Problem Decomposition and the Learning of Skills

Donald Michie[*]

University of Edinburgh
United Kingdom
E-Mail dm@aiai.ed.ac.uk

Abstract. One dimension of "divide and conquer" in problem solving concerns the domain and its subdomains. Humans learn the general structure of a domain while solving particular learning problems in it. Another dimension concerns the solver's goals and subgoals. Finding good decompositions is a major AI tactic both for defusing the combinatorial explosion and for ensuring a transparent end-product. In machine learning, pre-occupation with free-standing performance has led to comparative neglect of this resource, illustrated under the following headings. 1. Automatic manufacture of new attributes from primitives ("constructive induction"). 2. Machine learning within goal-subgoal hierarchies ("structured induction"). 3. Reconstruction of skills from human performance data ("behavioural cloning").

1 Introduction

Artificial Intelligence is the engineering science of concepts. We want machines to accept and use concepts from people. We want them to discover new concepts, and to communicate these back to people. We also want them to knit complex concepts together, and by combining observational data with pre-existing knowledge to build new theories of importance for science and technology.

At the lowest levels of machine-constructed theories, black boxes are acceptable. Above a certain level they are not. Reduction of complexity to transparently communicable theories demands more than simply following a symbolic paradigm. For non-trivial problems it usually requires structuring in some way or another, and in the present state of the art the intrusion (as purists might see it) of user interaction. But first I shall discuss a domain where interactive human brain-power reaps large rewards even from non-decompositional methods.

Concept Learning in Molecular Chemistry. In biomolecular modelling many important problems lie towards the simple end of the complexity scale, yet their solution is still beyond unaided human conjecture. Rewards from computer-aided discovery can be very high, for example in drug design. Following the development of efficient Inductive Logic Programming (ILP) algorithms [1], opportunities have multiplied. A recent co- development by members of the Programming

[*] Present address: 73-860 Shadow Mountain Drive, Unit 2, Palm Desert, CA 92260, USA.

Research Group, Oxford, and of the Imperial Cancer Research Fund, London, yielded significant advances based on (1) the expressive power of first order logic as a rule language, and (2) domain expert interaction in constructing "consensus rules" from induction products.

From existing biomolecular data it was required to extract theories able to suggest new compounds for trial syntheses. Rules must not only predict with good accuracy a specified biological activity but should also make sense to the scientists. The following case is taken from a summary of new applications of machine learning (ML) in structural molecular biology [2] and relates to the inhibition of E. coli dihydrofolate reductase by 2,4,-diamino-5-(substituted-benzyl) pyrimidines. Much previous study had previously been devoted to this problem using statistical and neural-net algorithms.

Here is a rule obtained by an ILP algorithm from a data-set describing 55 drugs for which results of tests for biological activity were available:

```
drug A is better than drug B if
drug A has a substitutent at position 3 with
hydrogen-bond donor = 0 and
p-acceptor = 0
and polarity > 0 and
size < 3 and
drug A has a substituent at position 4 and
drug B has no substituent at position 5.
```

In total 59 rules were found, from which seven consensus rules were formed manually by selecting the most commonly found features. These consensus rules were not only easier to understand than the automatically generated rules but in cross- validation their average predictivity was about 85%. This contrasted with about 67% for the products of stand-alone learners, whether ILP, neural net or regression. In contrast with these other learners ILP generated explicit forms that enabled both (a) formation of consensus rules and (b) a deepening of insight into the problem's stereochemistry. Sternberg and his co-workers summarize this approach to machine learning for scientific problems of this kind:

> There is an interactive cycle between human analysis and machine learning. Initially traditional methods process the data and develop representations that characterize the system and rules describing the relationship between the components of the system. Next machine learning uses these representations to identify new, and hopefully more powerful and incisive, rules. Then human intervention is required for interpretation of the rules and the cycle can be repeated.

Purists will say "Very nice for the practical or even the commercial ends of scientific clients! Meanwhile we in ML have our own science to do in characterising and studying algorithms. For this, as chemists with new compounds, we have to isolate, analyse and test them in pure form, uncontaminated by human interaction."

This analogy makes sense only in the context of a certain phase of organic chemistry. As soon as biochemistry, and later molecular chemistry, started treading on its heels, the need to keep living cells away from the reagents gave place to the need to bring reagents and cells together. So with ML. We should be in the business of designing ML algorithms so that human brains, not Martian, can use the products. How else are we to gather data for further automation? In the biomolecular example, interactive development of the technique of consensus rules necessarily preceded consideration of how to mechanise it.

2 Categories of Problem Decomposition

Decompositional approaches are discussed in the following contexts:

- automatic manufacture of new attributes;
- machine learning within goal-subgoal hierarchies;
- reconstruction of skills from human performance data.

2.1 Attribute manufacture

Experienced statistical data analysts routinely augment the "given" set of independent variables by variously transforming and combining them. ML workers in general neither do this themselves nor incorporate such procedures into their algorithms. Yet attribute manufacture can make tractable to propositional learners problems that otherwise require the full machinery of ILP. SOAR's "chunking" (see [3] for overview) has something of this flavour as an aid to problem solving. My discussion here is of aids to learning. Two examples will be given, namely "KRK illegality" and "Michalski trains".

KRK Illegality Problem. Automatic discovery of the concept "position illegal" in the elementary chess end-game king and rook against king was studied as a bench-mark [4]. With only primitive attributes supplied, ILP outperformed decision-tree and rule learning algorithms. Performance of human learners was also tested. Michie and Bain [5] then re-attacked the problem using a decision-tree learner with an attribute-manufacture pre-process. Without it, learning was dismal as before. With it, highly accurate trees were generated.

We denote the king-rook-king chess endgame as KRK. There are about a quarter of a million ways of placing on the chess-board the three pieces: White king, White rook and Black king. Training sets consisted of examples represented simply as a vector of six integer-attributes, each drawn from the interval (1, 2, ..., 8). The attribute-vectors are interpretable as a sequence of three co-ordinate pairs specifying the locations on the board of, respectively, the White king, the White rook and the Black king. The two decision classes were "illegal" and "legal". The latter (negative examples) outnumbered the former (positive examples) in the ratio of about two to one. What follows is based on Michie and Bain's follow-up of the earlier study by Muggleton and co-workers.

The theory "KRK-illegal", derived from the laws of chess, can be expressed using the "negation as failure" convention, i.e. "legal" is the default class.

```
if the White rook and the Black king
  either occupy the same file and the white king is not directly between
  or occupy the same rank and the white king is not directly between
then illegal;
if the two kings
  either are vertically adjacent
  or are horizontally adjacent
  or are diagonally adjacent
then illegal;
if two pieces are on the same square
then illegal;
otherwise legal.
```

This is one possible formulation of the theory to be discovered, and a fragment of training data might look like this:

```
2 6 7 4 2 3 no
7 6 3 7 3 8 yes
3 6 1 2 6 3 no
4 8 3 2 4 7 yes
8 8 5 4 8 2 no
```

The first six symbols in each line represent the three co-ordinate pairs needed to specify the squares on the board occupied by the three pieces, and the seventh represents the decision class, with "yes" for "KRK-illegal" and "no" for its negation, "KRK-legal".

Faced with raw data in this form humans were unable to extract any meaningful patterns in several hours. But as a side-test (not previously reported) a professional data analyst, Dr Jane Mitchell, from the Strathclyde University Statistics Department was invited to tackle the problem. She was allowed whatever analytical methods or computational tools she cared to employ. After two days' intensive work she announced a complete, correct and concise formulation logically equivalent to the theory given above.

So this seemingly toy-sized task is interesting. It completely defeats untrained humans, but succumbs to a professional analyst using statistical but no ML aids. Moreover, decision-trees induced from training sets of size 700 scored no better than the 67% achievable by the trivial rule that always guesses "legal". The decision-tree learner was then equipped with 15 additional attributes manufactured as pairwise differences of the six primitives. The effect was spectacular. The same algorithm now constructed theories that correctly classified about 98% of new cases.

An ILP algorithm (CIGOL, see Muggleton et al., cited above) had earlier been trained on five sets of 40 positive and 60 negative cases. Of the resulting five theories, one scored as high as 91%. In terms of performance, the logically redundant addition to background knowledge had enabled decision-tree learning to do at least as well as ILP. Yet on the transparency criterion, decision trees showed up badly. Each was a bulky and opaque structure of some 40-50 nodes.

ILP-generated theories were short, clear and explicit, even though incomplete. It would be nice to see transparency combined with new attribute formation, as in the following case.

Extended Michalski trains problem. Michalski's classical Eastbound-versus- Westbound discrimination first appeared in a graphical representation [6]. Trains differ in such attributes as the number and shapes of cars and loads, types of roofs etc. Last year a public challenge was issued by Michie and colleagues [7] to induce Prolog rules from facts about trains. The Prolog fact

```
eastbound([c(1, bucket, short, not_double, none, 2 l(circle,1)),
           c(2, rectangle, long, not_double, none, 3, l(triangle, 1)),
           c(3, rectangle, short, not_double, peaked, 2, l(triangle, 1)),
           c(4, rectangle, long, not_double, none, 2, l(rectangle, 2))]).
```

designates as eastbound a train with 4 cars, of which the first is bucket-shaped, short, single-walled, unroofed, and containing one circular load, the second is rectangular, long, etc. etc. A specimen rule in English looks like

If a train has a car with a peaked roof or a sequence of cars with the same load-shape that ends in a short car then Eastbound otherwise Westbound.

In "Trolog" (Prolog plus pre-supplied library of train-friendly predicates) the above rule reads:

```
eastbound(T):- has_car(T,C), arg(5,C,peaked);
               infront(T,C1,C2), short(C2), has_load0(C1,Shape),
                 has_load0(C2,Shape).
```

The "size" of this rule is 23, determined by counting the number of distinct symbols, omitting comma, bracket and full-stop. Page and Srinivasan, in an Appendix to [7], give a formal rationale of this measure. Experiments in progress by Hayes Michie [8] show that it broadly correlates with two measures of "psychological complexity", namely (a) "perceived complexity" derived from subjective rankings of sets of rules and (b) the times taken by subjects to score sets of trains using different rules.

The main competition presented a set of twenty trains, ten Eastbound and ten Westbound. The winner was Bernhard Pfahringer of the Austrian Research Institute for Artificial Intelligence, with an ILP program that can find the smallest rule if this lies below a length limit. A "computational cliff" was encountered after size 17, but a marginally smaller theory, of size = 16, lurked in the training set for his program to discover. The "official" rule originally used to classify the 20 trains, was of size 19, as also the four runners-up . One of these came from Peter Turney of the National Research Council's Institute for Information Technology, Canada. His program RL-ICET [9, 10] is conceptually related to LINUS [11]. It makes repeated decision-tree inductions guided by cost-related parameters. A genetic algorithm repeatedly selects among parameter-sets and returns the "fittest" tree found, fitness here being inverse to size. He and Dr. Pfahringer then both tried a newly generated 12-versus-12 discrimination task. Pfahringer's program, now searching beyond its computational cliff, found a rule of size 20. Once again Turney's RL-ICET ran close with a rule of size 23. What ingredients could so boost decision-tree learning?

First a pre-processor translates the Prolog relations and predicates of the example set into feature vector format. Then each feature is assigned a cost based on the sizes of the corresponding fragment of Prolog code. Decision tree induction is iterated on the example set guided by weighting parameters bearing a relation, to be empirically determined, to sizes of attributes. They thus function as attribute-selection biases. Tree-generation is performed repeatedly using different candidate bias vectors taken from a pool evolved by Grefenstette's [12] GENESIS genetic algorithm. The latter searches a space of bias vectors for one that minimises the Trolog complexity of the resulting tree. An initial pool of 50 vectors was seeded with one that had as it elements Prolog sizes of attributes. The other 49 were set randomly. Mutation and recombination (mating) of bias vectors followed the usual lines of genetic algorithms. Now for attribute manufacture.

Each attribute vector representing a train, and hence each bias vector, comprises altogether 1199 elements, each attribute being truth-valued. From 28 initial predicates applying to cars in a train (e.g. ellipse(C) when the car C has an elliptical shape), a train attribute is defined according to whether the train has such a car. Next 378 attributes are formed as the unordered pairs from the original 28. Using the relation infront(T,C1,C2), 784 ordered pairs can be formed. For example, u_shaped_infront_peaked_roof has the value 1 when the train has a u-shaped car in front of a car with a peaked roof, and 0 otherwise. Finally 9 more train attributes are added such as train_4 which has the value 1 when the train has exactly four cars. Thus an attribute vector has 28+378+784+9 = 1199 elements.

The rule language so defined is of course incomplete. But what I want to stress is that at first sight the size of the attribute set seems shockingly large. Yet relative to the pattern-memories used in human expert cognition it should seem shockingly small. The number of patterns stored in (largely subliminal) memory by a chess master comfortably exceeds 50,000. Yet complete specification of a chess position requires at most 32 co-ordinate-pairs expressible in a language of simple syntax using only eight primitives. Patterns, not board co-ordinates, are the building blocks of chess-master concepts. Moreover the intuitive store grows according to experience of different kinds of play.

This property of statistical adaptation is mirrored in the generic form of ICET (RL- ICET is "Relational Learning ICET"). Attributes can be declared as either "immediate" or "delayed", according to whether values are obtained at run time evaluation or by pre- evaluation. Under the train challenge's declarative rather than procedural measure of complexity it was rational to set RL-ICET's attributes to "delayed". Under the "immediate" option, the program would test each tree's fitness by actually running it over the training set, thus directly estimating its procedural complexity. The statistical properties of a domain sample determine the frequencies with which different parts of a rule are invoked. Hence without running the tree the same quantity can be estimated as the weighted mean costs associated with interior nodes (i.e. attribute calls). Only calls of attributes contribute to cost, so that weights are the relative probabilities during execution of traversing each given node. If the classification law remains constant but the statistical structure of the population changes, ICET will track these by picking differently from logically equivalent rules.

There is a complementary form of the problem, in which the logico-statistical structure of the domain remains fixed while the learner gains facility during successive trials. Biomolecular chemists become skilled in conjecturing new structure-activity relations for familiar classes of compounds. Familiarity grows with respect to logical as well as statistical regularities. Ways are needed for relational learning algorithms to add new patterns to their own background knowledge, allowing additions to persist from one discovery exercise to the next.

2.2 Structured Induction

The general notion of learning within a hierarchy was proposed as early as 1953 by Claude Shannon [13].

> Can we organize machines into a hierarchy of levels, as the brain appears to be organized, with the learning of the machine progressing up the hierarchy?

The example I shall examine illustrates the development of hierarchies of decision trees with backchaining links between the trees. The attribute invoked at each node of each tree can be either "primitive" or "procedural". The latter calls a lower-level concept also implemented as an induced tree. The method, termed "structured induction" by one of its originators, Alen Shapiro [14], can have spectacular effects on transparency. Shapiro demonstrated machine-mediated articulation of an elaborate chess concept subarticulately resident in every master player. He considered the 209,718 legal positions in which White has the move with king and pawn versus king and rook, with White's pawn on a7 threatening to queen immediately.

Without structuring, computer induction was still able to build a classifier directly from primitives, but in the form of a single unstructured decision-tree of 83 nodes, - an end-game theory for Martians. As a theory for humans it is useless. Concerning the desired human-type theory Shapiro noted that

1. no such representation had ever been constructed by human agency;
2. since the expertise is largely inaccessible to conscious review, no such representation, in the opinion of chess masters consulted, could ever be so constructed;
3. no representations (brain compatible or otherwise) could be constructed by known programming techniques other than by exhaustion of the domain;

4. because of the inaccessibility of expertise to conscious review ... the so-called "knowledge engineering" techniques of contemporary expert systems work would not be applicable;
5. in the absence of information about the structural features, conventional use of inductive learning techniques would be inadequate. Rules generated in this way might well be complete but not brain compatible.

By way of contrast to the black box decision tree, a structured theory was built interactively from the same vocabulary of 36 primitive attributes, supplemented with eight intermediate concepts named by the chess-master and individually implemented by computer induction. The machine was also responsible for rendering the whole into English. The top-level rule looks like this:

```
COMMENT:
This rule is used to decide if a White-to-move position of king and pawn
(on a7) versus king and rook is won for White or not. Bracketed notes refer
to the attribute-name associated with testing the corresponding condition.
END COMMENT.

Position is won for White if and only if
    the Black rook can be captured safely (primitive: rimmx)
    OR none of the following is true:
    there is a simple delay to White's queening the pawn (procedural: DQ)
    OR one or more Black pieces control the queening square (primitive: bxqsq)
    OR the WK is in stalemate (primitive: stlmt)
    OR there is a good delayed skewer threat (procedural: DS)
```

Each of the two procedural attributes called is similarly represented, and so on, spanning altogether four hierarchical levels.

During construction, the expert is shown positions and asked: "What properties of this case do you need to know in order to adjudicate it?" A list is made from his answers, and incremented by posing the same question for further selected cases. When the list of position-attributes has settled down, no further progress can be made on the given concept without machine aid. This is because once the half-dozen or so tests needed to classify a given case have been identified, the expert cannot give any further account that could be programmed of how he applies each attribute or of how he combines the results into a decision procedure. At this point computer induction takes over the task of articulation, returning to the above cycle as each given concept and sub-concept is induced.

When end-game specialists were allowed to interrogate this system, their impression was of contact with a sophisticated and resourceful mentality that exceeded their own grasp of this small domain at every level. Before concluding that structured induction is the only road to such transparency, mention should be made of a claim by B.R. Gaines [15] on behalf of his Exception Directed Acyclic Graphs as the basis for inducing from the same primitives a transparent solution to the same problem. The potential of the method is at an early stage of assessment.

Industrial impact. The technique of structured induction broadly escaped academic attention. Instead it found a world-wide niche in commercial applications. In industry and finance, numerous specific needs exist for knowledge-acquisition aids that allows tacit forms to be recovered and given explicit expression. Inaccessibility of intuitive expert know-how has been widely remarked. Sterling and Shapiro [16] give an account of their attempts to debrief an expert loan officer: "Our expert ... happily discussed the profiles of particular clients, and the outcome of their credit requests and loans, but was reluctant to generalize".

The induction engineer separately delegates each of the parts of the elicitation task: (i) the domain expert supplies or selects illustrative cases, (ii) machine learning generalizes them into decision trees. Large systems today can have a hundred or more small layered decision trees, comprising altogether thousands of rules. Early applications were reviewed in [17,18]. The flavour of contemporary use is well conveyed by snapshots of recent systems developed and installed by one small Swedish company. I am indebted to Mr Rudi Sillen, Director of Novacast AB, Ronneby, for permission to reproduce this information in an Appendix. References there to XpertRule denote a suite of structured induction development tools licensed from Attar Software, Leigh, UK.

A ten year track record now exists of accomplishments that are on any reckoning remarkable. So what factors inhibit ML research along this line? The most salient is that the methodology requires active availability of a domain specialist over long periods. When I was Technical Director of a company similar to Attar and Novacast we ordinarily put a clause into co-development contracts specifying an agreed number of hours per month of a domain specialist's time. Persuading a domain specialist to spend time in a software project is akin to persuading water to flow up-hill. It can be done, but needs pumping. In industry the pump is the contract. The specialist then does whatever the contract stipulates. Academic institutions cannot provide such pumps.

2.3 Behavioural cloning

Humans learn skilled real-time tasks partly by imitation. Successful beginnings have been made with ML to "clone" aspects of skilled behaviour from canned performance samples. Early success was recorded by Chambers and Michie [19] in 1969. They commented as follows concerning their pole-balancing task (they also studied their BOXES program under conditions of trial-and-error re-inforcement learning):

> On the face of it, it looks rather dull, since not much is involved beyond a passive transfer to the machine of experience acquired by the human. We might therefore think that a machine that has been educated in this way could not outperform its trainer when left to its own devices. This is a mistake. The human's performance is variable even within a given situation, ... and one false step in a real-time task can be fatal. By contrast, the BOXES program which has accumulated for a given box [logically

defined "situation"] the sum total of the human's experience, will consistently apply, whenever it visits this box, whichever decision is indicated by the balance of this experience. Thus if the machine makes the right kind of summary for itself of even quite low-performance human efforts, this summary can be made the basis of a much higher-performance strategy - the human's errors being "averaged out".

In new work with Sammut [22] using a desktop flight simulator, a diversity of control actions was available, in contrast to the "push-left, push-right" repertoire of pole-balancing. A blackboard architecture was adopted within which each agent was inductively trained from selected traces of expert input-output behaviour. Successes and shortcomings of behavioural cloning have been reviewed by Urbancic and Bratko [20]. They discuss the following problem domains: pole-and-cart balancing [19, 21], flight-simulator control [22], telephone-line scheduling [23] and their own work on crane-simulator control. Their conclusions are:

- Successful clones have been induced using standard ML techniques in all four domains.
- The clean-up effect, whereby the clone surpasses its original, has been observed in all four domains.
- In all domains best clones were obtained when examples from a single human only were used.
- The present approach lacks robustness in that they do not guarantee inducing with high probability a successful clone from given data.
- Typically, the induced clones are not sufficiently robust with respect to changes in the control task.
- Although the clones do provide some insight into the control strategy, they in general lack conceptual structure that would clearly capture the causal relations in the domain and the goal structure of the control strategy.

The chief conclusion is that learning from exemplary performance requires more than mindless imitation. The point can be vividly illustrated with a program (Polecat for Windows Research Version, developed by Mr. John White) for interactively testing human subjects. The pole-and-cart task is elaborated to include a third action, OFF, in addition to LEFT and RIGHT. The leading performance-scoring term is the number of times in a set period that the simulated system can be piloted across the track's mid-line. From numerous performances by an expert, the package builds an arbitrarily large training set of exemplary snapshots. From these a "clone" is induced in the form of a decision tree. When tested at run time, performance on the screen can seem exasperating. The clone mimicks the human exemplar by holding its modal position close to the mid-line, but differs in balancing the pole with more consistent accuracy (the clean-up effect). Hence errors and righting actions fail to carry the cart over the mid-line, so it scores poorly. How could it be so stupid? Of course, no-one has told the clone what else besides mimicry is the object of its exercise.

Current thinking centres on selective re-introduction of a cognitive level for goal setting and monitoring. Of considerable relevance are a number of essentially non-ML studies. Pearson, Huffman, Willis, Laird and Jones [24] worked with flight-simulator control (Cessna, on a Silicon Graphics workstation) at the University of Michigan. They made resourceful use of the SOAR architecture (see Newell [3]) while constructing skill models essentially by hand-crafting. Reported robustness to varied starting conditions is attributable, we believe, to the goal-directedness of their underlying architecture. Benson and Nilsson's [25] recent developments of the T-R formalism have demonstrated a particularly interesting integration of reactive control behaviour with hand-crafted goal-structures.

In this area too availability of domain specialists can be critical. For example, in behavioural cloning of piloting skills the closer the contact with professional pilots and flying instructors the better.

3 Human-Computer Learning

The crisis of unemployabilty in technically advanced nations has been aggravated by information technology's part in destabilising unskilled work. Yet research into intelligent uses of information technology to mend some of the damage is too long-term for most industrial and governmental sponsors. I see a role for studies of Human- Computer Learning, where agents of the two species interact in mutual enhancement of each other's knowledge-based skills.

With the idea that further advances might be directed towards trainability and self-trainability, the Human-Computer Learning Foundation, formed in 1994 by Rupert Macnee, Jean Hayes Michie and the author, was this year accepted as an educational charity by the UK Charities Commission. Mr Macnee lives in Los Angeles and is a documentary film maker. Ms. Hayes Michie is a cognitive psychologist. She and the author spend most of their research time in Palm Desert, California, in working relation with the Coachella campus of the University of California.

The Foundation's aim is to apply new advances in machine learning (ML) to computer-aided skill acquisition and self-instruction. Recent multimedia developments have found a growing market in home education. But the socially more sensitive area of skill training remains relatively untouched. The Board's intention is to attract sufficient benefactions to direct a significant flow of new work along this channel.

References

1. Muggleton, S. and Feng, C.: Efficient induction of logic programs. In Proc. First Conf. on Algorithmic Learning Theory (ed. S. Arikawa, S. Goto, S. Ohsuga and T. Yokomori, 1990), Tokyo: Japan. Soc. Art. Intell. pp. 368 - 381.
2. Sternberg, M.J.E., King, R.D., Lewis, R.A. and Muggleton,: S. Application of machine learning to structural molecular biology. Phil. Trans. R. Soc. Lond. B, 344 (1994) 365-371.

3. Newell, A.: A Unified Theory of Cognition. Boston, MA: Harvard University Press (1990).
4. Muggleton, S.H., Bain, M., Michie, D. and J. E. Hayes Michie.: An experimental comparison of human and machine learning formalisms. In Proc. Sixth Internat. Workshop on Machine Learning, (ed. A. Segre), Morgan Kaufman (1989) pp. 113-118.
5. Michie, D. and Bain, M. Machine acquisition of concepts from sample data.: In Artificial Intelligence and Intelligent Tutoring Systems, (eds. D. Kopec and R.B.Thompson), Ellis Horwood (1992) pp. 5-23.
6. Michalski, R.S. and Larson, J.B.: Inductive inference of VL decision rules. SIGART Newsletter, ACM, 63 (1977) 38-44.
7. Michie, D., Muggleton, S., Page, D. and Srinivasan., A.: To the International Computing Community: a New East-West Challenge (1994). Available as a Postscript file ml-chall.ps in URL ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/trains.tar.Z. See also URL ftp://ftp.comlab.ox.ac.uk/pub/Packages/ILP/results.tar.Z.
8. Hayes Michie, J.E.: Comparisons of Human and Machine Assessments of Complexity. Unpublished typescript, available from Ms Hayes Michie, 73860 Shadow Mountain Drive, Apt 2, Palm Desert, CA 92260, USA.
9. Turney, P.: Low size-complexity Inductive Logic Programming: the East-West clallenge considered as a problem in cost-sensitive classification. Submitted to IJCAI-95 (1995).
10. Turney, P.: Cost-sensitive classification:empirical evaluation of a hybrid decision tree induction algorithm. Jour. AI. Research. In press.
11. Lavrac, N. and Dzeroski, S.: Inductive Logic Programming: Techniques and Applications. New York: Ellis Horwood (1994).
12. Grefestette, J J.: Optimization of control parameters for genetic algorithms. IEEE Trans. on Systems, Man and Cybernetics 16 (1986) 122-128
13. Shannon, C.E.: Computers and automata. Proc. I.R.E, 41 No. 10 (1953).
14. Shapiro, A.D.: Structured Induction in Expert Systems. Addison Wesley (1987).
15. Gaines, B.R.: Inducing Knowledge. Unpublished typescript. Available from Prof. Gaines at Knowledge Science Institute, University of Calgary, Alberta, Canada T2N 1N4. Email: gaines@cpsc.ucalgary.ca (1995).
16. Sterling, L. and Shapiro, E.: The Art of Prolog. Cambridge, MA: The MIT Press (1986).
17. Michie, D.: New commercial opportunities using information technology. In Knowledge-based Systems: Third Internat Congr (1989).
18. Michie, D.: Problems of machine-aided concept formation. In Applications of Expert Systems, Vol 2 (ed. J.R. Quinlan), Addison-Wesley (1989) pp. 310-333.
19. Chambers, R. A. and Michie, D.: Man-machine co-operation on a learning task. In Computer Graphics - Techniques and Applications (eds. R. D. Parslow, R. W. Prowse and R. E. Green), Plenum Press (1969) pp. 179 - 185.
20. Urbancic, T. and Bratko, I.: Reconstructing human skill with machine learning. In Proc. Europ. Conf. on AI (ECAI 94), Amsterdam (1994).
21. Michie, D., Bain, M. and Michie, J.E.: Cognitive models from subcognitive skills. In Knowledge-Based Systems in Industrial Control (eds. M. Grimble, J. McGhee and P. Mowforth), Stevenage, Peter Peregrinus (1990).
22. Sammut, C., Hurst, S., Kedzier, D. and Michie, D.: In Proc. Ninth Internat. Workshop on Machine Learning, Morgan Kaufmann (1992) pp. 385 - 393.

23. Kibira, D.: Developing an expert controller of a black box simulation of a telephone line using machine induction. Unpublished report, Sydney: AI Lab, Univ. of New South Wales (1993).
24. Pearson, D. J., Huffman, S. B., Willis, M. B., Laird, J. E., and Jones, R. M.: A symbolic solution to intelligent real-time control. In Robotics and Autonomous Systems, 11, Elsevier (1993) pp. 279 - 291.
25. Benson, S. and Nilsson, N.: Reacting, planning, and learning in an autonomous agent. In Machine Intelligence 14 (eds. K. Furukawa, D. Michie and S. Muggleton), Oxford: the Clarendon Press, (in press).

A Seven case studies from Novacast AB

A.1 Adaptive Thermal Analysis System

System task: To control the melting and treatment of cast iron alloys, based on the time-temperature curve of serial samples of the melt. Metallugically important parameters of the curve are available from a standard algorithm.
Approach: Numerous test melts were made and evaluated. The results were used as examples for an inductive system (XpertRule Analyser) to produce the predictive rules and also to discover new relations between parameters.
Field experience: Installed in a Swedish foundry since April 1994, engineers use the system first to optimise the process. Once optimised, the system verifies it and suggests corrections.
Measured benefits of use: Reduced scrap, higher yield, less consumption of energy and of additives. Saving of the order of US$ 50 per ton.
Primary client: The Swedish Foundry Association.

A.2 Metal Painting Guide

System task: To give diagnosis and advice on rust-prevention and final coating.
Approach: Structured induction using XpertRule of more than 200 linked modules from expert-supplied examples.
Field experience: In commercial use since 1993.
Primary client: Coatech AB in Sweden.

A.3 Credit Scoring

System task: To interact with a large database covering all Swedish companies to make assessments of credit-worthiness on demand.
Approach: Example cases were collected partly from the client's experienced experts and partly from credit history. The system was built in a concept-oriented way using NovaCast's NovaLogic machine-learning program.
Field experience: In daily use since 1989.
Primary client: Soliditet, Sweden's major credit scoring company.

A.4 Evaluating Military Units

System task: Evaluating military capabilities of units.
Approach: Within an XpertRule structure, combinations of factors were automatically generated for combat unit commanders to evaluate ("truth table" method of Validation- Directed Induction described in [18]). An expert evaluation system was built that surpassed traditional assessment methods.
Field experience: In use since 1992.
Measured benefits of use: Savings of the order of 10 million Swedish crowns per year.
Primary client: The Swedish Defence Material Administration.

A.5 VAT Advisor

System task: To guide administrators through the regulatory labyrinth of VAT tax calculations, and in particular to enable inexperienced assessors to make expert assessments.
Approach: Concept-oriented structured induction using XpertRule, resulting in more than 150 task-modules comprising about 3000 rules in all.
Field experience: In use at several sites since 1992.
Measured benefits of use: Large reductions in the long, complex and costly training in VAT assessment that had previously been needed.
Primary client: The Swedish company DIALOG.

A.6 Predicting recurrence of breast cancer

System task: To predict likelihood of recurrence within 5 years of surgery.
Approach: A database was available covering some 180 mastectomies and the patient's state 5 years later. Three methods of analysis were applied, namely multivariate statistics, neural nets and rule induction (XpertRule Analyser). Best results were obtained with rule induction.
Field experience: in use since 1993.
Measured benefits of use: System's predictive accuracy is about 80%. Medical specialists achieved 40%. They also preferred the inductively generated rules because they could read them and thus understand and validate them.
Primary client: Central Hospital in Karlstad, Sweden.

A.7 Reduction of nitrogen in waste water

System task: To reduce the content of chemically bonded nitrogen in the waste water from large cities by predicting outcomes and suggesting corrective measures.
Approach: A complex biological method is in use in which bacteria transform the nitrogen to gas. In co-operation with a leading specialists in water purification, a set of knowledge- based modules was developed, each simulating a step in the control sequence. Output from one module is used as an input parameter to

the next. Each module was built with XpertRule from expert-supplied examples using the "truth table" method mentioned in case A.4.
Field experience: In use since 1993 at the waste water purification plant in Ronneby, Sweden.
Measured benefits of use: In addition to its use for error-diagnosis and control, the system has proved to be a cost-effective training tool for new operators. They use it on a "what if" basis by simulating postulated situations and testing alternative actions.
Primary client: Ronneby local city authority.