Refinement of the Structure of Knowledge Base by Learning

Chengjiang Mao
The First Academy, the Ministry of Aree-Space Industry
POB 9216, Beijing 100076, P. R. China

Jiarong Hong
Department of Computer Science, Harbin Institute of Technology
Harbin 150006, P. R. China

Abstract

In this paper, we argue that some failure in the existing expert systems mainly results from the ill-organized knowledge base, and propose an effective structure of knowledge base called solution-oriented structure, which is distinguished from the various existing kinds of structures of knowledge bases which we call systematic structure of knowledge base. Employing several strategies of machine learning, a expert system tool called THOUGHT can reorganize the systematic structure into solution-oriented structure. Based on the solution-oriented structure, THOUGHT can perform the three-level inference for solving different kinds of problems, which has some advantages similar to those of neural network, such as concurrency, error tolerance and generalization. The solution-oriented structure of knowledge base is specially efficient and effective on huge complicated knowledge bases and time critical situations.

1. Introduction

Some failure in existing expert systems, such as vulnerability [1-2], saturation [3], irreparable failure behaviour [2] etc., has been exposed. The failure can be viewed as resulting from the following causes: Luc Steels’ lack of knowledge [1], McDermott’s lack of rich problem solving methods [2], and Riet’s lack of structuring knowledge representation [4]. As McDermott stated, the first possibility can be ruled out since the repair is simply to add more knowledge. However, the second possibility also seems not essential because a problem solving method strongly depends on the organization and representation of knowledge. The structuring knowledge representation is a tempting way of the repair, but we are still reluctant to abandon the representation in the form of production rule because of its flexibility and generality. In this paper, we argue that some of the failure mainly results from the ill-organized knowledge bases.

To make complex knowledge bases manageable, we often need decompose them into loosely coupled subsystems [5]. The principle of the decomposition determines the structures of knowledge bases. The knowledge bases in existing expert systems are mostly organized along; (1) functional decomposition of the task to be performed, such as MYCIN [6] and diagnosis of motorcycle [7], (2) ingredient decomposition of the task to be performed, such as PROSPECTOR [8], (3) sequence decomposition of the process of problem solving, such as R1 [10] and HEARSAY-II [11]. The structure of knowledge base produced in these ways is referred to as systematic structure of knowledge base (SSK) in this paper. The blackboard system is a representative model of problem solving for SSK. The major advantage of SSK is that it can be directly built in cooperation with human experts, and has no redundancy. However, there exists some evidence [1-4] showing that this kind of organization (SSK) is not efficient in time-critical situations, and what is even worse, it may cause the irreparable failure and saturation for large complex expert systems.

Here we define all possible complete solutions to a problem as its solution space (SS). In this paper, we pro-
pose that the knowledge base for a problem could be orga-
nized along the decomposition of the problem solution
space, that is, we can partition the knowledge base into
everal (overlapped) subbases called knowledge sources,
each of which serves as the only knowledge base to solve
the problems within one subspace of solution space. The
structure of knowledge base organized in this way is called
collection-oriented structure of knowledge base (SSK) in
this paper. A salient feature of SOK is that it can restrict
the search (inference) space to a very limited one, while
SSK may allow the search space to be the whole knowledge
base. This makes SOK more effective and efficient than
SSK (The detailed differences between SSK and SOK can
be found in Section 2). Unfortunately, we have much diffi-
culty in acquiring SOK from human experts because the
process of the decomposition of solution space is often im-
PLICIT to us. Therefore, we must seek another way to ac-
quire SOK.

We know that humans change as a result of their ex-
perience, augmenting their store of knowledge, reorganiz-
ing their memories, forming generalizations [12], and that
learning ability is central to human intelligence. These
facts provide beneficial suggestion to acquire SOK. In this
paper, we describe a expert system tool called THOUGHT
which obtains SOK by employing more than four strategies
of machine learning to analyze the experience acquired
through a few performances of the expert system whose
structure of knowledge base is SSK. In other words,
THOUGHT can reorganize SSK into SOK. The process of
the reorganization comprises four major stages (see Section
3): acquiring experience which is partly similar to expla-
nation-based learning and ACT* [14], constructing the
skeleton or SOK by CLUSTER/2 [15], generating meta-
knowledge for guiding inference by AQ15 [16], and accu-
mulating of new experience by an incremental learning sys-
stem [16]. Also, THOUGHT has a knowledge discovery
system KD1 [17] for acquiring object-level knowledge,
which is not discussed in this paper. With SOK,
THOUGHT has the three-level inference for problem solv-
ing (see Section 4): (1) meta-inference for familiar prob-
lems which leads to their solutions with high confidence by
speedy retrieval such as dichotomizing search, (2) associ-
ative inference for near-familiar problems which is executed
on the cause-effect experience network with moderate speed
and confidence, (3) heuristic inference for unfamiliar
problems which is performed with lowest speed and confi-
dence on the SSK expert system which serves as a front
system of THOUGHT. We can see that it is similar to hu-
man behaviour that THOUGHT employs different strategies
of problem solving for different kinds of problems. Fur-
thermore, the inference on SOK has some advantages such
as concurrency, error tolerance and generalization which
are similar to those of neural network [18].

2. Differences between SSK and SOK

Without losing generality, we can consider that the
shape of the structure of knowledge base is tree-like, where
each individual leaf contains one knowledge source, and
the nodes between the leaves and the root load explicit
meta-knowledge to guide inference.

Usually, the activity of problem solving on SSK can
be viewed as occurring in three steps [3]: retrieving a set of
plausibly useful knowledge sources from a knowledge base,
refining the chosen knowledge source for a solution, and
executing the chosen knowledge source for the solution.
This process may cause saturation [3] that so many appli-
able knowledge sources are retrieved that it is unrealistic to
consider exhaustive invocation. In the worse cases, the
process may result in irreparable failure that no one knows
what the system does.

The above discussions lead to the idea that if the whole
process of problem solving invokes one knowledge source,
that is, the space of inference is limited to a very narrow
one, the inference will be more effective and efficient.
This requires that each knowledge source be satisfied with
solving the problems within one subspace of the solution
space without recourse to other knowledge sources. SOK
just meets this requirement.

For further comprehension, we use space and mapping
concepts to illustrate the relations between SSK and
SOK. Here we call all functions of an object its function
space (FS), and all knowledge about the object its knowl-
edge space (KS). Fig. 1 shows that SSK = f1 (FS) and the
elements (subspaces) of function space (FS) are disjoint,
and that SOK = g1 (SS) and the elements (subspaces) of
solution space (SS) are disjoint, and that g1 = THOUGHT
(f1, f2). Notice that f1 and g1 are one-to-one mapping,
f2 is many-to-one mapping, and g2 one-to-many map-
ning.

265
From Fig. 1, we can draw the following conclusions:

1. SOK is redundant, while SSK is not.
2. SOK is implicit because the decomposition of the solution space is not explicit while SSK is explicit due to the transparency of the function space decomposition.
3. SOK is highly effective and efficient because $g_1$ is one-to-one mapping (This means that the inference space is strongly narrowed), since $f_2$ is many-to-one mapping, SSK sometimes causes irreparable failure and saturation.

$$g_1: SS \rightarrow KS$$

$$g_2: KS \rightarrow FS$$

The first conclusion explains that the high efficiency of SOK is gained at the cost of memory; the second explains why SSK is popular while SOK has been rarely applied.

The differences between SSK and SOK can be best illustrated by an example. Suppose an expert system is used for diagnosing some malfunctions in an electric system. The knowledge base of the expert system may naturally be partitioned into two different (even disjoint) knowledge sources (SSK), one dealing with circuit troubles and the other dealing with element troubles. Given a concurrent trouble that a short-circuited causes some elements to be ruined, it will be detected by invocation of the two knowledge sources, that is, the whole knowledge base. Obviously, this organization of knowledge base (SSK) suffers heavy cost for problem solving. However, according to SOK, the knowledge base can be decomposed into three knowledge sources: one for dealing with (pure) circuit troubles, one (pure) element troubles, and one (pure) concurrent troubles. Therefore, for the trouble as mentioned above, the expert system based on SOK makes diagnosis with high effectiveness and efficiency by invoking only the knowledge source for dealing with concurrent troubles.

3. Construction of SOK

In this section, we only outline the procedures of automated construction of SOK. The interested reader can refer to [19] and [15-16] for details. The framework of THOUGHT is shown in Fig. 2.

![Fig. 2. Framework of THOUGHT](image-url)

3.1. Acquisition of Experience

Before we proceed, some terminologies need defining. In THOUGHT, rules and data are stored by using attribute-value pairs. Supposing that $x_i$ is the $j$th attribute, a relational statement of $[x_i = v_i]$ is called a selector, where $v_i \in D_j$, and $D_j$ is the domain of $x_i$. In THOUGHT, the premise of a rule is always a conjunction of selectors, and the conclusion of the rule contains a single selector.

The experience, which is the initial information for THOUGHT, can be generated by the inference on any existing rule-based expert systems. THOUGHT first employs an expert system tool, MEST [13], as its front system to solve problems, and then acquire experience from the executions of MEST and extend them. The method of acquisition of experience can be described by the following definitions.

(1) An evidence is the selector present in the premise of a rule, not in the conclusions of any rules. A hypothesis is the
selector which is the conclusion of a rule. A goal is the hypothesis absent in the premises of any rules.

(2). A rule is an alternative to another if they have the same conclusion part.

(3). An experience path \( P(G) \) is the sequence of the rules that follow each other in solving a particular problem whose solution is a goal \( G \). \( P(G) \) is denoted by a tuple of the names of the rules in the sequence.

(4). An experience rule \( R(G) \) is produced by a \( P(G) \). The premise of \( R(G) \) is a conjunction of all evidences present in the premises of all the rules in \( P(G) \), and the conclusion of \( R(G) \) is \( G \).

(5). An extended path \( E(G) \) produced by a \( P(G) \) is a set of names of the rules in \( P(G) \) and their alternatives.

(6). An extended rule \( E(G) \) is such an experience rule that is generated by a possible pseudo-experience path which is produced by replacing one rule in a \( P(G) \) with one of its alternatives. Note that the premise of an \( ER(G) \) should not include any hypothesises either.

(7). An experience network is the cause-effect network produced by the connection of experience paths.

**Example 1.** An unstructured knowledge base about recognition of animals is shown in Fig. 3, which is quoted from Winston’s book [9]. MEST first produces one solution for each animal and then THOUGHT will get the following experiences.

\[
P1(\text{Leopard}) = \langle 5, 1, 9 \rangle, \quad E1(\text{Leopard}) = \{1, 2, 5, 6, 9\}
\]

\[
R1(\text{Leopard}) = \text{Eat-Meat & Hair & Brown & Dull-Spot} \rightarrow \text{Leopard}
\]

\[
ER11(\text{Leopard}) = \text{Canine-Tooth & Claw & Look-Forward & Hair & Brown & Dull-Spot} \rightarrow \text{Leopard}
\]

\[
ER12(\text{Leopard}) = \text{Eat-Meat & Milk & Brown & Dull-Spot} \rightarrow \text{Leopard}
\]

\[
ER13(\text{Leopard}) = \text{Canine-Tooth & Claw & Look-Forward & Milk & Brown & Dull-Spot} \rightarrow \text{Leopard}
\]

\[
P2(\text{Tiger}) = \langle 6, 2, 10 \rangle, \quad E2(\text{Tiger}) = \{1, 2, 5, 6, 10\}
\]

\[
P3(\text{Giraffe}) = \langle 2, 7, 11 \rangle, \quad E3(\text{Giraffe}) = \{1, 2, 7, 8, 11\}
\]

\[
P4(\text{Zebra}) = \langle 1, 8, 12 \rangle, \quad E4(\text{Zebra}) = \{1, 2, 7, 8, 12\}
\]

\[
P5(\text{Ostrich}) = \langle 3, 13 \rangle, \quad E5(\text{Ostrich}) = \{3, 4, 13\}
\]

\[
P6(\text{Penguin}) = \langle 4, 14 \rangle, \quad E6(\text{Penguin}) = \{3, 4, 14\}
\]

\[
P7(\text{Albatross}) = \langle 3, 15 \rangle, \quad E7(\text{Albatross}) = \{3, 4, 15\}
\]

...  

3. 2. Construction of the Skeleton of SOK

THOUGHT adopts the well-known conceptual clustering system CLUSTER/2 to hierarchically classify the collection of extended paths into a classification tree, where the leaf contains a group of extended paths whose size is determined by special criterions [19], and an internal node of the tree represents a cluster of group of extended paths, and each branch of the tree is assigned a disjoint conceptual description which is used to gradually add new experience to the tree afterward (see Section 4.3). Now, the rules involved in the extended paths existing in one leaf of the tree just form an independent knowledge source. In addition, we chose two as the cluster number in every iteration.

**Example 2.** CLUSTER/2 partitions the seven extended paths (see Example 1.) into a hierarchy (Fig. 4).

3. 3. Generation of Meta-Knowledge for Controlling Inference

THOUGHT employs AQ15, a learning system from examples, to automatically generate conceptual descriptions about the capacity of problem solving of one knowledge source against the capacity of problem solving of the other. The examples input to AQ15 are the premises of extend-
FIG. 3. Knowledge Base about the Recognition of Animals

- evidence
- hypothesis
- goal

Knowledge base tree

Knowledge source 1

FIG. 5. Complete SOK Knowledge Base About the Recognition of Animals
ed rules which are capable of describing the capacity of problem solving [19]. Moreover, the examples that come from each knowledge source just form a separate category.

Example 3. The process of generating meta-knowledge by AQ15 is bottom-up in contrast with the top-down procedure of partitioning the knowledge base into a classification tree by CLUSTER/2. Using the premises of extended rules as input examples, AQ15 can generate meta-knowledge as shown in Fig. 5.

3.4. Organization of Experience Rules

To accelerate problem solving, THOUGHT employs CLUSTER/2 to partition the experience rules (whose premises serve as the input data to CLUSTER/2) within each individual knowledge source into a classification tree whose leaf contains one experience rule, and uses AQ15 to generate descriptions about the tree (The descriptions produced by CLUSTER/2 are too complex to be used). To avoid the confusion between the classification tree for partitioning knowledge base and the tree for organizing the experience rules, we refer to the former as knowledge base tree, and the latter as experience rule tree. Fig. 5 shows these two trees.

3.5. Construction of Experience Network

In this stage, we connect the experience paths within each knowledge source to form a network, a subnetwork of the cause-effect network in SSK, and name the intersecting node a net-knot.

4. Usage, Property, and Enrichment of SOK.

In this section, we will discuss the mechanism of inference on SOK, the properties of the inference and the enrichment of SOK by new experience.

4.1. Mechanism of Inference on SOK

THOUGHT has three kinds of inferences called three-level inference, meta-inference, associative inference, and heuristic inference.

4.1.1. Meta-Inference

Searching for an experience rule in SOK to match the input facts is called meta-inference. Meta-inference has three phases: searching in the knowledge base tree for one plausibly useful knowledge source, searching in the experience rule tree in the chosen knowledge source for an experience rule as an answer, and matching the input facts with the premise of the chosen experience rule to verify the answer. If the third phase is ignored, meta-inference will exhibit the ability of generalization and error tolerance similar to that of neural network [18] because the two preceding searching processes only catch the 'key' attributes which are abstracted by AQ15. Obviously, the problem solved by meta-inference, which is called a familiar problem, is experienced, and the solution is highly confident.

4.1.2. Associative Inference

If THOUGHT fails in searching for or matching experience rule's in the experience tree, it will invoke associative inference in an experience network, which is guided by searching net-knots ahead and matching the evidences within the premises of rules pointing to the net-knots with the input facts for pruning inference. Associative inference is much more efficient and effective than the heuristic inference in the front system MEST, not only because the inference is limited in a small space, but also because the experience network is a 'refined' cause-effect network, and conflict resolutions in it become easier than those in SSK. In addition, all the rules in the 'refined' cause-effect network are fired in the past inference. Therefore, the solution by associative inference gets more confident than that by heuristic inference because the rules ever used in past solutions are more confident than those never used. It is easy to find that the problem solved by associative inference, which is called a near-familiar problem, is relevant to experienced problems.

4.1.3. Heuristic Inference

If THOUGHT fails in searching knowledge tree in the stage of meta-inference or fails in associative inference, it will return to its front system MEST to make heuristic inference. Heuristic inference technique refers to any strategies used in current expert systems. We can find that the
problem solved by heuristic inference is unfamiliar to SOK, and the solution is less confident.

4.2. Advantages of Inference on SOK

The inference on SOK has many special properties, which results from the valuable features of CLUSTER/2, AQ15, extended paths and extended rules.

(1) Meta-inference searches for a solution without backtracking, and its complexity is \(O(log_2 N)\), where \(k\) is the number of branches of each node in knowledge base tree and experience rule tree, and \(N\) is the total number of experience rule.

(2) Meta-inference has the ability of generalization and error tolerance.

(3) The solution of one problem just exists in one knowledge source, Multi-problem can be solved parallelly in different knowledge sources.

(4) SOK has accurate self-knowledge ability.

(5) The behavior of the three-level inference for solving different kinds of problems with different methods is very similar to that of human expert.

Example 4. In this example, we will show some advantages of SOK by a sample experiment in the expert system shown in Fig. 3.

![Fig. 6. Improvement of Inference](image)

(1) Based on the SOK knowledge base, the average times of matching for meta-inference are 5, and those for associative inference are 9, while those for heuristic inference are 26 (Fig. 6).

(2) Though the SOK is constructed by analyzing only seven solutions, it covers the whole solution space (22 solutions) of the original knowledge base. The meta-inference on SOK can provide 7 solutions, and the associative inference the other 15 solutions.

(3) Surprisingly, the meta-inference ignoring the third phase (see Section 4.1.1) can give all the 22 solutions with average 4 times of matching and 100% accuracy, and can even guess more solutions than the original knowledge base offers.

These facts show that THOUGHT is capable of greatly improving the performance of SSK expert systems through a few executions of them.

4.3. Accumulation of Experience

All the solutions generated by associative inference and heuristic inference are new experiences to SOK. THOUGHT employs an incremental learning strategy to insert the experiences into SOK. This procedure has two operations: searching for an appropriate knowledge source which an experience adds to, modifying the conceptual descriptions attached in the knowledge base tree and the experience tree. This is a large topic (see [19] for details).

5. Conclusion

This paper describes the following items; definitions and properties of SSK and SOK, procedures of transforming SSK into SOK, three-level inference on SOK and its properties. After about three-year work of our group, THOUGHT has reached to a demonstration stage. Our further work focuses on overcoming the following limitations of the current version of THOUGHT: (1) THOUGHT is suitable only to rule-based knowledge bases; (2) THOUGHT has some difficulties in dealing with knowledge uncertainty; (3) THOUGHT involves much knowledge redundancy for reorganizing the SSKs which are organized by the sequence decomposition of the process of problem solving, such as R1.

Acknowledge

We are grateful to all the members of our group be-
cause of their hard relevant work. Also, we wish to thank Dr. Robert E. Stepp and Prof. Ryszard S. Michalski for their help and suggestion in replicating their CLUSTER/2.

References: