Quinlan, J.R., 1983) is an induction algorithm for generating a classification tree. Its classification criterion is an efficient search of the classification tree. Our operation presumption algorithm solves a classification criterion from given leaves of the classification tree, while ID3 generates the classification tree and leaves of the tree from a given classification criterion and items to be classified.

The MIS (Model Inference System)(Shapiro, E.Y., 1981)solves a horn clause by induction. It specifies a horn clause with the oracle by asking yes/no questions. Candidates of the horn clause are defined in the limited space of refinement operators. This search space is a kind of version spaces. The MIS prunes unnecessary candidates efficiently through skillful questions. Our operation presumption algorithm deals with more special forms of knowledge and more complex interaction than the MIS.

7. Current Problems

In the case that the expert cannot supply learning heuristics, the complexity of current operation presumption algorithm for selection criteria requires a large amount of computing power and is equal to the complexity of a total solution search of a version space. Therefore, it is important to use an elicitation method of learning heuristics. For this problem, the speciality of examples must be illustrated with a cluster tree to represent similarities between examples.

8. Summary

We introduced the knowledge acquisition approach and the knowledge acquisition support system(EPSILON/One) which makes hypotheses by induction and proposes candidates of knowledge. This research suggests a suitable integration of the engineering knowledge acquisition method and logical learning method. An important point is usage of operation types to make efficient questions for interviews and to limit forms of consequence of induction. In the cognitive view, proposals of knowledge are useful triggers of associative elicitation. We are planning to research more effective usage of proposals to extract knowledge.

Acknowledgement

We wish to express our thanks to members and the chief of ICOT 5th Laboratory, Mr. Kenji Ikoma, and members of the PGF and KSA working groups. We would also like to thank Mr. Marc Linster at GMD for his good suggestions. We must express our thanks to the Director of ICOT research Center, Dr. Kazuhiro Fuchi.

References

Boose, J.H. (1986). A Knowledge Acquisition Program for Expert Systems Based on Personal Construct Psychology, J. of Man-Machine Studies, Vol. 23, pp 495-525

Boose, J.H. and Bradshaw, J.M. (1987). Expertise Transfer and Complex Problem Using AQUINAS as A Knowledge-Acquisition Workbench for Knowlegde-Based Systems, J. of Man-Machine Studies, Vol.26, No.1, pp3-28

Boose, J. and Gaines, B. (Eds.) (1989a). A SUMMARY OF THE THIRD EUROPEAN WORKSHOP ON KNOWLEDGE ACQUISITION FOR KNOWLEDGE-BASED SYSTEMS, Knowledge Acquisition for Knowledge-based Systems Newsletter, Vol.1, No.2, Aug.

Boose, J. and Gaines, B. (Eds.) (1989b). A SUMMARY OF THE FOURTH KNOWLEDGE ACQUISITION FOR KNOWLEDGE-BASED SYSTEMS WORKSHOP, Knowledge Acquisition for Knowledge-based Systems Newsletter, Vol.1, No.3, Nov.

Breuker, J. and Wielinga, B. (1989). Models of Expertise in Knowledge Acquisition, Topics in Expert System Design, Guida, G. and Tasso, C. (Eds), Elsevier Science Publishers B.V. (North-Holland)

Chikayama, K., and et al. (1989). Overview of Parallel Inference Machine Operating System(PIMOS), ICOT TR No. 483

Date, C. J. (1986). An Introduction to Database Systems, Vol.1, Addison Wesley

Ford, K. M. and et al. (1989). An Approach to Knowledge Acquisition on The Structure of Personal Construct Systems, Proceedings of 4th Knowledge Acquisition for Knowledge-Based Systems Workshop, pp 11-1 - 11-20

Genesereth, M. and Nilsson, N. (1986). Logical Foundations of Artificial Intelligence, Los Altos, CA: Morgan-Kaufman

Hays-Roth, F. , Waterman, D. A. , Lenat,D. B. (1983). Building Expert Systems, Addison-Wesley

Inoue, K., Nagai, Y., Fujii, Y., Imamura, S. and Kojima, T. (1988). Analysis of Design Method of Machine Tools,ICOT Technical Memorandum, No.494 (in Japanese)

Kelly, G.A. (1955). The Psychology of Personal Constructs, New York, Norton

Linster, M. (1989). Report on my Visit to ICOT in August

Mitchell, T.M. (1978). Version Spaces: An Approach to Concept Learning, Stanford Technical Report STAN-CS-78-711, HPP-79-2

Nippon Seiko Corp.(1986). Roll Bearing (Catalogue) (in Japanese)

Quinlan, J.R. (1983). Learning Efficient Classification Procedures and Their Application to Chess End Games, in Machine Learning: An Artificial Intelligence Approach, R.S. Michalski, J.G. Carbonell, and T.M. Mitchell (Eds.), Tioga, Palo Alto, Calif.

Shapiro, E.Y. (1981). Inductive Inference of Theories from Facts, Technical Report 192, Yale University Computer Science Dept.

Taki, H., Tsubaki, K. and Iwashita, Y. (1987). Expert Model for Knowledge Acquisition, Proceedings of Proceedings of IEEE Expert Systems in Government Conference

Taki, H. (1989a). Functional Views of Production Rules, Proceedings of 9th Knowledge Engineering Symposium, SICE , pp95-98 (in Japanese)

Taki, H. (1989b). Knowledge Acquisition by Abductive and Inductive Explanation, Proceedings of 4th Knowledge Acquisition for Knowledge-Based Systems Workshop, pp 34-1 - 34-19

Taki, H. and Fujii, Y. (1989). Operation Presumption: Knowledge Acquisition by Induction, Proceedings of Third European Workshop on Knowledge Acquisition for Knowledge-Based Systems, pp 34-48

Taki, H. and Tsubaki, K. (1990). Expert Model: A Knowledge Representation for Knowledge Acquisition, Journal of Japanese Society for Artificial Intelligence, Vol.5, No.2 (in Japanese)

Taki, H. and Terasaki, S. (1990). Induction Programs for Operation Presumption in KL1 Language , ICOT Technical Memorandum (in Japanese)(to appear)

Kahn, G. , Nowlan, S. and McDermott, J. (1985). Strategies for Knowledge Acquisition, IEEE transactions on Pattern Analysis and Machine Intelligence 7(5)