Computer Science Journal of Moldova, vol.1, no.2(2), 1993

Expert games: means to acquire expert knowledge

G.Ginkul

Abstract

This article describes the idea and use of non-traditional technique designed to acquire Expert's knowledge for Expert Systems. The procedures, called "Expert Games", look outwardly as well-known computer "playthings" and at the same time are strong means for Knowledge Acquisition. If Expert wants to succeed, he must to use his professional knowledge. After analyzing game operations of an Expert it is possible to resolve reverse problem – to define which what kind of knowledge have been used for making Expert's decision during the game.

1 Introduction

A new wave of interest to Artificial Intelligence (AI) comes out of Expert System's (ES) existence. ES is a program that relies on the task usually performed only by a human Expert [1]. The strong point of ES derives from the system knowledge rather than from research algorithms and specific reasoning methods. Such characteristics allow to increase the power of ES in comparison with another AI systems to be wide-spread. The process of ES creation is called Knowledge Engineering.

The problem of Knowledge Acquisition is becoming the bottle-neck of Knowledge Engineering. In the broad sense, the Knowledge Acquisition is the transmission of the experience for resolving certain problems by means of AI system. In case of ES application the experienced human Expert is the main source of such information. It is necessary to elicit his professional knowledge and to represent it adequately in the Knowledge Base of ES. The person which does all these things and in general is responsible for creating of ES is called Knowledge Engineer. Knowledge Engineer is analogous to the systems analyst in traditional computer systems design.

Knowledge Acquisition is difficult and time consuming process, in the course of which Knowledge Engineer faces many problems, such as [2]:

- human knowledge is complex and ill-formulated (intelligence is easy to recognize, but more hard to define);
- humans find it difficult to articulate what knowledge they have and how they use that knowledge to solve problems;
- the more Expert someone becomes at a task, the more 'unconscious' his or her knowledge becomes and etc.

Incomplete current methods for working with Expert result the process of the transforming the Expert's knowledge into the knowledge base acording to "test and mistake" method.

Existing Knowledge Acquisition techniques conditionaly would be divided in automatized and non-automatized ones. The non-automatized techniques (interview techniques) supposed to elicit Expert's knowledge during the prolonged dialogue between Expert and Knowledge Engineer. Every interview technique creates appropriate context, in which it is more available for Expert to describe his knowledge, but means for creating such context are different for every technique. Consequently, situations for using each technique are different.

Knowledge Engineer is the leading person in dialogue with Expert. He chooses the next topic of the interview, formulates questions and later on forms rules, which are put in Knowledge Base. The advantage of non-automatized techniques is its simplicity and accessibility. The main disadvantage of interview is time consuming.

With Knowledge Base increasing it becomes more difficult for Knowledge Engineer to view all Knowledge Base and to estimate its correctness. Therefore it is necessary to entrust some functions of Knowledge Engineer to computer and so even if partly to introduce automatization into the process of Knowledge Acquisition.

Today there are two most widespread automated directions of Knowledge Acquisition: Interactive Automated Elicitation and Machine Learning. In the first case, the Expert directly interacts with computer program which assists him to realize and articulate the essence of his knowledge [3]. In Machine Learning the program forms Knowledge Base after analizing of the training set examples, and also the data of another type [4].

However, in spite of appearence of automatized tecniques the stage of Knowledge Acquisition remains non-effective. The especially narrow place of existing methods of Knowledge Acquisition is it's incapability to keep up the permanent Expert's interest to work with a Knowledge Engineer. Today the process of Knowledge Base forming is a very monotonous procedure containing no creative moments. Frequently, Expert is glad to work with Knowledge Engineer only during the first 2-3 minutes. Then the effectiveness of his work falls down. The dialogue with Knowledge Engineer has become the insufferable obligation for Expert and the stage of Knowledge Acquisition dragged on long time. Such circumstance demands stimulation the Expert's activity, to stimulate his work with Knowledge Engineer and to shorten the time for creating ES.

2 Expert Games

In present paper the non-traditional approach is proposed to resolve the problem of Knowledge Acquisition. It is based on the game simulation of Expert's professional activity. The main idea is to elicit the Expert's knowledge during an attractive computer game organized for investigating problem domain. Such games will be called as Expert Games Proposed approach allows to automatize the process of the eliciting the Expert's knowledge and to make it (process) more convenient for problem specialist.

The specificity of Expert Games depends upon it's dual nature: as the game and as the means of Knowledge Acquisition. The Expert Games are built on well known game principles and outwardly they looks as popular well-known computer games. As opposed parties may be either Expert and computer program or a few (two or more) Experts. During the game parties strive to make game goals better and faster than rival for gaining reward (for example, to increase the reserve of points). If one of parties is the computer program it either makes game obstacles for Expert or attempt to make game operations better than Expert.

At the same time Expert Games are based on problem terminology and are games with professional interest. Game situations createed for Expert conditions are similar to that one which he regularly faces at his work. If Expert wants to succeed, he must to use all his professional knowledges. After analyzing game operations of an Expert it is possible to resolve reverse problem – to define which what kind of knowledges have been used for making Expert's decision during the game.

It is necessary to mark the following condition: the Expert Games are similar as much as possible to attractive computer "playthings". Otherwise the game will degenerate in simple interview with Expert. The essence of Expert Games is to "lunge" Expert completely in the world of the game. Sometimes, it is useful not to report to Expert the original goal of the game.

All Expert's operations during the game are filed in the protocol of the game. Such protocol is of great value for Knowledge Engineer, but it has redunant information, so there is danger to stick in great number of inessential facts. Therefore, it is necessary to analyze protocol after the game. For that, for each of Expert Games the special procedure is being created. It processes the information from protocol, and represents it by some traditional method of knowledge representation, for example, as semantic network or as IF-THEN rules. Procedures of analysis are based on some assumptions about principles, which Expert used during the game. Taking it into account, procedure "restores" Expert's professional knowledge.

Every Expert Game allows to elicite information of some special type. At the same time, each Expert Game demands some initial information. It is supposed such information are in Knowledge Base. It may be received by Knowledge Engineer before from special literature or from protocols of another Expert Games. From the practice of Expert Games using it is evident that the Knowledge Acquisition is more flexible and effective when Knowledge Engineer has many Expert Games based on different game principles.

In [5] the Expert Game "Blackbox" was discussed in detail. In that game the test game principle was used. Test game principle based, for example, on guessing of some object (some word or some number) through direct asking a questions. The sequence of question-moves leads to the goal means, in essence, the test for guessing, and the task of player is to build the shortest test.

In the "Blackbox" the Expert allows to guess the name of object from problem domain, which the program has. At the beginning Expert has no information about that object. During the game the program reports the to Expert facts describing the object. On each step of the game program reports to the Expert only one fact. After that Expert must "make stakes", e.g. put ("stake") some part of its resourse on some object or objects. The game encouragement depends directly on the dimensions of stake made true.

So through made or taken away stakes Expert expresses his opinion about the correspondence or non-correspondence of known set of facts to objects from problem domain. And what is more, he expresses his opinion quantitatively!!! The procedure of protocol analysis for "Blackbox" forms IF-THEN rules and put them in Knowledge Base.

Below we propose the Expert Game called "Bulls and Cows". It bases on principle adopted from the the game with the same name for words and numbers. As a problem domain we shall use a toy animal identification problem.

3 The Expert Game “Bulls and Cows”

3.1 Rules of the game

INITIAL INFORMATION. For conducting the game it is necessary to enter the list of facts described the problem domain and objects into knowledge base. For example, it may be next:

ANIMAL EATS MEAT,
ANIMAL HAS CLAWS,
ANIMAL IS CARNIVORE,
ANIMAL IS GIRAFFE,
ANIMAL IS CROW,
ANIMAL IS TIGER, and etc.

Also it is necessary to have in knowledge base the description of one of the objects, for example, the next description of a CROW:

ANIMAL FLIES WELL,
ANIMAL HAS BEAK,
ANIMAL HAS BLACK COLOR,
ANIMAL LAYS EGGS,
ANIMAL CAN’T SWIM,
ANIMAL HAS FEATHERS.

The game allows to enrich the knowledge base with alternative descriptions of the same object and to form the rules determined the cause-reasoned connections between different facts or sets of facts.

In the beginning of the game program informs expert about the name of the object and the number of facts in the set:

ATTENTION !!!
I have some descriptions of A CROW. It includes
6 facts. Let’s try to guess these facts !

In other words, the facts describing a CROW are not reported to expert. The expert’s game goal is to define names of these facts by minimum moves. One move of Expert proposes a set of 6 facts. Responding the move program says how many facts from expert's inquiry are included in a set. For example, in reply to inquiry

ANIMAL FLIES WELL,
ANIMAL IS BIRD,
ANIMAL HAS BLACK COLOR,
ANIMAL HAS WINGS,
ANIMAL CROAKS,
ANIMAL LAYS EGGS.

program will respond: YOU GUESSED 3 FACTS.

After executed movements and received responses, expert makes the next inquiry and etc. The game will be finished when the expert guesses all 6 facts from the set, which are programmed, e.g. when he will receive next response:

CONGRATULATIONS!!! YOU GUESSED ALL 6 FACTS!!!

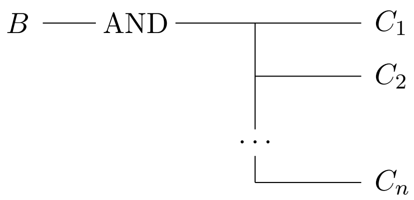
The sequence of expert moves is filed in protocol of the game.

3.2 Rule-based systems and AND/OR Tree

Before describing the procedure of analyzing protocols of games we will view descriptions of rule-based systems. In the present paper we will consider only attribute problem domains and only systems of rules (production systems) as a method of knowledge representation. In attribute problem domains the descriptions of any object or problem situation may be represented in the form of a set of attributes, having a finite number of incompatible values. In other words, there is a certain description with its value for each object of problem domain. For example, attribute SEX may have values MASCULINE and FEMININE. The pair <attribute>: <value> will be called Fact. In essence, the fact is a logic statement, e.g. proposition which is true or false. Remark, the attribute models are widely used in such problem domains as biology, medicine, sociology, etc. Rule-based systems knowledge is represented with the following rules

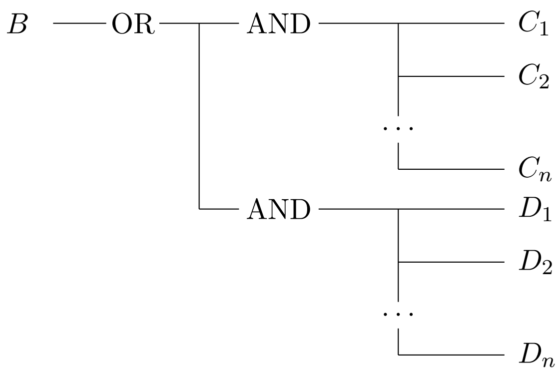
IF <condition>, THEN <conclusion>.

Such rules are called productions. The condition is one fact or conjunction of facts. Production is applicable if its condition is true. Conclusion is a fact which becomes true as a result of application of production. The set of rules forms the knowledge base of production system. Visually the production can be presented as a tree:



where B – conclusion, $C_1, C_2, \dots, C_n$ – conditions.

If there are several rules with the same conclusion, then use operation OR, we can unite a few trees in one:



Consequently, any set of rules can be repesented as the tree called AND/OR Tree. In downners nodes of AND/OR Tree visible facts take place, and in upper nodes – conclusions made by system. Logical inference in the production systems can be considered as a set of rules causing several conclusions and data on the base of which inference worked. Correspondently, the placing of nodes in AND/OR Tree indicates how we can receive conclusion with different sets of initial facts. Conclusion is confirmed if it is an opportunity to unite it with initial facts by the net (route) of confirmed nodes of AND/OR type. The route, indicates the reasons leading an expert to the final conclusion, is called the Decision Tree. It is possible to use different sets of initial facts choosing different ways in AND/OR nodes to achieve certain conclusion. Various decision trees will correspond them. So, AND/OR Tree is the mean allowing to indicate connections between facts and conclusions in some system of rules. Then we will use listed prefix record for designing the AND/OR trees, for example, the last tree will be noted as

$$(\text{OR } (\text{AND } C_1 \ldots C_n) \ (\text{AND } D_1 \ldots D_m)) \longrightarrow B$$

3.3 Analysis of game protocols

The special procedure – PROCEDURE of STRUCTURALIZATION – was created for processing the protocols of “Bulls and Cows”. It based on a few assumptions.

For example, during the game Expert made the next moves:

$$N_1 = \{f_1^1, \ldots, f_k^1\}, \ N_2 = \{f_1^2, \ldots, f_k^2\}, \ldots, \ N_n = \{f_1^n, \ldots, f_k^n\}.$$

Assumption 1 *The Expert's moves during the game were non other than several possible descriptions of guessed object (in our example – CROW).*

Let's discuss made assumption. Undoubtedly, such situation is possible when for examination of some his assumption Expert may to propose something else than the description of guessed object. Such non-adequate behaviour of player often takes place when there is some optimal strategy for game playing, for example as in the case with “Black-box” [8]. But in “Bulls and Cows” we have such situation when the large dimension of the space of possible sets of facts brings to nought all advantages of optimal strategy. In our case it is very easy for Expert to take over all known descriptions of guessed object. Taking that in account, Assumption 1 lookes very reliable.

Below we will sign rule "IF <condition>, THEN <conclusion>" as <condition>⟶<conclusion>. Then basing on Assumption 1 we can write Expert's moves as next the rules:

$$f_1^1 \ \& \ f_2^1 \ \& \dots \& \ f_k^1 \ \longrightarrow X$$

$$f_1^2 \ \& \ f_2^2 \ \& \dots \& \ f_k^2 \ \longrightarrow X$$

$$\dots$$

$$f_1^m \ \& \ f_2^m \ \& \dots \& \ f_k^m \longrightarrow X$$

Such rules can be represenred as AND/OR Tree, for example, in such way:

$$(\text{OR } (\text{AND } f_1^1 \dots f_k^1) \ (\text{AND } f_1^2 \dots f_k^2) \dots (\text{AND } f_1^m \dots f_k^m)) \longrightarrow X.$$

Such AND/OR Trees, formed after simple unification of several descriptions by disjunction we will call Trivial AND/OR Trees. In the some sense trivial trees indicate Expert's knowledge, but they aren't quite satisfactory for us, because they aren't structured and don't unmask steps of Expert's reasons.

Assumption 2 *During the expertise Expert operates not with separate facts, but with groups of connected facts. Such groups of facts define the integral Expert's judgements about guessed object –* **syndromes**. *From the point of view of our game that means that the Expert describing guessed object forms sets of facts not by chance. Most likely he points out the class (family) of object, its subclass (species), and at the end, individual features of object inside species. For that he evidently or non-evidently uses syndromes.*

Syndrome may have several definitions. For example, syndrome BIRD may be defined as
(<ANIMAL HAS FEATHERS> and <ANIMAL LAYS EGGS>)
or as
(<ANIMAL HAS BEAK> and <ANIMAL FLYS GOOD>) .

So "good" AND/OR Tree must have multilevel hierarchical structure in which the root node is the goal object of the problem domain and separated nodes of tree are facts or syndromes. For example:

```
(AND (OR (AND  A B  (OR
                         (AND C K H)
                         (AND E D)    )
          (AND  K L M)                  )
     (OR
     ...
```

At the moment (time) of defining the list of facts describing the problem domain we can not admit anybody to know about genus-species classes inside the problem domain. At the same time it is necessary to know it. With such approach the problem of knowledge acquisition (part of it) is to elicit names and definitions of syndromes and transform trivial trees into hierarchical. It is impossible to do this without expert, but by means of analyze of expert behaviour during the game it is possible to elicit sets of facts consisted of the definitions of classes. The next hypothesis is to make easier the analyze the filed information of the games.

Assumption 3 *When the expert makes the next move, he removes from the last definition of object facts describing some syndrome and put there another definition of the same syndrome. Doing this, expert removes the description of syndrome completely without dividing it into parts. Also it may be supposed, that, first of all he removes individual signs (facts), then – species signs, and in the last place – genus signs. For example, guessing the description of ALBATROSS, expert removes facts "ANIMAL FLYS GOOD" and "ANIMAL HAS BEAK " and replaces them on fact "ANIMAL HAS FEATHERS" and "ANIMAL LAYS EGGS ". In both cases expert wants to mark such circumstance that ALBATROSS is a BIRD, but he describes this syndrome in different ways.*

Proposing here the PROCEDURE of STRUCTURALIZATION receives the list of moves $N_1, N_2, \ldots, N_m$ and forms AND/OR Tree.

Middle nodes of tree are syndromes (OR-nodes) and their definitions (AND-nodes). The next work with built-up tree supposes naming of found syndromes. After that all, a tree can be translated into a set of rules. The base operations used by PROCEDURE of STRUCTURALIZATION are analyzed in accordance with replaces performed by an expert. Let's consider the game during which expert made 4 moves:

$$N_1 = \{a, b, c, d, e, f\}$$

$$N_2 = \{i, j, c, d, e, f\}$$

$$N_3 = \{i, j, c, k, l, m\}$$

$$N_4 = \{i, j, c, k, p, r\}$$

The first move can be shown as such trivial tree:

$$(\text{AND} \;\; a\ b\ c\ d\ e\ f) \longrightarrow X$$

With the second move expert replaces facts a and b on i and j. In accordance with Assumption 3 the sets of facts $\{a \ \& \ b\}$ and $\{i \ \& \ j\}$ are different definitions of the same syndrome, which name is S. The first move syndrome S was described by $\{a \ \& \ b\}$, the second one – by $\{i \ \& \ j\}$. So we suppose the existing some syndrome S and now our AND/OR Tree can be showen as

$$(\text{AND} \;\; (\text{OR} \;\; (\text{AND}\ a\ b)\ (\text{AND}\ i\ j))\ c\ d\ e\ f)) \longrightarrow X.$$

Such tree can be translated as the next set of rules:

$$a \ \& \ b \longrightarrow S$$
$$i \ \& \ j \longrightarrow S$$
$$S \ \& \ c \ \& \ d \ \& \ e \ \& \ f \longrightarrow X,$$

at that time for trivial AND/OR Tree we have:

$$a \ \& \ b \ \& \ c \ \& \ d \ \& \ e \ \& \ f \longrightarrow X,$$
$$i \ \& \ j \ \& \ c \ \& \ d \ \& \ e \ \& \ f \longrightarrow X.$$

The new tree AND/OR Tree has been gained after the PROCEDURE of STRUCTURALIZATION to regulated sequence of moves $N_1 = \{a, b, c, d, e, f\},\ N_2 = \{i, j, c, d, e, f\}$.

Go on to move No 3. Expert replaces facts $\{d, e, f\}$ into facts $\{k, l, m\}$. According to the analogy with the last step we have the next AND/OR Tree:

$$\begin{array}{l}(\text{AND } (\text{OR } (\text{AND } a\ b)\ (\text{AND } i\ j))\ c \\ \qquad (\text{OR } (\text{AND } d\ e\ f)\ (\text{AND } k\ l\ m))) \longrightarrow X.\end{array}$$

Note, the result tree has the next description of X: $\{a, b, c, k, l, m\}$, which has not been proposed by Expert during the game. In other words, the PROCEDURE of STRUCTURALIZATION not only structures moves, but sometimes forms new descriptions of object.

At last, on the move 4 the object was described as $\{i, j, c, k, p, r\}$. That means, the facts $\{i, m\}$ was replaced on $\{p, r\}$. The tree grows down:

$$\begin{array}{l}(\text{AND } (\text{OR } (\text{AND } a\ b)\ (\text{AND } i\ j)) \\ \qquad\qquad c \\ \qquad\qquad (\text{OR } (\text{AND } d\ e\ f)\ (\text{AND } k\ (\text{OR } (\text{AND } l\ m) \\ \qquad\qquad\qquad\qquad\qquad\qquad (\text{AND } p\ r))\ ))\ ) \quad \longrightarrow X.\end{array}$$

After move 4 the game was finished and formed tree is the result of work of procedure.

Note, sometimes the structure of tree is impossible. Such (trivial) situation take place when made moves haven't common facts, e.g. $N_i \cap N_j = \emptyset$ $(i, j = 1, 2, \ldots, m-1)$.

Particular case of trivial situation is when we have the only move $(m = 1)$ or every move have only one fact $(k = 1)$. Corresponding AND/OR Tree are:

$$(\text{AND } \ f_1^1\ f_2^1\ \ldots\ f_k^1) \qquad\qquad (\text{OR } \ f_1^1\ f_1^2\ \ldots\ f_1^m)$$

Case 1   Case 2

Describe the PROCEDURE of STRUCTURALIZATION more detail. It is recursive. To the procedure the sequence of moves is entered.

At the beginning every fact from first move is assumed it's unique index. Later indexes may be changed. During the work of procedure the sequential view of moves is made and for each of them two operations: Direct Inheritance and Confluence is made. The Direct Inheritance is the facts saved in current move in comparison with the last move save (inherit from last move) it's indexes.

The Confluence is all new facts is assumed new (same for all) index. Then this index is assumed to all facts from last moves, which (for facts) indexes was the same with old indexes of replaced facts. For example, if move N_t $(t = 1, 2, \ldots, m)$ was viewed and facts $\{f', f''\}$ had indexes i^* and i^{**} correspondingly was replaced by new facts, then the new view of moves from N_1 to N_t will be make and new index i^{***} will be assumed for all facts with indexes i^* and i^{**}. After analyzing of last move every fact have it's own final index.

At our example the PROCEDURE of STRUCTURALIZATION would be worked so.

Step 1. Every fact from first move is assume it's unique index:

$$\begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \{a, & b, & c, & d, & e, & f\} \end{matrix}$$

Step 2. Consider the move 2. Conduct the Direct Inheritance and Confluence; for facts remained from last move save "old" indexes and assume new index for all new facts:

$$\begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \\ \{a, & b, & c, & d, & e, & f\} \end{matrix} \qquad \begin{matrix} 7 & 7 & 3 & 4 & 5 & 6 \\ \{i, & j, & c, & d, & e, & f\} \end{matrix}$$

View all last moves and assume new index "7" for facts with indexes "1" and "2". Have:

$$\begin{matrix} 7 & 7 & 3 & 4 & 5 & 6 \\ \{a, & b, & c, & d, & e, & f\} \end{matrix} \qquad \begin{matrix} 7 & 7 & 3 & 4 & 5 & 6 \\ \{i, & j, & c, & d, & e, & f\} \end{matrix}$$

By analogy make for third and forth move.

Step 3.

$$\{\overset{7}{a}, \overset{7}{b}, \overset{3}{c}, \overset{8}{d}, \overset{8}{e}, \overset{8}{f}\} \quad \{\overset{7}{i}, \overset{7}{j}, \overset{3}{c}, \overset{8}{d}, \overset{8}{e}, \overset{8}{f}\}$$

$$\{\overset{7}{i}, \overset{7}{j}, \overset{3}{c}, \overset{8}{k}, \overset{8}{l}, \overset{8}{m}\}$$

Step 4.

$$\{\overset{7}{a}, \overset{7}{b}, \overset{3}{c}, \overset{9}{d}, \overset{9}{e}, \overset{9}{f}\} \quad \{\overset{7}{i}, \overset{7}{j}, \overset{3}{c}, \overset{9}{d}, \overset{9}{e}, \overset{9}{f}\}$$

$$\{\overset{7}{i}, \overset{7}{j}, \overset{3}{c}, \overset{9}{k}, \overset{9}{l}, \overset{9}{m}\} \quad \{\overset{7}{i}, \overset{7}{j}, \overset{3}{c}, \overset{9}{k}, \overset{9}{p}, \overset{9}{r}\}$$

Looking finish indexes we easy can restore AND/OR Tree. In it for groups of facts with same indexes the sub-trees united with OR will correspond to:

$$
\begin{array}{llllllllll}
\text{(AND} & \text{(OR} & \text{(AND} & & \overset{7}{a} & \overset{7}{b}) & \text{(AND} & & \overset{7}{i} & \overset{7}{j})) \\
 & \overset{3}{c} & & & & & & & & \\
 & \text{(OR} & \text{(AND} & \overset{9}{d} & \overset{9}{e} & \overset{9}{f}) & \text{(AND} & \overset{9}{k} & \overset{9}{l} & \overset{9}{m}) \\
 & & & & & & \text{(AND} & \overset{9}{k} & \overset{9}{p} & \overset{9}{r}))) \longrightarrow X
\end{array}
$$

In general case not each sub-tree is trivial. For example, the sub-tree

$$\text{(OR (AND } d\ e\ f\text{) (AND } k\ l\ m\text{) (AND } k\ p\ r\text{))}$$

isn't trivial. For structure such sub-tree described procedure use again.

For that in each of sets facts with the same indexes unit in groups:

$$\{<\overset{7}{a},\overset{7}{b}>,\ <\overset{3}{c}>,\ <\overset{9}{d},\overset{9}{e},\overset{9}{f}>\}\quad \{<\overset{7}{i},\overset{7}{j}>,\ <\overset{3}{c}>,\ <\overset{9}{d},\overset{9}{e},\overset{9}{f}>\}$$

$$\{<\overset{7}{i},\overset{7}{j}>,\ <\overset{3}{c}>,\ <\overset{9}{k},\overset{9}{l},\overset{9}{m}>\}\quad \{<\overset{7}{i},\overset{7}{j}>,\ <\overset{3}{c}>,\ <\overset{9}{k},\overset{9}{p},\overset{9}{r}>\}$$

Then new lists are forms from groups, consist facts with the same indexes:

$$\{<\overset{7}{a},\overset{7}{b}>\ \ <\overset{7}{i},\overset{7}{j}>\}\ \text{ and }\ \{<\overset{9}{k},\overset{9}{l},\overset{9}{m}>\ \ <\overset{9}{k},\overset{9}{l},\overset{9}{m}>\ \ <\overset{9}{k},\overset{9}{p},\overset{9}{r}>\}.$$

To each of such lists the PROCEDURE of STRUCTURALIZATION applies again. Have:

$\{<a,b><i,j>\}$ – trivial case,

$\{<c>\}$ – trivial case,

$\{<d,e,f><k,l,m><k,p,r>\}$ – future structuralization is necessary.

Final AND/OR Tree will be looked as:

$$\begin{array}{l}
\text{(AND (OR (AND } \overset{7}{a}\ \overset{7}{b}\text{)}\quad \text{(AND } \overset{7}{i}\ \overset{7}{j}\text{))}\\
\qquad\quad \overset{7}{c}\\
\qquad\quad \text{(OR (AND } \overset{14}{d}\ \overset{14}{e}\ \overset{14}{f}\text{)}\\
\qquad\qquad\quad \text{(AND } \overset{14}{k}\ \text{(OR}\ \text{(AND } \overset{15}{l}\ \overset{15}{m}\text{)}\\
\qquad\qquad\qquad\qquad\qquad\qquad\quad \text{(AND } \overset{15}{p}\ \overset{15}{r}\text{))))} \longrightarrow X
\end{array}$$

However with described procedure not everything it is possible to build non-trivial AND/OR Trees. Consider, for example, such situation. Suppose, Expert made five moves during the game:

$$N_1 = \{a, b, c, d, e, f\}$$
$$N_2 = \{i, j, c, d, e, f\}$$
$$N_3 = \{i, j, c, k, l, m\}$$
$$N_4 = \{i, j, c, k, p, r\}$$
$$N_5 = \{i, s, t, p, r, z\}.$$

In such case after conducting PROCEDURE of STRUCTURALIZATION we will have:

$$\{\overset{10}{a}, \overset{10}{b}, \overset{10}{c}, \overset{10}{d}, \overset{10}{e}, \overset{10}{f}\} \quad \{\overset{10}{i}, \overset{10}{j}, \overset{10}{c}, \overset{10}{d}, \overset{10}{e}, \overset{10}{f}\}$$

$$\{\overset{10}{i}, \overset{10}{j}, \overset{10}{c}, \overset{10}{k}, \overset{10}{l}, \overset{10}{m}\} \quad \{\overset{10}{i}, \overset{10}{j}, \overset{10}{c}, \overset{10}{k}, \overset{10}{p}, \overset{10}{r}\}$$

$$\{\overset{10}{i}, \overset{10}{s}, \overset{10}{t}, \overset{10}{p}, \overset{10}{r}, \overset{10}{z}\}$$

By another words, the trivial AND/OR Tree will be formed:

(OR (AND $a\ b\ c\ d\ e\ f$)
(AND $i\ j\ c\ d\ e\ f$)
(AND $i\ j\ c\ k\ l\ m$)
(AND $i\ j\ c\ k\ p\ r$)
(AND $i\ s\ t\ p\ r\ z$)) $\longrightarrow X$.

At the same time we have none of the trivial cases described above. Consider this situation more detail.

On the fifth move Expert replaces facts $\{j, k, c\}$ with facts $\{s, t, z\}$. As it was found after that move, it is impossible to build non-trivial tree for five moves and in spite of this kept strictly the conditions of Proposition 3. Really we can convince either any non-trivial tree which doesn't contain the decision tree for any step of game or it is in contradiction with the condition "Expert doesn't divide definitions of syndromes".

Reviewing that example we can conclude the following: the restoration of the tree is based on Assumption 3 strongly dependent on the order of moves. Really, each application (already the first one) of PROCEDURE of STRUCTURALIZATION sorts the set of facts.

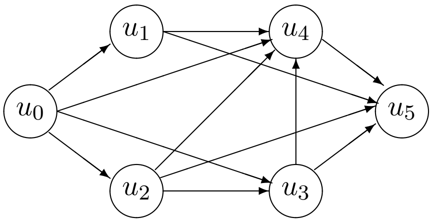
In the future some of moves demand unification causing the confluence of groups and the revision of gained structures. It is possible to construct the trivial AND/OR Tree in the result of such revision. In our example such situation arises after describing the fifth move. The Confluence of groups divided before may take place and with several sub-trees of AND/OR tree are built. In that case a minimal sub-tree becomes trivial. With the aim to remove this shortcoming, the PROCEDURE of STRUCTURALIZATION becomes possible with some changes.

Assumption 4 *During the game we can mark different stages in Expert's behaviour. These are stages of relative stability and moments when one or other reasons (unexpectedly greatly successful, or in contrary causes the great failure) are revised in his reasons. So if the sequence of moves doesn't allow to build non-trivial tree, it is necessary to divide it (sequence) on fragments, for each of which the building of non-trivial tree is possible and then unite building trees with node OR. How to divide initial sequence of moves? There are many possible variants. Let's consider one of them.*

Suppose, Expert makes 5 moves: N_1, N_2, N_3, N_4, N_5. We will build so-called "net of connections", which in general case consists of m+1 nodes. In our case we have:

$$U_0, U_1, \ldots, U_{m-1}, U_m.$$

Nodes U_i and U_j correspondent to the source and gutter of the net. The arc between nodes U_i and U_j $(i, j = 1, 2, \ldots, m-1;\ i < j)$ exists then and only when if it is possible to build non- trivial AND/OR Tree for sequence of moves $N_i, N_{i+1}, \ldots, N_j$. For example it isn't difficult to check that for the sequence of five moves from our example the next net of connections is corresponded:



Any route in the net from source to gutter corresponds to some divided sequence of moves on sequences for each of them it is possible to build non-trivial AND/OR Tree. Evidently, usually only one such route exists, and namely:

$$U_0, U_1, \ldots, U_{m-1}, U_m.$$

Supposed way for dividing the set of moves is to find the minimal route from the source to gutter in the net of connections. In our case we have few of such routes. Let choose one of them, for example, the route (U_0, U_4, U_5). The division corresponds it is: $(< N_0, N_1, N_2, N_3, N_4 >,$ $< N_4, N_5 >)$. For each of sequences $< N_0, N_1, N_2, N_3, N_4 >$ and $< N_4, N_5 >$ the trees AND/OR are built and then they are united with OR-node:

$$
\begin{array}{llll}
\text{(OR} & \text{(AND} & \text{(OR (AND } a\ b\ \text{)(AND } i\ j\ \text{))} & \\
 & & c & \\
 & & \text{(OR (AND } d\ e\ f\ \text{)} & \\
 & & \quad\quad \text{(AND } k\ \text{(OR (AND } l\ m\ \text{)} & \\
 & & \quad\quad\quad\quad\quad\quad\quad\quad \text{(AND } p\ r\ \text{)))))} & \\
 & \text{(AND} & i\ s\ t\ p\ r\ z\ \text{))} & \longrightarrow X.
\end{array}
$$

In practical realization of described procedure Knowledge Engineer may revise all AND/OR Trees, correspondent to all possible routes in the net and choosee one of them.

4 Conclusion

Knowledge acquisition is a difficult, time consuming and different process and, as yet, no formal methodologies have proved universal effectiveness. Knowledge Engineer should make the best use of existing techniques on different stages of knowledge acquisition. As a general principle the Expert should be encouraged to describe his Expertise in the way which is most natural to him. It is necessary to design new methods capable to stimulate Expert activity during the process of ES building.

In this paper we have proposed only one original technique for work with Expert, and described the example of using of such technique. The technique is based on the idea to acquire the Expert's knowledge during the attractive computer game organized in the problem domain. During such Expert Games the situation is similar with the real one. The Expert estimates it and makes decision according to his experience. After the game special procedures analyze protocols of games, "restored" Expert's knowledge and formes IF-THEN rules.

In the Laboratory of AI Systems of Academy of Sciences of Moldova the number of Expert Games were investigated. They are based on different game principles and allow to elicit different types of Expert knowledge. The created games included in the program system CAPRICE (written on LISP for IBM PC), which also include the number of subsidiary programs being effective for knowledge base forming.

References

[1] Parsaye K., Chignell M. Expert Systems for Experts. John Wesley & Sons, 1988.

[2] Hart A. Knowledge Acquisition for Expert Systems. London, Kogan Page, 1986.

[3] Motta E., Eisenstadt M., Pitman K., West M. Support for knowledge acquisition in the Knowledge Engineer's Assistant (KEATS). Expert Systems, 1988, vol.5, N 1, p.6-28.

[4] Machine Learning: An Artificial Intelligence Approach. Vol.1,2 /Ed. by R.Michalski, J.Carbonell, T.Mitchell. Morgan Kaufmann Publ., Inc., 1986.

[5] Ginkul G.P. The game approach to working with expert. Proceedings of 1-st All-Union AI conference. Pereslavl-Zalessky, 1988, part 1, pp.338-342. (Russian).

G.Gincul
Institute of Mathematics,
Academy of Sciences of Moldova,
5 Academiei str.,
Chişinău, 277028, Moldova
e-mail: 23*lsii@math.moldova.su*

Received April 1, 1993