AGASSISTANT: AN ARTIFICIAL INTELLIGENCE SYSTEM FOR DISCOVERING PATTERNS IN AGRICULTURAL KNOWLEDGE AND CREATING DIAGNOSTIC ADVISORY SYSTEMS

by

T. W. Fermanian
R. S. Michalski
B. Katz
J. Kelly

Agronomy Journal, Vol. 81, No. 2, pp. 306-312.

Reprinted from *Agronomy Journal*
Vol. 81, No. 2

AGASSISTANT: AN ARTIFICIAL INTELLIGENCE SYSTEM FOR DISCOVERING PATTERNS IN AGRICULTURAL KNOWLEDGE AND CREATING DIAGNOSTIC ADVISORY SYSTEMS

T. W. FERMANIAN, R. S. MICHALSKI, B. KATZ, AND J. KELLY

AGASSISTANT: AN ARTIFICIAL INTELLIGENCE SYSTEM FOR DISCOVERING PATTERNS IN AGRICULTURAL KNOWLEDGE AND CREATING DIAGNOSTIC ADVISORY SYSTEMS

T. W. Fermanian,* R. S. Michalski, B. Katz, and J. Kelly

Abstract

AGASSISTANT is a non-specialist artificial intelligence system for automatically determining, refining, and evaluating diagnostic rules and patterns for agricultural decision problems. It has the ability to create general decision rules from examples of expert decisions. It can provide advice using these self-created rules or rules supplied by a system creator, and can also serve as a tool for developing expert or advisory systems. AGASSISTANT was first applied to build WEEDER, a system for identifying 37 grass weed or turf species commonly found in turfs in the USA. To evaluate WEEDER's potential for exclusive identifications, a program was developed to produce all possible combinations of variable values which lead to an exclusive identification of any grass represented in the system. For most grasses, there were multiple ways ($\bar{x}$ = 4) of identifying each grass exclusively among all other grasses. Each identification required the selection of a value for an average of five variables. The maximum number of variable value decisions required for an identification was seven and the minimum was four. More than one-half (59%) of the identifications required a decision on five or less variables. WEEDER was found to be as efficient in the number of decisions required for an identification as the theoretical maximum efficiency (5 leads) for a dichotomous key covering the same species. WEEDER represents an improvement over current methods of plant identification (e.g. identification keys) due to its ability to identify specimens through multiple sets of plant variables, use uncertain knowledge, allow the user to answer questions in any order, and be easily modified to reflect local differences in plant populations. AGASSISTANT provided the total environment for the initial representation and organization of the WEEDER knowledge base and serves as its final delivery medium.

AGRICULTURAL scientists often provide advice to agricultural managers based on incomplete knowledge supplemented with their experience. Due to the variability of biological systems, agricultural knowledge is often unstable and may require periodic modification to reflect evolving conditions.

Expert or advisory systems are computer software applications that possess the capacity for reasoning and analysis within a narrowly defined area (Hayes-Roth et al., 1983). (Note: The terms expert or advisory system will be used interchangeably throughout this paper.) One of the major goals in developing such a system is to provide a reasoning tool which approaches the level of proficiency human experts exhibit in that field. Advisory systems, therefore, might provide agricultural scientists with a means for providing their expertise and knowledge to agricultural managers or other scientists, and to update any previously distributed knowledge bases in a consistent and timely manner.

Expert systems techniques were first applied in agriculture to diagnose symptoms of soybean diseases. PLANT/ds (Michalski et al., 1982) is a microcomputer based advisory system developed as an extension of the meta-expert system ADVISE, which may be used to construct new expert systems (Michalski and Baskin, 1983). While PLANT/ds provided the accessibility of a microcomputer-based expert system, a new domain (i.e. area of expertise) cannot directly utilize its inference engine (i.e. general search and evaluate algorithms). In order for a new system to be developed the knowledge base has to be developed and tested on a minicomputer, and then interpreted to the appropriate Pascal code for the PLANT/ds system on an IBM PC (International Business Machines Corporation, Endicott, NY.)

AGASSISTANT was developed to provide a total environment for the initial representation of agricultural knowledge and its organization into a final delivery system. The design of the system was specifically oriented toward users not trained in computer or expert systems technology. Some of the novel features incorporated into AGASSISTANT are multiple means of creating and refining knowledge, including a module for learning by example and the ability to use uncertain knowledge. The system is IBM PC-based and offers menu-driven screens for ease of operation.

Learning from examples is one of the most explored areas in machine learning (Carbonell et al., 1983). In this form of learning, a teacher provides characteristic examples and their representative decision classes to a learner. The task is then to create a set of rules which generalize the given examples. While simple in principle, this method of building descriptions of concepts is often powerful in practice. For example, it has been shown that inductively derived rules for soybean disease diagnosis outperformed expert derived rules (Michalski and Chilausky, 1980).

In order to test and possibly modify the capabilities of AGASSISTANT, a prototype advisory system was designed. WEEDER, an advisory system for the identification of grass weed and turf species found in turf (Fermanian and Michalski, 1988), was built, using AGASSISTANT, to provide assistance to individuals without extensive grass identification knowledge, such as turf managers, county extension advisors, students,

T.W. Fermanian, Dep. of Horticulture, and B. Katz, Dep. of Comp. Sci., Univ of Illinois at Urbana-Champaign, Urbana, IL 61801; R.S. Michalski, Dep. of Comp. Sci., George Mason Univ, Fairfax, VA 22030-4444; J. Kelly, Bolt, Beranek, and Newman, Cambridge, MA 02238. Contribution from the Horticulture Dep., Univ. of Illinois at Urbana-Champaign. This study was supported in part by an Univ. of Illinois Res. Board grant, Int. Intelligent Systems, Inc. and was part of project no. 65-0357 of the Agric. Exp. Stn., College of Agric., Univ. of Illinois at Urbana-Champaign, Urbana, IL 61801. Received 22 April 1988. *Corresponding author.

Published in Agron. J. 81:306–312 (1989).

etc. The domain consists of 37 grass species, each represented by eleven or fewer morphological or growth character-variables. Hereafter, character-variables will be referred to as variables. With the development of this system we wished to evaluate the following objectives; (i) to provide a comprehensive advisory system development environment in which all phases of development were conducted on the delivery hardware system; (ii) for the grass domain specifically, to provide multiple means for identifying an unknown species exclusively from all other species in the knowledge base; and (iii) to provide a consistent mechanism of plant identification which was, theoretically, at least as efficient as a dichotomous identification key.

Materials and Methods

Overview of Program

AGASSISTANT consists of a series of modules accessible through menu-driven screens. AGASSISTANT was written in Turbo Pascal, ver. 3.0 (Borland International, Scotts Valley, CA) under PC-DOS 3.1 (International Business Machines Corporation, Endicott, NY.) Due to the nature of the Turbo Pascal compiler, which restricts program and data code to 64K segments, AGASSISTANT is a chain of twelve files for a total 400 KB of compiled code. Therefore, it requires a hard disk to run on an IBM PC. An additional minimum requirement for its operation is 512 KB of memory. A color monitor is highly recommended. AGASSISTANT runs on an IBM PC, XT and compatible machines under PC-DOS 3.x, and is partially functional on an IBM AT.

Figure 1 is an overview of the components and data modules of AGASSISTANT. The advisor module takes a compiled knowledge base as its input, and presents questions to the user, as well as offers advice on the basis of answers provided by the user. The compiled knowledge base is created by another module, a compiler, which in addition to parsing rules for correct syntax, creates a more compact compiled version of the system for faster execution. Rules may be created by hand through an external editor or created through induction from examples of previous decisions. The inductive engine calls the program NEWGEM (Reinke, 1984) to operate on examples and variable definitions to produce rules. Additionally, this module may start with existing rules and improve them with new examples (incremental learning) or optimize them according to criteria selected by the user. System data can be found in AGASSISTANT as compiled rules, interaction text to provide help to the user, uncompiled rules in a rule base, variable definitions, and tables of examples for producing rules. Knowledge is represented in AGASSISTANT primarily in the form of modified VL1 (variable-valued logic) rules (Michalski, 1974). These rules take the generalized form of:

If condition *then* conclusion

More specifically, if a set of conditions is satisfied, then a particular conclusion is acted upon. A set of conditions or a complex is conjunctively connected (logical AND). A single condition in a complex contains a single variable, which may have multiple values. For example:

Weed is Large Crabgrass if:	confidence level
1. Ligule is toothed or acute,	65
2. Blade width is coarse,	60
3. Sheath is compressed,	50
4. Habit is bunch,	40

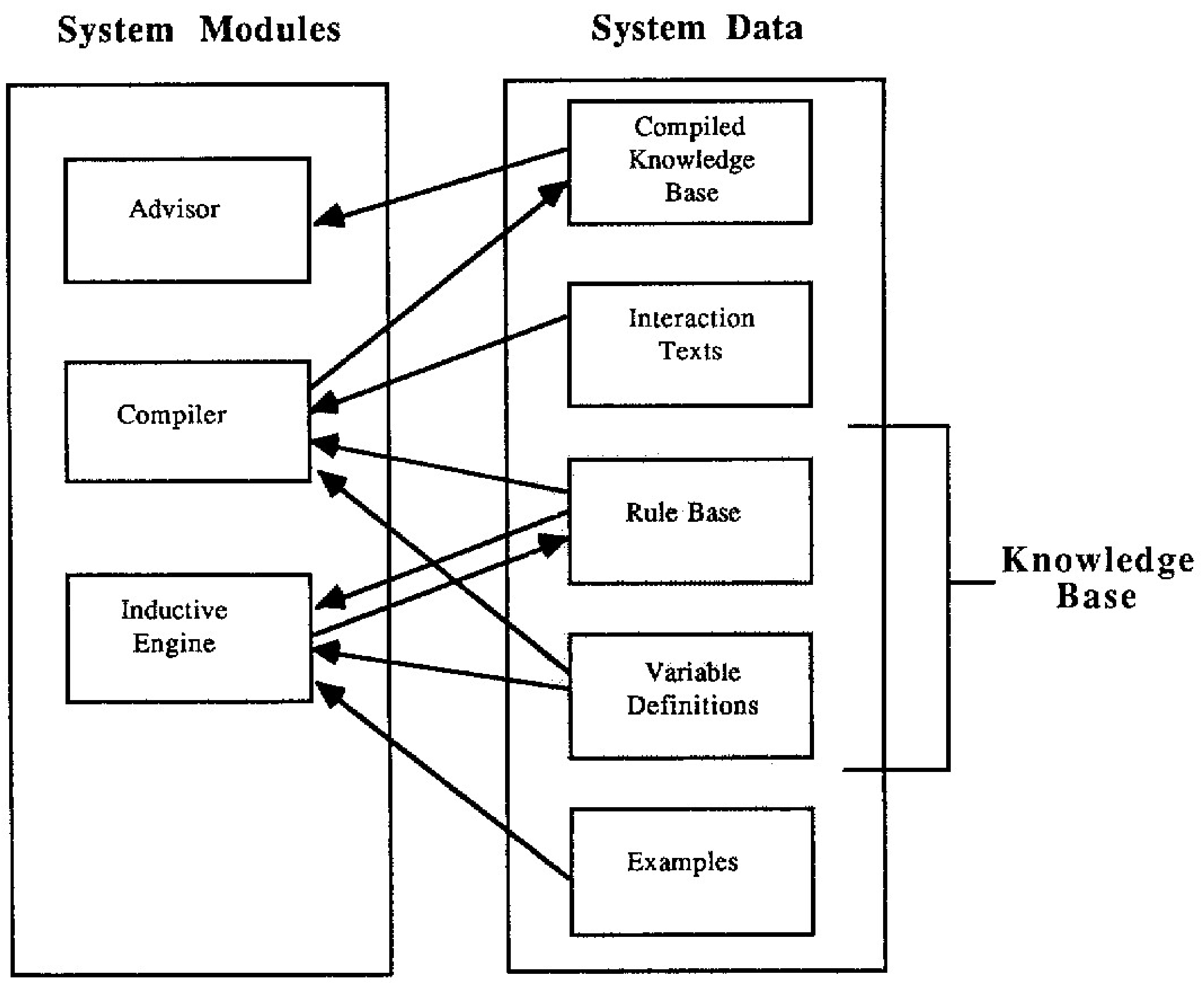


Fig. 1. Components of AGASSISTANT, an expert system development environment.

5. Vernation is rolled,	35
6. Collar is broad,	35
OR	
1. Disarticulation is below,	40
2. Flower is spike,	60
3. Florets is 1,	35
4. Vernation is rolled,	35
5. Glumes are shorter.	20

As shown in the above example, rules can have multiple complexes disjunctively (logical OR) connected. In order for the rule conclusion to be satisfied, not all conditions in a complex of conditions must be satisfied. For each variable and value set condition a value between 0 and 100 is assigned to represent the expert's confidence level (CL) in the importance of that variable-value pair to conclude the entire rule alone without additional evidence. A value of zero represents no support, while a value of 100 represents absolute support. As each condition is satisfied, support is added to the evidence for confirming the conclusion. When a preset threshold of evidence is supported the rule is concluded (fired). Rules with multiple complexes are concluded if one or more complex is satisfied. Rules are initially prepared in an English-like syntax and saved in an ASCII file.

Variables in the system are defined as one of five possible types; (i) nominal variables have a set of unordered values (e.g. *color*: red, green, blue.); (ii) linear variables may be non-numeric, but require an order defined in the variable definition table (e.g. *size*: small, medium, large.) (iii) integer, (iv) numeric variables represent either a range of integers or real numbers, respectively. Numeric variables, however, are divided into eight equal segments for the numeric range stated; and (v) structured variables are unique to AGASSISTANT and allow a hierarchical set of values to be associated with a variable (e.g. *shape*: round [circle, oval] box [square, rectangle].) This structure must be explicitly defined in the variable definition table.

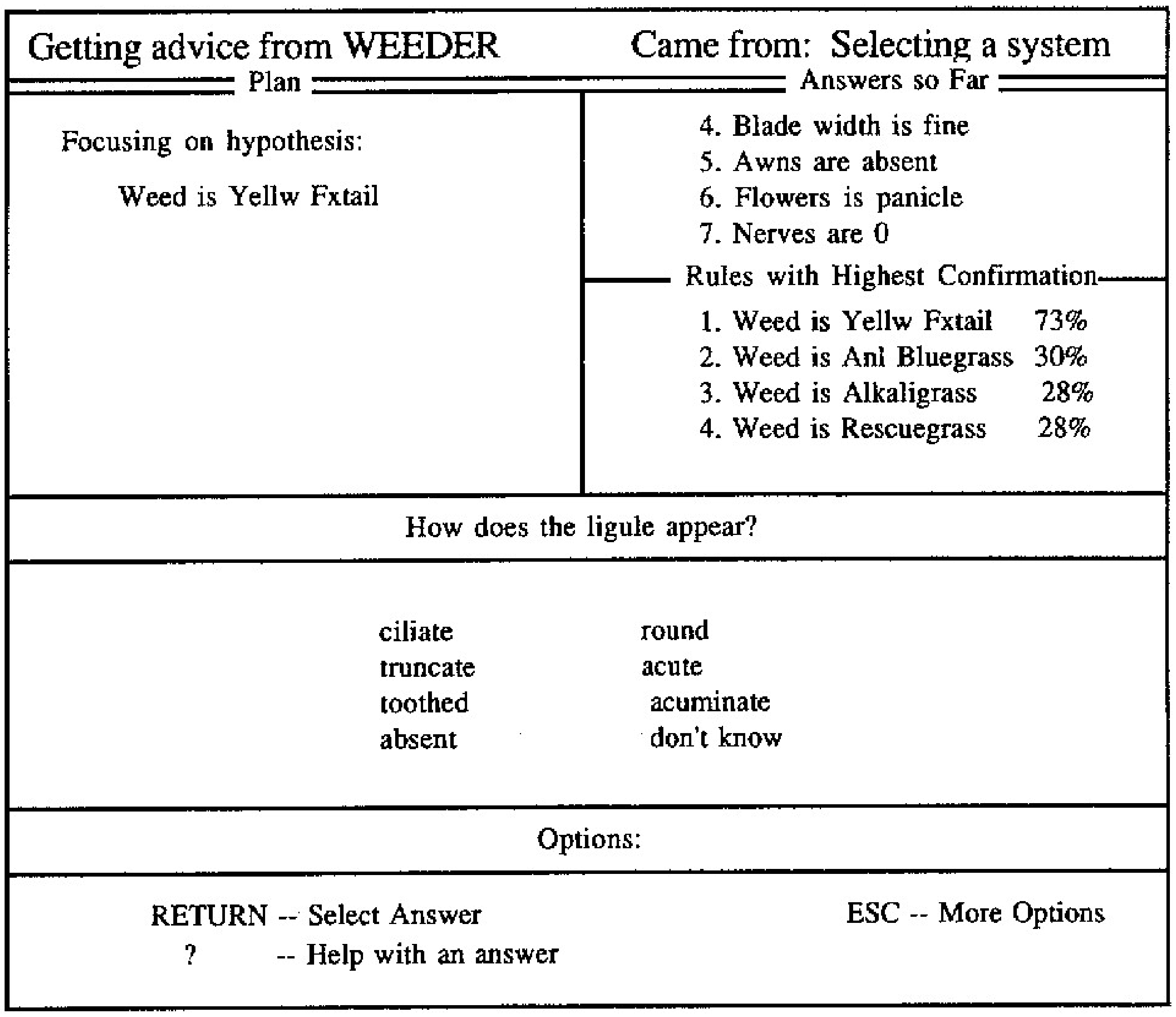
Along with the generation of rules, variable types with their range of values must be declared in the variable definition table. A mechanism for this operation is provided within AGASSISTANT's variable table editor.

Rules are then compiled into the final advisory knowledge base. The compiled knowledge base consists of the following parts; (i) a list of all variables, their values, and relevant information; (ii) a listing of all the rules with integers indicating the ordering of the variables which have the greatest possibility of confirming a rule; (iii) a listing of arithmetic expressions and questions to be used as prompts for the values of any variable. The variables are listed in their relative order of utility, (i.e., their ability to provide discriminating knowledge.) Each variable is listed with its corresponding declared range of values. For each value listed, one or more tuples (i.e. a number set to associate values with elements in a list) containing four elements is provided. The first element is the associated rule number, the second is the complex number within that rule, the third element is the condition number within the complex, and the fourth element of the tuple is the associated CL, with three digits of significance to reduce round-off error (Katz et al., 1987).

Rules are compiled for two reasons; (i) to check for syntactic errors or omissions in the knowledge base (e.g. a variable is used in a rule but not defined); and (ii) the knowledge base is compacted allowing quicker execution of the advisory system.

Advisory Module

User interface. The user interface in the advisory system presents a screen of five separately operating windows (Fig. 2). The "*Plan*" window presents the current focus of the inference plan, which initially is a query regarding the variable with the greatest utility for discriminating between all possible decisions, and finally, on the rule which becomes best supported (highest CL). The "*Answers so Far*" window


Getting advice from WEEDER   Came from: Selecting a system

Plan

Focusing on hypothesis:

Weed is Yellw Fxtail

Answers so Far

4. Blade width is fine
5. Awns are absent
6. Flowers is panicle
7. Nerves are 0

Rules with Highest Confirmation

1. Weed is Yellw Fxtail 73%
2. Weed is Anl Bluegrass 30%
3. Weed is Alkaligrass 28%
4. Weed is Rescuegrass 28%

How does the ligule appear?

ciliate   round
truncate   acute
toothed   acuminate
absent   don't know

Options:

RETURN -- Select Answer   ESC -- More Options
? -- Help with an answer

Fig. 2. Typical computer screen of an advisory session in AGASSISTANT using the WEEDER advisory system.

lists the previous four values selected for prompted variables. Below that window the "*Rules with Highest Confirmation*" window lists the values of confirmed or unconfirmed rules with the highest CL. In the middle window a user is prompted to select a value for a variable. The user can use keyboard arrow keys to move the highlight area to the appropriate value for selection. For each prompted variable a "don't know" value is added for the user to select if they are uncertain of another appropriate value. The bottom, or "*Options*" window can be activated with the Escape key, allowing the user to select additional options.

Inference engine. A CL sum is maintained for each complex (set of conditions) of each rule as associated pieces of evidence are provided through the selection of a value for a prompted variable. This sum is augmented by the CL associated with the satisfied variable-value pair. Variables with internal disjunction (e.g. Ligule is toothed *or* acute) are satisfied if any one of their values is chosen. The general equation for evaluating a complex is:

$$\text{Complex CL} = \Sigma\,(\text{CL of satisfied conditions})/\Sigma\,(\text{all CL's in the complex}) \qquad [1]$$

Evaluation of a rule with multiple complexes is performed by recursively taking the probabilistic sum (PSUM) of the nth complex with the PSUM of the first n−1 complexes. The formula for this is:

$$\text{PSUM}[V(n), V(n-1),\ldots V(1)] = V(n) + \text{PSUM}\,[V(n-1),\ldots V(1) - V(n) \times \text{PSUM}\,(V(n-1))] \qquad [2]$$

where $V(n)$ is the evaluated CL of complex n. The virtue of the PSUM scheme is that it fits well into the intuitive notion that if any complex is completely satisfied the rule will also be, as well as the mathematical notion that the probability of two independent events occurring is the PSUM of each events probability.

A rule is considered to be satisfied if the CL for any one of its complexes goes above a preset threshold. This is currently set at 85 for the WEEDER advisory system (Fermanian and Michalski, 1988), but can be changed. Variables which are in turn actions of another rule receive the final CL value of the other rule multiplied by the CL of the single condition. A hierarchical rule base contains a partial ordering between rules. The rule evaluation module topologically sorts the rules according to this partial ordering before the advisory session begins and uses the resulting order to evaluate all rules after each new value is entered. This ensures that all CLs are propagated in the correct sequence. If all conditions of all rules receive equal CLs, and if the threshold of rule firing is 100%, the inference scheme emulates the standard forward chaining inference engine which has not been directly implemented in AGASSISTANT.

AGASSISTANT has two control mechanisms, that is, schemes which determine the order in which questions are asked. The first method applies to rule bases which are flat, or non-hierarchical, and is known as the utility scheme. The second mechanism for hierarchical rule bases is a backward tracking scheme. At the start of the advisory session questions are asked for variables in order of their utility regardless of the control mechanism.

The utility measure, precalculated and found in the compiled system file, reflects the degree to which the variable will affect the confirmation level of all rules. Those variables appearing in the most places in the rules are given the highest utility. Since knowing the value of these high utility variables will reduce the number of alternate conclusions most effectively, they are the focus of the initial queries to a user. The advisory session continues using the highest utility variables until the confirmation level of any rule goes beyond a minimum CL, which is determined as 100 − threshold CL (e.g. in the WEEDER system, the minimum CL = 100 − 85 or 15). When this occurs, the system will focus on that rule by asking for the values of all variables relevant to the rule. This continues until the rule is either rejected or confirmed, or all variables for that rule are exhausted. At this point, the system will focus on another rule which is above the minimum CL. If none exist it will return to the utility measure. The process continues until all rules are rejected or confirmed. This may require that all variables be queried, but usually occurs much sooner.

The backtracking scheme is automatically invoked if the rule base is hierarchical. The system begins by asking questions for variables with the highest utility and continues in this fashion until the user selects the "don't know" value for a variable, which also appears in the action of some other rule in the system. The process then proceeds recursively to conclude that rule until the system is able to find a value for the original variable it was focusing on. Then the system will continue to query the user until all rules are either rejected or confirmed.

Knowledge Acquisition Facilities

Rules can be acquired in four possible ways. They can be learned from examples, improved with new examples, optimized according to selected criterion, or edited directly. (Fig. 3) These methods can also be used in combination. An example of this might be, editing rules directly and then optimizing them to remove any extraneous information. Direct editing of rules is the simplest way to enter information into the system. The system creator expresses the rules in modified VL1 form in an external editor, then the rule set is saved in an ASCII file. AGASSISTANT can only form rules with a hierarchical structure by direct editing.

Rules can also be learned from examples of previous decisions utilizing AGASSISTANT's learning module. This module includes NEWGEM, an AQ quasi-optimal covering procedure that works by attempting to find a rule which covers all the positive events in a group of examples, while none of the negative events for the same group of variables in other examples. Positive events are those which belong to the decision variable value under consideration, while negative events belong to all other values of the decision variable. AGASSISTANT combines variable definitions found in the variable table, events found in the data table and parameters as set in the parameter table as input to the NEWGEM learning program.

A random event is selected within a set of positive events and extended against successive negative events until it covers none of the negative events. Extending a partial cover against the negative events simply means specializing it so that it no longer covers the negative event, if indeed it did to begin with. The process is continued until a cover or disjunct of covers is produced for all the original positive events.

Parameters for the learning program determine various aspects of the rule creation process, including the breadth of the beam search (i.e. restricted breadth-first search) used by the AQ algorithm, the lexicographic evaluation functional (i.e. a sequence of criterion-tolerance pairs) (Michalski, 1975), which is used in sorting candidate hypotheses, and the complexity of generated rules. Learning parameters are set through a control screen.

Each single condition produced in learned rules is associated with a CL. These CLs are calculated by the following formula and then normalized so that the CL of a given complex equals 100:

$$CL = pe/(pe + ne) \quad [3]$$

Where pe is the number of positive events covered and ne is the number of negative events covered by the variable. Thus, the CL produced represents the degree of evidence that the given value supports the variable in that class of examples.

Example of an AGASSISTANT Application

To test the abilities of AGASSISTANT to produce functional advisory systems, WEEDER, an advisory system to identify weed and turfgrass species commonly found in turf, was developed. The knowledge base of WEEDER represents 37 grass species. Each grass is identified in a single rule consisting of less than 11 identifying variables. The variables selected were those thought to be most easily recognized in the field without supportive equipment (Shurtleff et al., 1987).

A feature of WEEDER is its ability to first subdivide the knowledge base with an initial question. This was necessary because floral variables used in identification are often not available on grasses found in turf due to frequent mowing. "Does not apply" questions were established, which are always asked first to provide a meaningful subset of variables for WEEDER to act on. The question "Are seedheads or flowers present?" to which the user responds "yes", "no", or "don't know" begins each session. If a "don't know" answer is given, then all identifying variables are asked. If "yes" is the answer, then 11 of the possible variables are presented to the user. If "no" is the answer, which is the usual situation for turf, then seven variables are presented; only those pertaining to vegetative portions of the plant.

Gower and Barrett (1971) state that the most efficient determination of an unknown species using an identification key is to use identifying variables which divide potential species into equal binary groups. Therefore, they suggested an equation to represent the minimum theoretical number of decisions necessary when using a dichotomous identification key.

$$\text{Minimum number of decisions} = \log_2 n \quad [4]$$

Where n represents the total number of species considered. If a dichotomous key was constructed to identify the 37 grasses of WEEDER, a minimum of five variable decisions would have to be made for a positive identification. This minimal number of decisions would, therefore, provide the most efficient use of the key. This would require each decision to equally divide the species, which is not possible for a key using the chosen variables for the 37 species in WEEDER. In practice most identification keys do not provide this optimum efficiency (Pankhurst, 1978).

In order to evaluate the efficiency of WEEDER, it was necessary to develop a program to determine the minimum number of variables required to identify each species. This program determined which groups of variables would provide a CL of threshold or greater value for a chosen rule. This program was run external to AGASSISTANT and was used to determine the maximum set of variable combinations for each rule in WEEDER.

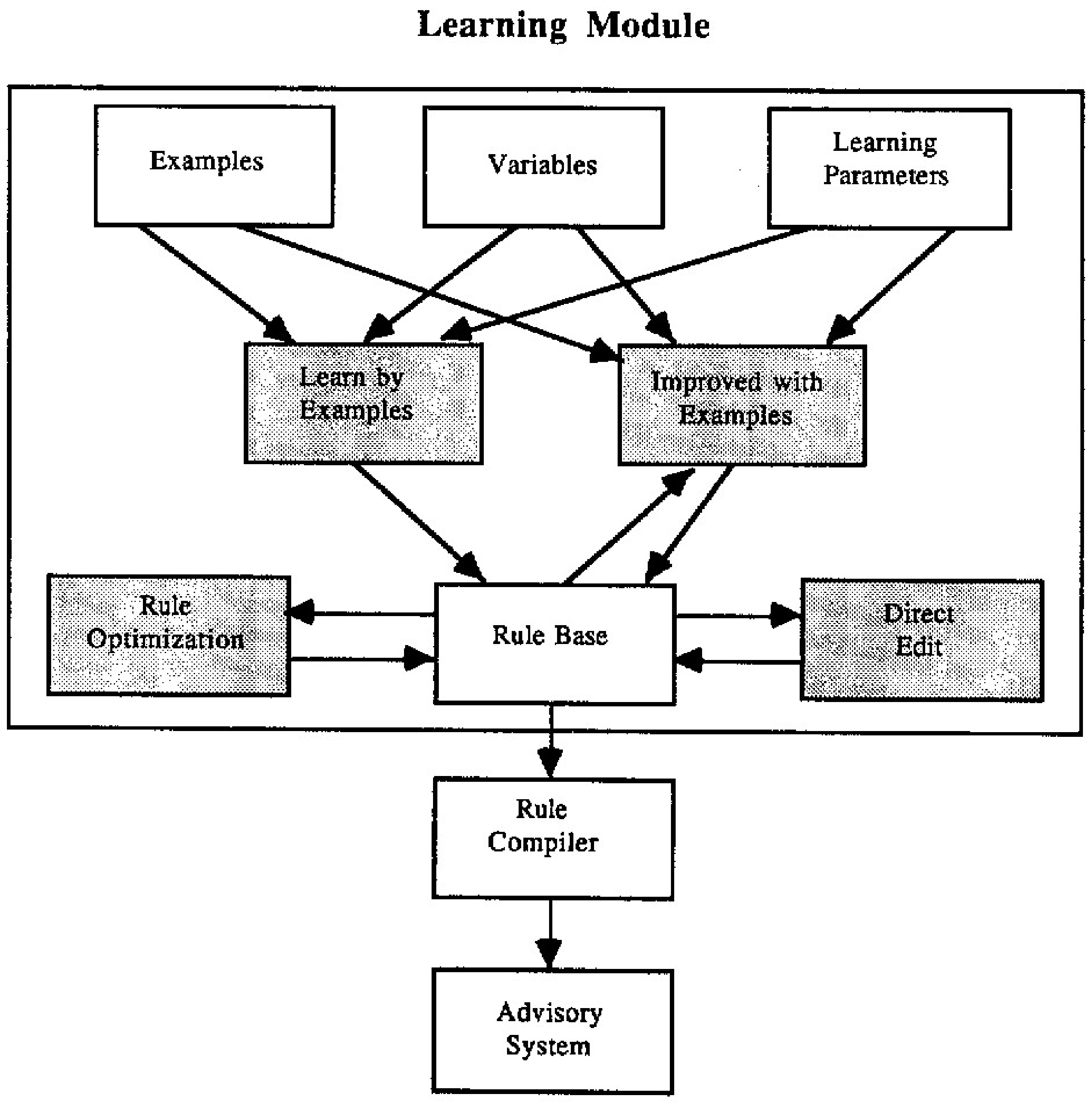


Fig. 3. Relationships between data and components of the learning module of AGASSISTANT. Shaded components represent the four methods of creating rules.

Results and Discussion

Only the subset of variables describing vegetative characters was used in the evaluation of WEEDER. For the 37 grasses there was a maximum of 16 sets of variables providing for the correct identification of any one grass. For four species only a single set of variables provided its identification. There were a total of 145 different sets of variables which represented a correct identification of any species with a mean of four sets for each species. The average number of variables necessary to correctly identify each grass species and its accompanying mean CL is listed in Table 1. An identification was made when the CL was 85 or greater. The mean CL for all identifications was 92.

With a mean number of five variables required for each identification, WEEDER's efficiency was similar to the theoretical maximum efficiency for a dichotomous key to identify the same species. A dichotomous key with this level of efficiency has not been constructed using the same set of variables and values for the species in WEEDER. The maximum number of variables required for a correct identification of any species was seven. Fig. 4 shows the distribution of the number of variables necessary for the 145 sets of variables which represent a correct identification in WEEDER. Over one-half (59%) of the identifications

Table 1. Mean number of variables and average CL required to identify any one of 37 grass species using WEEDER.†

	No. of variables‡	CL
Mean	5§	92
Minimum	4	88
Maximum	6	99
SE¶	0.1	0.5
CV(%)#	11	3

† For each unknown specimen an identification was considered correct when the CL was ≥ 85.
‡ Mean number of variables necessary for the identification of each species.
§ Mean of 37 species means.
¶ Standard error of mean.
Coefficient of variation.

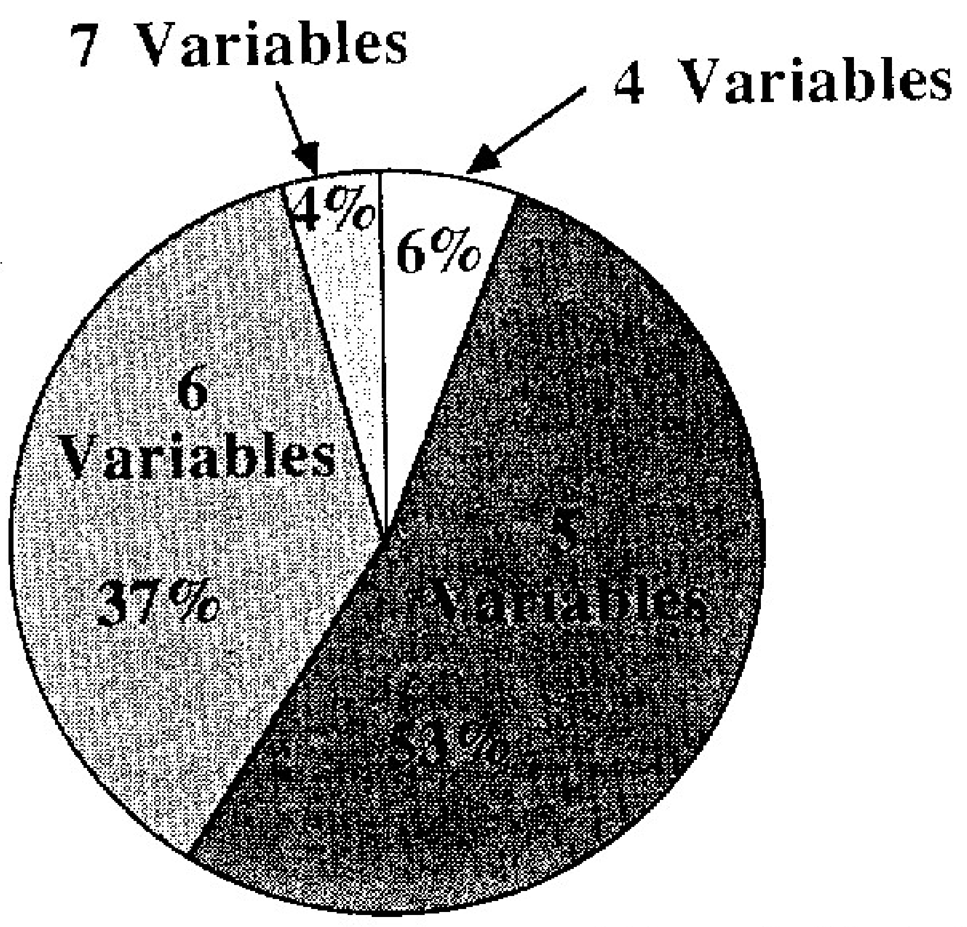


Fig. 4. Distribution of the number of variables required for the 145 sets of variables which represent a correct identification of a grass species in the WEEDER advisory system. Each pie section represents the percentage of the 145 possible correct identifications.

required five or fewer variables. It is evident that for most species (96%) the recognition of a maximum of six variables were required for its identification. Since dichotomous keys generally do not perform at the theoretical maximum efficiency for identifying a species, WEEDER shows excellent potential as a grass identification tool.

The WEEDER advisory system provided an excellent exemplary system and helped to test some of the proposed capabilities of AGASSISTANT. Specifically, WEEDER was able to provide a means for the satisfactory identification of 37 grass species commonly found in turf through a minimal number of decisions. In most cases, multiple sets of variables could be used to identify a single species. A more rigorous evaluation of WEEDER is presented in Fermanian and Michalski (1988) where it was used to identify unknown grasses.

It should be restated that, throughout the duration of this study, the only computing environment used for the construction, evaluation, and refinement of WEEDER was the IBM-PC, using AGASSISTANT. AGASSISTANT provided all the necessary capabilities for evaluating knowledge, inducing properly represented rules through examples of classes of variables, and providing advice for their identification. Although the focus of this paper is not on the implementation details of AGASSISTANT, the system includes a number of important features, including explanation and help facilities. It was designed as a user-friendly system for non-specialists who are either computer proficient or non-proficient. Further implementation details, a User's Guide, and the AGASSISTANT software may be obtained from the authors for a minimal cost ($30) to cover handling and shipping. Send orders to: Dr. T. W. Fermanian, Univ of Illinois, 1201 S. Dorner Drive, Urbana, IL 61801, USA. At present, several of the planned capabilities of AGASSISTANT have not been completed or fully evaluated, however, the program is fully capable of operating on domains of equal size and complexity as WEEDER.

In order to truly evaluate the performance and capabilities of AGASSISTANT it will be necessary to design and implement an advisory system which is hierarchical in structure. In addition, the size of the knowledge base must be expanded to allow for the inclusion of additional rules and/or variables. Current work is proceeding in both of these directions as well as in preparing WEEDER for additional field evaluation.

References

Carbonell, J.G., R.S. Michalski, and T.M. Mitchell. 1983. An overview of machine learning. *In* Michalski, R.S., J.G. Carbonell, and T.M. Mitchell (ed.). Machine learning an artificial intelligence approach. Tioga Publishing Co. Palo Alto, CA.

Fermanian, T.W., and R.S. Michalski. 1988. WEEDER: An advisory system for the identification of grasses in turf. Agron. J.81:312–316.

Gower, J.C., and J.A. Barrett. 1971. Selecting tests in diagnostic keys with unknown responses. Nature. 232:491–93.

Hayes-Roth, F., D.A. Waterman, and D.B. Lenat. 1983. Building expert systems. Addison-Wesley Publishing Co., Reading, MA.

Katz, B., T.W. Fermanian, and R.S. Michalski. 1987. AGASSISTANT: An experimental expert system builder for agricultural applications ISG 87–16, UIUCDCS–F–87–978, Dep. of Comp. Sci., Univ. of Illinois, Urbana, IL.

Michalski, R.S. 1974. VARIABLE-VALUED LOGIC: system VL1. p. 323–346. *In* Proc. of the 1974 symposium on multi-valued

logic, Morgantown, WV, 29–31 May 1974 West Virginia Univ., Morgantown, WV.
———. 1975. Synthesis of optimal and quasi-optimal variable-valued logic formulas. p. 76–87. *In* Proc. of the 1975 International Symposium on Multiple-Valued Logic. Bloomington, IN, 13–16 May 1975. IEEE Computer Society, Long Beach, CA.
———, and A.B. Baskin. 1983. Integrating multiple knowledge representations and learning capabilities in an expert system: the ADVISE system. p. 256–258. *In* Proc. of the 8th IJACAI, Karlruhe, W. Germany, 8–12 Aug. 1983. Morgan Kaufmann Publishers, Inc., Palo Alto, CA.
———, and R.L. Chilausky. 1980. Learning by being told and learning from examples: an experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. Int. J. of Policy Anal. and Inform. System. 4:125–160.
———, J.H. Davis, V.S. Bisht, and J.B. Sinclair. 1982. PLANT/DS: an expert consulting system for the diagnosis of soybean diseases. p. 133–138. *In* Proc. of the 1982 European conference on artificial intelligence. Orsay, France, 12–14 July 1982.
Pankhurst, R.J. 1978. Biological identification Univ Park Press, Baltimore, MD.
Reinke, R.E. 1984. Knowledge acquisition and refinement tools for the ADVISE meta-expert system. M.S. Thesis ISG 84-4, UIUCDCS-F-84-921, Dep of Comp Sci, Univ. of Illinois. July, 1984.
Shurtleff, M.C., T.W. Fermanian, and R. Randell. 1987. Controlling turfgrass pests. Prentice-Hall, Inc. Englewood Cliffs, NJ.