知識工学と人工知能　59－11
（１９８８．７．７）

プロトコル解析の為の帰納的学習

滝　寛和

（財）新世代コンピュータ技術開発機構

本論文は、専門家作業の時系列情報（プロトコル）からエキスパートシステムの知識ベースを構築する方法である解釈に基づく学習（IBL: Interpretation-Based Learning)について述べる。ＩＢＬでは、専門家作業を「状況の把握」＋「動作」として、その因果関係（「状況」－＞「動作」）を帰納的にルール化することで、専門家の問題解決知識を獲得する。ＩＢＬは、獲得対象の「概念」、「概念間の関係」と「概念の意味する範囲」を先行知識として持つ。先行知識は、専門家作業の記号化、獲得したルールの不要項目の除去や一般化・特殊化に利用する。先行知識を利用する学習には、この他に、説明に基づく学習（ＥＢＬ）があるが、それは、先行知識として、領域理論（Domain Theory）と操作性規範（Operationality Criteria）を持つ。ＩＢＬの「概念の意味する範囲」はこの操作性規範に対応する。ＩＢＬは、この領域理論も獲得できるため、領域理論が不十分な場合により有効である。

Inductive Learning for Protocol Analysis

Hirokazu TAKI

ICOT Research Center,
Institute for New Generation Computer Technology,
4-28, Mita 1-Chome, Minato-ku, Tokyo 108, Japan
csnet: htaki%icot.jp@relay.cs.net,
junet: htaki@icot.junet

Abstract

This paper describes a learning method for building knowledge bases. There are two types of knowledge acquisition systems which extract knowledge from human experts, interactive and non-interactive. This paper describes a non-interactive knowledge acquisition system which acquires a human expert's knowledge by observation. It learns the human expert's problem solving strategies and makes logical rules from temporal sequential data. The learning method of the knowledge acquisition system is interpretation based learning (IBL), which uses advance knowledge in the learning process. Advance knowledge of IBL consists of domain concepts, concept relations and interpretation knowledge, which translates observed data into internal concepts. Although explanation based learning (EBL) also uses advance knowledge, which consists of domain theory and operationality criteria, it learns knowledge using the domain theory, but IBL learns the domain theory itself. IBL is a useful knowledge acquisition method when a domain theory has not been prepared.

1. Introduction

One major problem in building expert systems is removal of the knowledge acquisition bottle-neck. Knowledge acquisition systems have been developed to solve this problem. Most of them are interactive knowledge acquisition systems. This type of system has an interview sub-system. The interview system can access a human expert directly to ask necessary information about his job. Therefore, this type of system is called an active knowledge acquisition system (AKAS)[Boose 84][Boose 87][Taki 87][Kahn 85]. There are many cases or situations in knowledge acquisition environments. Sometimes, the human expert is too busy to answer questions which are asked by the interview system. In this case, the knowledge acquisition method is the observation only. This type of system cannot ask the human expert any questions. This type of system is called a passive knowledge acquisition system (PKAS)[Taki 88]. The AKAS obtains symbolic data (e.g., language representation) interactively from the human expert, but the PKAS obtains not only symbolic data but also numerical data. Therefore, the PKAS must extract numerical data and translate it into symbolic data. The PKAS must build knowledge base inductively from observations only. Most inductive learning systems (i.e., similarity based learning systems) require positive and negative examples. However, only positive examples can be obtained from the observations of human expert operations. One learning system which acquires knowledge from positive examples only. The explanation based learning (EBL) system [Mitchell 85][Mitchell 86], which extracts knowledge effectively using advance knowledge: domain theory and operationality. However, EBL learns only goal concepts which are constructed according to the domain theory. It cannot be used if a domain theory is not prepared. The PKAS must learn the domain theory, too. We are developing a PKAS which has a knowledge-oriented learning mechanism, called interpretation based learning (IBL). IBL learns the domain theory inductively by observation. It has three items of advance knowledge: symbolic concepts, ranges of values of concepts and interpretation knowledge. In IBL, there are some learning strategies, such as translation from sensed information to symbolic concepts, inductive rule generation and noise reduction. The following sections discuss the environment of knowledge acquisition by observation, advance knowledge of IBL, inductive rule generation, and knowledge generalization.

2. Characteristics of PKAS

The PKAS can observe the actions of human experts and the situations in which those actions occurs as shown in figure 1. Normally, this symbolic information is translated from data extracted by sensors. Therefore, the PKAS must be able to interpret the sensed data as internal symbolic representation data. Sometimes, there are ambiguities and noise (useless information) in the sensed data. The PKAS must be able to handle various meanings in the ambiguities in building a knowledge base. Generally, this noise is very harmful. It makes acquired knowledge too specific. The PKAS must have a function which chooses only suitable situations related to actions. Example 1 is a noisy situation.

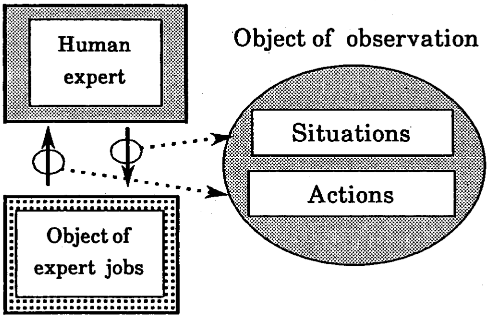


Figure1 PKAS overview

Example 1:

Situation information:

{It rained, and
the output voltage of the amplifier was too low.}

Action information:

{An expert changed an output transistor.}

Result of induction:

(Weather = Rain)
& (Amp-output-voltage = Low)
→ (Change Amp-output-transistor)

This result is too specific to be used in real amplifier maintenance, because the weather is not related to amplifier maintenance. Therefore, the PKAS must choose situations related to the with actions. It has to make the following rule (example 2).

Example 2:
Result of induction:
(Amp-output-voltage = Low)
→ (Change Amp-output-transistor)

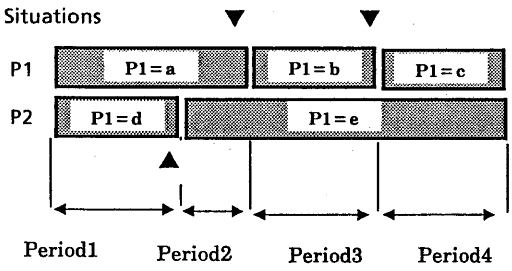
This check is done with domain knowledge which contains relations between situations and actions in a target domain.

2.1 Problems in Knowledge Acquisition by Observation

There are some problems related to ambiguity in the PKAS. They appear in the interpretation process of observed information.

(1) Problem of dividing sensed data
Sensed data is continuously collected at every sampling or when a sampling trigger is detected. Sensed data is temporal sequential data. To symbolize series data, the PKAS divides the data into parts. If sensed data contains some ambiguity, there are many ways of dividing it. Therefore, the PKAS must have knowledge for dividing it to reduce the number of alternatives. The results of dividing data must be matched with internal symbolic concepts. The cause of ambiguity in sensed data is sensor capacity. The sensor has an limitative capacity of detection and detects noise. Figure 2 shows how to make situation data. In this figure, parameter 1 is divided into three parts. Parameter 1 has three values (a, b and c). If parameter 1 changes critically in these three values, it is easy to divide parameter 1. However, generally, parameter 1 does not always change stepwise (i.e., it can be a middle value between a and b) but changes continuously. Therefore, it is difficult to decide the points of change of parameter 1. If more detailed changes are considered, parameter 1 is divided into more parts, and the PKAS obtains more detailed situation information. In this case, the PKAS must be able to handle many concepts related to dividing criteria. the PKAS must have knowledge which divides sensed data into useful level granules corresponding to internal concepts.

(2) Problem of symbolizing divided data
Normally, fragments of sensed data are translated two types: internal symbolic concept representation and parametric information, which consists of a parameter and a range of its value. Parametoric information contains numerical data and a specific instance.



$$
\begin{aligned}
Situation\mathbf{1} &= (P\mathbf{1} = \mathbf{a},\ P\mathbf{2} = \mathbf{d}) \\
Situation\mathbf{2} &= (P\mathbf{1} = \mathbf{a},\ P\mathbf{2} = \mathbf{e}) \\
Situation\mathbf{3} &= (P\mathbf{1} = \mathbf{b},\ P\mathbf{2} = \mathbf{e}) \\
Situation\mathbf{4} &= (P\mathbf{1} = \mathbf{c},\ P\mathbf{2} = \mathbf{e}) \\
Situation\mathbf{i} &= (P\mathbf{1} = p\mathbf{1i}, P\mathbf{2} = p\mathbf{2i}, \\
&\quad \partial P\mathbf{1} = \partial p\mathbf{1i}, \int P\mathbf{1} = \int p\mathbf{1i}, \ldots..)
\end{aligned}
$$

Figure 2 Data division and situation generation

Example 3: The human expert measured register5 with voltage-tester1, and this tester detected from 0 V to 3 mV. Then, he changed register5.

Symbolic data:
use(Voltage-tester), detect(low-voltage, Register)
→ change(Register).

Parametric data:
Voltage-tester = voltage-tester1
(*this variable can be matched only with a voltage-tester*),
Register = register5
(*this varible can be matched only with a register*),
0 V ≦ low-voltage ≦ 3 mV
(*this symbol means low voltage*
(the word "low" is a fuzzy expression)).

To symbolize the sensed data, the PKAS must check matching real data with internal symbolic concepts. In example 3, voltage-tester1 matches the value "Voltage-tester", register5 matches the value "Register", and the real voltage matches "low-voltage". In this case, the PKAS contains concepts of "Voltage-tester", "Register" and "low-voltage". If the PKAS has only concepts of "Tester", "Device" and "no-voltage(or detect-no-voltage)", the symbolic expression is changed as follows:

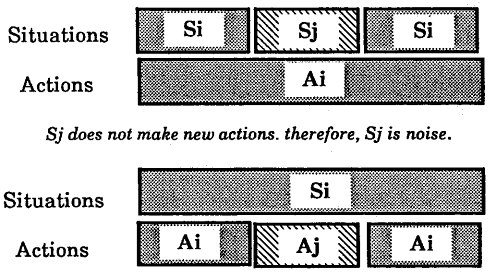
Example 4:
Symbolic data:
use(Tester), detect(no-voltage, Device)
→ change(Device).

Parametric data:
Tester = voltage-tester1
(*this variable can be matched only with a tester*),
Device = register5
(*this variable can be matched only with a device*),
0 V $\leqq$ no-voltage $\leqq$ 3mV
(*this symbol means zero voltage*
(the word "zero" is a fuzzy expression)).

The PKAS must have appropriate concept sets of the target domain. Generally, a concept consists of some sub-concepts. In figure 2, a situation (or a concept) contains two parameters. There is other information in this example, combination information of temporal variable data, which can be thought of as differentiation and integration information. The necessity of higher order differentiation depends on the target domain. The PKAS must have internal symbolic concepts, internal concept sets, and internal parametric definitions.

(3) Problem of combining situations and actions
Situations cause actions in the human expert's tasks. Therefore, at a certain time, there is some causality between situations and actions. The inductive learning system makes rules from these situations and actions. However, there is some noise in these symbols, as shown in example 1. The PKAS must select appropriate situations and actions as shown in figure 2. Normally, there is a time delay in the causality. The PKAS must combine situations and actions carefully. Figure 3 shows noise reduction examples. The first example has situation noise. The situation changed, but the action did not change, and the situation returned the same state. Therefore, Sj must be noise. In the same way, the second example shows action noise. Aj may be noise. The PKAS must have symbolic concept relations to make appropriate rules, and must have a noise reduction mechanism.

3. Interpretation Based Learning



Sj does not make new actions. therefore, Sj is noise.

Aj may be noise, because Si is the same and Aj returns to Ai.

Figure 3 IBL system noise reduction

This section describes an interpretation based learning system and explains the learning flow and mechanism.

3.1 Learning Input and Output

Examples are given as samples of an expert's jobs. They are temporal sequential data. They contain problem solving strategy knowledge of the expert. IBL learns problem solving rules. The following examples show input and output.

Example 5: *Input contents*
Sensing parameters at time t0: p1(t0),p2(t0),...,pn(t0)
Values of the parameter: numerical data, symbol or logical values.

Example 6: *Output contents*
Implication rules: S1&S2&....Sj → a1&a2&...am
An expression Si(i = 1,...,j) is a variable (i = 1,...,j).
An expression ai(i = 1,...,m) is a function with one or more variables.
The variables of the action part are shown as Ai(i = 1,...,k).

Variable boundary and range:
The values of variables (Si/Aj) are numerical values, symbols, or logical values. The variation of the range of a variable, V, is shown as follows:
Equality: V = number/symbol/logical values (e.g., true/false)
Upper limit: V $\leqq$ number-1
Lower limit: V $\geqq$ number-2

Upper and lower limits:
number-2 ≦ V ≦ number-1
A sub-set: V ⊆ {symbol-1, symbol-2,....}

3.2 IBL Learning Strategies

There are seven learning steps in IBL. Figure 4 shows an overview of this flow and advance knowledge.

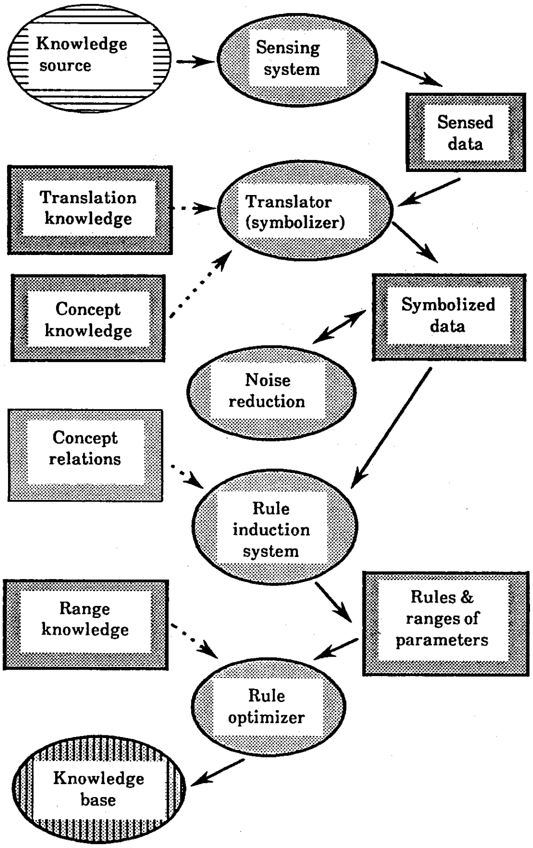


Figure 4 IBL system structure

Step 1: Dividing sensed data
Sensed data consists of many parameters. Each parameter has temporal variable values. IBL checks the value change of each parameter, and divides data in the time scale.

Step 2: Matching sensed data with internal concepts
Here, data is separated into symbolic concepts and parameter instances. Symbolic concepts are set as situations and actions.

Step 3: Reducing noise in situations and actions
There are some relations between situations and actions. Therefore, a certain action data which is independent of situations must be noise. In the same way, a situation data which is independent of actions must also be noise.

Step 4: Making symbolic rules
Rules are made to combine situations and actions. A rule consists of an "if-part" and a "then-part". Situations match the if-part, and actions match the then-part. Sometimes, generated rules also have useless information, as shown in example 1. Therefore, the relationships between situations and actions in all rules must be checked. Unnecessary situations or actions are removed.

Step 5: Optimizing values of parameters
A parameter has an instance value and a range of its value. This instance value is collected from sensed data. It is only one example; therefore, it must be generalized and optimized. It must be changed into the mean value or typical value.

Step 6: Combining parameter data and rules
Rules are very general knowledge in IBL. Specification knowledge of a rules is the parametric range of the values.

Step 7: Generalizing rules and parameter data by multiple examples
IBL learns rules and parameter ranges from step 1 to step 6, when it obtains one example. IBL acquires other knowledge from other examples. Then, it checks and compares rules with the same form. If their actions are the same, the two situations are reformed into a more general situation. The ranges of parameters are also generalized. For example, a parameter consists of "Voltage" as a symbolic name and "0 V ≦ Voltage ≦ 15 V" as the range of its value. A new example brings IBL a new range of its value, i.e., "3 V ≦ Voltage ≦ 20 V". IBL makes a new parameter

which contains "Voltage" as the name and "0 V ≦ Voltage ≦ 20 V" as the range of its value.

4. Advance Knowledge for IBL

One of the most important components of learning systems is advance knowledge. Advance knowledge controls the learning flow. It limits and stimulates the knowledge acquisition system to induce knowledge from examples. In EBL, there are two types of advance knowledge: domain theory and operationality knowledge. Domain knowledge attempts to explain the examples. If an example is implied from the domain knowledge, it is explained and EBL recognizes it as a positive example. An example is given to the EBL system as a goal concept; therefore, it learns how to construct the goal concept from domain knowledge. Operationality knowledge controls the generalization level of explained knowledge. It limits generalization of that knowledge. There are two learning steps in EBL. The first step is the explanation step to check whether an example is positive or not, and the second step is the generalization step to generalize knowledge. As shown in section 3.2, IBL does "interpretation" instead of "explanation", and therefore does not use the domain theory, but it uses domain concepts and relations of concepts.

4.1 Domain Concept Knowledge

The domain concept knowledge means atom level concepts and relations among these concepts. Atom level symbolic concepts mean symbolized situations and actions, and parameter expressions. They also contain ranges of the parameter values. Another type of domain concept knowledge is relation knowledge, which contains relations among symbolic concepts. Each concept has a range of its value. This information is used for parameter generalization and optimization. It is a generalization limit. This range depends on the target domain.

Example 5: *Symbolic concepts*
Symbolic concepts: voltage1, register5, capacitor3.
Parameter expressions: Voltage, Time-delay,
Voltage-tester.

In IBL, a form "register5(Voltage)" is expressed "Voltage-register5" or "Voltage5". A concept (i.e., Voltage5) is sometimes made from multiple concepts (i.e., register5, Voltage).

Example 6: *Range knowledge of concepts*
Parameter range: 0 V ≦ Voltage ≦ 12 V,
3 mA ≦ Ampere8 ≦ 1 A.
In logic circuits, the voltage range is from 0 V to 5 V. This range is 0 V or 5 V in the logical meaning.

Acquired knowledge must be more specific than advance knowledge because IBL must obtain efficient problem solving knowledge.

4.2 Concept Relation Knowledge

Relations among concepts may be positive (e.g., same class concepts and positive relativity), negative (e.g., contrary relativity), no relations, or equations.

Example 7: . *Concept relations (*about force feedback
robot control)
Positive relations:
pair(Movement direction, Velocity vector)
in position control
Negative relations:
pair(X-axis velocity, X-axis pressure)

Note: If the robot's grip touches a wall, a tactile sensordetects pressure in the opposite direction to which it is moving.

No relations: pair(X-axis velocity, Y-axis pressure)
Equations: Velocity = Initial-Velocity * time

4.3 Interpretation Knowledge

Interpretation knowledge is used for translating sensed data into symbolic concepts and parameters. It also contains dividing knowledge for sensed data because divided data must be matched with internal concepts.

Dividing knowledge:
IF | p1(ti)-p1(ti+1) | ≧ e1,
THEN divide parameter p1 at ti.
IF | p1(ti)-p1(tj) | ≧ g1,
THEN divide parameter p1 at tj-1.
e1 and g1 are special knowledge for dividing data.

Symbolizing knowledge (translation knowledge):
IF f1 ≧ p1(from ti to tj) ≧ f2,
THEN p1(from ti to tj) is a concept, "X".
IF p1(from ti to tj) = f3,

THEN p1(from ti to tj) is a concept, "Y".
The range of "X" is from f1 to f2. The value of "Y" is f3.

5. Generation and Generalization

This section describes how to make and optimize rules. It describes the induction method, noise reduction and relation check.

5.1 Rule Generation

Situations and actions are extracted each time. They are represented by symbolic expressions and parameters. IBL makes implication sets (i.e., rules) from situations and actions to select a good set of situations as shown in figure 5. Temporal information shows a sequential rule evaluation flow. rule(ti), which is made from situations and actions that occurred at time ti, makes a new environment which matchs situations of rule(ti+1). Therefore, IBL adds situations made by actions of rule(ti) to the situations of rule(ti+1) shown in example 8.

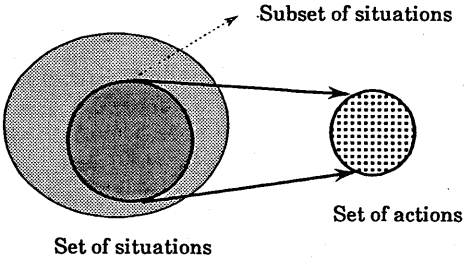


Figure 5 IBL induction

Example 8: . *Rule generation considering temporal information*
Situations: S1, S2 and S3 are observed at time ti+1.
S1 = symbol-1, 0 ≦ S2 ≦ 15 and S3 = symbol-2.
Actions: a1 is done by the human expert at time ti+1.
The parameter of "a1" is A1, and A1 = 20.
Action of rule(ti): a2 is done.

Generated rule(ti+1):
S1& S2& S3 & side-effect of a2 → a1(A1),
in context (S1=symbol-1, 0 ≦ S2 ≦ 15
and S3=symbol-2)

5.2 Noise Reduction

Real noise is caused by sensors and human experts' errors. This noise must be removed. It is unnecessary data in expert jobs. For example, in spite of a sensor detecting a situation, a human expert sometimes does not react to that situation. That situation information is useless data. IBL detects this noise as shown in figure 3. Both situations and actions have some causality with each other. Therefore, data that have no causality must be removed.

5.3 Concept Relation Check

As shown in example 1, sensed data contains most concepts of the target domain. Therefore, generated rules contain unnecessary situations in their "if-part". Each situation must have some causality which depends on the target domain; this causality is dealt with as the concept relation knowledge. IBL uses this concept relation knowledge to reduce the amount of unnecessary information.

5.4. Generalization, Specification, and Optimization

In one learning process, only parameters are generalized or optimized. However, structures of rules are not generalized in one observation, but by multiple examples.

(1) Generalization (optimization) for ranges of parameters

Range expressions are shown in example 6. They show the generalization criteria. Strictly speaking, range information contains a lower case and an upper case. The lower case is used for parameter generalization and the upper case for parameter specialization.

Example 9: *Range optimization*
Lower case (narrow range): 3 ≦ V ≦ 4
Upper case (wide range) : 1 ≦ V ≦ 5
Acquired range: 0.5 ≦ V ≦ 3.5
Optimum range: 1 ≦ V ≦ 4

The lower limit of value "V" must be more than 1 and less than 3; therefore, the acquired range is changed to "1 ≦ V ≦ 3.5". The higher limit of value "V" must be more than 4 and less than 5; therefore, the acquired range is translated into "1 ≦ V ≦ 4".

If an acquired range is within the limits of a lower case, it must be rewritten as a lower case. If it is beyond the limits of an upper case, it must be rewritten as an upper case. A range of an instance is generalized or specialized in order to fit it into a range between the upper case and the lower case. It becomes an optimized range as shown in figure 6.

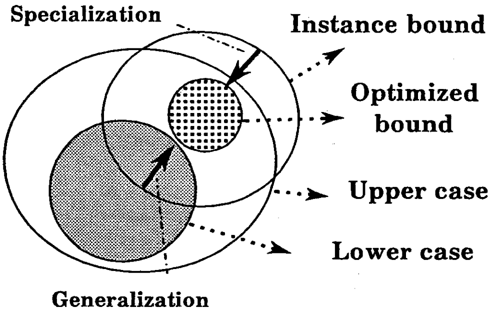


Figure 6 Range optimization

(2) Generalization by multiple examples
There are many rules in one expert task. However, general rules and special rules are mixed in the task. Taking other expert tasks into consideration, some of the same rules are extracted. Both old and new rules have some differences from each other. To use these differences, rules can be generalized. We explain the generalization of IBL for each difference.

Case 1: There are no symbolic differences in the if-parts between new and old rules, and each then-part is the same. However, the values of the new parameters of the if-parts are different from old ones. IBL generalizes the ranges of situation parameters.

Case 2: There are symbolic differences in the if-parts between new and old rules, and each if-part is the same. IBL applies the logical-OR operator to these if-parts and makes a new rule.

Case 3: There is no difference in the if-parts but the new then-part is different from the old one. IBL applies the logical-AND operator to these then-parts and makes a new rule.

6.Acquiered Knowledge and Object Model

The human expert knowledge is made from situations and actions inductively. A form of acquiered knowledge is an implication rule (i.e., situations → actions).These actions are occured by situations in the human expert. Next situations are occured by these actions in an object of expert jobs. Therefore, an implication form (i.e., actions → next situations) represents a sub-model of the object. IBL can also obtain the sub-model of the object. If a detail model (e.g., deep knowledge) of the object is given, we can know a coverage of acquiered knowledge to compare the detail model and the sub-model.

7. Conclusion

IBL learns the human expert's problem solving knowledge by observation. It acquires knowledge in logical form and the range information of the values in logical rules. It cannot obtain general rules from one observation, but it has a function which optimizes parameter ranges. In order to acquire general knowledge, multiple task examples are given to this system. It has not been implemented, but a subset of its functions was developed for a robot skill acquisition system [Taki 85], and it was proved that the major functions of this system are useful for skill acquisition by observation in that system. We believe that it is also useful to extract not only skills but also knowledge of human experts. This paper does not deal with the treatment of alternative interpretations (translations). The TMS [Doyle 79][de Kleer 86] mechanism is useful to maintain the acquired rule base. Acquired knowledge is a logical form; therefore, the partial evaluation techniques in logic programming [Fujita 87] are useful for these rules to reform effective rule sets.

8. Acknowledgment

I would like to thank Mr. Yuichi Fujii, the chief of the fifth laboratory at ICOT, who helped me to further knowledge acquisition research. I wish to