N92-16602

VALIDATION OF AN EXPERT SYSTEM INTENDED FOR RESEARCH IN DISTRIBUTED ARTIFICIAL INTELLIGENCE

C. Grossner, J. Lyons, and T. Radhakrishnan

Concordia University
Deparment of Computer Science
Montreal, Quebec
Canada H3G 1M8

Abstract. The expert system discussed in this paper is designed to function as a testbed for research on cooperating expert systems. Cooperating expert systems are members of an organization which dictates the manner in which the expert systems will interact when solving a problem. The Blackbox Expert described in this paper has been constructed using CLIPS, C^{++}, and X windowing environment. Clips is embedded in a C^{++} program which provides objects that are used to maintain the state of the Blackbox puzzle. These objects are accessed by CLIPS rules through user-defined function calls. The performance of the Blackbox Expert is validated by experimentation. A group of people are asked to solve a set of test cases for the Blackbox puzzle. A metric has been devised which evaluates the "correctness" of a solution proposed for a test case of Blackbox. Using this metric and the solutions proposed by the humans, each person receives a rating for their ability to solve the Blackbox puzzle. The Blackbox Expert solves the same set of test cases and is assigned a rating for its ability. Then the rating obtained by the Blackbox Expert is compared with the ratings of the people, thus establishing the skill level of our expert system.

INTRODUCTION

Distributed Artificial Intelligence or DAI is the branch of AI that is concerned with the problems of coordinating the actions of multiple intelligent agents, in order to solve a large and complex problem (Gasser 1987). The agents could be expert systems or other types of AI programs. Two factors that could lead to the distribution of such agents are a geographic distribution required due to the intrinsic properties of the problem being solved and a functional distribution of the problem. The Distributed Vehicle Monitoring Testbed or DVMT distributes its agents based on a geographic distribution of sensors (Durfee 1987) whereas the Distributed Blackbox Testbed described in (Pitula 1980) is based on data partitioning. In this paper, we are concerned with the validation of an expert system (called Blackbox Expert) that solves the Blackbox puzzle. This expert system will be used as an agent in our DAI research. Our laboratory contains a Distributed Computing Facility that is composed of the MACH Distributed Operating System Kernel, a network of SUN workstations, C++, and CLIPS.

The resource constraints of a university environment will permit the development of low cost prototypes only. This normally has several conflicting requirements:

(a) The test environment should be "rich" in problems but not too complex to solve in a realistic time with the available resources.
(b) The test data required by the experiment should be easy to obtain but not trivial.
(c) Exercising and evaluating the prototype should reveal "more than obvious" behaviour of the modeled system.

The prototype systems used thus far in DAI research have been very complex and have required several man years of effort. For example, the manpower expended in the DVMT project is estimated to be 15 to 20 man years. In this context, the advantages and disadvantages of using Blackbox as an experimental test case are described in (Pitula 1990).

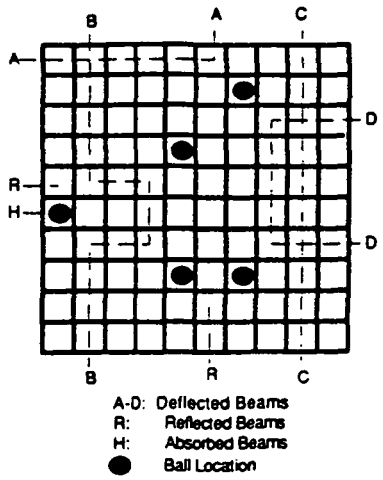


Figure 1: Beam Behaviour in Blackbox

Validation of a system substantiates that it performs with an acceptable level of accuracy (O'Keefe 1987). Developing a validated expert system for DAI research is a non-trivial task and must face several issues:

(a) Acquiring the knowledge and building the knowledge base.
(b) Arriving at an appropriate metric for performance evaluation.
(c) Finding an adequate number of human experts to participate in the validation experiment.
(d) Designing an experiment for statistical evaluation of the performance of the expert system compared with that of humans.

Unlike the case of other well known games and puzzles such as chess, there are no known ways of rating human problem-solvers of Blackbox. On the contrary, Blackbox is simple to learn, the correct solutions are known, and knowledge acquisition is not too expensive. One of the outcomes of our validation experiment is that we now have a set of test cases that are placed into multiple groups of increasing complexity. The performance of the expert system is compared with that of humans using this test set.

BLACKBOX AND DAI

The Blackbox puzzle[1] consists of an opaque square grid (box) with a number of balls hidden in the grid squares. The puzzle solver can fire beams into the box. These beams interact with the balls, allowing the puzzle solver to determine the contents of the box based on the entry and exit points of the beams. As illustrated in Figure 1, the beams may be fired from any of the four sides of the box (along one of the grid rows or columns) and follow four simple rules:

(a) If a beam hits a ball, it is absorbed. (Labelled by 'H')
(b) If a beam tries to pass next to a ball, it is reflected 90 degrees away from the ball in the square diagonally next to the ball. (Labelled Alphabetically except for 'H' and 'R')
(c) If a beam tries to enter the grid at a square adjacent to a border square that contains a ball, it is reflected back out the way it came in. (Labelled by 'R')
(d) If a beam tries to pass between two balls, it is reflected back 180 degrees. (Labelled by 'R')

[1]In previous publications Blackbox was referred to as a game. We have now decided to refer to it as a puzzle, because the word "game" implies an element of luck. The word "puzzle" implies that a skill is needed to find the solution to a problem, which is definitely the case with Blackbox.

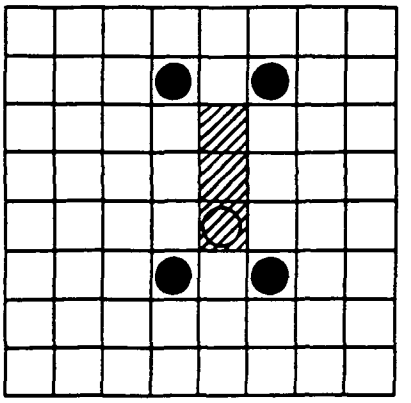


Figure 2: An Example of a Shielded Region.

The objective of the Blackbox puzzle-solver is to determine the contents of as many of the grid squares as possible, while minimizing the total *value* of beams fired. Beams that are absorbed or reflected have a value of one point, while deflections have a value of two points. The puzzle is solved iteratively by firing beams and observing their exit point from the grid. The information obtained from observing the exit points of the beams, and the problem solver's knowledge of how the beam can be affected by the balls within the box are used to create hypotheses about the contents of the box. The amount of information to be processed when solving the puzzle increases with the grid size and the total number of balls.(Pitula 1990).

A region is called a *shielded region* if it is a proper subset of the Blackbox square, it contains at least one ball, and is shielded by other balls so that no beams can penetrate into the region. An example of a shielded region is shown in Figure 2, wherein the shielded region is shaded. No beam can penetrate into the shaded region. The balls contained in this region are called "unmappable balls". In the case of a shielded region, a person can only state that the contents of each square in the region remains unknown.

According to (Parsaye and Chignell 1988), expert system tasks may be categorized into four well defined classes that require different approaches and methodologies: diagnosis and repair, monitoring and control, design and configuration, and intelligent tutoring. The tasks performed by the Blackbox Expert fall into the first category. Diagnosis may be viewed as the task of discovering the relationships between symptoms and faults (diseases). A single symptom, at a given level of granularity, may be the result of many different interacting faults (diseases). On the other hand, a single fault (disease) may produce many symptoms. These one-to-many non-independent relationships make the task of diagnosis quite difficult. Often the diagnosis expert is required to suggest a method of remedy or repair. As an analogy, consider the following pairs of subtasks performed by an expert system for medical diagnosis and the Blackbox Expert respectively:

(a) Observing the symptoms of a disease or the results of a medical test versus observing the exit points of the beams fired.
(b) Suggesting new medical tests versus selecting one or more beams to fire before hypothesising where the balls are located.
(c) Ruling out the possibility of certain diseases versus marking the squares as empty.
(d) Deciding how serious the potential errors in diagnosis are versus setting the weights for the terms in the SCORE function (described in the section on Validation of the Blackbox Expert).
(e) Taking into account the past treatment procedures followed for a disease versus considering the hypotheses generated from the sequence of beams that have been fired.

These types of analogies help in transferring the methodologies developed and the results of research from one problem domain to another.

Several areas that are of interest to the DAI community include negotiation protocols for expert systems (Smith 1980), blackboard architectures (Jagannathan et al. 1989), the sharing of

information among multiple expert systems (Durfee 1987), and coordination of multiple expert systems (Ginsberg 1987). When multiple experts are used to solve a single problem, they can be members of an *organization* (Grossner 1990). An organization defines the type of control flow and data flow permissible among the experts. When dealing with Ill-Structured problems (Simon 1973) there are two phases required for problem-solving, namely the planning phase and the execution phase (Durfee 1986). The control and data flow constraints may be applied in each of these phases as necessary. In (Grossner 1990), it is shown through three different example organizations that Blackbox is an interesting problem for DAI research.

THE BLACKBOX EXPERT

The primary goal in the design of the Blackbox Expert (Lyons 1990a, Lyons 1990b) was to provide an expert system that could later be used in our experiments with *organizations* of cooperating expert systems. Any system to be used for such a purpose requires the following features:

(a) It must be easy to modify the knowledge base.
(b) It must be possible to modify the data structures used by the Blackbox Expert without affecting the knowledge base.
(c) As the data structures used by the Blackbox Expert will be moved from its working memory to a blackboard when the Blackbox Expert becomes a member of an organization, they should be designed to minimize the effects of this move.
(d) It must be possible to monitor which rules are fired as the Blackbox puzzle is solved.
(e) It must be possible to monitor the number and type of accesses to the data structures used by the Blackbox Expert as it solves test cases of the Blackbox puzzle.

Easy modification of the knowledge base provides the flexibility we require for development of the Blackbox Expert and our DAI experiments. In the development stage of the Blackbox Expert, the knowledge needed to solve the Blackbox puzzle will be extracted from human experts. Human experts tend to disagree on the strategies that should be used to solve Blackbox. Thus, many of the rules in the knowledge base will be discovered incrementally by monitoring the performance of a prototype of the Blackbox Expert. When the Blackbox Expert is used in an organization, modifying its knowledge base would permit experimentation with the effects of updates or deficiencies in a single expert's knowledge base on the performance of the organization. The modularity of the knowledge base will also allow construction of different organizations.

Easy modification of the data structures used by the Blackbox Expert is required to support organizations and incremental development of the knowledge base. When the Blackbox Expert is a member of an organization and the data structures are put on the blackboard, the method required to access these data structures will change. Access to a data structure in the working memory requires a local access whereas access to a blackboard will require the Blackbox Expert to generate a request to an independent process, perhaps on a remote computer. Thus, the interface between the rules of the Blackbox Expert and the data structures must be consistent whether a data structure is located locally or on the blackboard. During the development phase of the Blackbox Expert, the functionality and implementation of the data structures will undergo many changes. These changes must be as transparent as possible to the rules in the knowledge base.

Our experiments using the Blackbox Expert for DAI research will depend on the ability to monitor the activities of individual experts. Experimental research conducted to evaluate the effectiveness of proposed solutions to the problems in DAI requires a comprehensive facility to monitor and record the pattern of rules fired and data structures accessed.

In order to meet the design requirements, we decided to use a combination of an expert system shell (Hayes-Roth et al. 1983) and object-oriented design techniques (Budd 1991). The flexibility that was required for incremental growth and modularity of the knowledge base was available as features of several expert system shells. Object-oriented design techniques were able to provide the data encapsulation required to permit the migration of the data structures used by the Blackbox Expert from its local working memory to the Blackboard. Object-oriented techniques also ensured that a modification to the data structures would not require global changes to the knowledge base.

The CLIPS expert system shell (Culbert 1989) and the C^{++} object-oriented programming language (Stroustrup 1987) were chosen for implementing the Blackbox Expert (Lyons 1990c,

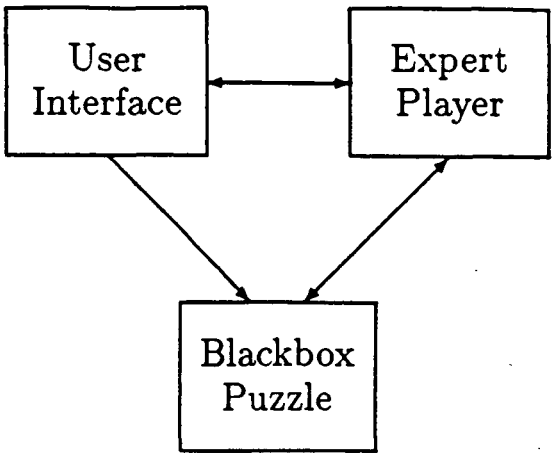


Figure 3: Components of the Blackbox Expert

Lyons 1990d). CLIPS and C^{++} were chosen together because CLIPS is written in C, the source code of CLIPS is available, CLIPS provides a flexible interface to external procedures, and C^{++} provides the data encapsulation facilities. CLIPS also provides the flexibility required for the development of the knowledge base and monitoring the rules that fire when the Blackbox Expert is operating.

As shown in Figure 3, the Blackbox Expert is composed of three major components: the User Interface, the Blackbox Puzzle, and the Expert Player. In this manner, each component of the problem solving system is encapsulated. The arrows depict the flow of data between the modules. The User Interface and the Blackbox Puzzle are implemented using C^{++} and the Expert Player is based on both C^{++} and CLIPS.

The Blackbox Puzzle is responsible for simulating the problem domain. It contains the rules describing the basic principles of the interactions that can occur between the beams and the balls in the Blackbox Puzzle. The Blackbox Puzzle maintains the data structure that contains the location of the balls within the Blackbox grid. It will receive the (X, Y) coordinates of the entry point for beams that are fired by the Expert Player. It determines the trajectory of beams and returns the (X, Y) coordinates of the exit point for the beams.

The Expert Player is responsible for solving the Blackbox puzzle. As shown in Figure 4, the Expert Player is composed of four modules, the Expert Manager, Working Memory, the Rule Base, and the CLIPS Inference Engine. These modules provide the rule base for solving the puzzle, the inferencing capability required to draw conclusions from the rule base, the data structures used by the rule base to maintain its current hypothesis as to the contents of the Blackbox, and an interface between the CLIPS Inference Engine and the other modules of the system.

The Expert Manager maintains control over the operation of the CLIPS Inference Engine and provides the interface between the components of the Expert Player, the Blackbox Puzzle, and the User Interface. It is responsible for the following: responding to commands from the User Interface, instructing the Rule Base to select a beam to fire, instructing the Blackbox Puzzle to fire the beam selected by the Rule Base, and instructing the Rule Base to analyze the result of a beam firing. The different operating modes for the Blackbox Expert, such as *single step*, are controlled by the Expert manager.

Working Memory stores the data structures that are required by the Expert Player. It contains the Expert Player's current hypothesis about the contents of the Blackbox, the number of balls hidden in the grid, the number of balls that have been found thus far, the size of the grid, and a list of the beams that have been fired. The Rule Base accesses Working Memory through a set of user-defined functions that have been added to the CLIPS shell. These functions allow the Rule Base to set the hypothesis for the contents of a square of the Blackbox, check the contents proposed for a grid square, or check a region of the grid to see if it is known to be empty, etc.

The Rule Base used for the Blackbox Expert is divided into 4 groups: Beam Selection, Beam

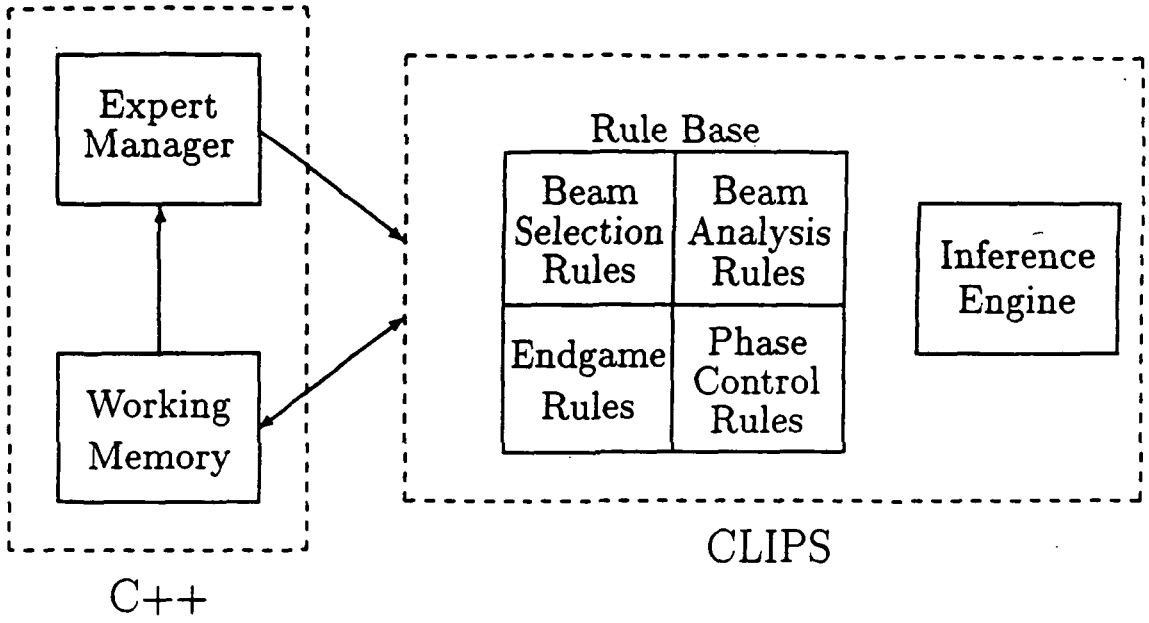


Figure 4: The Components of the Expert Player

Analysis, End Game, and Phase Control. These groupings reflect the different phases required for solving the Blackbox puzzle. The opening strategy for solving Blackbox requires a combination of both firing beams and analyzing the information gained from observing their exit points. The ending strategy for solving Blackbox requires different criteria for determining the contents of the remaining grid squares because there may not be any additional beams that can be fired which will provide more information.

The Beam Selection rules are responsible for determining the best beams for the Blackbox Expert to fire. The Expert Player maintains a list of beams that have yet to be fired along with a rating for each beam. The beam selection rules adjust these ratings based on the entry/exit locations observed for previous beams and the current hypothesis about the contents of the Blackbox grid.

Figure 5 shows a sample beam selection rule. This rule will fire when there is a beam that has not been fired which would potentially pass between two balls and the area of the grid that the beam would pass through is known to be empty. Once the rule fires, it asserts a fact that will cause an adjustment to the beam's rating. The user-defined function *isempty* is used by this rule to determine if the area of the hypothesis grid where the beam would pass before it reaches the ball is empty.

Beam analysis rules are responsible for determining the contents of the grid. Their findings are based on the data observed from the beams that have been fired, and the current contents of the hypothesis grid. Rules can determine that a grid square is empty or that it contains a ball. There are rules that can determine that the position of a ball placed on the hypothesis grid is known to be certain. When two or more rules present contradictory hypotheses, a *conflict* is detected. Thus, conflict resolution rules are required to determine which of the contradictory hypotheses is correct.

The Phase Control rules determine when the Blackbox Expert has finished the current phase. For example, when all the beam selection rules that can fire have fired, the phase control rules will take the highest rated beam and return it to the Expert Manager. The Expert Manager will then fire that beam, place the results in Working Memory and have the Expert Player enter the Beam Analysis phase. The Phase Control rules can also detect the start of a new phase such as the End Game.

The End Game rules are activated when there are very few beams left that the Blackbox Expert will fire. These rules will cause the Blackbox Expert to draw more "risky" conclusions as to the contents of the puzzle. This is justified because during this phase the Blackbox Expert is aware that it has observed most of the beams and it is unlikely to obtain any additional information by firing more beams. The End Game rules will also decide when the Blackbox Expert has finished

```
(defrule W91-24-left
  (phase selection)                    ; fire during beam selection phase
  (poss-ball-found ? ?row1 ?col $?)
  (poss-ball-found ? ?row2&:
            (>= ?row2 (+ ?row1 2))     ; the balls should be in the same
               ?col $?)                ; column, at least two rows apart
  (SHOTLEFT =(+ ?row1 1) 0 $?)         ; a beam has not been fired that
                                       ; will pass next to the upper ball
  (not (ADJUSTED-SHOT =(+ row?1 1)
                      0 W91-24-left)   ; only adjust the beam once for
  (test (isempty ?row1 1 ?row2
                      (-?col 1)))      ; Is grid empty between the balls
                                       ; and the edge of the box
=>
  (assert (ADJUST-SHOT =(+ ?row1 1) 0 50 W91-24-left 0))
                                       ; then adjust the value of the beam
)
```

Figure 5: Sample Beam Selection Rule

solving the puzzle.

The User Interface is responsible for handling the interaction between humans and the Blackbox Expert. It allows a person to monitor and assert control over the Expert's progress as it solves the Blackbox puzzle. The User Interface is written in a combination of both C and C^{++} and runs under the X windowing environment. Both C and C^{++} are required because the widgets used in the X windowing environment are written in C while the procedural modules of the Blackbox Expert which carry out the functions requested by the humans are written in C^{++}. Figure 6 shows the Blackbox Expert's User Interface, with a sample game in progress.

The User Interface consists of four areas: the Real Grid, the Hypothesis Grid, the Dialog Window, and the Command buttons. These four areas allow a human to view the contents of the Blackbox grid as well as the current hypothesis of the Blackbox Expert as it solves the puzzle, allow the Blackbox Expert to annotate the actions it takes to solve the puzzle, and allow a human to issue commands to the Blackbox Expert. Both the Real Grid and Hypothesis Grid are custom made widgets designed specifically for the Blackbox Expert. The Dialog Window and the command buttons are implemented using the ATHENA widget set (O'Reilly and Assoc. 1990).

The Real Grid serves two purposes: it displays the contents of the Blackbox grid, and it allows humans to enter test cases for the Blackbox Expert to solve. Entering test cases is as easy as pointing the mouse cursor at the grid square where a ball is to be placed and pressing the left button. The right button of the mouse is used to remove a ball. Once a new test case has been entered into the Real Grid, the Save command button can be used to store the placement of the balls in a file.

The Hypothesis Grid displays the Blackbox Expert's current hypothesis about the contents of the Blackbox. It also displays the beams that have been fired by the Blackbox Expert. Areas of the grid about which the expert has not made any conclusions are marked with the letter U, for Unknown. Grid squares that the Blackbox Expert concludes contain a ball are marked with the letter B, while squares that the expert concludes are empty are marked blank. When the Blackbox Expert cannot definitely determine the contents of a grid square because two or more rules are in conflict over it, that square is marked with the letter C.

The Dialog Window is used by the Blackbox Expert to display messages that indicate the actions it is taking to solve the current test case of Blackbox. Messages are displayed in a variety of situations such as when beams are fired, and when conflicts are detected or resolved.

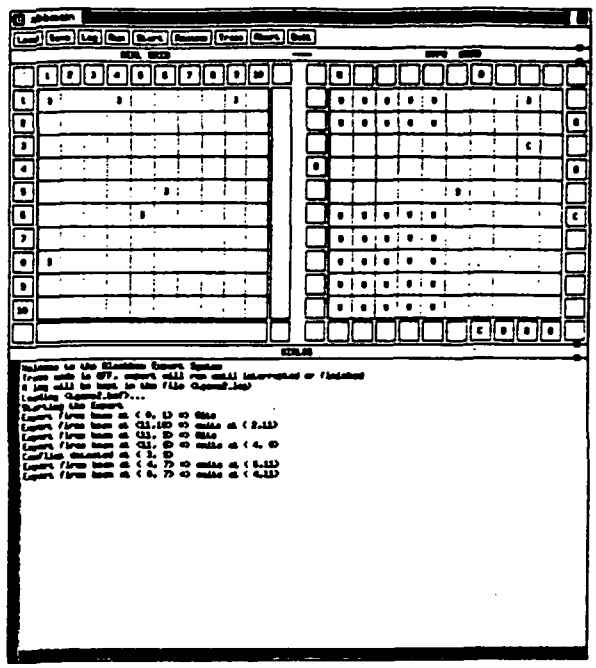


Figure 6: The Blackbox Expert's User Interface

The Command buttons perform the following functions:

Load: Loads a Blackbox test case from a file. A prompt is given to get the name of the file to load.
Save: Saves the current Blackbox test case to a file. A prompt is given to obtain the file in which to save the location of the balls.
Log: Toggles the logging function. If logging is turned on, a list of all the actions performed by the Blackbox Expert while solving the current test case of Blackbox is saved in a file.
Run: Allows the identification of a batch file from which the interface will take all its commands.
Start: Causes the Blackbox Expert to start solving the current test case of Blackbox.
Resume: When in trace mode, the Resume button causes the expert to fire the next beam and then analyze the results.
Trace: Toggles trace mode. When tracing is turned ON, the Blackbox Expert will stop after each beam is fired and analyzed; execution will resume upon the receipt of a Resume Command.
Abort: Aborts a game in progress.
Quit: Terminates the Blackbox Expert.

VALIDATION OF THE BLACKBOX EXPERT

The validation of computer software systems refers to the process of determining if the system satisfies the need for which it was designed (Boehm 1979). Validation of software systems is concerned with the factors that determine the system's usefulness such as: correctness of the results the system produces, its speed, its efficiency, its cost effectiveness, and many human factors. For our purposes, we will be concerned *only* with the correctness of the results produced by the system.

The validation of expert systems is analogous to the validation of general software systems but there are additional problems. The types of problems that are tackled by expert systems, called

Ill-Structured Problems (Simon 1973), do not have a complete specification. Thus, while the problem can seem to be intuitively clear, the criteria to be used for determining the accuracy of the solutions produced by an expert system, or even what constitutes an acceptable solution, are not clear. The lack of a "complete" specification for the problems solved by expert systems is a serious obstacle for the validation process.

The validation of expert systems strongly relies on the opinions of human experts. This has its problems (O'Keefe 1987). Achieving a consensus among a group of human experts is difficult. Generally, a human expert's time is limited and expensive. Producing a "complete" set of test cases for validating the expert system is typically not feasible.

Several models have been proposed for the development lifecycle of an expert system which include validation phases. The spiral model used for general software systems (Boehm 1988) is adapted to include phases that set the acceptable level of performance that is expected from the expert system at different stages of development(O'Keefe and Lee 1990). The heuristic testing approach of (Miller 1990) defines ten prioritized classes of fault types that can occur in an expert system and proposes a method for automatic generation of test cases. These test cases will test the expert system for the types of faults defined by each fault class. Generic tasks are used to decompose a knowledge base into conceptual units (Harrison and Ratcliffe 1991) in order to derive a standard methodology for knowledge base validation.

The development of the Blackbox Expert follows the modified spiral model of (O'Keefe and Lee 1990). Currently we have a "working" research prototype. The initial requirements analysis, requirements verification, and setting of acceptable levels of performance phases are complete. The following describes the validation phase we performed for our research prototype.

The Blackbox puzzle includes several features that facilitate the validation of the Blackbox Expert. Any solution that is proposed by a person for a test case of the Blackbox puzzle can be evaluated because the correct solution to the puzzle is known to another person. The time consumed by a computer or humans to solve each test case of Blackbox is between 10 minutes for someone with a lot of experience, and 30 minutes for a beginner. Therefore validation of the Blackbox Expert is not costly. Developing a population of human experts against whose performance the Blackbox Expert can be validated is simple because the effort required by a human to become skilled at solving Blackbox is not too large.

In consultation with a group of Blackbox experts, a metric has been devised to evaluate the *correctness* of a solution that is proposed for any test case of the Blackbox puzzle. The factors that were chosen to determine the correctness of a solution are: the number of balls that were correctly located, the number of locations of the grid (other than those which contain balls) whose contents are correctly identified, and the total value of the beams fired to solve the puzzle. As stated in the objectives of Blackbox, the best solution would have all the balls and locations correctly identified as well as a minimum total value for the beams that were fired.

The metric that was adopted is as follows:

$$SCORE = \left(2 - \frac{B^C - B^W}{B^M} - \frac{L^C - L^W}{L^T} + \frac{b^V}{b^T}\right) \times 100$$

where:

B^C: The number of correctly located mappable balls.
B^W: The number of incorrectly positioned balls.
B^M: The total number of mappable balls.
L^C: The number of locations of the grid which do not contain a mappable ball that are correctly identified.
L^W: The number of locations of the grid which do not contain a mappable ball that are incorrectly identified.
L^T: The total number of locations of the grid which do not contain mappable balls.
b^V: The total value of the beams fired to solve the puzzle.
b^T: The total number of entry/exit positions of the Blackbox.

This metric assigns a numerical value to a proposed solution of any test case of Blackbox and is used as a measure of the degree of its correctness. A solution with a lower $SCORE$ is considered

to be more correct than a solution with a higher score. This metric examines each of the factors th human experts identified as being relevant to assessing the correctness of a proposed solution o Blackbox. The SCORE function is sensitive both to correct and incorrect responses. The weight placed on each factor rank the number of balls correctly identified as the most important facto followed by the value of the beams fired and the number of correctly identified locations (thos not containing balls) is the least important. The minimum *SCORE* possible is zero and it occur when all balls are found, no beams are fired, and all locations are correctly identified. The maximun *SCORE* possible is 500 and it occurs when no balls are found, all shots are fired, and all location are incorrectly identified.

The Blackbox Expert is validated by comparing the correctness of the solutions it produce with human performance. A group of people, whose familiarity with Blackbox ranges from a fev hours of exposure to several years of exposure, are asked to solve a set of test cases for Blackbox Using the metric for evaluating the correctness of the solutions proposed by these people, eacl problem-solver receives a rating for their ability to solve Blackbox puzzles. The Blackbox Exper solves the same set of test cases and is assigned a rating for its ability. Then the rating obtained by the Blackbox Expert is compared with the ratings of the people, thus establishing the skill level o the Blackbox Expert.

Two people with several years of experience in solving Blackbox developed the set of tes cases to be used for validating the Blackbox Expert. The puzzles in the test set were given a rating o easy, medium, or hard. The two people who developed the test set participated in a group discussior with several other people who also had a lot of experience with Blackbox. The group focused its discussion on the factors that would determine the degree of difficulty of a test case of the Blackbox puzzle. Using the input from this discussion, the two people responsible for the test set determined the criteria used to develop the test set and place each test case that was developed into one of the three categories.

The people who created the test set decided that the following features would contribute to the complexity of a test case:

(a) The presence of unmappable grid squares.
(b) Beam entry and exit points that can be accounted for by many different trajectories through the grid.
(c) The presence of balls in the corners of the box.
(d) A positioning of balls that results in a large number of Hits and Reflections.

The presence of unmappable grid squares increases the complexity of a test case because it makes it difficult to decide when a solution has been found. Many people seem to have an aversion to leaving parts of the Blackbox grid unknown. They will actually convince themselves that they are able to pinpoint the locations of balls which actually are unmappable. People tend to always choose the simplest solution. Thus, when there are many possible trajectories that can account for the entry and exit points of a beam people tend to make errors. If the puzzle-solvers do not confirm their choices for the locations of the balls by firing more beams, they risk making mistakes. The strategy used by many people when solving Blackbox is to work from the edges of the grid towards the center. Balls in the corners of the grid prevent a person from following this strategy thereby increasing the difficulty in solving the puzzle. Deflections provide a lot of information to the puzzle-solver because they often pinpoint the location of a ball and indicate many empty grid squares. A positioning of the balls that results in many hits and reflections is very difficult to solve as there is very little information with which one can determine the contents of the grid.

The rating of each person who participated in our validation experiment was done in two stages. The first stage was designed to be a learning phase and the second stage was the solution of the test set. The learning stage included a set of instructions explaining the basic principles of the Blackbox puzzle as well as the metric used for assessing the correctness of a proposed solution, a demonstration of how a person would solve the puzzle, and a set of sample games designed to demonstrate the principles and the metric described in the instructions. The test phase required each subject to solve the test cases which were presented to them in a random order. Even the subjects with a lot of previous exposure to Blackbox were required to go through the learning phase in order to ensure that they fully understood the metric.

RESULTS: VALIDATION EXPERIMENT

Fifteen people participated in our validation experiment. They solved the 17 test cases in our

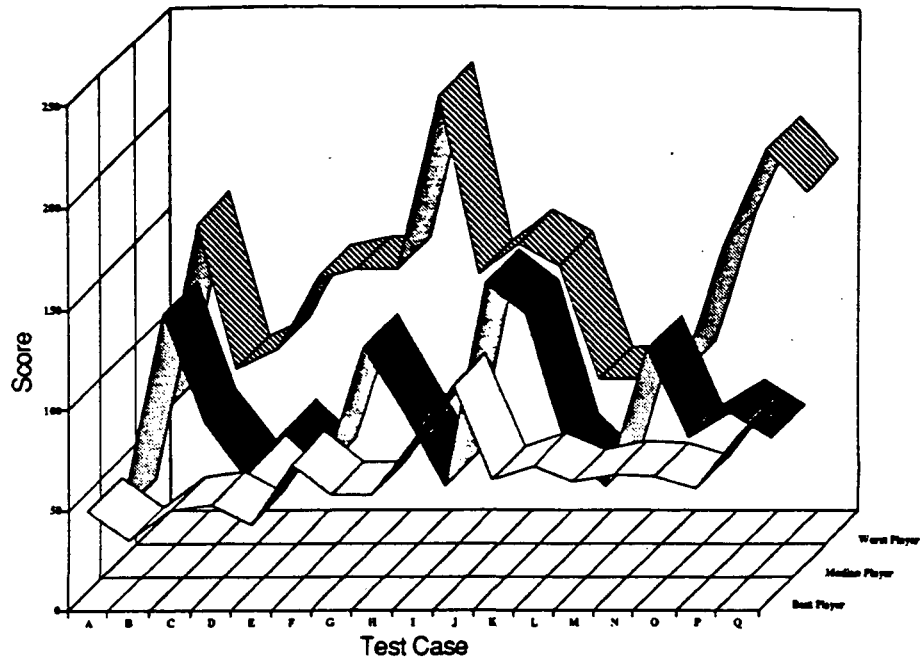


Figure 7: Best, Median, and Worst Player

Blackbox test set. Figure 7 shows the scores obtained by the people who had the best (lowest), median (middle), and worst (highest) average score for all seventeen test cases. Test cases A, B, and C were placed in the easy category, cases D, E, F, and G were placed in the medium category, and the other test cases were placed in the hard category. The person with the best score was also found to be the most consistent puzzle-solver. This consistency is evident from the narrow fluctuation in the scores for the person. The scores for the best person tend to rise slowly from easy to hard test cases. The person at the median has scores that fluctuate more widely than the best player. The person with the worst average score also experiences the largest variation in score.

The best, worst, and median scores for each test case are shown in Figure 8. Again, the lowest scores obtained by any person exhibit the least variation. The median scores vary more than the best scores and the worst scores have the largest variation. These scores also exhibit an upward trend when the easy, medium, and hard test cases are compared.

The average scores and total number of errors made in placing balls in the Blackbox grid by the people who solved the test set are shown in Table 1. Both the average score and total errors made in placing the balls increase when comparing the easy, medium, and hard test cases. As expected, this trend seems to suggest that the performance of the people when solving the test cases from each of the three groups in our test set is different. In order to validate this assumption, we performed an analysis of variance to determine if the difference that is observed in the mean scores of the three groups can be accounted for by the variance in the scores of all the test cases solved. The ANOVA table is shown in Table 2. The F ratio obtained with 2 and 252 degrees of freedom is 13.42. An F ratio of 4.69 or greater is needed for significance with a confidence level of 99%, thus we can reject the null hypothesis that $\mu_{easy} = \mu_{medium} = \mu_{hard}$.

Having determined that the average scores for the three groups are statistically different, we must now examine the individual differences between the groups. Table 2 shows the confidence intervals for the pairwise comparisons of the means of the groups in the test set. These comparisons are done using an F value of $F_{2,252,.95} = 3.035$. As the confidence intervals for $\mu_{easy} - \mu_{medium}$ and $\mu_{easy} - \mu_{hard}$ do not contain zero we can reject the null hypotheses $(\mu_{easy} - \mu_{medium}) = 0$ and $(\mu_{easy} - \mu_{hard}) = 0$. Thus, $\mu_{easy} < \mu_{medium}$ and $\mu_{easy} < \mu_{hard}$. However, the confidence interval for $\mu_{medium} - \mu_{hard}$ does contain zero, which does not permit us to reject the null hypothesis $(\mu_{medium} - \mu_{hard}) = 0$. Thus, there is a statistical difference between the easy and medium groups,

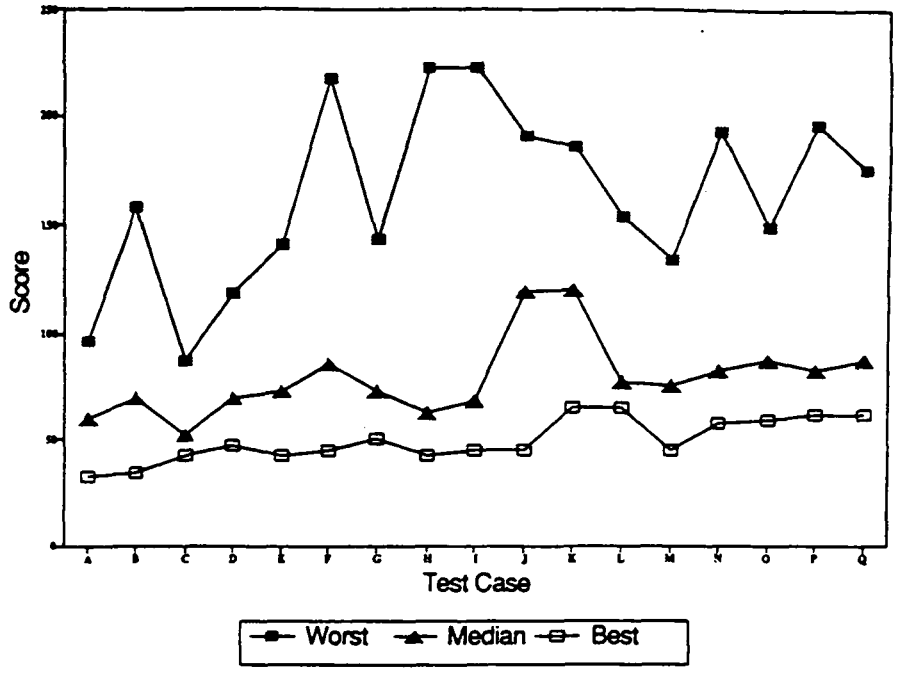


Figure 8: Best, Median, and Worst Scores

	Total Ball Errors			
SUBJECT	EASY	MED	HARD	ALL
1	8.0	4.0	18.0	30.0
2	0.0	0.0	2.0	2.0
3	0.0	0.0	9.0	9.0
4	0.0	0.0	4.0	4.0
5	2.0	1.0	9.0	12.0
6	2.0	2.0	23.0	27.0
7	0.0	0.0	6.0	6.0
8	0.0	6.0	8.0	14.0
9	0.0	0.0	7.0	7.0
10	0.0	6.0	21.0	27.0
11	0.0	8.0	17.0	25.0
12	0.0	4.0	13.0	17.0
13	2.0	20.0	33.0	55.0
14	8.0	15.0	30.0	53.0
15	10.0	22.0	27.0	59.0
AVG	2.1	5.9	15.1	23.1

	AVG SCORE			
SUBJECT	EASY	MED	HARD	ALL
1	80.0	72.2	84.7	80.9
2	60.0	66.3	69.3	66.9
3	45.0	56.3	73.5	64.4
4	69.2	74.4	79.1	76.2
5	64.7	80.1	82.7	78.9
6	50.0	65.2	97.8	81.7
7	57.5	62.5	78.2	70.8
8	61.7	107.0	88.1	87.9
9	59.2	62.5	78.6	71.4
10	59.2	80.8	93.3	84.3
11	66.7	93.9	114.7	101.4
12	54.2	76.6	93.5	82.6
13	46.7	137.4	130.4	117.3
14	86.1	118.2	124.4	116.2
15	102.6	125.2	141.1	130.6
AVG	64.2	85.2	95.3	87.4

Table 1: Average Score and Total Errors in Placing Balls

Source of Variation	Sum of Squares	df	Mean Square	F
Between group	33915.97	2.00	16957.99	13.42
Within group	318506.48	252.00	1263.91	
Total	352422.45	254.00		

	Confidence Interval	
Pairwise Comparison	Lower Limit	Upper Limit
Easy-Medium	-38.34	-3.79
Medium-Hard	-23.44	3.32
Easy-Hard	-46.01	-16.24

	Confidence Interval	
Group Comparison	Lower Limit	Upper Limit
Easy-(Medium,Hard)	-40.77	-11.42
Hard-(Easy,Medium)	9.38	31.81

Table 2: Analyses of Variance

	Average Score			
	EASY	MEDIUM	HARD	ALL
PEOPLE	64.2	85.2	95.3	87.4
EXPERT	64.2	73.6	88.9	80.9
	Average Ball Error			
PEOPLE	2.1	5.9	15.1	23.1
EXPERT	0.0	2.0	12.0	14.0

Table 3: Average Score and Ball Errors

the easy and hard groups, but not the medium and hard groups. The factors we used to place the different test cases into the groups are valid. However, the difference between the medium and hard groups is not confirmed.

The last set of tests to be performed on our test set are the group comparisons shown in Table 2. The confidence intervals are given for $\mu_{easy} - (\frac{\mu_{medium}+\mu_{hard}}{2})$ and $\mu_{hard} - (\frac{\mu_{easy}+\mu_{medium}}{2})$ using an F value of $F_{2,252,.95} = 3.035$. In both cases, we can reject the null hypothesis that $(\mu_{easy} - (\frac{\mu_{medium}+\mu_{hard}}{2})) = 0$ and $(\mu_{hard} - (\frac{\mu_{easy}+\mu_{medium}}{2})) = 0$. Thus, the easy group is different from the average of the medium and hard groups and the hard group is different from the average of the easy and medium groups.

The performance of the Blackbox Expert on the test cases is shown in Figure 9. The Blackbox Expert is compared to that of the people who were rated as the best, median, and worst players. Except for one test case (I), the best player performed better than the Blackbox Expert. The Blackbox Expert performed better than the worst player in 15 of the 17 test cases. The Blackbox Expert performed better than the median player in 7 of the 17 test cases. The two test cases (C and P) where the Blackbox Expert performed poorly compared to the worst player indicate a deficiency in the knowledge base. Both test cases C and P have balls located near the corners of the Blackbox grid. The knowledge base of the Blackbox Expert is lacking rules to determine when balls are located near the corners of the Blackbox.

The average score and the total number of errors made placing balls by the humans and the Blackbox Expert are shown in Table 3. The Blackbox Expert on average made fewer errors locating balls than the humans. The lowest total number of errors (2 errors in 17 test cases) was made by a person with several years of experience solving the Blackbox puzzle as seen in Table 1. Also, the Blackbox Expert had a better average score on each group of test cases in the test set except on the easy test cases where it had the same average score. When the total number of errors in locating balls is considered, the Blackbox Expert ranks 7^{th} compared to the people.

The average score obtained by the Blackbox Expert for each group of test cases as well as the entire test set is compared to the average score obtained by the people in Figure 10. The Blackbox Expert ranks 10^{th} on the easy test cases, 7^{th} on the medium and hard test cases, and 7^{th}

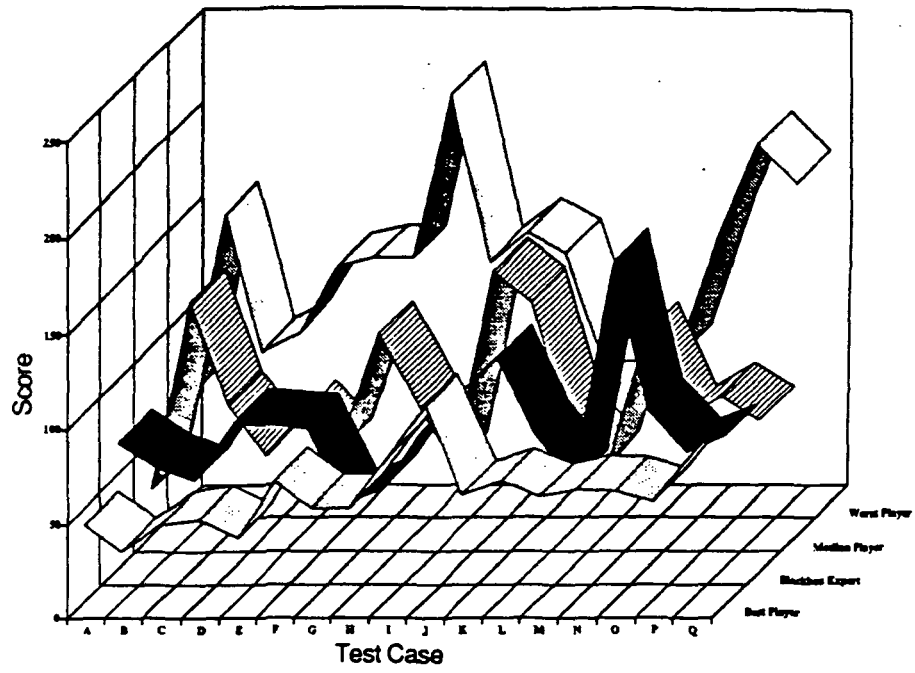


Figure 9: Scores of the Blackbox Expert

on the entire test set. The improvement observed in the Blackbox Expert's ranking on the medium and hard test cases occurs because even though the Blackbox Expert and the humans can find all the balls in the easy test cases, the Blackbox Expert requires more beams to solve the test cases. In the case of the medium and hard test cases, the Blackbox Expert still tends to fire more beams than the humans. However, on average it makes fewer errors which allows it to improve its position.

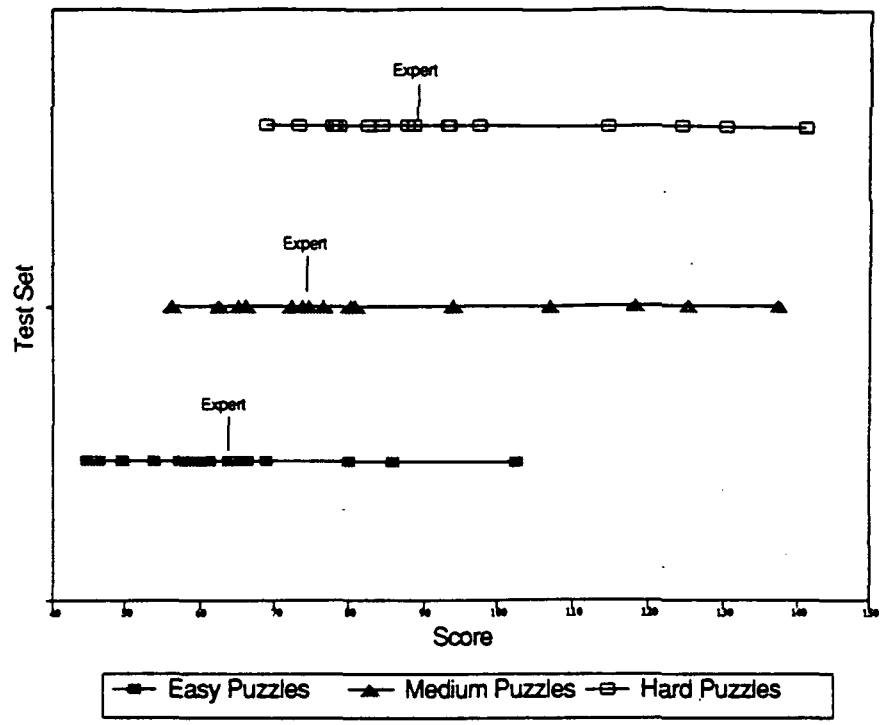
CONCLUSIONS AND FURTHER RESEARCH

In order to validate the expert system described in this paper, we created 17 test cases. Using a set of criteria extracted from human Blackbox experts, the test cases were placed into three groups easy, medium, and hard. Based on a metric defined in this paper and the results of the 15 people solving each of the 17 test cases, the groups in the test set were statistically tested to determine if the mean scores for the test cases in each group were different. It was found that the means of the easy and hard groups are statistically different as are the means of the easy and medium groups. However, the difference between the means of the medium and hard groups is not significant. Further statistical tests have shown that the medium group can be combined either with the hard or with the easy group to form two statistically different groups of test cases.

We have briefly described the structure and design of the Blackbox Expert in this paper. It consists of 300 rules and 8000 lines of source code of which approximately 25% is comments. Its performance is compared with that of the 15 people. This comparison included both the SCORE metric developed in the paper and the total number of errors committed in placing balls. Overall the Blackbox Expert ranks 7^{th} with respect to SCORE and 7^{th} with respect to total errors placing balls.

By analyzing the games in which the people performed better than the Blackbox Expert, we notice that the expert system can be improved in three ways: the rule base can be enhanced to account for the cases where balls are placed in corner squares; the beam selection rules can be improved; and the conflict resolution rules can be improved by studying the ball placement errors committed by the expert system. The additional knowledge required for the improvements to the rule base of the Blackbox Expert can now be obtained from the people who solved the test set. During the next phases of the modified spiral model used for the development of expert systems, this information will be useful.

The Blackbox Expert, the test cases, and the details of the humans solving the test cases



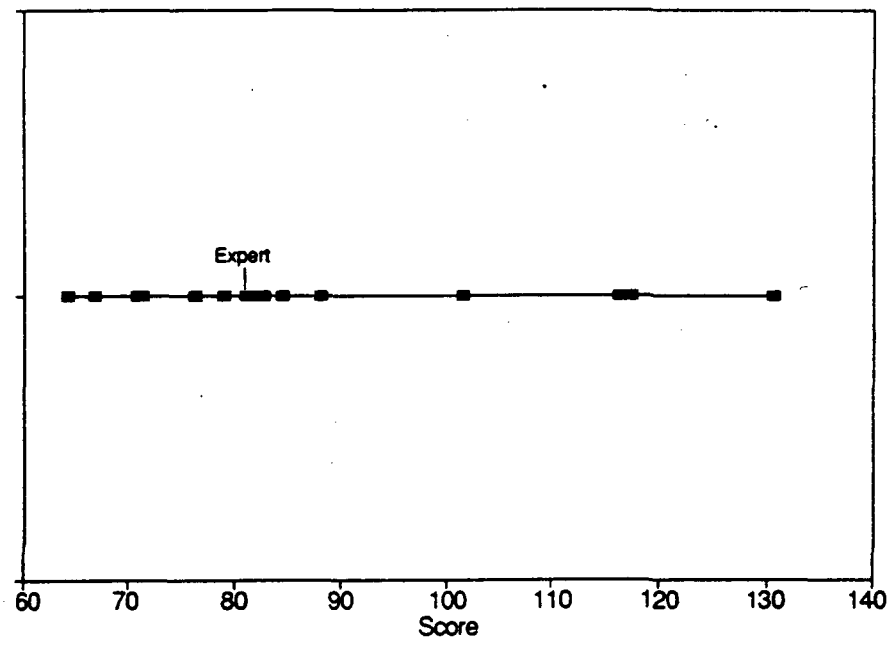


Figure 10: Blackbox Expert versus Humans

will all be useful for several experimental research projects. In particular, we are interested in co-operative problem solving with multiple expert systems. The optimality of different organizations effects of "Information Deficit", and the effectiveness of different planning and communicatior protocols using the blackboard architecture are some of the problems that our group of two faculty members and five graduate students are investigating.

ACKNOWLEDGEMENTS

There are many people who helped us to collect the data for this experiment by solving the test cases of Blackbox. We thank them for their participation. The indispensable help of Le Hoc Duong in the development of the rule base for the Blackbox Expert is gratefully acknowledged. We thank Kristina Pitula who made her Blackbox expertise available to us for developing the test set. We are grateful to Dr. Alun Preece for his valuable suggestions and comments concerning our validation procedure.

REFERENCES

Boehm, B.W. (1979). Software Engineering: R & D trends and defense needs, *Research Directions in Software Technology*, Wegener, P. (Ed), M. I. T. Press, Cambridge Mass.

Boehm, B.W. (1988). A Spiral Model of Software Development and Enhancement. *IEEE Computer*, May, pp. 61-72.

Budd, T. (1991). *Object-Oriented Programming*, Addison Wesley.

Culbert, C. (1989). CLIPS Reference Manual, *Artificial Intelligence Section, Johnson Space Center*, Houston.

Durfee, E.H. and Lesser, V.R. (1986). Incremental Planning to Control a Blackboard Based Problem-Solver, *AAAI-86*, pp. 58-64.

Durfee, E.H., Lesser, V.R. and Corkill D.C. (1987). Cooperation Through Communication in a Distributed Problem Solving Network, *in Distributed Artificial Intelligence*, Morgan Kaufmann.

Gasser, L. (1987) The 1985 Workshop on Distributed Artificial Intelligence, *AI Magazine*, Vol. 8, No. 2.

Ginsberg, M.L. (1987). Decision Procedures, *in Distributed Artificial Intelligence*, Morgan Kaufmann, California.

Grossner, C. and Radhakrishnan (1990). Organizations for Cooperating Expert Systems, *22nd Southeast Symposium on System Theory*, Tenn.

Harrison, R.P., and Ratcliffe, P.A. (1991). Towards Standards for the Validation of Expert Systems, *Expert Systems with Applications*, Vol. 2, No. 4, pp. 251-258.

Hayes-Roth, F., Waterman, D.A. and Lenat, D.B. (Eds.) (1983). *Building Expert Systems*, Addison Wesley.

Jagannathan, V., Dodhiawala, R. and Baum, L.S. (1989). *Blackboard Architectures and Applications*, Academic Press.

Lyons, J. and Grossner, C (1990). A Blackbox Expert System: User Requirements, *Technical Report*, Computer Science Dept., Concordia University.

Lyons, J. and Grossner, C (1990). A Blackbox Expert System: Software Requirements Specification, *Technical Report*, Computer Science Dept., Concordia University.

Lyons, J. and Grossner, C (1990). A Blackbox Expert System: Preliminary Design, *Technical Report*, Computer Science Dept., Concordia University.

Lyons, J. and Grossner, C (1990). A Blackbox Expert System: Detailed Design, *Technical Report*, Computer Science Dept., Concordia University.

Miller, L.A. (1990). Dynamic Testing of Knowledge Bases Using the Heuristic Testing Approach, *Expert Systems with Applications*, Vol. 1, No. 3, pp. 249-270.

O'Keefe, R.M., Balci, O. and Smith, P.E. (1987). Validating Expert System Performance, *IEEE Expert*.

O'Keefe, R.M. and Lee, S. (1990). An Integrative Model of Expert System Verification and Validation, *Expert Systems with Applications*, Vol. 1, No. 3, pp. 231-236.

O'Reilly-Staff (1990). X Toolkit Intrinsics Reference Manual. *O'Reilly and Associates*, Vol. 5.
Parsaye, K. and Chignell, M. (1988). *Expert Systems for Experts*, John Wiley.
Pitula, K., Radhakrishnan, T. and Grossner, C. (1990). Distributed Blackbox: A Test Bed for Distributed Problem Solving, *Int. Phoenix Conf. on Computers and Communications*, Phoenix.
Simon, H. (1973). The Structure of Ill-Structured Problems, *Artificial Intelligence* , Vol. 4, 1973.
Smith, R.G. (1980). The Contract Net Protocol: High Level Communication and Control in a Distributed Problem Solver, *IEEE Transactions on Computers*, Vol. C-29, No. 12, pp. 1104-1113.
Stroustrup, B. (1987). *The C++ Programming Language*, Addison Wesley.