CSEL: A LANGUAGE FOR EXPERT SYSTEMS FOR DIAGNOSIS

Tom Bylander, Sanjay Mittal , and B. Chandrasekaran
Artificial Intelligence Group
Department of Computer and Information Science
The Ohio State University
Columbus, OH 43210 USA

Abstract

We present CSRL (Conceptual Structures Representation Language) as a language to facilitate the development of expert diagnosis systems based on a paradigm of "cooperating diagnostic specialists." MDX, the medical diagnosis system that has been developed in our laboratory over the past few years is based on this paradigm. In our approach, diagnostic reasoning is one of several generic tasks, each of which calls for a particular organizational and problem solving structure. A diagnostic structure is composed of a collection of specialists, each of which corresponds to a node or "concept" in a diagnostic hierarchy, e.g., a classification of diseases. A top-down strategy called establish-refine is used in which either a specialist establishes and then refines itself, or the specialist rejects itself, pruning the hierarchy that it heads. CSRL is a language for representing the concepts of a diagnostic hierarchy and for implementing the establish-refine process. The body of a concept specifies how it will respond to different messages from its superconcept. The knowledge to establish or reject a concept is factored into knowledge groups, which correspond to specific decisions in the diagnosis. We also introduce the concept of a family of languages in which different languages for diagnosis are designed for different kinds of end users.

I Introduction

Many kinds of problem solving for expert systems have been proposed within the AI community. Whatever the approach, there is a need to acquire the knowledge in a given domain and implement it in the spirit of the problem solving paradigm. Reducing the time to implement a system usually involves the creation of a high level language which reflects the intended method of problem solving. For example, EMYCIN was created for building systems based on MYCIN-like problem solving. Such languages are also intended to speed up the knowledge acquisition process by allowing domain experts to input knowledge in a form close to their conceptual level. Another goal is to make it easier to enforce consistency between the expert's knowledge and its implementation. In this paper, we present CSRL (Conceptual Structures Representation Language) as a language to facilitate the development of expert diagnosis systems based on the MDX approach to diagnostic problem solving [4, 8], an approach that has been developed in our laboratory over the past few years. In addition, we introduce the concept of a family of languages in which different languages are designed for different kinds of end users.

First, we will overview the relationship of MDX to our overall theory of problem solving types, the diagnostic problem solving that underlies MDX, and the differences between our approach and the knowledge base/inference engine approach. We then present CSRL in relationship to diagnosis and illustrate many of its constructs. Next, we discuss the family of languages concept. Finally, our immediate plans for using CSRL are listed. Due to space limitations, some understanding of how MDX performs diagnosis is assumed.

Currently at Knowledge Systems Area, Xerox PARC, 3333 Coyote Hill Rd., Palo Alto, CA 94304 USA

II Overview of MDX

A. Types of Problem Solving

Our group at Ohio State has been concerned with how knowledge is organized for expert problem solving. We propose that there are well-defined generic tasks each of which calls for a particular organizational and problem solving structure [3]. Some tasks that we have identified are diagnosis, consequence finding, and knowledge-directed data retrieval. The knowledge of a given domain that applies to a given task can be compiled into a knowledge structure which is tuned for that task. This structure is composed of a collection of specialists, each of which perform the same problem solving, but specialize in different concepts of the domain. Also, each task is associated with a problem solving regime, i.e., how the specialists coordinate for problem solving. The implementation of MDX is based on the diagnostic task.

B. The Diagnostic Task

The diagnostic task is the identification of a case description with a specific node in pre-determined diagnostic hierarchy. The idea of a diagnostic hierarchy is well-established in medicine in the form of disease classification.

(We will use medical terminology in the following, but the reader should keep in mind that the diagnostic task also applies to other domains, e.g., cars, computers, and power plants.) For example, figure 1 shows that cholestasis, cirrosis, and hepatitis are subclasses of liver disease. Cholestasis can be further refined into extra-hepatic and intra-hepatic cholestasis. In the diagnostic task, each disease is associated with a specialist that evaluates its presence or absence in a patient. Specialists in MDX, for example, attempt to classify a cholestatic case according to its etiology.

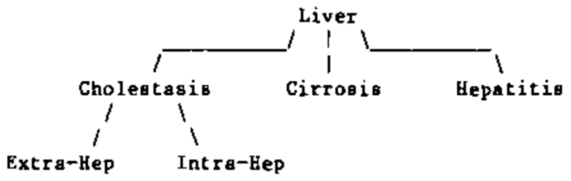

Figure 1: Fragment of a diagnostic hierarchy

A top-down strategy, which we call <u>establish-refine</u>, is used for this task. In relation to figure 1, a simple version of this strategy follows. First the Liver specialist determines if it is <u>established</u>, i.e., if liver disease is likely. If so, Liver <u>refines</u> itself by invoking its subspecialists. Each succeeding level of specialists performs the same establish and refine functions. On the other hand, if the Liver specialist <u>rejects</u> itself, the whole hierarchy of liver diseases can be pruned. This strategy, in combination with the diagnostic hierarchy, is the problem solving regime of the diagnostic task. For a detailed analysis of diagnostic problem solving, see Gomez and Chandrasekaran [7].

An important companion to the diagnostic hierarchy is a data base assistant which organizes the findings in a relevant manner [8, 9]. For example, to determine if a patient has been exposed to anesthetics, the data base, if necessary, can infer this from other data, e.g., major surgery or exposure to ether. Thus the diagnostic structure is insulated from solving problems about finding-finding relationships, avoiding a potentially combinatorial explosion of finding-disease relationships in the specialists of the diagnostic structure.

C. Differences

The usual approach to building knowledge based systems is to emphasize a general knowledge representation structure and different problem solvers which use that knowledge. One difference in the MDX approach is that the organization of its knowledge is not intended as a general representation for all problems. Rather it is tuned specifically for diagnosis. By limiting the type of problem to be solved, a specific organizational technique (classification hierarchy) and problem solving strategy (establish-refine) can be used to provide focus and control in the problem solving process.

Another difference is that the specialists in the hierarchy are not a static collection of knowledge. The knowledge of how to establish or reject is embedded within the specialists. Each specialist can then be viewed as a individual problem solver with its own knowledge base. The entire collection of specialists engages in distributed problem-solving.

III CSRL

CSRL is a language for defining a diagnostic hierarchy and for implementing the establish-refine process. A diagnostic hierarchy is represented by defining <u>concepts</u>. Relationships to neighboring concepts are specified in the <u>declarations</u> of the concept. Establish-refine is implemented within CSRL via message passing. Each concept has a <u>body</u> which specifies how the concept will respond to different messages, and which contains the statements which invoke other concepts with messages. The knowledge to establish or reject a concept is factored into <u>knowledge groups</u>, which determine how the case description relates to specific decisions in the diagnosis. For a complete description of CSRL, see Bylander [2],

A. Body and Message Blocks of a Concept

The body of a concept contains a list of message blocks, which specify how the concept will respond to different messages from its superconcept. The message block contains a message pattern, which is matched against the incoming message, and a sequence of CSRL statements, which are executed if the match succeeds. In figure 2, the body of Cholestasis contains two message blocks. The first one will be activated if an "Establish Cholestasis" message is sent from its superconcept, Liver (declared in the Declarations section), and the second, for a "Refine Cholestasis" message. The literal "Self" is bound to the name of the concept.

```
(Define-Concept Cholestasis
    (Declarations (Subconcept-of Liver)
                  ...)
    (Knowledge-Groups ...)
    (Body
        (Message-Block (Establish Self)
            ...)
        (Message-Block (Refine Self)
            ...)))
```

Figure 2: Message blocks in Cholestasis

Message blocks for establish messages are relatively simple since the knowledge groups (described below) do most of the work. Figure 3 shows how one would look for the Stone concept. The knowledge groups are named Xray, Physical, History, and Summary. Within the (Establish Self) message block, an Execute statement runs all the knowledge groups, and then an Establish-Reply statement asserts the value of Summary as the establish value of Stone. The establish value is an integer from -3 to 3, which represents symbolic probabilities from "definitely not" to "definite." A value of 2 or 3 means that the concept has been established. This value is written on a blackboard [6], which other concepts can access.

```
(Define-Concept Stone
    (Declarations (Subconcept-of Extra-Hep)
                  ...)
    (Knowledge-Groups
        (Xray ...)
        (History ...)
        (Physical ...)
        (Summary ...))
    (Body
        (Message-Block (Establish Self)
            (Execute Xray History
                     Physical Summary)
            (Establish-Reply Summary))))
```

Figure 3: Statements for establishing Stone

Refining a concept is more complicated since the message block must be carefully tailored to follow the establish-refine strategy. In figure 4, the (Refine Self) message block contains two Callexpert statements. The first one calls each subconcept with an establish message (Subconcepts is bound to the declared list of subconcepts). The second Callexpert statement calls each subconcept that was established with a refine message.

Message passing is appropriate for the diagnostic task since the establish-refine regime easily translates into a message protocol, in which the messages clearly indicate the important activities of the concept. Also note that although each concept would have an establish message block in this formulation, the way that a concept establishes itself is concept-specific, i.e., a concept has its own knowledge groups.

B. Knowledge Groups

The Knowledge-Groups section contains a list of knowledge groups, which are used to evaluate how the case description relates to the establish value of a concept. A knowledge group (kg) can be thought of as a cluster of production rules which map the values of a list of conditions (boolean and arithmetic operations on data) to some conclusion on a discrete, symbolic scale. Different types of kg's perform this mapping differently, e.g., directly mapping values to conclusion, or having each rule add or subtract a set number of "confidence" units. Generally, the knowledge in a concept is factored into several kg's, and other kg's are used to combine their results. See [5] for a discussion on combining diagnostic knowledge in this way, as well as reasoning with uncertain data.

Stone is a subconcept of Extra-Hep in MDX. It represents the disease "stone causing extra-hepatic cholestasis."

```
(Define-Concept Liver
    (Declarations (Subconcepts Cholestasis
                               Cirrosis
                               Hepatitis)
                  ...)
    (Knowledge-Groups ...)
    (Body
        (Message-Block (Refine Self)
            (Callexpert (E in Subconcepts)
                (With-Message (Establish E)))
            (Callexpert (E in Subconcepts)
                (With-Message
                    (Cond ((Established? E)
                           (Refine E))))))
        ...))
```

Figure 4: Statements for refining Liver

As an example, figure 5 is the Physical kg of the Stone concept presented above. The conditions query the data base (not defined in CSRL) for whether the patient has cholangitis, colicky pain in the liver, or has been vomiting. Each rule in the Match section is evaluated until one "matches." The value corresponding to this rule becomes the value of the kg. For example, the first rule tests whether the first and second conditions are true (the "?" means doesn't matter). If so, then 3 becomes the value of the knowledge group. Otherwise, other rules are evaluated. The resulting value of the table measures the strength of physical evidence towards establishing the Stone concept. The Xray and History kg's of Stone similarly evaluate the radiological and historical evidence. The Summary kg combines their results (the values of the other kg's are the conditions of Summary) into the establish value of Stone.

```
(Physical
   (Options (End-After (Match 1)))
   (Table (Conditions (Present? Cholangitis)
                      (Pain? Abdomen Colicky)
                      (Present? Vomit))
          (Match (If (T T ?) Then 3)
                 (If (? T T) Then 2)
                 (If (? T ?) Then 1)
                 (If (T ? ?) Then 1)
                 (If (? ? ?) Then -1))))
```

Figure 5: Example of a knowledge group

Factoring the knowledge of a concept in this manner has many advantages. Only the relevant knowledge gets invoked. It allows knowledge to be acquired more easily from domain experts because you can focus their attention on some specific subtask. It also allows knowledge to be debugged because it is easier to see what purpose is being served by a knowledge group. This factoring would make it easier for experts to directly enter the knowledge at some future time.

C. Implementation of CSRL

CSRL is implemented on a DEC 20/60 using ELISP, a dialect of LISP developed at Rutgers, and a local version of FRL (Frame Representation Language). The CSRL interpreter and environment takes up an additional 33K words of storage. The environment includes a thorough syntax check when concepts are defined, commands to invoke any concept with any message, and a simple trace facility. CSRL currently allows little user interaction while it is running, but in the future we plan to add a simple explanation facility and to allow the user to "advise" the system during execution.

IV Family of Languages

Designing languages for knowledge representation often has to face conflicting requirements. At one end, they should be powerful enough to allow different kinds of knowledge and control to be expressed. The power is needed in the form of flexibility in the programming constructs available. At the other end, the language should be simple enough so that non-programmers such as domain experts can directly encode their knowledge without having to worry about the representation in the machine.

We are studying how to do this for the diagnostic task by using CSRL to experiment with the notion of "family of languages." The basic idea is that the same task is embedded in all languages in the family. However, some of the languages make stronger commitments to a particular message passing protocol or structuring of knowledge. Thus at the lowest level we have message passing and knowledge grouping but no commitment to any set of messages or any types of knowledge groups. In this regard, the language would become a general-purpose language such as LOOPS [1]. In fact, we are considering using LOOPS as the bottom-level of the diagnosis family.

The higher-level languages in the family would begin to tie these general facilities to the specifics of the diagnostic task. For example, a fixed set of message types may be allowed to carry out the message passing protocol of MDX. The highest levels may go so far as to create types of concepts, with built in templates for the knowledge groups and body. This would allow users to pick out the appropriate template and concentrate only on filling in the knowledge.

CSRL fits into this framework in the following way. A strong commitment is made concerning the types of knowledge groups that are available, but no commitment is made as to the set of messages that must be used. However, the flow of control is definetly restricted to be top-down.

V Current Plans

Our group at Ohio State is currently using CSRL in a variety of domains including blood type analysis, cars, and nuclear power plants. We are also translating MDX's diagnostic structure from the present LISP code to CSRL. We also plan to implement a diagnosis language which non-programmers can use with minimal training to implement prototype diagnostic systems.

Acknowledgments

We would like to acknowledge Jack Smith and Jon Sticklen for several discussions during CSRL's design phase. The language development is funded by a grant from the Battelle Memorial Laboratories University Distribution Program, and experimentation and application in different domains is supported by AFOSR grant 82-0255, and NSF grant MCS-8103480 .

References

1 D. Bobrow and M. Stefik, "The LOOPS Manual," Tech. Report KB-VLSI-81-13, Xerox Palo Alto Research Center, 1982.

2 T. Bylander, "The CSRL Manual," in preparation, 1983.

3 B. Chandrasekaran, "Towards a Taxonomy of Problem Solving Types," AI Magazine. Vol. 4, No. 1, Winter/Spring 1983.

4 B. Chandrasekaran, F. Gomez, S. Mittal, and J. W. Smith, "An Approach to Medical Diagnosis Based on Conceptual Structures," in Proc. IJCAI-79. 1979.

5 B. Chandrasekaran, S. Mittal, and J. W. Smith, "Reasoning with Uncertain Knowledge: The MDX Approach," in Proc. 1st Ann. Joint Conf. of the American Medical Informatics Association. May 1982.

6 L. D. Erman and V. R. Lesser, "A Multi-level Organization for Problem-solving using Many Diverse Cooperating Sources of Knowledge," in Proc. IJCAI-75. 1975.

7 F. Gomez and B. Chandrasekaran, "Knowledge Organization and Distribution for Medical Diagnosis." IEEE Trans. SM&C. Vol. SMC-11, No. 1, pp. 34-42, January 1981.

8 S. Mittal, "Design of a Distributed Medical Diagnosis and Database System," Ph.D. Thesis, Department of Computer and Information Science, The Ohio State University, 1980.

9 S. Mittal and B. Chandrasekaran, "Conceptual Representation of Patient Data Bases," J. of Medical Systems. 1981.