anised Thinning with a Waratah Grapple Harvester and Timberjack Forwarder. Logging Industry Research Organisation Report, Vol. 17 No. 15 1992.
Evanson, T. 1995. A Survey of the Quality of Mechanised Log-making in New Zealand. Logging Industry Research Organisation Report, Vol. 20 No. 6 1995.
Gadd, J. and T. Sowerby. 1995. The Waratah 240 HTH – Debarking and Logmaking Tree-Length *Eucalyptus regnans*. Logging Industry Research Organisation Report, Vol. 20 No. 1 1995.
Gleason, A.P. 1982. Mechanisation of Felling and Delimbing in New Zealand. In Logging Machinery Seminar Proceedings. Logging Industry Research Association. 1982.
Gordon, R. 1986. Early attempts at Mechanised Felling/Delimbing in New Zealand 'Ground Based Logging' seminar Proceedings. Logging Industry Research Association. 1986.
Hall, P. 1994. Timberline ST3530 Stroke Delimber: Kaingaroa. Logging Industry Research Organisation Unpublished Report, P1.1 1994.
Hall, P. 1994. Waste Wood at Logging Landings. Logging Industry Research Organisation Report, Vol. 19 No. 15 1994.
Hill, S. 1995. Trinder Static Delimber. Logging Industry Research Organisation Report, in preparation. 1995.
Jones, G. and T. Evanson. 1992. The Bell Static Delimber in a Cable Clearfell Operation. Logging Industry Research Organisation Report, Vol. 17 No. 21 1992.
McConchie, M. 1994. Timbco T445 Feller Buncher. Logging Industry Research Organisation Unpublished Report, P1.1 1994.
McConchie, M. 1994. Trinder Static Delimber. Logging Industry Research Organisation Unpublished Report, P1.1 1994.
McConchie, M., and T. Evanson. 1995. An Evaluation of a Waratah HTH 234 Felling and Tree-Length Delimbing in Radiata Pine Clearfell. Logging Industry Research Organisation Report. Vol. 20 No. 2 1995.
McConchie, M. and T. Evanson. 1995. Delimbing with a PC400 pulling Full Trees through a Large Static Delimber. Logging Industry Research Organisation Unpublished Report, P1.7 1995.
McMahon, S., and T. Evanson. 1994. The Effect of Slash Cover in Reducing Soil Compaction Resulting from Vehicle Passage. Logging Industry Research Organisation Report. Vol. 19 No. 1 1994.
Parker, R. 1994. Analysis of Lost Time Accidents – 1993 Logging (Accident Reporting Scheme Statistics). Logging Industry Research Organisation Report, Vol. 19 No. 9 1994.
Raymond, O. 1986. The Success of Mechanised Pine Felling and Delimbing. 'Ground Based Logging' seminar Proceedings. Logging Industry Research Association. 1986.
Raymond, K., and T. Evanson. 1994. Clearfelling of Radiata Pine with the Hultdens F850 Feller-Director. Logging Industry Research Organisation Unpublished Report, P1.7 1994.
Riddle, A. 1994. Business Management for Logging. Logging Industry Research Organisation 1994.
Robinson, D., and T. Evanson. 1994. A Mechanised Swing Yarder Operation in New Zealand. Logging Industry Research Organisation Report, Vol. 17 No. 22 1992.

Decision-support systems – new management tools

Euan G. Mason*

ABSTRACT

Decision-support systems are integrated software packages comprising tools for processing both numerical and qualitative information (ideas). They are likely to be essential in future for forestry companies, as they facilitate consistent, thorough, and rapid decision-making. They differ from traditional forestry software packages in that they can process ideas, keep track of uncertainty, have higher levels of integration among software tools, and are easy to use. The development of programming tools for processing knowledge is reviewed. Steps in the development of decision-support systems are described, with emphasis on those required for knowledge-based software. Two decision-support systems designed for forestry are outlined.

INTRODUCTION

'Decision-support system' (DSS) is one of those buzz phrases that arise from time to time, and one might suppose that a DSS would accompany 'upskilling', 'outcome', 'competitive environment', 'level playing field' and 'separation of providers and funders', and have 'upmarket' status. However, unlike most of those other novelties, decision-support systems offer managers a useful new set of tools for decision-making and therefore have a future.

This paper sets out to explain what DSSs are, what is new about them, how they are made, and to indicate some of their uses in forestry.

DECISIONS, DECISIONS ...

Those faced with decisions require ways of predicting results of alternative courses of action, as well as ways of judging the results. There is often plenty of information available, but turning this information into relevant knowledge by selecting the right facts is crucial. Moreover, simply selecting appropriate information is not enough; it often requires processing in order to be useful.

Forest managers are well acquainted with computer programs which select information and produce knowledge in numerical form. This knowledge may be simply attributes, such as volumes of wood per hectare, or it may be active knowledge, such as a model of changes in characteristics of a stand given alternative tending regimes.

Some of the less commonly recognised knowledge used for decision-making is qualitative (non-numerical). As with numerical knowledge, qualitative knowledge, here referred to as ideas[1], may simply comprise descriptions of attributes or be active in the form of logical constructions. The most important knowledge of all, that which includes *how to make the decision*, is almost always qualitative.

It has been suggested that coding ideas into a computer program is a poor substitute for numerical coding, as science could eventually reduce all decisions to numbers (Oscar Garcia pers. comm.). Given unlimited research finance, plenty of lead-time for decisions, and vast computing power, this might be plausible. In the real world, however, budgets are limited, decisions are needed now, and computers have speed limitations. Some knowledge may never be represented in numerical form. Consider such simple examples as the frost-flat regime for establishing radiata pine, or the idea that woody weeds should be controlled prior to the establishment of a tree crop if at all possible. These two heuristics are essential components of decision-making about plantation estab-

* *School of Forestry, University of Canterbury.*
[1] *It is recognised that ideas may be numerical, but for the purposes of this paper, the word "ideas" is used to refer to qualitative knowledge only.*

lishment, and while they could conceivably be represented numerically for a single site, they are unlikely to be generalised numerically. Representing them in logical constructions within computer programs requires techniques quite different from those used for numerical computation.

Rarely can decision-makers claim to have enough knowledge to predict the future perfectly, so there is always some uncertainty associated with a decision. Uncertainty may arise from measurement error, inadequate information, and/or a lack of understanding of the system being modelled. This latter cause of uncertainty is especially prevalent in forestry. For example, who would claim to know how improved breeds of radiata pine are managing to grow faster than those left behind in the bulk seedlot? Without such knowledge, extrapolations of growth trends of improved breeds are fraught with uncertainty. Identifying and keeping track of the uncertainty associated with a decision can be vital for an assessment of risk, and may also identify new topics for research.

Decision-support systems should help people generate both quantitative and qualitative knowledge relevant to a decision, and should provide an indication of the uncertainty associated with the knowledge they generate.

WHAT'S NEW ABOUT DECISION-SUPPORT SYSTEMS?

Foresters are already familiar with many of the likely components of decision-support systems used in forestry. Databases, geographic information systems, statistical tools, financial analysis packages, growth and yield models, and operations research techniques are commonly part of a forester's arsenal of decision tools. All of these techniques use mainly numerical computation, and are readily represented within computer programs.

An ability to process ideas is one of the new features included in a DSS. Knowledge generated from this process may be entirely new, and uniquely generated for a given decision by relatively sophisticated logical processing. It might include such important things as how the decision should be made, what knowledge is relevant, what techniques are required, and suggestions as to what might be decided. The final decision should be made jointly by the software and the user, and should be the responsibility of the user.

A second new feature included in many DSSs is a representation of the uncertainty of knowledge generated by logical processing. This may be statistical uncertainty, but more often it is an expression of subjective probability generated by the computer during inferencing (computer reasoning).

Some decision-support systems can explain why a question is being asked or why a particular piece of advice has been offered. This is known as transparency, and it is another feature not usually found in applications using mainly numerical techniques.

Integrating diverse sources of information and ways of processing information is essential for effective decision-making. A typical DSS might include several different kinds of models, databases, knowledge-based tools, and programs for numerical optimisation. Techniques for processing ideas can facilitate integration and provide a user-friendly environment with knowledge that is easily accessible. An integrated decision environment can promote consistent, thorough and rapid decision-making.

The term 'decision-support system' has sometimes been applied to programs containing wholly numerical methods (e.g. Davis & Martell, 1993). Programs using numerical computation certainly help with decision-making, but they are unlikely to represent all of the information necessary for a decision, nor can they usually supply explanations. In addition, they often lack representations of uncertainty, and linkages between components are likely to be relatively trivial.

DSSs need to process both numbers and ideas. Foresters are familiar with many commonly used numerical methods. They are likely to be unfamiliar with techniques used for processing ideas, however, and these are described in the next section.

KNOWLEDGE-BASED PROGRAMMING

Knowledge-based programming is a set of computer programming techniques which enables machines to represent and process ideas in a logical way. The techniques have been borrowed from research into artificial intelligence. Saarenmaa (1989) comprehensively outlined these techniques within a forestry context, and Mason (1991) summarised opportunities for them within New Zealand forestry.

Development of artificial intelligence techniques

'Artificial intelligence' (AI) is now commonly used to describe a wide range of computer applications, including machine learning, robotics, vision, natural language processing, logic, planning/scheduling, and knowledge-based systems (Saarenmaa, 1989). Much of the important theoretical development of AI has been conducted by researchers who attended a meeting in Dartmouth College during the summer of 1956, their students, or their students' students (McCorduck, 1979, Stock, 1987). Four stages of AI development were identified by Forsyth (1984):

(i) neural nets during the 1950s;
(ii) generalised heuristic search during the 1960s;
(iii) knowledge-representation during the 1970s;
(iv) machine learning during the 1980s.

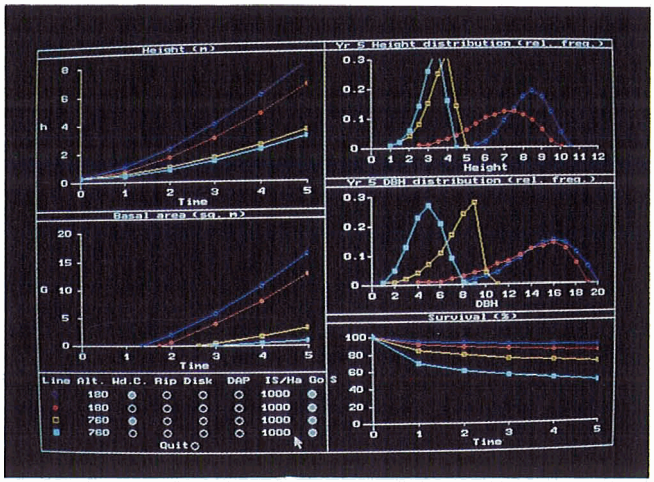
Neural nets during the 1950s. Initial attempts to create intelligent machines involved building machine or program architectures which duplicated structures found in human brains. They consist of massively interconnected structures known as neurodes, which output a weighted sum of incoming signals. Knowledge is contained in the weights between connections, and these can be taught to the net by a learning scheme known as the Delta rule. At the time, success was limited with this approach, and it was later shown that existing algorithms for 'training' the networks were extremely limited from a theoretical point of view (Minsky & Papert, 1969).

Heuristic search. Following the limited success of early neural nets, it was proposed that intelligence might be built from object-manipulating abilities, such as pattern matching, searching, and modifying semantic structures. The 'General Problem Solver' (GPS) program was built with these capabilities, founded on a 'depth-first-search' approach, where a problem is successively decomposed until a point is reached where solutions can be found (Newell & Simon, 1963). Its success was limited to small domains such as puzzles where rules were well defined and numbers of problem states were very limited (Forsyth, 1984).

Knowledge-representation. 'Expert systems', as they have been labelled in the popular literature, were founded on the idea that there was an inverse relationship between the generality (breadth of problems addressable) of an AI program and its sucess. It was thought that if the heuristics (rules of thumb) and relations employed by an expert in a very narrow domain (problem area addressed by the program) could be represented in a program, then a successful machine expert would result. The first such program was DENDRAL, which successfully acted as an expert in certain aspects of organic chemistry (Feigenbaum *et al.*, 1971).

One of the most influential expert systems was MYCIN, developed by Shortliffe (1976) to diagnose and suggest treatments for blood diseases caused by bacteria. The system represented knowledge in the form of a large number of if/then rules, with conditions for higher level rules satisfied by lower level rules.

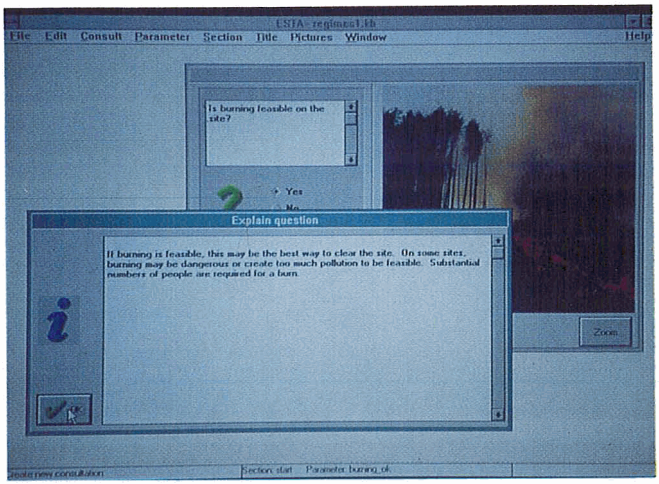
Establishment practices can make a big difference on some sites. Trees on the right are growing in cultivated soil at Cape Karikari, while soil on the left has not been cultivated. The small trees in the foreground are the same age as those in the background, but they have not received any rock phosphate.



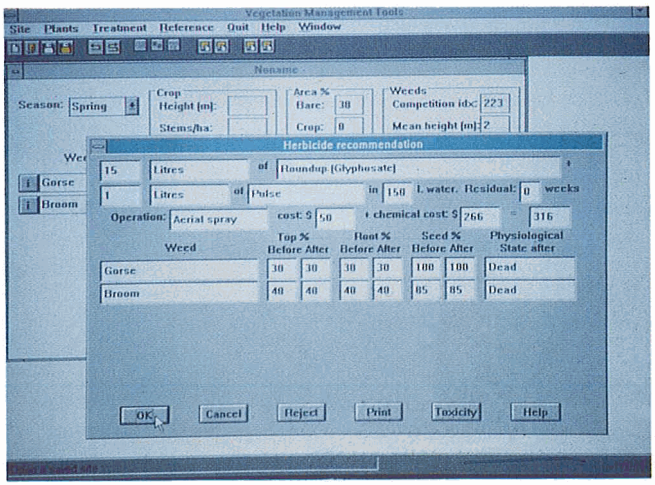
The initial growth model software allows users to compare the effects of alternative establishment treatments for radiata pine in the Central North Island. The control box at lower left is used to specify altitude, weed control, ripping, mounding, fertilisation with diammonium phosphate, and initial stocking. The expected height growth, basal area growth, size distributions at age five, and survival are then displayed in the other windows.

Important features of MYCIN were the inclusion of 'certainty factors' to represent the probability nature of inferred knowledge, the ability to explain reasoning in response to user queries, and a sophisticated, friendly user interface.

Well-constructed expert systems have the potential to provide a permanent source of accurate, easily accessible, and reproducible advice on specific topics. The phrase 'expert sytem', however, tends to promise more human-like behaviour than most programs currently deliver, and can divert attention from the ways in which computer capability exceeds human mental ability. 'Knowledge-based programming', on the other hand, is a term that conveys the idea of representing and processing ideas; it neither promises too much nor restricts use of a range of pro-



A module of the decision-support system helps people select establishment regimes. On this screen a user has asked why a question is being asked.



When an establishment regime has been selected, the system helps users select specific ways to implement treatments. In this case the user has given a detailed description of vegetation on a site, and the system has recommended a herbicide mixture.

gramming paradigms to those which directly mimic human problem solving methods.

Machine learning. Forsyth (1984) identified machine learning as the main AI topic of the 1980s. The example he quoted was a program called EURISKO (Lenat, 1982), which improved its own body of heuristic rules automatically, on the basis of experience. Automatic 'induction' from tables is a feature of some empty expert system 'shells'. What Forsyth could not have foreseen was the return to neural nets following a theoretical improvement in learning, known as 'backpropagation' (Rumelhart *et al.*, 1986, Caudill & Butler, 1990).

Although some commercial applications using neural nets have been built, their use is currently limited to problems with specific types of inputs and solutions.

The 1980s saw the extension of knowledge-based programming into many commercial enterprises, including forestry (Mason, 1991). Most commercial applications are based on the narrow-domain, knowledge-representation approach of Feigenbaum *et al.* (1971) & Shortliffe

(1976), with increasingly sophisticated mixtures of knowledge-representation techniques. Mixtures of techniques are likely to work best for the representation of knowledge relating to forestry. For example, the establishment decision-support system designed by Mason (1994) has components which use backward chaining, forward chaining, scripts, frames, and numerical techniques.

DEVELOPING KNOWLEDGE-BASED PROGRAMS

Domain selection

Careful identification of the domain in which the program will work is the first step towards a successful development. Stock (1987) identified the following desirable characteristics of domains for knowledge-based programs.

(i) Expertise should be scarce and time-consuming to learn, but the task should take only a few hours or days.
(ii) The problem domain should be narrow, but deep (highly specialised), and there should be a large number of possible solutions.
(iii) The problem solution should require heuristics (rules of thumb), i.e. a set of equations could not be used to arrive at a satisfactory solution.
(iv) Competent experts must be available and willing to help with development.
(v) The problem should be financially important enough to warrant building the system.
(vi) Experts in the area should agree.
(vii) Ample data, test cases, and potential users should be available for testing the system.

An additional set of criteria, provided by Gordon *et al.* (1987), suggests that boring tasks, whether easy or hard, should be assigned to a machine, tasks which are both interesting and easy should be left to humans, while computer assistance should be provided to humans doing interesting but difficult tasks.

Given the current lack of sophisticated computer-learning capabilities, acquiring knowledge about how to solve a problem is usually the bottleneck which restricts development once a domain has been identified (Stock, 1987).

Knowledge acquisition

The purpose of knowledge acquisition is to identify and organise the knowledge needed for decision-making within a chosen domain. Often, those coding knowledge into a computer are not experts in the domain, and a number of different methods have been proposed. The key tasks were summarised by Kidd (1987) as:

(i) obtaining data (usually verbally) from an expert;
(ii) interpreting these verbal data to infer the underlying knowledge and reasoning;
(iii) using the interpretation to build a model or language that describes the expert's performance.

Knowledge acquisition should be completed before programming tools are selected. Poor systems would likely result from attempts to mould 'reality' to fit a particular scheme for knowledge representation.

Elicitation of knowledge generally involves several different types of interaction with experts. The types chosen may vary, depending on the type of knowledge being elicited.

Interviews with experts are by far the most common means of eliciting knowledge. Interviews should be conducted in a structured manner, with continuous audio taping if possible. Conversation theory can be very helpful in identifying interview strategies, and in ensuring that communication is efficient. Transcription of interviewing sessions provides a permanent record of the process, ensures that no information is overlooked, and provides a basis for formulating questions for future interviews.

Verbal protocol is a complementary way of eliciting knowledge. An expert is asked to perform a task while providing a running commentary about the decision-making involved (Kuiper & Kassirer, 1987, Fox *et al.*, 1987).

Emphasis should be placed on capturing an expert's conceptual structure, not just procedural skills (Johnson & Johnson, 1987). To this end, several ways of modelling an expert's psychological profile have been proposed. For example, psychological theory has been applied to identify associations between parts of a problem as viewed by an experienced problem solver (Shaw & Gaines, 1987).

As knowledge is gathered, it should be interpreted. An attempt has been made to structure the task of knowledge acquisition (Breuker & Wielinga, 1987), emphasising the separation of information elicitation from information analysis. A classification was made of 'epistemological primitives', or types of knowlege, which might be expected in any domain. Once the information was classified, mapping into appropriate knowledge-representation schemes was easier.

Knowledge-representation

Several ways of representing qualitative knowledge have been developed, and these are often combined within a program. Some commonly used ways include backward chaining, forward chaining, and frames and scripts.

Backward chaining. Backward chaining is a common representation, ideally suited to diagnostic programs such as MYCIN (Shortliffe, 1976). Given a logical set of 'if/then' rules and facts, the algorithm satisfies a goal through pattern matching beginning with potential answers and attempting to 'prove' they fit the problem at hand (Parsaye & Chignell, 1988).

The author has built a system for diagnosing nutritional deficiencies of radiata pine using backward chaining (Mason, 1995a). Drs Don Mead, of Lincoln University, and Graham Will, formerly head of Soils and Site Productivity group at the Forest Research Institute, provided most of the knowledge represented in this system.

Forward chaining. When many rules but few facts are available, forward chaining may be useful. A forward chaining interpreter makes several passes through a set of rules, identifying new facts during each pass (Parsaye & Chignell, 1988). This strategy is often used when several different conclusions are desired. A forest plantation species-selection, knowledge-based program has been written using forward chaining inference (Rice *et al.*, 1989).

Frames and scripts. Many advanced knowledge-based programs, especially those where a solution must be constructed (as opposed to being selected), employ frame-based knowledge representation, and a script to define the problem-solving procedure (e.g. Mittal *et al.*, 1986). Rule-based representations (using backward or forward chaining) are often combined with frames (Parsaye & Chignell, 1988).

Minsky (1975) developed the concept of frames for representing knowledge. Essentially, a frame is a data structure with 'slots' which can contain either properties (scalars), or procedural knowledge (vectors). The structure of a frame is designed as a way of generalising about something in the outside world. Frame-based systems usually employ a network of frames in a hierarchy, with lower-level frames inheriting information and program code from higher-level frames. This inheritance allows a very efficient representation of knowledge. A script describes what is expected, how expectations should be adapted to reality, what should be done at any stage in design, and in what order the stages should occur.

Representation of uncertainty. A decision situation consists of information plus uncertainty. Uncertainty can arise from measurement error, lack of knowledge about particular variables or relations, and 'fuzziness" of semantic associations. As knowledge-based programs are often aimed at reducing the uncertainty associated with a decision, representing uncertainty is an important, but often

underemphasised, aspect of program development. For dealing with uncertainty, two basic steps have been identified (Parsaye & Chignell, 1988):

(i) determining the uncertainty of a basic set of events;
(ii) compounding the values obtained in (i) to arrive at the uncertainty of compound or complex events.

Several alternative ways of coping with uncertainty have been used. In deciding on a representation scheme for MYCIN, Shortliffe (1976) considered and discarded systems based on Shannon's (1949) information theory and Bayesian probability on the grounds that they were impractical and required some untenable assumptions. 'Certainty factors' devised by Shortliffe (1976) fitted well with backward chaining inference, allowing the incremental combination of subjective estimates of uncertainty surrounding conclusions in a rule with that of users' replies. Both positive and negative certainty factors were calculated, the latter representing measures of disbelief.

To represent uncertainty of semantic associations, such as 'tallness', or degree of membership in a set (e.g. is a 'vanette' a van, a minibus, or a stationwagon?), Zadeh (1965) devised fuzzy set theory and fuzzy logic. In fuzzy set theory, membership of a set is denoted by a number between 0 and 1, implying degrees of membership. Statements in fuzzy logic are associated with a continuum of relative truth values, instead of being absolutely true or false as in Boolean logic.

The type of representation chosen varies with the domain concerned and probably with the prejudices of system developers. There are clearly theoretical problems with most approaches so far devised, and many tools for developing knowledge-based systems allow several alternative systems for managing uncertainty.

Neural nets. Neural nets are software or hardware structures which can learn by example, given particular types of problems (Rumelhart *et al.*, 1986, Caudill & Butler, 1990). Many different types of configurations have been demonstrated, but the most practically useful arrangements appear to be layers of neurodes with information moving from an input layer, through one or more middle layers, to an output layer. 1950 vintage nets were only capable of trivial tasks (Minsky & Papert 1969), but the development of backpropagation, a technique used to adjust the weights of middle layer connections when the net is learning (Rumelhart *et al.*, 1986), makes modern nets extremely powerful.

Problems suitable for neural nets are those where many historical examples are documented, the inputs and outputs can be expressed numerically, and the logic governing the outcome of the decision process cannot be identified by humans currently making the decision. Neural nets have also been capable of pattern and voice recognition.

Software tools for knowledge-based representation. In many respects programming is the easiest task required for the development of a knowledge-based program. Planning the scope of the system, eliciting knowledge from experts and potential users, and designing the system are much more difficult. However, programming techniques used are different from those that forest managers are usually familiar with, so these are outlined below.

Traditional procedural software languages such as FORTRAN, Pascal and Basic could theoretically be used for knowledge-based programs, but developers find that higher-level languages allow more emphasis on and clarity of the logic and knowledge embodied in the system.

Languages which manipulate symbols, such as LISP and PROLOG, are often used for rule-based systems (Stock, 1987). PROLOG is a declarative language; that is, programmers can write the essential logical elements of a problem, and the language will find answers to queries through a built-in backtracking mechanism (Prolog Development Centre, 1990).

Rule-based 'shells' are also available. The first was EMYCIN, a version of MYCIN with all the specific knowledge removed (Stock, 1987). These are often very useful for straight diagnostic problems, but reliance on them can lead developers to constrain knowledge representation to fit shell capability instead of devising representation schemes which properly fit the problems of domains.

Frames can be represented adequately with PROLOG (Jay & Knaus, 1989), but most frame-based systems exploit object-oriented programming languages such as Smalltalk (Digitalk, 1991), and C++ (Borland, 1990). Object-oriented languages are programmed with hierarchical classes. Classes include data and program code specific to themselves, but also inherit data and code from higher-level classes. Emphasis is placed on getting 'objects' to respond to commands, instead of doing operations *on* objects as in traditional languages.

Forest technology includes many non-numerical representations, and knowledge-based programming provides a way of clarifying, permanently recording, and using this knowledge. Whilst many of the techniques described here have been successfully employed in forestry (Mason, 1991), they need to be combined with numerical representations, especially growth and yield modelling and forest estate modelling in order to effectively support decision-makers.

SOME FORESTRY-RELATED DSSs

Very few fully-fledged DSSs have been built for forestry.

One of the best-known DSSs is used for coordinating fire-control resources in Ontario. It was developed at the Petawawa National Forestry Institute of Canada (Kourtz, 1990). The system uses geographic information, weather data, fire reports, models of fire behaviour, and dynamic programming to optimise the despatch of fire-suppression equipment and crews. It can handle up to 50 fires simultaneously, and suggests a response to a fire report within approximately 30 seconds. The core of the system is written in PROLOG, using non-numerical techniques, while dynamic programming is done by a PROLOG-controlled FORTRAN routine. A parallel system has been designed to predict the daily occurrence of man- and lightning-caused fires in the region. This latter program combines large rule bases with fuzzy-set theory. Both programs are now used routinely for fire fighting in parts of Canada.

In New Zealand, a DSS is being built to assist with planning plantation establishment. The research is a collaborative project between the author, the New Zealand Forest Research Institute Ltd, Tasman Forestry Ltd, Carter Holt Harvey Ltd, and the Forestry Corporation of New Zealand Ltd. Funding and expertise in vegetation management have been provided by the companies and the Forest Research Institute, while the design and construction of the system and growth modelling are being undertaken at the School of Forestry, University of Canterbury. The DSS is planned to comprise: 1) models capable of representing the crop and other components of a site over entire rotations; 2) a module which assists with the design of establishment regimes; 3) a module which helps plan specific establishment operations; 4) a module which analyses cash flows; and 5) linkages to other components of a forest management information system, such as compartment records, a geographic information system, and decisions made with estate planning tools (Mason, 1995a, 1995b). Most of the system's components have been built and are available for managers to use, but they are not yet fully integrated. Participation of forest managers in the project has been absolutely essential, helping to ensure that the system is accurate, relevant and usable.

CONCLUSIONS

Decision-support systems contain a mixture of old and new programming techniques, and offer frameworks for rapid, thorough and consistent decision-making in forestry. New features of DSSs include non-numerical information coded in consistent and logical ways, high levels of integration of decision tools, representations of uncertainty, and the rapid assembly of knowledge from large amounts of information. Development of these systems requires the active participation of forest managers.

ACKNOWLEDGEMENTS

Many thanks to Janet Scott for her helpful criticism of the manuscript.

REFERENCES

Borland. 1990. Turbo C++ User Guide, Borland International Inc., 264 p.

Breuker, J., and B. Wielinga. 1987. Use of models in interpreting verbal data. *In* Kidd, A.L., (Ed.), Knowledge acquisition for expert systems – an introductory framework. Plenum Press, New York: 17-44.

Caudill, M., and C. Butler. 1990. Naturally intelligent systems, The MIT Press, Cambridge, Massachusetts, 304 p.

Davis, R.G., and D.L. Martell. 1993. A decision-support system that links short-term silvicultural operating plans with long-term forest-level strategic plans. Canadian Journal of Forestry Science. 23: 1078-1095.

Digitalk. 1991. Smalltalk/V DOS Object-oriented programming system – tutorial and programming handbook, Digitalk Inc. 529 p.

Feigenbaum, E.A.; B.G. Buchanan, and J. Lederberg. 1971. On generality and problem solving: A case study using the DENDRAL program. Machine Intelligence. 6: 165-190.

Forsyth, R. 1984. The expert systems phenomenon. *In:* Forsyth, R. (Ed.), Expert systems – principles and case studies. Chapman & Hall Ltd., London: 3-8.

Fox, J.; C.D. Myers, M.F. Greaves and S. Pegram. 1987. A systematic study of knowledge-based refinement in the diagnosis of leukaemia. *In* Kidd, A.L. (Ed.), Knowledge acquisition for expert systems – an introductory framework. Plenum Press, New York: 73-90.

Gordon, S.E.; R.T. Gill, and T.A. Dingus. 1987. Designing for the user: The role of human factors in expert system development, A1 applications in natural resource management 1(1): 35-46.

Jay, C. and R. Knaus. 1989. Expert's toolbox: Frames in Prolog, A1 expert. 4(3): 1924.

Johnson, L. and N. Johnson. 1987. Knowledge elicitation involving teachback interviewing. *In* Kidd, A.L., (Ed.), Knowledge acquisition for expert systems – an introductory framework. Plenum Press, New York: 91-108.

Kidd, A.L. 1987. Knowledge acquisition – An introductory framework. *In* Kidd, A.L., (Ed.). Knowledge acquisition for expert systems – an introductory framework. Plenum Press, New York: 1-16.

Kourtz, P. 1990. Artificial intelligence: a new tool for management. Canadian journal of forestry science. 20: 428-437.

Kuiper, B. and J.P. Kassirer. 1987. Knowledge acquisition by analysis of verbal protocols. *In* Kidd, A.L., (Ed.), Knowledge acquisition for expert systems – an introductory framework. Plenum Press, New York: 45-71.

Lenat, D. 1982. Eurisko: a program that learns new heuristics and domain concepts. Artificial intelligence. 21.

McCorduck. P. 1979. Machines who think. W.H. Freeman & Company, San Francisco, 375 p.

Mason, E.G. 1991. Expert systems for New Zealand's forest managers. *In* Menzies, M.I., G.E. Parrot and L.J. Whitehouse (Eds.), Proceedings of the IUFRO symposium on "Efficiency of stand establishment operations", September 1989, New Zealand Forest Research Institute Bulletin. 156: 87-93.

Mason, E.G. 1995a. A computerised decision-support system for planning forest establishment operations. *In* Gaskin, R.E., and J.A. Zabkeiwicz, Popular summaries from second international conference on forest vegetation management. Forest Research Institute Bulletin 192: 277-279.

Mason, E.G. 1995b. Planning forest establishment operations with a computerised decision-support system: A case study analysis of decision-making over a full rotation. *In* Gaskin, R.E., and J.A. Zabkeiwicz, Popular summaries from second international conference on forest vegetation management. Forest Research Institute Bulletin. 192: 137-139.

Minsky, M. 1975. A framework for representing knowledge. *In* The psychology of computer vision, Edited by P.H. Winston, McGrawHill, NY: 211-280.

Minsky, M. and S. Papert. 1969, Perceptron; an introduction to computational geometry, MIT Press, Massachusetts.

Mittal, S.: C.L. Dym and M. Morjaria. 1986. PRIDE: An expert system for the design of paper handling systems. IEEE computer special issue on expert systems for engineering applications. 1986.

Newell, A. and H.A. Simon. 1963. GPS, a program that simulates human thought. *In* Feigenbaum, E.A., and J. Feldman (Eds.), Computers and thought, McGraw Hill Book Co. Inc, New York: 279-293.

Parsaye, K., and M. Chignell. 1988. Expert systems for experts, John Wiley & Sons Inc., New York. 462 pp.

Prolog Development Centre. 1990. PDC Prolog V3.2: User's guide, Prolog Development Centre, Copenhagen, Denmark.

Rice, D.A.; S.J. Nix and A.J.R. Gillespie. 1989. A simple rule-based system for species selection in artificial regeneration. A1 applications in natural resource management. 3(1): 38-43.

Rumelhart, D.E.; J.L. McClelland and the PDP Research Group. 1986. Parallel distributed processing: Explorations in the microstructure of cognition. Vol. 1. Foundations, MIT Press/Bradford books, Cambridge, Massachusetts.

Saarenmaa, H.J. 1989. Artificial intelligence concepts, techniques, and applications for forestry. *In* Burkhart, H.E., H.M. Rauscher and K. Johann (Eds.), Proceedings of the IUFRO meeting on artificial intelligence and growth models for forest management decisions, Vienna, Austria: 190-201.

Shannon, C.E. 1949. The mathematical theory of communication, University of Illinois Press, Urbana, Illinois.

Shaw, M.L.G. and B.R. Gaines. 1987. An interactive knowledge elicitation technique using personal construct technology. *In* Kidd, A.L., (Ed.), Knowledge acquisition for expert systems – an introductory framework. Plenum Press, New York: 109-136.

Shortliffe, E.H. 1976. Computer-based medical consultations: MYCIN, Elsvier Computer Science Library, New York. 264 p.

Stock, M. 1987. AI and expert systems: an overview, A1 applications in natural resource management. 1(1): 917.

Zadeh, L.A. 1965. Fuzzy sets. Information and control 8.