CONTROL AND COOPERATION IN DISTRIBUTED EXPERT SYSTEMS

MICHAEL N. HUHNS, LARRY M. STEPHENS, and RONALD D. BONNELL

Electrical and Computer Engineering Department
University of South Carolina
Columbia, SC 29208

ABSTRACT

This paper presents an investigation of knowledge engineering techniques that support problem-solving activity on a network of workstations. The research takes into consideration goals and constraints derived from an application domain, an organization of the intended users, and the specific problem to be solved. The results of the research would enable the implementation of a network of autonomous expert systems capable of solving problems in a variety of domains.

The major difficulty in reaching these objectives is to control the multiple expert systems so that they cooperate in solving a problem and arrive at a global consensus. This difficulty has been solved by the use of a question-and-answer paradigm, in which the interactions occur by means of a blackboard data structure. The blackboard, implemented at each workstation, also contains meta-knowledge about the questions and answers in the form of values for question difficulty, question importance, and answer confidence. This use of meta-knowledge is a novel stratagem which has not been previously investigated.

I. INTRODUCTION

In its most general form, a distributed expert system consists of multiple processing nodes with one or more expert systems at each node. Distributed processing is now economically feasible because of recent developments in VLSI technology and network technology. Many envision the proliferation of networks of powerful individual workstations in office environments, engineering design (CAD) environments, and distributed sensor environments. The potential advantages of distributed processing are increased reliability and flexibility, enhanced real-time response, lower communication costs, lower processing costs, and reduced software complexity. These potential advantages have yet to be exploited because little research has been done on the semantics and programming methodologies for distributed information processing systems. In particular, the following problems must be addressed:

* Control: the system must achieve coherent behavior even though its control may be distributed among a number of autonomous processing nodes.

* Communications: to cooperate in the solution of a global problem, these nodes must be able to communicate data, tentative results, subproblems, and queries. Knowledge integrity must be maintained system-wide, but the amount of information exchanged should be minimized to reduce the required communications bandwidth.

* Problem Partitioning: the global problem itself must somehow be partitioned and distributed among the processing nodes.

* User Interfaces: to be useful, the system must be able to provide information to its users concerning the status of the problem-solving process and both intermediate and final results. This requires a decision as to which processors communicate information out of the system and which processors formulate the highest level and most abstract hypotheses.

* System Flexibility: the system must be adaptable to the solution of a wide range of problems and must be readily extensible by the addition of more experts and/or more processing nodes.

This paper presents a methodology for control and communication among distributed expert systems.

II. ANALYSIS

A distributed problem solving system operates best according to a question-and-answer paradigm in working towards a global solution or goal. This mimics the cooperative behavior of humans. The communication of a question between knowledge sources (KS´s) is the equivalent of task-sharing, as analyzed in [1]. A question actually constitutes a new subproblem for the system, and this subproblem can potentially be solved by any KS. In fact, a KS can work on any unanswered question or can work on a question that has been specifically directed to it. To prevent every KS from attempting to solve every subproblem, a process of negotiation known as a contract net has been suggested [1,4]. However, this process is inefficient due to excessive overhead in cases where subproblems are very simple.

To eliminate this inefficiency, two values known as the estimated difficulty (D) and the estimated importance (I) are attached to each subproblem. These values provide meta-knowledge about the subproblem and enable implicit control of subproblem solutions. If a problem has a low degree of difficulty and is within the range of expertise of a number of KS´s, then it is simpler and more efficient for the problem to be solved by more than one KS. This has the additional advantage of reducing the number of processors which need to be informed of the results of the subproblem solution. If a subproblem has a high degree of difficulty, then its solution may be postponed until its difficulty decreases (as described below), or it may be worked on by idle processors, or it may be worked on by an appropriate KS immediately. This latter possibility occurs when the estimated importance attached to the subproblem is large.

The values for the estimated difficulty of a subproblem are based on the following two factors:

1. amount of time already spent on the problem

2. solution time for similar problems.

The values for the estimated importance of a subproblem are based on

1. the estimated importance of the parent problem

2. the number of solution paths through the subproblem

3. the number of solution paths around the subproblem

4. the number of other problems dependent on the solution of the subproblem

5. the scope, or level of abstraction, of the subproblem.

These values are initially calculated by the KS which first suggests the subproblem. However, any KS which possesses additional information about the subproblem can recommend modifications to the values. For example, if another KS also requires a solution to the same subproblem, then its estimated importance will increase. From a problem-solving viewpoint, the value calculation attempts to estimate the impact of the information that is expected to be generated by the solution.

The second half of the question-answer paradigm is the communication of answers and results between KS´s. This result-sharing is a form of cooperation in which KS´s assist each other by sharing partial and possibly inexact results, based on different perspectives of an overall problem. The type of control that occurs is data directed and implicit. It can also be considered as a propagation of constraints between KS´s.

When a KS receives results, it must decide whether to accept them, what credibility to associate with them, what goals it should achieve, and what tasks it should execute to accomplish the goals. Similarly, each KS decides which results should be transmitted and when, based on its estimate of the state of the problem solving activity in the network.

These decisions are made according to a reliability measure. Each result transmitted has a value attached to it known as an estimated confidence (C). The value is dependent on

1. the confidence in the data which was used to calculate the result,

2. the confidence in the procedure used to calculate the result, and

3. an estimate from prior experience about the reliability of the result.

A KS can adjust the confidence level for any result for which the KS has additional information. The confidence level provides meta-information about each result. For example, a result with a very low confidence level would be essentially ignored and not communicated between KS's.

The specific mechanism for storing tasks, goals, and results and for interacting with their confidence, difficulty, and importance measures is a blackboard data structure, physically located at each processing node. Although there may be different expert systems at each node, the blackboard mechanism of each is identical, as are the procedures for accessing it.

Figure 1 shows the communication paths for distributing goals (asking questions) and sharing results (answering questions). This represents the flow of knowledge and meta-knowledge in a hierarchically distributed system. Note that this is a symmetric digraph in which there are direct communication paths among the descendents of a node but not among the descendents of different nodes. Also note that the descendents of a given node are all interconnected to form a problem-solving heterarchy.

Figures 2 through 4 describe the communication paths for transmitting changes in meta-knowledge among KS's. Changes in goal difficulty (Figure 2) are transmitted along the same paths as the original goals. This permits a descendent to directly modify the difficulty of a goal created by its ancestor as it works towards that goal.

Figure 3 shows that changes in goal importance can not be communicated from lower levels of the tree to higher levels, corresponding to the evaluation procedure for goals that occurs in most organizational hierarchies.

Figure 4 indicates which KS's are allowed to modify the confidence levels of results calculated by neighboring KS's. Modifications occur as additional results are obtained.

III. CONCLUSIONS

We believe the introduction of the meta-knowledge parameters D, I, and C will enhance the cooperation of expert systems and lead to efficient implementations of them. We are demonstrating this conjecture in three parallel efforts. In the first effort, we are developing the application layer database which will permit the communication of knowledge between the various experts in the system. The network architecture will be hierarchical, because the distributed expert system will be used in environments that are already structured in a hierarchy, such as engineering design teams, management teams, military situations, and factories. Unsolved subproblems with large values for difficulty and importance will be transmitted to higher levels in the hierarchy.

In the second effort we are developing general software tools for the design and implementation of distributed KS's. The third effort is to demonstrate a complete system by implementing a multiple-expert system for curriculum advisement of engineering students. In addition to expert curriculum advisors for each of the engineeering diciplines, higher-level expert systems for scheduling, classroom assignments, and teaching assignments are being developed. The system can then model the interaction and mutual satisfaction of goals at different levels in a hierarchy. The resultant multiple-expert system will provide a basis for evaluating the communications and control structures described above.

IV. REFERENCES

[1] R. G. Smith and R. Davis, "Frameworks for Cooperation in Distributed Problem Solving," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, January 1981, pp. 61-69.

[2] V. R. Lesser and D. D. Corkill, "Functionally Accurate, Cooperative Distributed Systems," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11, No. 1, January 1981, pp. 81-96.

[3] D. B. Lenat, "BEINGS:, Knowledge as Interacting Experts," Proceedings of the Fourth International Joint Conference on Artificial Intelligence, 1975, pp.126-133.

[4] R. G.Smith, "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver," IEEE Transactions on Computers, Vol. C-29, No. 12, December 1980, pp. 1104-1113.

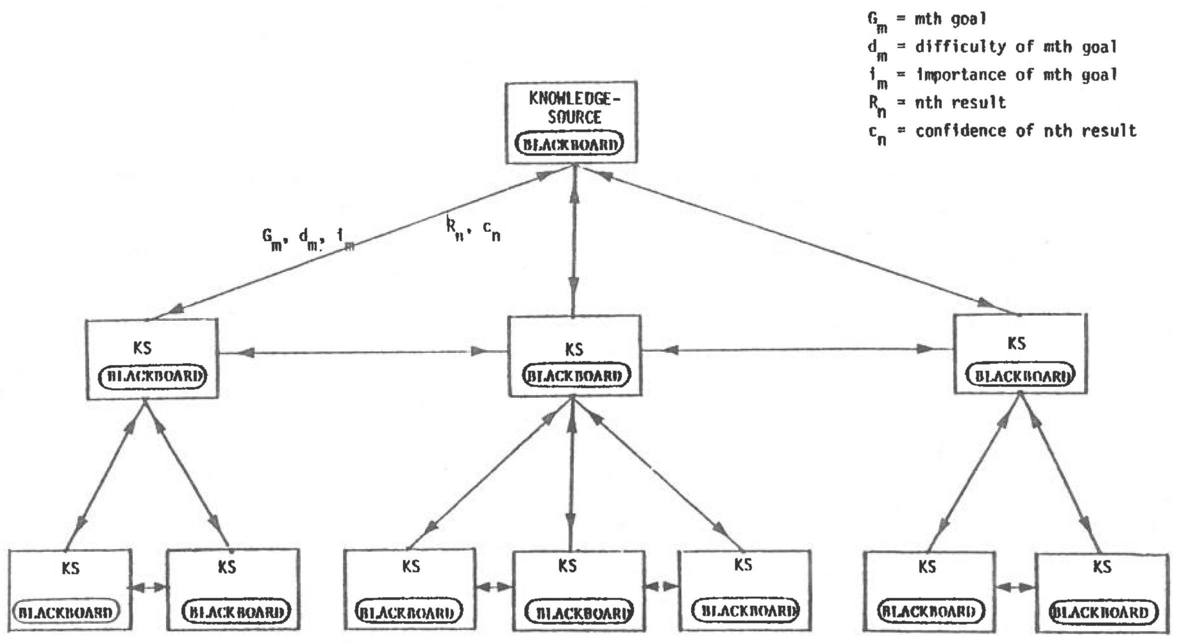

FIGURE 1. Communication Paths for Goal Distribution and Result Sharing.

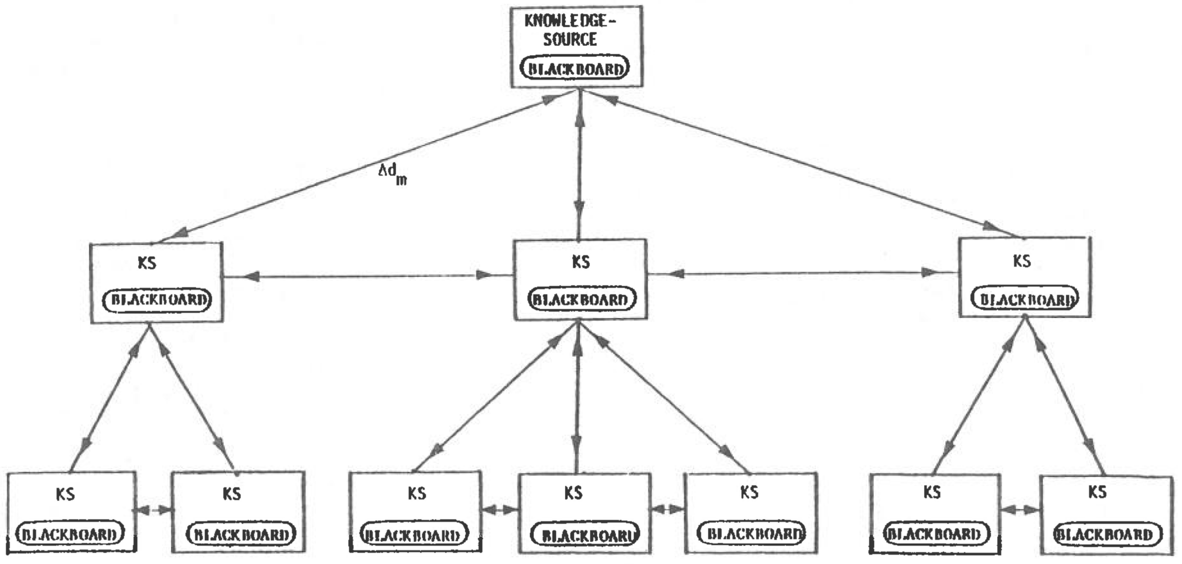

FIGURE 2. Communication Paths for Δd_m, Changes in Goal Difficulty.

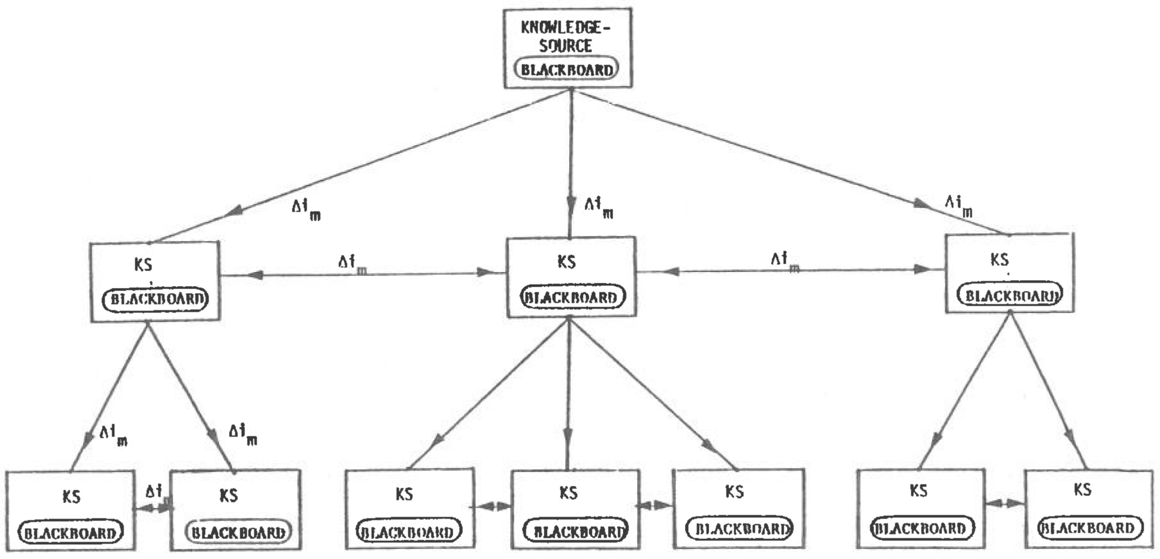

FIGURE 3. Communication Paths for Δi_m, Changes in Goal Importance.

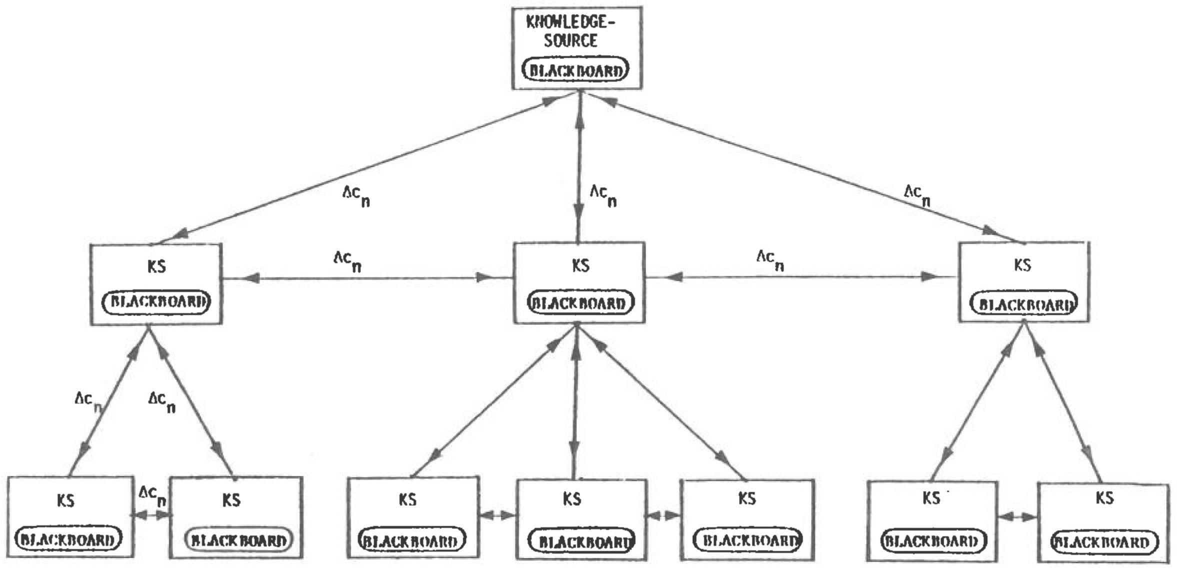

FIGURE 4. Communication Paths for Δc_n, Changes in Result Confidence.

Dr. Michael N. Huhns

CONFERENCE PROCEEDINGS

IEEE SOUTHEASTCON '83

SHERATON TWIN TOWERS
ORLANDO, FLORIDA
APRIL 11–14, 1983

83CH1899-4