MACHINE LEARNING: Applications in Expert Systems and Information Retrieval
RICHARD FORSYTH, Technical Director, Warm Boot Limited, London, and ROY RADA, Research Officer, National Library of Medicine, Bethesda, Maryland, USA

This overview of the artificial intelligence sub-field of machine learning ties together a number of disparate studies within a coherent framework, identifying common themes and significant differences. The book makes good a gap in the literature, taking a strong practical approach, and highlighting the 'how to do it' aspects of machine learning.

ARTIFICIAL INTELLIGENCE: Applications in the Future of Software Engineering
D. PARTRIDGE, Department of Computer Science, New Mexico University, Las Cruces, USA

This exploration of the relationships between artificial intelligence (AI) and software engineering circumscribes the limitations of current commercial AI, and investigates the full potential of AI for practical software.

THE MIND AND THE MACHINE: Philosophical Aspects of Artificial Intelligence
Editor: S. TORRANCE, Senior Lecturer in Philosophy, Middlesex Polytechnic

This book asks an important question: What light do current developments in computing shed on philosophical and psychological understanding of the human mind? A colloquium of authors from philosophy, artificial intelligence, linguistics, psychology and computing provide their answers.

"a good book, well worth reading" — A. D. Vella in *Computer Bulletin*

LOGICS FOR ARTIFICIAL INTELLIGENCE
R. TURNER, Director of Cognitive Studies Centre, University of Essex

"the book is a winner. Logicians will like it and even non-logicians will get some idea of what is going on and why it matters to them if they are working on artificial intelligence programs. It would be hard to come away from this book without having gained at least a few bright ideas. I thoroughly enjoyed it" — Chris Naylor in *Computer Weekly*

"useful . . . timely guide to an important area of study" — A. M. Andrew, Viable Systems, Lifton, Devon, in *Robotica*

COMPUTER INTERPRETATION OF NATURAL LANGUAGE DESCRIPTIONS
C. S. MELLISH, School of Social Sciences, University of Sussex

This book describes how a computer program can build up an internal model of a world described in natural language. It concentrates on questions involving the *timing* of this process and asks: how does a natural language text convey meaning, and to what extent is it possible to build an understanding *gradually* as the text progresses?

INTELLIGENT SYSTEMS: The Unprecedented Opportunity
Editors: J. E. HAYES and D. MICHIE, The Turing Institute, Glasgow

"I thoroughly enjoyed reading this book" — C. W. Painter, Library and Information Services Committee, in *The Production Engineer*

"very good . . . well worth having" — A. M. Andrew, Viable Systems, Lifton, Devon, in *Robotica*

published by
ELLIS HORWOOD LIMITED
Publishers · Chichester

distributed by
HALSTED PRESS a division of
JOHN WILEY & SONS
New York · Chichester · Brisbane · Toronto

Halsted Press Library Edition ISBN 0-470-20335-8

MICHIE

ON MACHINE INTELLIGENCE

Ellis Horwood Series in **ARTIFICIAL INTELLIGENCE**

ON MACHINE INTELLIGENCE

Second Edition

DONALD MICHIE

Foreword by Professor Sir Hermann Bondi, K.C.B., F.R.S.

Donald Michie was among the pioneers of Bletchley Park from 1942–1945, working with the Colossus series of computers. After a post-war career in biology, for which Oxford University awarded him the D.Sc., he formed the Experimental Programming Unit at Edinburgh in 1965, and conceived and directed the FREDDY robot project, recognised as a world first. He held the chair of Machine Intelligence at Edinburgh from 1967–1985, and is Editor-in-Chief of the acclaimed Machine Intelligence series, as well as co-editing *Intelligent Systems* (Ellis Horwood, 1983). He is Chief Scientist and Director of Research at the Turing Institute and is a Fellow of the Royal Society of Edinburgh and of the British Computer Society.

Ellis Horwood Series in
ARTIFICIAL INTELLIGENCE

Series Editor: Professor John Campbell, Department of Computer Science, University College London

ON MACHINE INTELLIGENCE, Second Edition

DONALD MICHIE, The Turing Institute, Glasgow

This lucid commentary on twenty-five years with machines which learn and plan traces and illuminates the latest progress towards the goal of the intelligent machine. It surveys an integrated sequence of researches undertaken by Professor Michie and his colleagues, while at the same time providing an authoritative introduction to the subject.

This second edition, which has been completely revised and expanded, incorporates much stimulating new material, resulting in a new book rather than a new edition. Half of the present text has been newly written, and fresh introductory notes supplied for each of the book's four main sections. Every chapter is individual and self-sufficient, offering the reader the freedom to select topics to suit his own needs, and helping towards both theoretical and practical approaches to representing cognitive processes in machines.

It is a thought-provoking review which will help satisfy the thirst for information and advice in the fast-growing areas of artificial intelligence, from the pen of an internationally respected leader in the field. It is fascinating to read, elegant and persuasive in its presentation, and will be widely read not only by scientists and engineers, but by all interested in the role of computers today.

Readership: Scientists, engineers and researchers in artificial intelligence, computing, machine intelligence, robotics, machine learning; knowledge engineering; cognitive science; information science; software engineering; electronics.

ON MACHINE INTELLIGENCE

Second Edition

ELLIS HORWOOD SERIES IN ARTIFICIAL INTELLIGENCE

Series Editor: Professor JOHN CAMPBELL, University College, London

S. Abramsky & C. Hankin (editors)	**Abstract Interpretation of Declarative Languages**
B.K. Boguraev	**Natural Language Interfaces to Computational Systems**
M.A. Bramer (editor)	**Computer Game Playing: Theory and Practice**
E.J. Briscoe	**Computational Speech Processing: Syntax and Prosody**
J.A. Campbell (editor)	**Implementations of PROLOG**
J.A. Campbell, R. Forsyth, A. Narayanan, & M. Teague	**Dictionary of Artificial Intelligence**
R. Davies (editor)	**Intelligent Library and Information Systems**
E.W. Elcock & D. Michie (editors)	**Machine Intelligence 8: Machine Representations of Knowledge**
J. B. Evans	**Discrete System Simulation**
R. Forsyth & R. Rada	**Machine Learning: Applications in Expert Systems and Information Retrieval**
D.M. Gabbay	**Elementary Logic: The Procedural Perspective**
T. Gergely & I. Futó	**Artificial Intelligence in Simulation**
J. Glicksman	**Image Understanding and Machine Vision**
H.W. Gottinger	**Artificial Intelligence: The Commercialisation of Intelligent Systems**
R. Hawley (editor)	**Artificial Intelligence Programming Environments**
J.E. Hayes, D. Michie & L.I. Mikulich (editors)	**Machine Intelligence 9: Machine Expertise and the Human Interface**
J.E. Hayes, D. Michie & Y.-H. Pao (editors)	**Machine Intelligence 10: Intelligent Systems: Practice and Perspective**
J.E. Hayes & D. Michie (editors)	**Intelligent Systems: The Unprecedented Opportunity**
J.R.W. Hunter, N.M. Gotts & R.K.E.W. Sinnhuber	**Artificial Intelligence in Medicine**
W.J. Hutchins	**Machine Translation: Past, Present and Future**
C. Mellish	**Computer Interpretation of Natural Language Descriptions**
D. Michie	**On Machine Intelligence, Second Edition**
D. Partridge	**Artificial Intelligence: Applications in the Future of Software Engineering**
P.E. Slatter	**Cognitive Emulation in Expert Systems**
L. Spacek	**Advanced Programming in PROLOG**
K. Sparck Jones & Y. Wilks (editors)	**Automatic Natural Language Parsing**
L. Steels & J.A. Campbell (editors)	**Progress in Artificial Intelligence**
S. Torrance (editor)	**The Mind and the Machine**
R. Turner	**Logics for Artificial Intelligence**
M. Wallace	**Communicating with Databases in Natural Language**
H. Wertz	**Automatic Correction and Improvement of Programs**
M. Yazdani (editor)	**New Horizons in Educational Computing**
M. Yazdani & A. Narayanan (editors)	**Artificial Intelligence: Human Effects**

ON MACHINE INTELLIGENCE

Second Edition

DONALD MICHIE
The Turing Institute, Glasgow

ELLIS HORWOOD LIMITED
Publishers · Chichester

Halsted Press: a division of
JOHN WILEY & SONS
New York · Chichester · Brisbane · Toronto

This edition first published in 1986 by
ELLIS HORWOOD LIMITED
Market Cross House, Cooper Street,
Chichester, West Sussex, PO19 1EB, England
The publisher's colophon is reproduced from James Gillison's drawing of the ancient Market Cross, Chichester.

Distributors:
Australia and New Zealand:
JACARANDA WILEY LIMITED
GPO Box 859, Brisbane, Queensland 4001, Australia
Canada:
JOHN WILEY & SONS CANADA LIMITED
22 Worcester Road, Rexdale, Ontario, Canada
Europe and Africa:
JOHN WILEY & SONS LIMITED
Baffins Lane, Chichester, West Sussex, England
North and South America and the rest of the world:
Halsted Press: a division of
JOHN WILEY & SONS
605 Third Avenue, New York, NY 10158, USA

British Library Cataloguing in Publication Data
On machine intelligence. — 2nd Edition
1. Artificial intellignece
I. Michie, Donald
006.3 Q335

Library of Congress Card No. 86–9885

ISBN 0–7458–0084–X (Ellis Horwood Limited)
ISBN 0–470–20335–8 (Halsted Press)

Phototypeset in Times by Ellis Horwood Limited

Printed in Great Britain by The Camelot Press, Southampton

Contents

Foreword

In the course of some of my jobs I visited quite modest-sized industrial laboratories. The director of the laboratory, perhaps a little overawed by a visit from someone he thought of as an eminent scientist, was liable to start his description of the work of the laboratory with the remark 'What we do here is really not science at all, it is just trial and error'. I used to respond, perhaps a trifle unkindly, that I did not know that there was anything to science other than trial and error.

This identity of science and trial and error is often obscured by the opacity of jargon and of sophisticated mathematics, but Dr. Michie, in this book as elsewhere, demonstrates with charm on every page that he appreciates that identity. The charm owes much to another truth that he appreciates, that *playing* is an excellent way and often the best way of learning. This fact is often hidden by an abnormal distinction that is drawn between playing and 'the serious work of learning'. Dr. Michie is never under that apprehension. The joy of reading through this volume is precisely that one is never left in doubt that he is enormously enjoying his games, and learning a very great deal in the process, learning that is, with the pleasure of playing, vicariously transferred to us, his readers. I was myself involved in space affairs when, in-April 1970, a serious malfunction in the Apollo 13 mission to the Moon led to great anxiety for the safety of the crew. By a rapidly devised brilliant strategy, the crew returned to the Earth safe and sound, albeit without having landed on the Moon. When I expressed my astonishment to my friends at NASA that this strategy had been thought up and adopted in the very short time available, I was told that this had only been possible because the staff at Mission Control had been spending all their time playing games with their equipment. Rescue from disaster had been one of the games, and so they were familiar with what was needed, though of course the exact problem that actually occurred had not been foreseen. A less wise management would not have allowed their staff to play games with their expensive equipment (perhaps to save the taxpayer's money?) and

then the rescue could not possibly have occurred. Our play instinct is always something to be fostered and Dr. Michie shows us once again how fruitful this indulgence can be.

In my younger days there were still plenty of people around in the university world, largely, though not wholly in the humanities, who would dismiss study of this or that with the words: 'But of course it is a *mere* machine'. It is worth stressing to-day that such views were nonsense already fifty or even a hundred years ago, though their absurdity was only made plain to everybody by the development of the computer. Yet even less blinkered people took a long time to appreciate that there were difficulties in understanding how a system containing even a quite modest number of switches could act in circumstances not envisaged by the designer. Control engineering is a new subject not because the need for it is new, but because we humans were so slow to appreciate this need.

Dr. Michie's subject of machine intelligence is seen by him, to our benefit and enjoyment, very largely in this light. Putting a few devices of relative simplicity together makes a system the responses of which cannot be forecast but have to be explored. Nowhere is this more pleasantly displayed than in his MENACE machine for playing noughts and crosses, where machine learning through trial and error led with such speed to excellent results. But altogether the sections on machine learning are very fascinating. However, there are worrying aspects too, especially the point so well brought out in this volume that by such or other means programs are developed which are effective in practice but inscrutable in the sense that it is not clear what the program is and therefore how it would handle unusual situations. It is a matter of concern that, in the future, issues of real importance may be decided (or at least decisions greatly influenced) in, say, military matters by such inscrutable programs. If I may digress for a moment, this is precisely the point that is perhaps most worrying to me about non-democratic systems of government. In a democracy, the range of opinions and attitudes is manifest. Their changes and the cross-currents are there for all to see so that the response of government and opposition parties to a developing new situation is, if not always predictable, yet is invariably understandable. In the Soviet system or in that of, say, Franco Spain there is no such decision taking with full public coverage. Decisions are reached quite possibly efficiently but the process is opaque and cannot generally be understood, let alone predicted.

To come back to my remarks about trial and error at the beginning, development through trial and error is necessarily messy and follows a zizag course. Thus, as Polanyi has said, science does not progress like a steam-roller, much as this fact surprises many of our non-scientific fellow citizens. The progress of science is piece by piece, and even the most brilliant contributions are only keystones to arches of many irregularly laid other stones.

Of course this is true of work on machine intelligence, but Dr. Michie makes the intriguing and to me convincing point that it is likely also to be true of human intelligence, quoting Herbert Spencer and stressing the

essential messiness of evolutionary advances. Therefore the insights and hints on the nature of human intelligence that may be gained from work on machine intelligence are likely to be of just the right incremental type and not of the 'sudden light' kind so much hoped for by non-scientists and so unlikely to be really helpful.

I hope and trust that most readers of this volume will share my delight in it and therefore will agree with me in wishing more power to the elbow of Dr. Michie and others in the field who will surely advance it with many hesitations and false turnings, but overall progress, in the truly human manner of all science.

Sir Hermann Bondi
April 1986

Foreword to the first edition

During a recent visit to China I was fascinated by the dexterity of the abacus users in banks, shops and factories. This simple device, whose moveable beads are the digits to be manipulated in accordance with the usual processes of arithmetic, illustrates how ancient is man's urge to supplement the digits of his hands for calculating purposes. In the 1850s Babbage designed an ambitious computing engine in which the processing of the digits was performed by mechanical elements. Alas! his ideas were too far ahead of the available technology and his success was limited. Nevertheless, Babbage's design concepts were sound, and by the use of thermionic tubes in place of mechanical linkages the electronic engineers of World War II were able to produce successful digital computing machines in response to the military need for high-speed calculation.

The invention of the transistor in 1948 removed the power-consuming tubes from the first generation computer and ultimately led to the modern high-speed digital electronic computer in which the active elements are solid-state integrated circuits. Such calculating engines have become indispensable, not only in scientific research but in all aspects of engineering and in the handling of business data. But the modern computer is not merely a mammoth abacus, rather is it to be regarded as a general processor of information. In order to perform the arithmetical operations the program or sequence of instructions necessary to execute the calculation is held in store in the form of coded binary digits, as are also the numerical data to be processed. Other types of information may be similarly coded and stored and ultimately subjected to a sequence of logical operations in accordance with the program. Thus the modern electronic computer is concerned with information processing in the broadest sense and has become the essential tool of the age of Information Engineering in which we live — a small but significant step towards the Intelligent Machine?

Computer programs and their 'authors' have become of crucial importance in this broadened field of use. In modern jargon we say that the

'software' which is to direct a mathematical investigation, the design of an engineering component or the control of an automated chemical plant now transcends in importance the 'hardware' itself, which is the computer and its associated input and output devices. How is software engineering to be advanced? Surely by the perfecting of computer languages which must far exceed in power and flexibility the simple sequences of binary coded instructions which formed the early programs. To be sure, new techniques of programming will be developed as part of the specific engineering projects they are intended to serve, but there is need also for broader research into the whole software problem; this is just what the university team is ideally fitted to perform. Improved computer languages have already stemmed from the Edinburgh work, and these are finding application in fields remote from the study of Machine Intelligence and computerized games which prompted them.

I have been deeply involved in the design and operation of large radar systems as used for airspace invigilation and control of civil air traffic. Here the problem is to detect all aircraft flying within the region of surveillance and to form the aircraft echoes as displayed upon the plan position indicator into tracks that may be associated with the flight plans of the aircraft as filed by the captains. The programs so far devised for computer-controlled target recognition and tracking show many points of similarity with the trial and error computer learning systems developed by Professor Michie and his colleagues. Certainly, the radar problem is very much a case of the engineer playing a game against the environment and there is need for the flexibility of the programs to be such that a veritable learning process is required if the computer tracker is to match the performance of the experienced radar operator.

Again, the integrated circuit which is the essential component of a modern computer can fulfil a multiplicity of very complex functions whether of logic or storage, and a small wafer of silicon, the size of a pin-head, may contain a large number of active elements. The design of such a device, and the layout of the interconnections, is an extremely difficult topological problem. It is therefore singularly appropriate that the computer itself should be invoked to help design the vital elements of which it is composed. CAD, i.e. computer-aided design, is now an essential activity within the semiconductor industry, but the programs required are very complex and tedious to write; new methods for preparing them are urgently needed.

In his essay on 'Machine Intelligence as Technology' Professor Michie discusses the possible practical applications of the results of research conducted in his own laboratory and others like it. Certainly Machine Intelligence should not be regarded as the only approach to the automatic factory, nevertheless it has a great contribution to make to the better understanding of the role which the modern electronic computer can play in such automated systems. In this essay one of the major goals of Machine Intelligence research is identified as the discovery of 'better design principles for teachable programming systems'. I recognize the need for deep study of programming techniques and also for methods of representing within the

computer that knowledge of a limited part of the outside world which it is the aim of the program to influence. Studies in university departments which lead to better understanding of the total modelling process will be extremely valuable to scientists and engineers in industry.

I have found these papers fascinating to read, elegant and persuasive in their presentation of a complex subject, yet stimulating in their relevance to my own technological problems. I trust that they will be widely read, not only by scientists and engineers, but by all those interested in the role of the computer in our modern world.

Sir Eric Eastwood
December 1973

Preface

A collection of mine was published in 1974 by Edinburgh University Press under the title *On Machine Intelligence*, but sold out without reprinting. The publishers of this new edition acquired the rights of the old one and suggested that it be revised and extended by incorporating new material. With the encouragement of John Campbell as Series Editor I took this in hand, initially as something of a chore. But I found myself becoming engrossed. In the event it has become more a new book than a new edition. About half by bulk is newly written since the earlier publication. The Introduction comes more or less unchanged from the earlier book, but introductory notes have also been supplied for each of the new book's four main Sections.

I have not attempted to stamp out the duplications of topic which inevitably crop up in this kind of collection. Such stringency would make sense if I were expecting the reader to start at the beginning and to proceed from left to right. But I see no reason to be officious. He should feel free to hop about, if he prefers, or to read the book backwards. If he does, he will still, I hope, find every Chapter self-sufficient.

Those who find themselves stimulated to pursue these topics further, whether along academic or commercial lines, should know of various institutions through which applied AI has become professionalized, in particular the British Computer Society's Expert Systems Specialist Group and the American Association for Artificial Intelligence. These societies coordinate a range of activities, including seminars, conferences, and publication of periodic newsletters. In addition the National Computer Centre offers a range of information services and other forms of assistance.

The thirst for information and advice in these areas is growing. Expositions of topics in artificial intelligence are a correspondingly urgent need, admirably addressed by this series published by Ellis Horwood under Professor John Campbell's distinguished editorship. I am pleased to have become a part of their endeavour.

Donald Michie
January 1986

Introduction

Certain tasks are generally agreed to require intelligence, for example playing chess, translating from one language to another, building a boat, or doing mathematics. Preceding and accompanying any of the physical transactions involved in the above are certain operations of thought. Since the earliest recorded times attempts have been made to construct systematic rules for thinking. The high-speed digital computer has enabled us now to discover that these attempts have carried us only a negligible part of the way towards the desired objective. It is possible in principle to take any sufficiently well specified theory of 'how it is done' and, by translation into a programming language, to 'run it on the machine'. When we turn, however, to a chess primer, a grammar, a boat-builder's manual, or a mathematics text-book we encounter an uncomfortable hiatus. Even the rules proposed by scholars with a special and systematic interest in the formalization of 'how it is done', such as de Groot in chess, Chomsky in linguistics, and Polya in mathematics, fail disappointingly to bridge the void. After the first flush of excitement comes the question: 'How would I program it?' The conviction follows that although here perhaps is a foothill or two, the mountain is yet to climb.

We are faced, then, with an intriguing possibility, and it is one from which Machine Intelligence derives its name and aim. If we can form a sufficiently complete and precise theory of any given aspect of intelligence, then we can convert it into a computer program. The program itself constitutes an expression of the theory, but it should also, if the theory is valid, have the power to cause the computer to manifest behaviour entirely similar to that which the theory purports to describe. If we believe that we really and truly understand Euclid, or cookery for that matter, there is an acid test. We can be asked to convert our understanding into program, and so cause the machine to do geometry or compose recipes as the case may be. We must certainly own, from the present level of achievement in computer programming for complex tasks, that we do *not* yet understand either Euclid

or cookery: we may possess a kind of instinctual 'understanding' of such tasks, analogous to that by which a high-jumper gets himself over the bar or an acrobat balances on a wire, but we have not achieved understanding of the understanding. If we had, we could program it. If we cannot, then although, as *Homo sapiens*, we may display this or that capability, we cannot claim truly to understand, in the given respect, what it is to be human.

The question of what most distinguishes man from the beasts remains an open one. Man is not, for example, the only tool-using animal. In addition to the recorded uses of tools by birds and mammals, some highly elaborate procedures have been observed in the insect world. The tree ants of Africa and Australia fasten leaves together with silk. A number of ants pull the edge of a leaf into position. Others pick up mature larvae, squeeze them so that they secrete liquid silk, and use them as we would a tube of glue to fasten the leaf down. Other larvae are used as shuttles to weave the nest itself. A. B. & E. B. Klots, from whom my account is taken, comment: 'As far as is known, this extraordinary habit is unique in the animal kingdom, the nearest thing to it being the exploitation of child labour by humans'.

Nor is man the only language user, as recent studies of the use of sign-language by the chimpanzee have established. It is even in doubt whether the use of syntax, as opposed to association of signs without regard to order, may possibly be unique to man.

Man is undoubtedly 'wise' or 'intelligent' (the right translation of *sapiens* is hard to hit), but comparison with horses, dogs, dolphins, and apes seems to reveal a difference in degree rather than kind. According to Tennyson, it was not so much wisdom that Pallas Athene offered to Paris as

> 'Self-reverence, self-knowledge, self-control'.

To frame from this a distinctive picture for our species, Tennyson's line should perhaps be capped:

> 'And self-preoccupation most of all'.

Man worries about himself. On the high philosophical plane: 'Who am I? Where do I come from? Where am I going? What is my nature? How should I live?' On the plane of daily intimacy: 'How do I look? What do I feel? What sort of person am I?' And in his leisure life of books, music, magazines, plays, cinema, and television, there is blended with the purely cultural and the purely frivolous the same perpetual quest for mirrors, mirrors to enlarge, mirrors to elucidate, mirrors to produce and to present himself to himself.

In his loving and anxious quest there is no professional skill which has not been enlisted. Yet man remains

> 'Most ignorant of what he's most assur'd—
> His glassy essence.'

In the centuries which have passed, man's ignorance of almost everything else has been lessened or abolished. But the stubborn persistence of self-ignorance has actually now come to endanger him. Man may or may not

survive the next two hundred years without obliteration in war or strangulation through industrial and population growth. Experts differ on magnitudes of disaster and on time-scales. But on one point they seem to be in agreement; that the need is not for more physics, for more chemistry, or for more of the old-style industrial technology, but for better understanding of the physiology, psychology, sociology, and ecology of our own species.

Thus Machine Intelligence is an enterprise which may eventually offer yet one more mirror for man, in the form of a mathematical model of knowledge and reasoning. From such work we may perhaps learn a little more about our own capacities. When one speaks of Machine Intelligence, one speaks of a collective venture to build 'knowledge machines'; but one also speaks of an unintended consequence: to fashion a mirror for the first knowledge machine of all, the machine within the skull.

This book consists of a selection of semi-popular essays written from time to time over the past twenty five years. Others may discern thematic development. My own criterion for inclusion has mainly been that if I enjoyed writing the essay in the first place, and if now I enjoy re-reading it, then I put it in, and otherwise not. If it impels some of my readers to learn more of this new subject, then I am content. Man's cultural and intellectual environment in the 21st century may possibly be conditioned more by developments from this one field of enquiry than by any single pre-existing branch of science or technology. So portentous-sounding a statement deserves a solid basis, so I have included, in the last Chapter or two of each Section, various distillations which bear on the question.

Acknowledgements

'Trial and error' appeared in *Science Survey,* part 2, 129–45. Copyright Penguin Books Ltd, 1961. 'Puzzle-learning versus game-learning in studies of behaviour' appeared in *The Scientist Speculates* (ed. I. J. Good), 90–100. Heinemann 1962. 'Machines that play and plan' appeared in *Science Journal,* **4** (1968), 83–8. 'Computer—servant or master' appeared in *Theoria to Theory,* **2** (1968), 242–9 (then published by Pergamon Press). 'Future for Integrated Cognitive Systems' appeared in *Nature,* **228** (1970), 717–22. 'Tokyo-Edinburgh dialogue on robots in Artificial Intelligence research' appeared in *The Computer Journal,* **14** no. 1 (1971), 91–5. 'Artificial Intelligence' appeared in *New Society,* 26 August 1971, 370–3. 'Machine Intelligence at Edinburgh' appeared in *Management Informatics,* **2** (1973), 7–12. 'Machines and the theory of intelligence' appeared in *Nature,* **241** (1973), 507–12. 'Memory mechanisms and machine learning' appeared in *Simple Nervous Systems* (eds. B. R. Newth & P. N. R. Usherwood). Edward Arnold 1974. 'Knowledge engineering' appeared in *Kybernetes, International Journal of Cybernetics and General Systems,* **2** no. 4 (1973), 197–200. 'Machine Intelligence as technology' is abridged from a paper which appeared in *Proceedings of the Conference on Shop Floor Automation,* National Engineering Laboratory, East Kilbride, 11–13 December 1973. 'Machine models of perceptual and intellectural skills' appeared in *Scientific models and man* (ed. H. Harris) Oxford: Clarendon Press, 1979.

'Evluative comments in chess' appeared in *The Computer Journal,* **24** (1981), 278–286. 'High-road and low-road programs' appeared in the *AI Magazine,* **3** (1981), 21–22. 'Computer chess and the humanisation of technology' appeared in *Nature,* **299** (1982), 391–394. 'Measuring the knowledge-content of expert programs' appeared in the Bulletin of the *Institute of Mathematics and its Applications,* **18** (1982), 216–220. 'Automating the synthesis of expertise' appeared in *Aslib Proceedings,* **36** (1984), 337–343. 'Steps towards robot intelligence' appeared in the *Proceedings of the Royal Institution of Great Britain,* **87** (1985), 151–165. 'Towards a knowledge accelerator' appeared in *Advances in Computer Chess 4* (ed. D. Beal), Oxford: Pergamon Press, 1986.

The papers are reproduced by permission of the original publishers, to whom grateful acknowledgement is made.

Section 1 Computer game playing

INTRODUCTORY NOTE TO SECTION 1

The time-honoured tension between artist and patron is by no means to be explained by the follies or knaveries of the two parties, much as each would have you believe so of the other. The contradictions are intrinsic. Raising the stakes, as when 'research scientist' and 'institutional sponsor' are substituted for 'artist' and 'patron', only heightens the contradictions. It finally becomes a wonder when these partnerships advance at all.

So what are the problems? I believe that there are two. Being myself a scientist I can only expound them from a certain point of view, in which the sponsors are of course the villains. Equally seeing eyes, in the heads of others, will perceive the same two problems in terms destructive of the scientists' rather than of the sponsors' credit.

The first contradiction is that scientists prefer to be given the money first, so as to use it to do the work. To sponsors it is obvious that funding is a *reward*, which by its nature belongs after the event. Samuel Johnson's account of this phenomemon is apt:

> Is not a Patron, my Lord, one who looks with unconcern on a man struggling in the water, and, when he has reached ground, encumbers him with help? (letter to the Earl of Chesterfield, 1755).

There are no known formal solutions to this problem. An informal solution is to make a practice of handing in for this year's help the work completed by spending last year's. This depends, of course, on an initial 'float' which must come from somewhere.

The second contradiction surfaces after broad topics and goals have been agreed and materials and applications are being chosen. The scientist wants to choose with a view to the discoveries or demonstrations which he is after. The sponsor knows, however, that it is precisely the materials and applications which will be picked up by the technical and other media, to form the image which he is buying with his money.

How can the two agree? They cannot. Again, though, constructive guile may bridge the gap. With luck, good work can be carried on the back of a sufficient mass of other activity. In AI the matter comes to its sharpest focus in the computer emulation of game-playing skills, How would sponsors look if it were revealed in a Parliamentary or Congressional debate that taxpayers' money had been going on chess?

They would of course look bad. Not surprisingly, then, only an infinitesimal fraction of national AI budgets is available for what is by far the most searching proving-ground for experimental advance. Even this infinitesimal expenditure, though, can be consequential. The papers in this first Section, apart I hope from diverting the reader, can be used for assessing this claim. Let us preview them in turn, picking out points on which light was thrown.

'Trial and Error' was an archetype of what the knowledge engineering industry sees today as a design platitude: top-down decomposition into sub-problems, with a rule-structured solution for each individual sub-problem. This is the platitude, or in modern jargon the paradigm, of 'rule-based programming'. As a key move, the humble MENACE machine added a crude form of rule-learning shown viable for serious problems by the BOXES adaptive pole-balancer described in Chapter 3. A remote descendant of BOXES, supplied by the author's laboratory, is today keeping a Pittsburgh nuclear fuels factory in balance with estimated savings in excess of $10M per year.

The fuel-refining process is sufficiently puzzle-like as opposed to game-like in structure, to use the terminology of Chapter 2, that a deterministic form of rules-from-examples learning proved adequate. While re-reading this Chapter I recalled many an industrial problem where this was not so, and which cried out for a control automation capable of probabilistic inference. Chapter 2 poses the problem, using the animal psychologist's hard-worked experimental subjects to model it. Chapter 4 elaborates the same problem, using the chess-player as model, and introduces an operational test: can the automated controller in an uncertain world not only make good decisions but also evince some understanding of what is going on, in the form of evaluative comments? Leading up through applications to software technique in Chapter 5, Chapter 6 places the need for machine articulacy in a context of social urgency. Failure by either side of a man-machine partnership to form operational models of the other's decision-taking under uncertainty could seriously damage the planet's health.

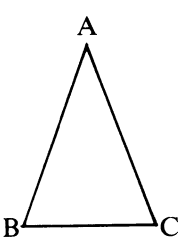
These partnerships today control power stations, military warning systems, air traffic and the like. On commission from the EEC's programme for Forecasting and Assessment of Science and Technology, Danny Kopec and I reported on the prevalent mismatch between machine representations and human concepts (FAST series no. 9, 1983) as a spreading emergency. Subsequent 'Star Wars' talk of military ventures in space has further sharpened that argument.

1

Trial and error (1961)

Can machines think? The short answer is 'Yes: there are machines which can do what we would call thinking, if it were done by a human being.'

Consider the well-known theorem of Euclid, which states that the two angles at the base of an isosceles triangle are equal to each other. Most of us can probably remember, or reconstruct, Euclid's own proof, which requires as construction that a straight line be drawn from the apex to the base. Can you devise an alternative proof *which requires no construction*? You may spend hours or days of painful thought on this and will probably not find a solution. As far as I know no human being has ever succeeded in doing so. But Marvin Minsky recently gave a computing machine a simple program for Euclidean geometry and it produced a new proof [1] which has the above-mentioned property: it is construction-free. It is also shorter and simpler than Euclid's, and has an additional quality which an impartial geometer might well describe as 'brilliance'. Here is the proof:



$$
\begin{aligned}
AB &= AC \text{ (given)} \\
AC &= AB \text{ (given)} \\
\angle BAC &= \angle CAB \\
\therefore \Delta ABC &= \Delta ACB \\
\therefore \angle ABC &= \angle ACB \quad \text{Q.E.D.}
\end{aligned}
$$

It is even possible to read this through a few times without getting the point, so daring is the ruse of treating triangles A B C and A C B as separate entities for the purposes of proof, but a single entity for the purposes of the conclusion.

If you or I had made this achievement, no one would grudge us the credit of having done some thinking: indeed, thinking of a rather spectacular quality. On the other hand a machine might conceivably arrive at the same result by some set of procedures quite different from those involved in human thought. From this point of view the use of the word 'thinking' could be as misleading as to say that a boat swims or that a porpoise sails. We might even decide to define 'thinking' to include the subjective experiences of the thinker; it would then follow automatically that insentient beings, which might be held to include machines, cannot think.

The argument is, of course, purely linguistic. Since boats have existed long enough for there to be a separate word for their motion through water, we are willing to say that they 'sail' rather than swim, and thus reap a gain in precision. Aeroplanes, on the other hand, are such recent innovations that we are content, for the time being, to say that they 'fly', although their method of doing so has little in common with that of birds, bats, or bees. We are in the same quandary with the even more recent development of complex computing machinery. It will therefore not be through perversity, but through need, if in describing mechanical processes I intermittently borrow words from the vocabulary of human or animal psychology.

A much more interesting objection is sometimes made to comparisons between human thought and mechanical reasoning. The objectors allege that a machine can 'in principle' perform calculations only by rote, that is, by following slavishly the train of thought dictated by a human master. It is often alleged that however fast and accurately a machine can perform the arithmetical or logical operations built or fed into it, it could never simulate the two most important components of human intellectual activity, namely (1) originality, and (2) the ability to learn. By learning I mean here the modification of behaviour, in the light of experience, in a 'purposive' or 'goal-seeking' fashion.

The geometrical proof which was cited earlier should be sufficient to dispose of the objection concerning originality. This chapter is devoted mainly to discussion of the second question, concerning the nature of learning and the possibility of simulating it mechanically.

THE MECHANICS OF LEARNING

There are two main reasons why a biologist like myself should be interested in learning machines. The first is that being a biologist he is (pending the development of mechanical biologists) also a man, and as such can expect to have his habitat transformed by the advent of such machines, possibly during his lifetime. The post-war development of electronic computers has already had a resounding impact upon science, industry, and military engineering. Yet most of the research effort has so far been limited to improving the speed and storage capacity of what are in essence no more than glorified desk calculating machines, or 'high-speed idiots' as they have been called. Attention is now turning to the development of machines which improve their own procedures as they go along, from machines which learn

to recognize and distinguish symbols, letters, or pictures, to machines which learn to play a passable game of draughts. The technical revolution which is brewing is not to be compared with such events as the transition from sailing-boats to steamers, for at some point a tearaway process is likely to get under way: learning machines will be harnessed to the job of designing better learning machines.

The second point of interest for biologists is more strictly professional. Will the design and investigation of learning machines throw light on the mechanisms of learning in the central nervous systems of man and animals? There is a way in which a model of a biological function can be illuminating, and a way in which it can offer a dangerous temptation. The temptation is to construct a device which performs a given bodily function, and then to exclaim: '*That* must be how the body does it!' No biologist in his senses would look at a modern aeroplane and conclude that birds, despite appearances, must work on a jet-propelled fixed-wing principle, but the temptation sometimes presents itself in more subtle guises. All that we have a right to expect from a model is that it may deepen our understanding of the matrix of physical laws within which both the model and the biological system have to work. In this sense the study of aeroplane flight *can* advance our understanding of animal flight, not directly, but by elucidating aerodynamic laws to which flying animals are also subject.

During the coming decades the machine-builders will be forced to analyse in increasing depth and detail the logical and mathematical structure of learning processes. The biologist will be able to use the results of these analyses to sharpen his investigation of living nervous systems, which quite possibly operate through entirely different mechanisms. At the same time, whenever a learning machine exhibits a striking parallel with human or animal behaviour, the biologist should be on the alert: it *may* be a clue to a biological mechanism.

This last point is part of my justification for the construction of the simple learning machine which I shall later describe. The starting-point was to divide certain forms of trial-and-error learning into two components: one which is difficult to simulate, and was therefore evaded, and one which is easy. The two components may be termed *classification of the stimulus* and *reinforcement of the response*. Classification of the stimulus is essential to any form for learning, for if you cannot classify a situation as similar to one previously encountered, how can you profit by your past encounters? If Mr A raises his fist at Mr B, the latter is faced with a situation which he has never met before in exactly that form. Even if Mr A has frequently menaced him in such a fashion, even wearing the same clothes with an identical posture and facial expression, he has never before produced precisely the same pattern of stimulation on Mr A's retina, owing to differences in lighting, background, position in Mr B's field of view, and so on. Yet Mr B 'instinctively' raises his arm to ward off the blow. Actually instinct is precisely what is *not* involved. Mr B has learnt the response from the many occasions, probably in his boyhood, when a raised fist was followed by a blow.

The problems posed by such a feat of classification are quite extraordi-

narily complicated. It is the central problem facing those who are developing machines to read a printed text—a highly desirable accomplishment for the translating machines of the future, which will otherwise have to be spoon-fed with texts laboriously punched on to teleprint tape by human typists. The fact that it is difficult enough even to make a machine spot that 'O' is the same letter as '*o*', underlines the magnitude of the problem.

The second problem, reinforcement of the response, is much more tractable. The response leads to an outcome (for example, Mr B is either struck or not struck) which produces sensations in the responder which are to some degree agreeable or disagreeable. The outcome can thus be said to have a *value* which expresses in numerical terms the degree of pleasure or displeasure associated with it. The probability of the person responding in the same way when the 'same' stimulus is presented later depends on the value of the last outcome of this response. If it has a positive value, the probability is increased. If it has a negative value, the probability is decreased, and the probabilities of alternative responses (if inaction is included as a form of 'response') are accordingly raised. The word 'reinforcement' will be used for the change of probability, with the understanding that a decrease in probability represents a *negative* reinforcement.

THE MATCHBOX MODEL

We now have a conceptual blueprint for devising a simple learning machine, provided that the problem of classification can be side-stepped. For this, the number of discrete situations encountered in the task which the machine is to learn must be sufficiently small for them all to be separately enumerated.

The task which I wish to consider from this point of view is that of learning to play the game of noughts and crosses, known in America as tic-tac-toe, but apparently unknown on the continent of Europe.

It would be easy to devise a machine which would play impeccable noughts and crosses from the outset, but that is not the point. The point is to construct a machine which starts with no prior information about how to play, apart from the rules, but which will become an effective player through practice. Such a machine would embark on its career making its moves entirely at random, and end as an acknowledged expert.

An extremely simple machine of this sort is shown in Fig. 1.1. It was made by glueing some three hundred empty matchboxes together so as to form a chest-of-drawers, and placing different numbers of variously coloured small glass beads in the various boxes. In addition, each box has a V-shaped cardboard fence fixed in the front, so that when the box is tilted forward, one of the contained beads is selected by chance through being the first to roll into the apex.

This machine is always allowed the opening move. For each of the three hundred or so distinct positions with which Nought (by convention the opening player) can be confronted, there is a corresponding box bearing on its front a drawing of the position, together with a code number for ease of reference. All three hundred boxes can thus be arranged in numerical order

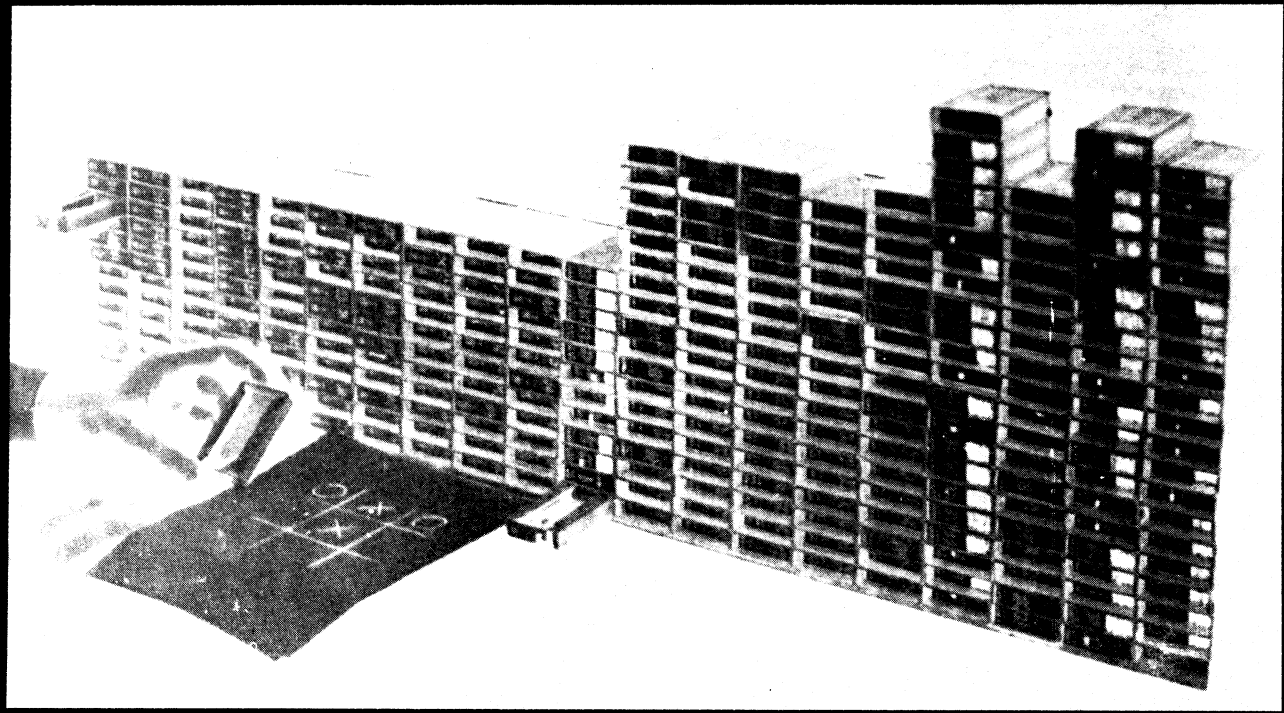
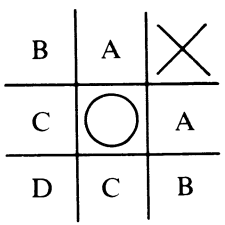


Fig. 1.1 — The original matchbox version of MENACE

in the chest-of-drawers: this has not in fact been done entirely consistently in the model shown in the photograph, and its castellated appearance is an unnecessary refinement thought at one stage to facilitate indexing.

Each box contains a number of beads of assorted colours: there are nine colours, corresponding to the nine squares of the board, and the selection of a bead of a particular colour signifies a move to be made to the corresponding square. A given box contains only beads of colours corresponding to unoccupied squares, so that only legal moves can be made. Knowledge of the rule defining legal moves is thus 'built-in' to the machine. If it were not, the machine would simply learn the rule as it went along, but this would complicate the problem unnecessarily. Moves which, owing to symmetry of the position, are equivalent to each other are not separately represented. For example, the figure below represents a position which is symmetrical about one diagonal. It is Nought's turn to play, and at first sight he appears to have seven alternative moves, as there are seven squares unoccupied. But the symmetry of the position makes the two squares labelled A equivalent to one another, also the two labelled B, also the two labelled C. So a choice need only be made between four alternatives, A, B, C, and D. Similarly there are only three essentially distinct opening moves (corner, side, and centre squares). The first box of the matchbox model therefore contains beads of three colours only.



Suppose we wish to play against the machine. We ascertain its opening move by taking out the first box, shaking it so as to randomize the positions of the beads in it, and tilting it forwards so that the beads run to the front. If the colour of the bead arriving in the apex of the cardboard fence is, say, pink, we place a nought on the machine's behalf in the centre (square 4). We now replace the box in the chest-of-drawers, but for applying the reinforcements at the end of the play (a 'play' is the complete series of moves leading from the initial state—all squares empty—to an outcome—win, draw, or lose) it is convenient to leave the drawer open. We reply with a cross to, say, the top left-hand square (square 1). The position is now 51 in the code which was adopted, and we must take out, shake, and tilt the box with this code number in order to ascertain the machine's next move; and so on until the end of the play.

We now apply the 'reinforcements'. If the machine has lost, we confiscate the apical bead from each of the three or four boxes which have been

left open, thus making it less probable in each case that the same move will be repeated when the same position recurs in the course of future play. If the machine has done well (when it is playing against an expert, we call it 'doing well' if it draws) each open box is given a bonus of one bead of the same colour as the apical bead, thus encouraging repetition of the move concerned. The open drawers are pushed shut and the machine is now ready for the next play.

A little reflection should convince the reader that such a machine cannot help improving its standard of play, after a fashion and to some degree. But we have as yet said nothing about how many times each colour is replicated in the various boxes. This matter is of great importance since it determines the rate at which the probabilities are changed by the system of unit forfeits and bonuses. With so crude a mechanical contrivance we cannot hope to make its reinforcement system fully rational, and indeed the reinforcement problem, as applied even to a much more simple system than the machine under discussion, remains unsolved by mathematicians. A reasonably workable system was arrived at in the present case along the following lines. It is clear that if the machine's fourth move (stage 7 of the game) is followed by defeat, it is a bad move without qualification, and there is no point in its ever being repeated. Hence the boxes at stage 7 should have only one replicate of each legal move, so that the confiscation of one bead expunges the offending move for ever. It is equally clear that a defeat should be regarded as a black mark against the move which was made at stage 5, but the evidence is not so strong as against the stage 7 move. In like manner the weight of evidence from a given outcome must continue to lessen as we approach the beginning of the game. But even an opening move should receive *some* reinforcement in the light of the ultimate outcome.

For the trial run, the simplest possible system was adopted, as follows:

stage	machine's move	number of replicates
1	1st	4
3	2nd	3
5	3rd	2
7	4th	1

It turned out that the allotment of only twelve beads to the first box (three legal moves quadruplicated equals twelve) gave the machine scarcely sufficient resources to withstand repeated discouragements in the early stages of play against an expert. On more than one occasion the first box nearly ran out of beads: if it had actually done so, we should have understood the machine to be refusing to play. It sometimes happened that a box at later stages became empty, but this was as it should be: it is reasonable to resign in a hopelessly lost position. But at all events this reinforcement

system in all its crudity was sufficient to make a surprisingly good showing when the time came for the machine to be challenged by its inventor. This we shall now see.

MAN VERSUS MENACE

For its maiden tournament the machine, which was given the name MENACE (Matchbox Educable Noughts And Crosses Engine), was set up as already described. The forfeit for a defeat was confiscation of the apical bead from each open box. The reward for a draw was the addition of one bead of the same colour as the apical bead. Against best strategy, as made clear by D. W. Davies in a recent article [2], it is impossible to win at noughts and crosses. One might therefore think, assuming that its human opponent would adopt best strategy, that the question of rewarding MENACE for victories would not arise. But in practice the machine quickly found a safe drawing line of play against best strategy, so that its human opponent had to resort to unsound variations, risking machine victories in the hope of trapping it into a more than compensating number of defeats. This possibility had been foreseen (although not the speed with which it matured) and the bonus for a *win* was fixed at three beads added to each open box. The bonuses and forfeits can be regarded as equivalent to the value of the outcome, if we take a defeat to have the value -1, a draw $+1$, and a win $+3$.

The tournament lasted for 220 plays of the game, occupying two eight-hour sessions on successive days. By the end of the first twenty plays the machine was settling into a stereotyped line which ensured a draw in face of 'best strategy'. I therefore resorted to a series of theoretically unsound variations, in order to draw the machine into unfamiliar territory. Each of these paid off for a time, but after 150 plays the machine had become capable of coping with anything, in the sense that whatever variations I employed I could not get a better average result against it than a draw. In fact after this point I did much worse than this, by unwisely continuing to manoeuvre in various ways. The machine was by then exploiting unsound variations with increasing acumen, so that I would have done better to return to 'best strategy' and put up with an endless series of draws, or retire from the tournament. This I eventually did after sustaining eight defeats in ten successive games. At every stage, I used what tactics I judged to be the most hopeful. It is likely, however, that my judgement was sometimes impaired by fatigue.

The progress of the tournament is shown graphically in Fig. 1.2. The line of dots gives a complete representation of the outcomes throughout the tournament: the line jumps one level down for each losing outcome, one level up for each draw, and three levels up for each win. The angle of the line with the horizontal at a given stage of the tournament shows how well the machine was doing at that stage. An upward slope of 45° corresponds to an average drawing result, which is the best that the machine can do against 'best strategy'. This is exemplified by the 'testing run' of twenty plays which was made at the end of the tournament (see Fig. 1.2).

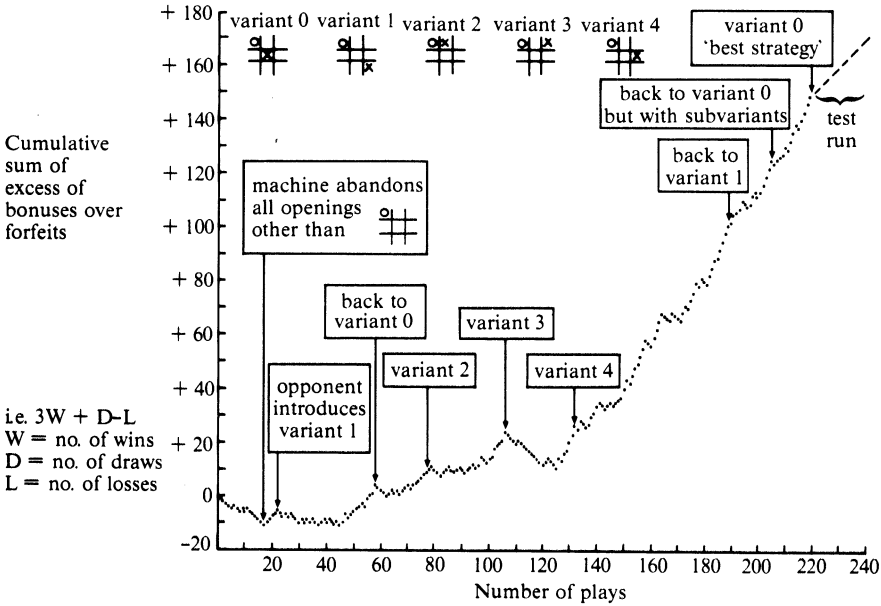


Fig. 1.2 — Performance of the MENACE learning machine in its first noughts and crosses tournament

We shall now turn to the consideration of what parallels may possibly exist between the mechanical model and phenomena encountered in biology.

PARALLELS WITH ANIMAL LEARNING

We shall adopt W. H. Thorpe's [3] division of animal learning into five main categories: habituation, conditioning (or the formation of conditional reflexes), trial-and-error learning, insight learning, and imprinting. In all but the simplest acts of learning, an animal will of course combine several of these mechanisms.

Habituation describes the 'Wolf! Wolf!' situation. A sight or sound or other stimulus arouses our attention the first time we meet it; but on repetition we get used to it and take no notice. This happens only if it is not accompanied by any happening significant for us, such as the offer of something we want, or the infliction of pain. In the terminology which we have used earlier, the original response leads to an outcome of zero value (noxious outcomes have negative values). If the matter is put in these terms, MENACE clearly does not show habituation, if only because the reinforcement system does not allow zero outcome values.

Conditioning. In a conditional reflex the most prominent feature is that a

response such as blinking or secreting saliva comes to be evoked by a stimulus, such as a sound, which did not evoke it at first. The animal becomes 'conditioned' to respond in a particular way. This has no parallel in the workings of MENACE, although it is well simulated by some other learning machines, such as that designed by A. M. Uttley [4].

Trial-and-error learning is defined by Pringle [5] as follows: 'Essentially, the animal makes more-or-less random movements and selects, in the sense that it subsequently repeats, those which produced the "desired" result . . .' This description seems tailor-made for the matchbox model. Indeed, MENACE constitutes a model of trial-and-error learning in so pure a form, that when it shows elements of other categories of learning we may reasonably suspect these of contamination with a trial-and-error component. To illustrate this point, it is convenient to take next the last category listed by Thorpe, namely imprinting.

Imprinting has chiefly been described and studied in certain bird species. A single experience at a critical phase of the animal's development may result in permanent modifications in its behaviour. A famous case is that of the greylag goslings studied by Konrad Lorenz. The hatchlings 'unquestioningly accept the first living being whom they meet as their mother, and run confidently after him'. One is reminded of Oberon's threat concerning Titania in *A Midsummer Night's Dream*:

> 'The next thing then she waking looks upon,
> Be it on lion, bear, or wolf, or bull,
> On meddling monkey or on busy ape,
> She shall pursue it with the soul of love.'

In analogous fashion Lorenz contrived to gain the devotion of newly-hatched goslings.

Imprinting is not quite as specific, or as long-lasting, as this description suggests, but there is no doubt of the existence of the phenomenon. At first sight it has nothing in common with trial-and-error learning. But consider what would happen if, at some stage in the tournament, MENACE had been given an over-riding reinforcement of the response to some stimulus, in the shape of an arbitrarily large bonus of beads for a move made in some particular position. From then on, whenever it encountered the same position again, the machine would almost inevitably produce the same move, just as the ducklings could not help following Lorenz. Is it possible that imprinting operates through a mechanism analogous to that of trial-and-error learning, differing only in that the reinforcement is very large?

If so, we must ask what changes occurring during imprinting could result in an overriding reinforcement. One possibility is that the process is associated with the relief of acute anxiety. Support for this idea would seem to be lent by the fact that imprinting in hatchlings does not occur, or occurs less strongly, if they are pre-medicated with tranquillizing drugs. There are

also suggestive resemblances between imprinting and some types of human neurotic behaviour.

Insight learning, Thorpe's remaining category, is perhaps the most interesting and the most inaccessible to analysis. I can best illustrate it by describing the results of testing human subjects under exactly the same conditions as those under which MENACE was tested in the tournament described earlier. All three subjects were Continental scientists with no previous knowledge of the game of noughts and crosses. For each test the subject was informed of the rules of the game, but not of its object. I then invited him or her to a series of plays of the game, the subject having the first move of each play. At the end of each play I announced 'You have lost' or 'It is a draw', but gave no other information. The results in the three cases were closely similar. After three or four plays the subject spotted the point of the game, and after five or six plays had adopted best strategy.

Whatever else it involves, insight learning must certainly depend upon highly sophisticated feats of *stimulus-classification*, and also upon sustained flights of *logical deduction.* It seems likely that it also involves a process of *silent* trial-and-error: imaginary stimuli are self-administered, trial responses are performed in fantasy, their outcomes are envisaged, and the appropriate reinforcements are applied internally. The idea can be illustrated by a description of how a trial-and-error machine could simulate insight learning.

With the aid of Mr John Martin of Ferranti Ltd, the matchbox model has been programmed for a digital computer. The advantages, apart from a thousandfold increase in speed of operation, are that both sides of the game can be mechanized (so that a mechanical Nought plays against a mechanical Cross), and either side can be set to play as an expert, as a random player, as any sort of hybrid between the two, or as a self-improving player with any desired degree of previous experience of play. In this way the properties and merits of an unlimited variety of reinforcement functions can be tested over a wide range of fully specified conditions. Could 'insight learning' be exhibited by the electronic version of MENACE?

During play the computer produces a printed record, the printout, of every move made. In fact, the printing of symbols on paper constitutes the only *overt behaviour* of the machine. If nothing is being printed, then there is no play in progress, according to the criteria which we apply to human players. Let us now imagine that we set the Cross side of the program to make its moves at random (excluding illegal moves), and let Nought run as a mechanized learner as was done for the test of the matchbox model. We now start the program running, *but omit to switch on the printout mechanism.* What happens? Nought proceeds to run off a series of trial plays 'in his head', without making any marks on paper, but, just as though he were engaged in actual play, the outcomes of these phantom plays leave their mark on his strategic habits through the reinforcement system. We might then interrupt the machine's reverie and, leaving the Nought side undisturbed, set the Cross side to play at some fixed level of skill (for example,

expert). When now tested (with the printout switched on of course), Nought will come out with a standard of play which is fair, good, or excellent, depending on how long a period was previously allowed for him to gain 'insight' through silent meditation.

If any parallel at all exists with the behaviour of the aforementioned Continental scientists, it is with the second phase of their behaviour, during the rapid transition from knowledge of the values of different outcomes to adoption of best strategy. It seems likely that part of what was going on in their minds consisted of trial sequences of the form: 'If I do this, then he may do that, to which I might reply thus, and so on.' But it is damaging to the case which we are here considering to put much weight on any alleged parallel with human insight learning, since the part played by thought processes other than those of trial-and-error is so preponderant in our species. When we consider insight learning as it is found in lower animals, the example afforded by the behaviour of the computer with its printout switched off seems less obviously objectionable. Defining insight learning Pringle remarks that '. . . the animal in this case appears to be able to work out the consequences of a number of possible alternative responses to sensory stimuli without actually performing them, and then, on the addition of a further stimulus, to perform only the one which leads to a favourable result'.

LEARNING AS AN EVOLUTIONARY PROCESS

An evolutionary process is usually thought of as characterized by such features as increase of complexity, and increase of orderliness. On such a definition, learning is just as much an evolutionary process as is organic evolution, that is the evolution of living organisms. It is therefore natural to wonder whether the mechanisms by which the two processes operate have anything in common; whether, in particular, as Pringle has suggested, learning can profitably be studied with the concepts of Darwinian theory.

It is quite possible to think of the matchbox model as a Darwinian system. The reader can divert himself by thinking of the boxes as discrete habitats through which is dispersed a species of organism which takes the form of glass beads. Each habitat is occupied by several varieties, one of which, by reason of greater fitness to that particular habitat, usually displaces the other and becomes the local type. Of more than trivial interest is the fact that equally Darwinian systems are found incorporated in other learning machines under current development. Thus, O. G. Selfridge's [6] machine Pandemonium, which has the task of learning to read Morse Code as tapped out by human operators and converting it into typewritten English, operates through a hierarchy of demons and subdemons (a demon is a computing operation): subdemons which prove unworthy are eliminated by the program, and their place is taken by new subdemons procreated by the worthy survivors. The essence of subdemon selection had already been foreshadowed in A. L. Samuel's machine which learns to play checkers

(*anglice*: draughts) [7]. Whether or not Samuel's work influenced Selfridge, it emphasizes how direct can be the road leading from games-playing problems to applications of practical importance.

ACKNOWLEDGEMENT

I am indebted to Dr G. H. Begbie for reawakening my interest in the problem, and for drawing my attention to most of the references which I consulted while preparing this article.

NOTES AND REFERENCES

[1] Such a proof was in fact obtained by the Greek geometer Pappus, but was later forgotten (author's note, 1986).

[2] D. W. Davies (1950) A theory of chess and noughts and crosses. *Science News,* **16**, 40–64. This article on elementary game theory should certainly be consulted by anyone interested in the subject.

[3] W. H. Thorpe (1950) The concepts of learning and their relation to those of instinct. *Symposium of the Society for Experimental Biology,* **4**, 387–408.

[4] A. M. Uttley (1956) A theory of the mechanism of learning based on the computation of conditional probabilities. *Proc. First Intern. Congress of Cybernetics,* Namur, 830–56.

[5] J. W. S. Pringle (1951) On the parallel between learning and evolution. *Behaviour,* **3**, 174–215. Some parts of this paper on biological learning systems are comprehensible to the non-specialist, and worth a try; but much of it is very technical.

[6] O. G. Selfridge (1959) Pandemonium: a paradigm for learning. *National Physical Laboratory Symposium no. 10, Mechanization of Thought Processes*, 511–31. London: HMSO. A racy account, at a semi-technical level, of the morse-reading learning machine discussed in the last section of the present article.

[7] A. L. Samuel (1955) Some studies in machine learning, using the game of checkers. *IBM Journal,* **3**, 210–29. A technical account of Samuel's machine that learns to play checkers (draughts).

See also:

U. Neisser & O. G. Selfridge (1960) Pattern recognition by machine. *Scientific American,* **203**, 60–00. A good and readable popular essay which deals with the stimulus-classification problem, and hence is complementary to the present article.

C. E. Shannon (1955) Games-playing machines. *J. Franklin Institute,* **260**, 447–54. An entertaining lecture on games-playing machines.

2

Puzzle-learning versus game-learning in studies of behaviour (1962)

This chapter is concerned with an elementary distinction which, it will be argued, is of crucial importance in the study of animal and human learning. The distinction is that between a one-person game or 'puzzle', and a two-person game[1].

In a puzzle, as the term is here used, the player makes a succession of moves, each of which results in a changed state of the apparatus. After each move the rules tell the player whether the puzzle has now been solved or not. They may tell him that there are no more legal moves available to him, even though the puzzle has still not been solved, in which case he must restore the apparatus to its initial state and start again. In this event the value assigned to the outcome of his attempt can be conventionally described as a minus quantity, say -1. The outcome of a successful attempt may be given the value $+1$. One could imagine a more complex case where different valid solutions of the puzzle were graded (for example according to economy in the number of moves) on a quantitative scale, having values $+1$, $+2$, $+3$, etc, and that unsuccessful outcomes might be assigned similar gradations on a scale of negative numbers. We shall, however, take the simpler two-valued case for ease of exposition.

The distinguishing feature of a puzzle, as opposed to a game, is that the change effected in the apparatus by a given move is fully determined. In a game, on the other hand, the player has to take into account not one single necessary consequence of his move, but a range of alternative possible consequences, a selection from which will be made by his opponent's move or moves before it is again his turn to move.

A game, it will be contended, summarizes the essential features of most of the situations with which animals are confronted in real life, whereas most of the problems given to animals in conventional studies of learning are 'puzzles'. I shall attempt to show that mechanisms of learning which give high efficiency for games are ill-suited to puzzles, and vice versa.

Consequently it is possible to arrive at misleading conclusions if one presents puzzles, as is the custom of experimental psychologists, to central nervous systems which have been adapted by natural selection to cope with games.

ANIMAL–ENVIRONMENT INTERACTION AS A TWO-PERSON GAME

In the language of games the two 'persons' are the animal and its environment. Moves are made alternately. The environment's moves are called 'stimuli' and the animal's moves are called 'responses'. The outcome of the game is assessed after some number of moves. The object, from the animal's point of view, is to secure an outcome of the highest positive value, outcomes being valued according as they contribute to the satisfaction of the animal's needs, or, on the negative side of the balance sheet, to the causation of discomfort, pain, etc. It is not, of course, supposed in any literal sense that the *environment's* play is guided by any object, although in special circumstances it may be, as when the effective environment consists of a human being or another animal. On the other hand, it will necessarily be subject, as are the animal's responses, to the 'laws of nature', which correspond to the rules of the game. They determine what alternative moves are possible (legal) in a given situation. Additional restrictions on the animal's moves are imposed by the limited range of responses to a given stimulus allowed by its innate behaviour patterns. Purring is not a possible move for a cat to make on receipt of a painful blow.

The essence of trial-and-error learning consists in whittling down the range of innate potential responses to a small number which become habitual. The process of selection in the light of experience can be likened to that which transforms a beginner at chess, who may (as Black) make any one of the twenty legal replies to 'White 1. P–K4...', into an expert who would not seriously consider more than at most eight, and who probably uses only three or four of these habitually. This analogy should not be pressed further than its simple illustrative purpose demands, since human learning of a game like chess makes heavy use of 'insight learning' in addition to 'trial-and-error'; this chapter is concerned only with the latter category.

Before examining the main thesis: that optimal mechanisms of trial-and-error learning are fundamentally different according to whether the task constitutes a puzzle or a game, it remains briefly to substantiate the claim that most real-life situations are games rather than puzzles. This can be seen to be true as soon as we recognize that at any point in time an animal is responding *not* to the total actual situation (which it is in no position fully to assess) *but to the total stimulus-situation.* Thus, the sight at noon on Monday of a pine-tree from a westerly point at five metres distance constitutes the same stimulus-situation as the sight of the same tree from the same vantage-point at noon on Tuesday, assuming reasonable constancy of climatic and other conditions. This is so even though every pine-needle of the tree has meanwhile moved. But the actual situation underlying the same stimulus-situation on the two occasions may be very different. A mountain

lion may be in the tree on Tuesday which was not there on Monday, with the consequence that a response which leads to a favourable outcome on one occasion (e.g. using the tree for shade) may have a disastrous outcome on another occasion. This multiplicity of actual situations underlying a single stimulus-situation, and the resulting multiplicity of consequences attendant upon a given response, is a sufficient criterion of game-like rather than puzzle-like structure. The animal's ignorance of the actual situation which underlies the current stimulus-situation corresponds to the predicament of the chess-player, who is inevitably ignorant of his opponent's strategy although fully aware of the current position on the board to which that strategy has contributed.

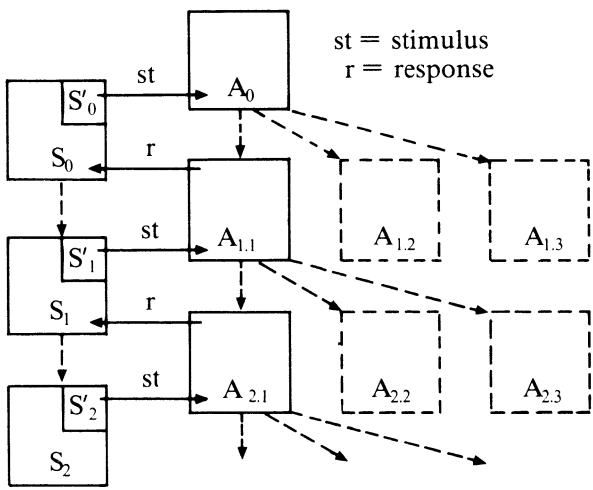


Fig. 2.1 — The interaction between animal and environment represented in terms of a puzzle. S_0, S_1, S_2 denote successive states of the environment or 'actual situation' with the corresponding 'stimulus situation' denoted by S'_0, S'_1 and S'_2. The sequence of states of the animal A_0, $A_{1.1}$, $A_{2.1}$, shows the path actually taken, with alternative choices leading to potential states $A_{1.2}$, $A_{1.3}$and $A_{2.2}$, $A_{2.3}$, etc.

To summarize the ideas outlined above, we present two diagrams. Fig. 2.1 depicts a sequence of choices made by an animal confronted with a puzzle. In this case (unlike that of a game) the stimulus-situation contains all relevant features of the actual situation. In Fig. 2.2 the diagram reproduced in Fig. 2.1 is reproduced with a new feature added, which converts the puzzle into a game. The new feature is the existence of alternative *potential* transitions of the environment of which the animal must take account. These transitions are entirely compatible with the stimulus-situation, although not with the actual situation of which the animal is necessarily unaware.

We shall now consider the kinds of learning behaviour which would be appropriate to these two very different kinds of problem.

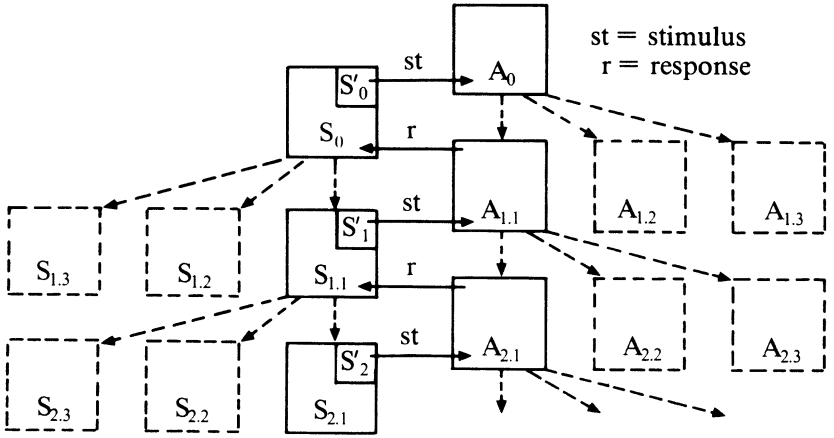
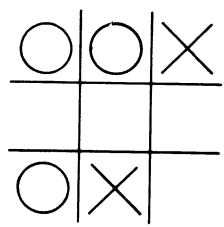


Fig. 2.2 — The interaction between animal and environment represented in terms of a game. The new feature as compared with Fig. 2.1, is that a given response, say that arising from $A_{1.1}$, may trigger off any one of several alternative changes in the environment for example those leading to states $S_{1.1}$, $S_{1.2}$, or $S_{1.3}$. These paths $S_0 \rightarrow S_{1.1} \rightarrow S_{2.1}$, and $A_0 \rightarrow A_{1.1} \rightarrow A_{2.1}$ represent those taken by environment and animal respectively on a given occasion.

LEARNING TO PLAY A GAME VERSUS LEARNING TO SOLVE A PUZZLE

Within the strict context of trial-and-error learning there is no alternative, when faced with a new game or a new puzzle, but to embark on a series of randomly-chosen moves. Sooner or later the series of moves will terminate in an outcome, favourable or unfavourable; this is where learning begins. It is obvious that the probability of the terminal move, the next time that the same position is encountered, must be modified in some way. More specifically, if the outcome-value was negative the probability must be reduced, and if it was positive, it must be increased. But by how much? We here come upon the first important contrast between puzzle-learning and game-learning. It can be seen at once that, in a puzzle, the probability change, or 'reinforcement', should be maximal. That is to say, if the move has immediately led to a solution of the puzzle (outcome-value $= +1$), then the probability of repetition should be adjusted to zero.

This is only true of a *game* where the outcome immediately follows the last move, without an intervening move by the opponent. When a delay, occupied by the opponent's move, precedes the outcome the state of affairs is entirely different. Consider the following example from the game of noughts and crosses (otherwise known as tic-tac-toe):



On the first occasion when this position arises Cross places his move in the lower right-hand corner, and is defeated when his opponent promptly completes the left-hand column of noughts. In a world of puzzles this would be sufficient for Cross to 'learn his lesson' and never again repeat the fatal move. But in a world of games this move *may* be the one which in the long run gives the best results. One unfavourable result is not sufficient to excludethe remote possibility that Nought has a 'blind spot' which causes him on most, but not all, occasions to reply to the centre square and thus to allow Cross to win. Evidently Cross should be chary of repeating the losing move, but should not discard it completely until further evidence on its actuarial risk has accumulated: the probability of this particular response *should* be reduced, but not to zero. This principle does not hold (an artificial case from the real-life point of view) when the opponent's play is guided by 'best strategy'.

In real-life situations the outcome even of a terminal move is frequently indeterminate, as in the case of Cross's move in the above example. All that can be attached to it in the light of the animal's accumulating experience is an increasingly well-charted frequency-distribution estimating the relative probabilities of the various possible outcomes. Yet this is *not* true of the laboratory conditions under which learning behaviour is commonly tested. In the typical and simplest case, the animal is rewarded if it turns left and punished if it turns right, and this rigid connexion between move and outcome-value is held invariant throughout the experiment. The animal, however, is not to know this. If, therefore, it requires a substantial number of trials before settling decisively for the left rather than the right turn, its sluggishness should not be imputed to imperfect learning powers: it may merely indicate that the animal has a better grasp than has the experimenter on the realities of its own daily life.

The second major contrast between game-learning and puzzle-learning concerns the relation between temporal sequence and the strength of reinforcement. In formulating his classical Law of Effect, Thorndike [2] drew attention to 'the effect of increasing the interval between the response and the satisfaction or discomfort', namely a diminution of the amount by which the probability of response is modified in the light of the outcome.

In terms of game-learning we interpret 'interval' as meaning the number of further moves intervening before an outcome is reached. An efficient game-learning mechanism should modify the probability not only of the move immediately preceding the outcome, but, in diminishing degree, also that of the penultimate move, the antepenultimate move, and so on. This principle has been utilized in constructing a simple machine which 'learns' to play noughts and crosses[3]. Even though the desirability of applying non-zero reinforcement to pre-terminal moves may seem obvious, we have to ask ourselves what is it *precise* justification. As a first approach we can frame our answer in rather loose language: the *rationale* of discouraging earlier moves which have led to one particular unfavourable final outcome (or encouraging those which have led to a particular favourable outcome) is that we take the outcome-value as evidence that a given earlier move was bad (or good)

in some more general sense than that it happened to lead to this particular result in this particular case. More rigorously, we must consider the formal representation of a game or puzzle as a branching tree. The principle of 'guilt by association' to which we have given a loose expression above can be expressed by saying that twigs of the same branch tend to bear similar fruit: that is to say, the total variation in outcome-values, instead of being distributed at random over the terminal spots, shows a trend towards homogeneity within families of spots, and heterogeneity between families.

The principle is well known to game-players, and is a commonplace of real life. But is it also true of puzzles? Doubtless it is true of many puzzles, but it can easily be seen that it need not be true of any given puzzle, and that there is no reason at all why it should be true of the particular puzzles which experimental psychologists devise for their animals.

In the first place, there may be no opportunity for the principle to operate owing to insufficient variation of outcome values. This is so in a maze in which only one terminal spot contains a reward, the remainder carrying punishments. A simple maze which does not allow re-tracing is shown in Fig. 2.3. The terminal spot containing the reward is boxed in the

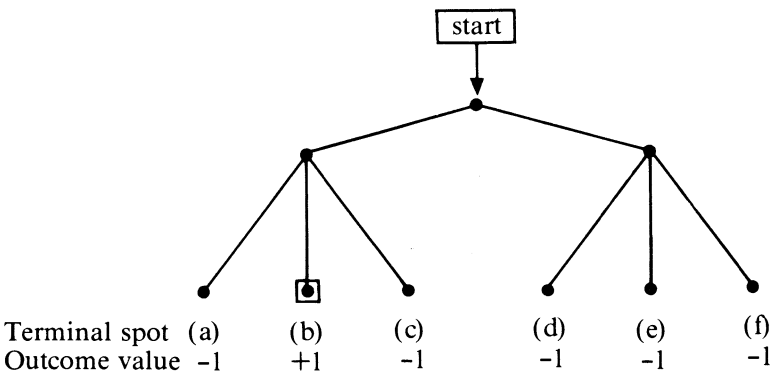


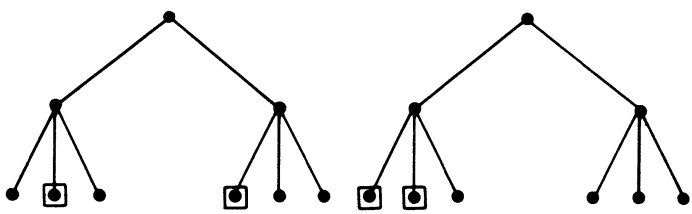
Fig. 2.3 — A simple maze without retracing, drawn and labelled in such a way as to exhibit the formal structure of a puzzle.

diagram.

The three pre-terminal spots represent choice-points, of which the first can be termed 'primary' and the othe two 'secondary'. Since this is a puzzle and not a game, the most efficient learning procedure will, as we have seen in an earlier section, discard immediately and irrevocably any secondary choice which has once been followed by a negative outcome. A real animal, as we have also seen earlier, will not do this because, in our submission, it is adapted to game-learning rather than puzzle-learning. It will also display another behavioural feature irrelevant to puzzle-learning, namely a modification of *primary* choice consequent upon a negative outcome. If its first run of the maze took it to terminal spot (a) where it received a punishment, it will tend on the next occasion to make a left rather than a right turn at the

primary choice-point. This spread of reinforcement to a preterminal move is an adaptive mechanism in game-learning, owing to family likeness of terminal spots. Yet in the puzzle under consideration the average number of trials needed for solution of this puzzle (given maximal reinforcement of secondary choices) is exactly 3.5, and this expectation is completely unaffected by any spread of reinforcement to the primary choice.

A different version of the same maze might contain two reward-boxes, and these could be disposed in two essentially different ways, as shown in Fig. 2.4. Here it is less obvious that spread of reinforcement is ineffective in contributing to learning-speed. Everything, in fact, depends on whether the puzzle belongs to type A or type B. For type B, which exemplifies the 'family likeness' of outcome characteristic of games, a negative outcome should indeed result in a negative reinforcement of the primary choice: if twigs of the same branch tend to bear similar fruit, it is better, after a disappointment, to try another branch! But for type A, a negative outcome should result in a *positive* reinforcement if a maximum efficiency is required. This is because having eliminated one negative outcome in the family, we except a corresponding higher proportion of positives in the surviving members of the same family. In such a case a fundamental feature of learning which forms a normal and necessary part of animal behaviour would not only be useless to the animal, but would be actively harmful and serve only to lead it into trouble.



Type A
(homogeneity between families)

Type B
(heterogeneity between families)

Fig. 2.4 — Simplified representation of two contrasting types of puzzle, one in which twigs of the same branch tend to bear similar fruit (type B) and one in which fruits are dispersed evenly over branches. For more general application of this distinction we would have to consider sub-branches, sub-sub-branches, etc., before arriving at the twigs, or 'terminal spots'.

The main ideas that have been advanced can be summarized as follows:

(1) Real life has the structure of a game rather than of a puzzle.
(2) Efficient game-learning by trial-and-error requires two fundamental features in the reinforcement system: (a) partial rather than absolute reinforcement of the terminal move, and (b) spread of reinforcement to preterminal moves.
(3) Both these features are exemplified by trial-and-error learning in animals.

(4) By testing experimental animals with puzzles rather than games these features can be nullified and may even unwittingly be turned to the discredit of the animal's estimated power of learning.

REFERENCES

[1] D. W. Davies (1950) A theory of chess and noughts and crosses. *Science News,* **16**, 40–64.

[2] E. L. Thorndike (1911) *Animal Intelligence,* 241–50. New York: The Macmillan Company.

[3] D. Michie (1961) Trial and error. *Penguin Science Survey,* part 2, 129–44.

3

Machines that play and plan (1968)

Proposals to construct man-like machines are nothing new. The following particularly charming excerpt from the *Scotsman* newspaper of 100 years ago recently came to my attention:

> A STEAM MAN — The 'Newark Advertiser' (New Jersey) describes the very extraordinary invention of a machine which, moved by steam, will perform some of the most important functions of humanity — stand upright, walk or run, as he is bid, in any direction, and at almost any rate of speed, drawing after him a load whose weight would tax the strength of three stout draught horses. In order to prevent the 'giant' from frightening horses by its wonderful appearance the inventor intends to clothe it and give it as nearly as possible a likeness to the rest of humanity. The boilers and such parts as are necessarily heated will be encased in felt or woollen garments. Pantaloons, coat and vest, of the latest styles, are provided. Whenever the fires need coaling, which is every two or three hours, the driver stops the machine, descends from his seat, unbuttons 'Damel's' vest, opens a door, shovels in the fuel, buttons up the vest, and drives on.

Here the attempt is dominated by the ideas of motion and force central to nineteenth century technology. In the twentieth century our technology revolves increasingly around the notion of information, and it is the rational rather than the muscular faculties of man which now challenge the machine builders.

Games of mental skill devised by humans for amusement provide the research worker in machine intelligence with ideal material. In such exercises as programming a computer to play draughts (checkers), or chess, or Kalah or Go, all the intellectual faculties on which we pride ourselves are brought into play. Among these I would place high on the list the distinctively human ability to look ahead, predicting the consequences of alterna-

tive actions. This activity is normally called planning, and the design and testing of machines that can play and plan is a central interest in all those laboratories around the world dedicated to the machine intelligence objective.

During the war I was a member of Max Newman's group at Bletchley, working with prototypes of what we now call the electronic digital computer. One of the people I came most in contact with was Alan Turing, a founder of the mathematical theory of computation and the first apostle of the idea of designing a machine to think as intelligently as a man. He was also much interested in chess. But he was so profoundly intrigued by the deep principles of the game that he could never keep his mind for long on the tactical details. Beng one of the few people in the Bletchley environment bad enough to give him a reasonably even game, I became his regular sparring partner. After the war he and I engaged in some experiments with the mechanization of chess which I think were the earliest to be conducted in this field.

In those days we attached considerable polemical importance to showing that even one non-trivial exercise of thought, and chess certainly qualifies as that, could be convincingly mechanized. Looking back, I am not at all sure why we thought this so important. The mental solution of differential equations or the inversion of matrices constitute equally non-trivial tasks and yet they can be solved by algorithms; that is, they can be clearly stated and solved by applying a sequence of specified operations. No-one doubted even in the 1940s that computers could outgun most human beings in these feats of numerical mathematics. Consequently the discovery of equivalently powerful algorithms in non-numerical problems such as chess should not logically convey any added conviction.

So indeed it has turned out. The Greenblatt chess program is today operating in America, under the professional name MacHack, as a tournament player of reasonable competence. But no-one hails MacHack as the world's first intelligent machine. Rightly so, since MacHack would be as useless at solving differential equations as someone else's differential equations program would be at playing chess. The human intellect is marked not so much for any special brilliance at some particular task but rather for its ability to make a plausible shot at almost anything. We accordingly suspend judgement about the idea of an intelligent machine, waiting until one arises with the versatility and the powers of integrative behaviour which we demand of our colleagues.

Machine intelligence is not about chess nor about any particular mental task. It is about what can be generalized, including procedures initially developed on a task-oriented basis but possessing the seed of generality. Into this category fall various tree-searching techniques initially developed in mechanized game-playing. They include such fundamental features of planning as the generation of possible future states by a look-ahead process followed by assessment of the relative desirability of these states by approximate rules of strategic evaluation. The object of machine intelligence work is to tie together all essential general components of cognition

into a single library of computer programs held in the machine's store in such a way that the machine can respond to human interrogation in a coherent and resourceful fashion. In Table 3.1 are listed the topics which I regard as

Table 3.1 — Design topics which must be studied in depth if intelligent machines are to be developed are listed below. The first item is in the nature of supporting technology: it is an essential 'springboard' for the development of adaptive and problem-solving computer programs. Last item leads into the realm of robots — intelligent machines able to see, feel, and move about.

Time-sharing systems and 'conversational' programming languages
Learning by rote
Learning by trial and error
Learning by generalization
Elementary deductions about a simple world
Tree-searching and automatic problem-solving
Theorem-proving by machine
Theorem-proving representations of the problem to be solved
How many library routines make a mind?
Talking to the library
Linguistic skills: syntax and semantics
Associative storage and retrieval
Sense organs: pattern perception
Exploratory behaviour and theory formation

the minimum set to be studied in depth. Each poses a design problem for which solutions are necessary before the parts can be assembled into something with which we might hope to hold a usefully intelligent conversation.

The primary motive of work on machine intelligence is an engineering one: we want to make a machine which is man-like in certain respects. At the same time there is every reason to hope that spin-off may be produced which the brain scientists can use. An excellent example of this kind of spin-off can be found in the fact that we now have an understanding of the principles of flight in birds, particularly in relation to the evolutionary changes in anatomy which can be found in the fossil record. This new understanding has been largely achieved through the work of John Maynard Smith, whose application of the engineering concepts of feedback and aerodynamic instability to birds and other flying animals was made possible by the fact that he spent the war years working as an aircraft designer.

Learning by rote is perhaps the simplest of all cognitive aptitudes. I shall use it to illustrate the theme of taking over a trick from someone's special-purpose program in order to fashion a general-purpose implement.

A. L. Samuel's learning program for the game of draughts now plays at respectable county championship level, although Samuel believes that it is still a long way from attaining the calibre of a world champion. A fundamental feature of Samuel's program is the evaluation of a board position by

means of a 'scoring polynomial', in which the terms describe different strategic properties of the board position and the coefficients denote the respective weights to be attached to them. In addition, two learning mechanisms operate in parallel: 'rote learning' and 'learning by generalization'.

The first of these bases itself upon a dictionary of previously encountered board positions held on magnetic tape. A position is added to the tape with its value as calculated by the scoring polynomial. The dictionary thus acts as a look-up table for evaluating positions. If the position can be found on the tape, then the value is obtained relatively quickly; otherwise the scoring routine is entered and the value obtained by calculation. In the latter case the new position–value pair is added to the dictionary at the end of the evaluation. There is also a 'refreshing and forgetting' scheme whereby the least used entries are allowed to drop out of the dictionary whenever it is necessary to economize on storage space. As a result of this simple rote learning system, evaluations are performed with increasing speed as experience accumulates on the tape. Time is thus freed for pushing the look-ahead analysis deeper, with a gain in playing strength, and other learning effects of a more subtle kind accrue.

Another illustration, this time from work by R. A. Chambers and myself, is the surprising efficacy of crude rote learning for enabling a computer to master a difficult control task: balancing a pole on a motor-driven car under 'black box' conditions (see Fig. 3.1). The task, in common

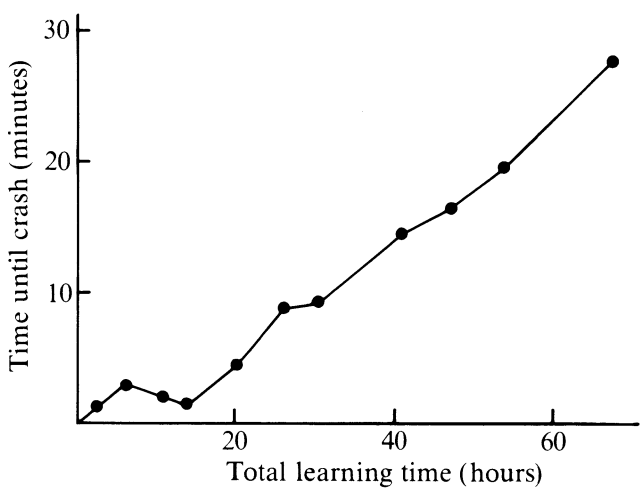


Fig. 3.1 — Trial-and-error learning enables a computer to balance a pole on a motor-driven cart. The task has the same structure as that of learning to play a game, with nature as the opponent. The computer learns by experience from an initial state of ignorance.

with adaptive control tasks generally, has the same formal structure as the task of learning to play a game; a 'game against nature'. The opponent's 'strategy' is determined by the causal laws which govern the responses of the unstable system to the control signals. The control signals themselves can be thought of as the 'moves' made by our side, and the successive state signals correspond to successive 'board states' in the play of the game.

What we need now is a way of generalizing these simple ideas, adding them to the mental furniture, so to speak, of general-purpose computing systems. We would like computers to learn from experience not only when doing unusual things like balancing poles but even when engaged on the run-of-the-mill tasks of ordinary arithmetic.

Contemporary programming languages provide for the definition of mathematical functions, subject to more-or-less awkward restrictions which I shall not discuss here. But existing languages such as ALGOL make no provision for the fact that when a certain function is applied to a given argument for the second time it may be more expeditious to recall the result from memory than to work it out again from scratch. The means of making any function into a 'memo function', complete with attached memory, has now been provided in our Multi-POP system at Edinburgh. Typing the appropriate instruction will attach to any function a memory with space for a specified number of entries. There are other refinements analogous to Samuel's 'refreshing' and 'forgetting'. The observed effect of the memo facility is in the expected direction: with increasing experience of using a given function, the computer carries out its task faster and faster. The self-improvement effect turns out to be substantial, and speed-ups of the order of tenfold are easily attainable in appropriate cases.

Now I want to consider the automation of mental processes more sophisticated than ordinary arithmetic. I shall restrict my remarks about graph-searching and problem-solving to a certain family of problems first treated along these general lines by Alan Newell and Herbert Simon at the Carnegie Institute of Technology. These are problems which can be represented as a set of discrete states and a rule book. The rule book specifies a set of operators — 'legal moves' — by which some states can be transformed into others. For example, in a sliding-block puzzle like the one illustrated in Fig. 3.2 the operators are the physical sliding movements of individual square blocks. Although I speak of this as a restricted class of problem, it is rich enough to provide formal representations of a wide variety of problems in which one is required to find a sequence of transformations leading from some initial state to a goal state, defined either explicitly or by possession of some desired property.

The activity known as problem-solving can be divided into two levels. The higher level is concerned with finding improved representation of the problem to be solved. This means in general replacing a problem graph — a diagram of nodes, representing the states in the problem, linked by arcs representing transformations — with another graph containing fewer nodes. An example using the eight-piece sliding puzzle might involve identifying states with symmetry classes instead of with individual board configurations.

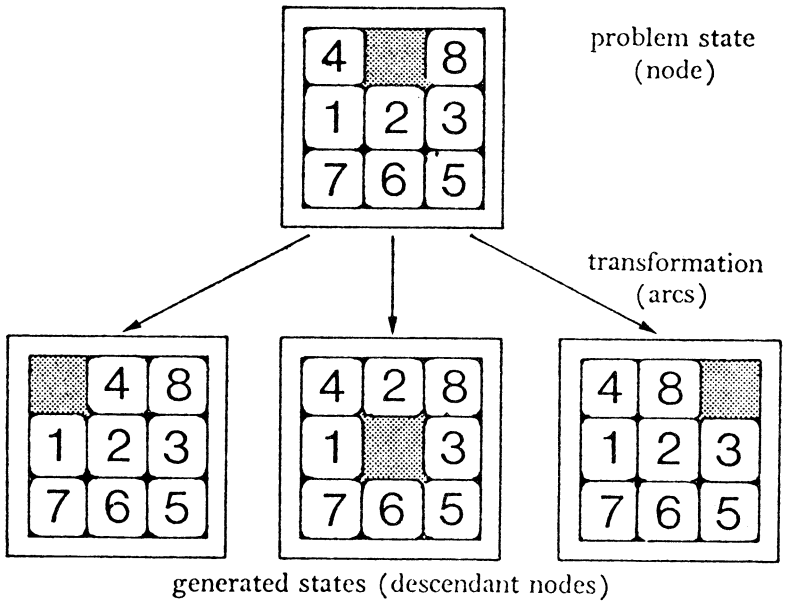


Fig. 3.2 — Sliding-block puzzle illustrates the treatment of problems which can be represented as a set of discrete states and a rule book of transformations or 'legal moves'. All such problems can be represented in the form of a graph of nodes, representing states, connected by arcs representing transformations, as in figure 3.4.

Another example with this type of puzzle could consist of replacing the simple moves of the rule book with compound moves — for example, the eight 'corner twists'. (A corner twist is a cyclic sequence of four simple moves confined to one corner which rearranges the three pieces occupying this corner. The use of this operator set reduces the nodes of the problem graph by a factor of nine, that is, to those representing 'centre empty' configurations.) I am not going to talk further about this higher level, which is concerned with problem representation. Impressive work is being done in this vital area by Saul Amarel of the Radio Corporation of America.

The lower level of problem-solving concerns what you do with your graph once you have it. J. E. Doran and I have developed a simple search algorithm in the form of a computer program called the Graph Traverser (Fig. 3.3). To set it to work, the user must give it definitions for two of its functions. These are, first, the 'evaluate' procedure which when applied to a state produces a score intended to estimate, however imperfectly, the distance of that state from the goal; and, second, the 'develop' procedure which when applied to a state selects an operator and uses it to produce a descendant state. While the 'develop' procedure thus embodies the rule book, the 'evaluate' procedure embodies whatever information or notions may be available concerning desirable or undesirable features of intermediate states (Fig. 3.4).

The real interest of the Graph Traverser idea lies in the possibility that it

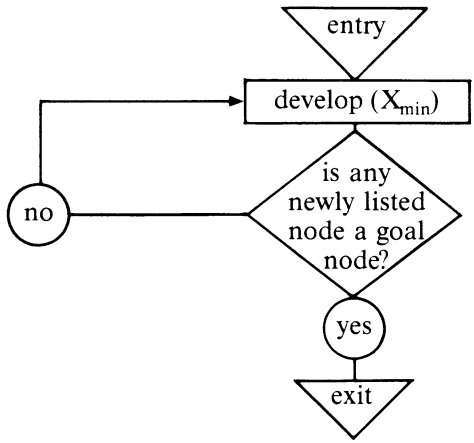


Fig. 3.3 — Graph traverser search algorithm is a computer program developed to solve problems represented as graphs of nodes and arcs. By 'develop' is meant select an operation from the rule book, apply it to the state in question, and evaluate the descendant state so produced. X_{min} denotes the minimum valued state of all those currently known to the program, that is, the most 'promising' of the states.

might be able to improve its own search strategy by exploiting in some way its accumulating experience of the problem. The 'develop' and 'evaluate' procedures constitute two separate points of entry for the introduction of learning ability. It is the 'develop' procedure which is concerned with the order of preference in which operators are selected. In our experiments with the Graph Traverser the program has always selected at random, but in the next phase we shall allow it to re-order the set of operators by promoting on the list those which turn out retrospectively to lie on the main path. A stage beyond this very simple learning method lies the attempt to set up a 'plausible move generator' based on abstracted features of the current problem state. This is done on a non-learning basis by the Greenblatt chess program MacHack, but there is no reason why the process should not be made adaptive. The preference orderings attached to the different classes of problem state defined by the abstracted features need not be fixed for all time but would be revised by a promotion process.

As for the 'evaluate' procedure, the problem here is essentially that called by Samuel in his checkers program 'learning by generalization'. His scoring polynomial is a weighted sum of terms which measure various strategic features of the board position; the program's problem is how to adjust these weights so as to improve its playing ability. Assuming that the Graph Traverser's evaluation function likewise takes the form of a weighted sum of terms, is there any way of adapting Samuel's approach to our needs? The key idea is that, as search proceeds, past experience accumulates in the form of a stored search tree with labelled nodes, the labels being the values assigned by the evaluation function. To the extent that the function is a good

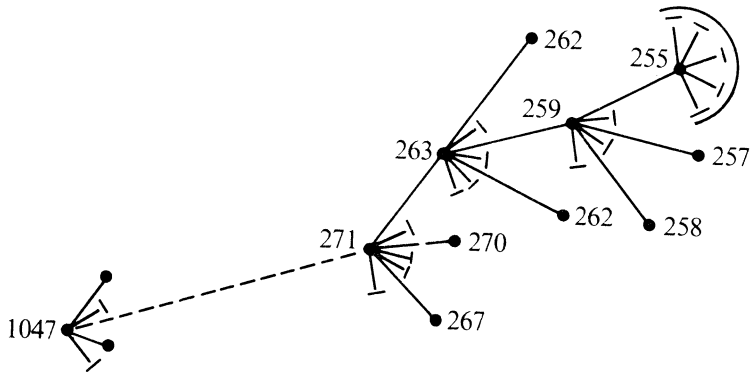


Fig. 3.4 — Search method used by Graph Traverser program is shown diagrammatically. Back-tracking automatically occurs when all arcs leading from a node are blocked. These blocked arcs denote developments which fail to produce improvements in the form of a reduction of the numerical values which are attached to the nodes.

one, the numerical values of the labels will correlate well with the distances from each other of the corresponding nodes. Moreover, the program is free to 'ruminate' over this stored record, adjusting the scoring function so as to improve this correlation. R. Ross and I have made preliminary tests of this idea, using sliding-block puzzles, with promising results.

Why did the hen cross the road? Actually there were three hens. The first was a clockwork hen and it crossed the road because its owner had wound it up and pointed it in that direction. The second hen crossed the road because an experimental psychologist was using it to illustrate a 'taxis' to his behaviour class: the hither side of the road was in darkness and the visual response to the illumination of the far side, together with reflex locomotor responses to tactile and proprioceptive inputs from its limbs, were sufficient to unroll a chain of actions which got it across the road. The third hen crossed the road in order to get to the other side. The explanation for this behaviour in a member of so unintellectual a species turned out to be that this hen was an intelligent robot. Hence it was able to operate upon an internal model of external reality, alternately updating the model inductively and using it deductively to foresee the consequences of its actions.

My engineer-psychologist colleague Richard Gregory points out that this predictive processing of stored information can be regarded as a way of exploiting the redundancy present in the real world, and he lists the following advantages which hen number three would enjoy:

(1) It can achieve high performance with limited information transmission rate... The gain results because perception of objects (which are always redundant) requires identification of only certain key features of each object...
(2) It is essentially predictive. In suitable circumstances can cut reaction time to virtually zero.
(3) It can continue to function in the temporary absence of any input, e.g. turning the music page, blinking or sneezing while driving...

(4) It can continue to function when the input changes in kind. Thus in maze learning, rats can continue to run a maze once learned though each sensory input in turn is denied it — vision, smell, kinaesthetics etc...
(5) It can extract signals from 'noise', if internal models are highly redundant. They can be called up with minimal sensory information. This means that the models can enormously improve the effective signal/noise ratio of sensory systems.
(6) Provided a particular situation is similar to the situations for which a 'model' was developed, behaviour will generally be appropriate. This, in the language of experimental psychology, is 'positive transfer of training'.

As disadvantages, he lists:

(1) When the current situation is sufficiently similar to past situations which have been selected and combined to give an internal model, but the current situation differs in crucial respects, then the system will be *systematically misled by its model.* This is 'negative transfer'.
(2) Internal model systems will be essentially conservative (showing inertial drag to change), for internal models must reflect the past rather than the present.

Gregory's notion of 'internal models' is decomposable into interpretative models, according to which the pattern of sensory stimulation is reduced to 'features' and 'objects', and predictive or planning models, according to which the likely consequences of applying alternative actions are computed by combining these interpretations of current sensory input with stored experience from the past. Fig. 3.5, which indicates where Gregory's internal

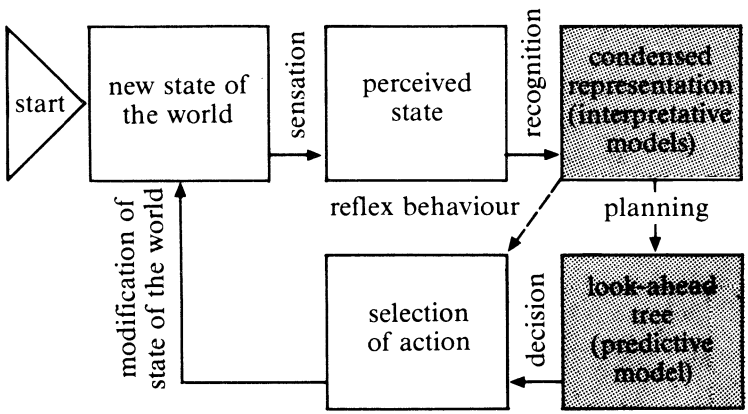


Fig. 3.5 — Intelligent robot's cycle of activity as it reacts with its environment is shown schematically. It operates by means of an internal model of external reality (grey boxes). This internal model is composed of an interpretative model, which reduces sensory stimulation to features and objects, and a predictive model which combines these interpretations with past experience to determine the likely consequences of alternative actions. The short circuit from 'recognition' to 'action' (dotted arrow) is for skills which are either inborn or have become automatic by means of practice.

models may be thought to reside, bears a close relationship to the scheme followed by my colleague J. E. Doran in his computer simulations of robots able to explore, learn, and plan. The 'predictive model' box of the diagram is mimicked in some essential respects by the Graph Traverser program discussed earlier. When the program grows a search 'tree' it is using an internal model to construct a plan, but when it is restricted to growing a 'bamboo', corresponding to what I have elsewhere called a 'conditional choice strategy', it is operating in reflex mode (a bamboo stem has nodes but no branches). The correspondence between Gregory's formulations in the behavioural realm and those of Graph Traverser design turn out to be engagingly simple. Those which most immediately leap to mind are listed in Table 3.2

Table 3.2 — Correspondence of the processes of exploration and learning in the biological world and in machines is remarkably direct if biological exploration and learning as formulated by R. Gregory is compared with the notation used in the Graph Traverser program.

Biological exploration and learning (Gregory's formulations)	*Machine exploration and learning (Graph Traverser notation)*
State of environment	*State of problem (for example, sliding block puzzle configuration)*
Perceived state	*Node on problem graph*
Repertoire of acts	*Set of operators*
Pleasure–pain associations	*Evaluation function*
Predictive model	*The 'develop' function of the Graph Traverser program*
Use of internal model to construct a plan	*Application of develop function to grow a partial search tree*
Selection of action	*Printout of partial path under 'dynamic pruning' regime*
Chain of reflex actions	*Conditional choice strategy*

Of particular interest are ideas which we are investigating experimentally for enabling the Graph Traverser to apply increasingly severe pruning procedures to its tree-growing activity, as its experience enlarges, until it has converted its operations from 'tree' mode to 'bamboo' mode. The biological analogy is with the embedding into fixed chains of habitual actions of patterns of behaviour which the organism originally elaborates for itself on a trial and error basis. When I first learned to tie my tie, the process was painful and fumbling with many false starts, backtracks and abandoned experiments. Now the sequence of actions proceeds in smooth stereotype,

each action creating a new state which in turn infallibly elicits a unique successor action, until it has run to completion. Only if I take a seriously wrong turn — if, for example, I have selected a bow tie by mistake — am I thrown back into 'tree-growing', as opposed to 'bamboo' mode, until I have worked back to a state sufficiently familiar to allow stereotyped habits to resume control.

Aggregation into larger units so as to exploit redundancy is called 'chunking' by George Miller. The extreme product of this process in linguistic behaviour is the cliché, the immense benefits of which in terms of neural economy is evidenced by the cliché-ridden speech of the general citizen. Chunking involves both input and output streams. Output chunking corresponds to what are sometimes called 'compound moves' or 'macro-moves' in the literature of automatic problem-solving.

Some of this discussion has been vague. The time for generalities is, however, drawing to an end, as laboratories in different parts of the world embark on the concrete task of constructing intelligent robots. In our own laboratory we plan to construct a FREDERICK (Family Robot for Entertainment, Discussion and Education, the Retrieval of Information, and the Collation of Knowledge). In future time the reader will be rightly impatient of any treatment of these topics which does not include accounts of the exploits by brain, eye, and limb of actual machines perambulating the laboratory proving ground. When that time comes I believe that certain fundamental capabilities will be found common to all such machines, including rote learning, generalization, the growing of look-ahead trees, tree-to-bamboo conversion, and the inductive up-dating of predictive rules in the light of accumulating sensory experience. Although the first explorations of these design topics were made in the attempts to program computers to play games, the nascent planning abilities of intelligent machines will increasingly be devoted to playing 'the game against nature'.

FURTHER READING

J. E. Doran (1968) New developments of the Graph Traverser. *Machine Intelligence 2* (eds E. Dale & D. Michie), 119–35. Edinburgh: Oliver and Boyd.

J. E. Doran (1968) Experiments with a pleasure seeking automaton. *Machine Intelligence 3* (ed D. Michie), 195–216. Edinburgh: Edinburgh University Press.

J. E. Doran (1969) Planning by generalization in a simulated robot. *Machine Intelligence 4* (eds B. Meltzer & D. Michie), 433–54. Edinburgh: Edinburgh University Press.

R. L. Gregory (1968) On how so little information controls so much behaviour. *Bionics Research Report no. 1.* University of Edinburgh: Department of Machine Intelligence and Perception.

D. Michie (1966) Game playing and game learning automata. *Advances in programming and non-numerical computation* (ed L. Fox), 183–96. London: Pergamon.

D. Michie (1968) Memo functions and machine learning. *Nature,* **218,** 19–22.

D. Michie, J. G. Fleming, & J. V. Oldfield (1968) A comparison of heuristic, interactive and unaided methods of solving a shortest-route problem. *Machine Intelligence 3* (ed D. Michie), 245–55. Edinburgh: Edinburgh University Press.

G. A. Miller (1956) Human memory and the storage of information. *IRE Transactions on Information Theory,* **IT–2**, 128–37.

A. L. Samuel (1959) Some studies in machine learning using the game of checkers. *IBM J. Res. Dev.*, **3**, 210–29.

A. L. Samuel (1960) Programming computers to play games. *Advances in Computers 1,* 165–92. London: Academic Press.

A. L. Samuel (1968) Some studies in machine learning using the game of checkers; II recent progress. *IBM J. Res. Dev.*, **11**, 601–17.

J. Maynard Smith (1952) The importance of the nervous system in the evolution of animal flight. *Evolution,* **6**, 127–9. Reprinted in *On Evolution,* 29. Edinburgh: Edinburgh University Press.

4

Evaluative comments in chess (1981)

Classical game theory partitions the set of legal chess positions into three evaluative categories: won, drawn and lost. Yet chess commentators employ a much larger repertoire of evaluative terms than this, distinguishing (for example) a 'drawn' from a 'balanced' position, a 'decisive' from a 'slight' advantage, an 'inaccuracy' from a 'mistake', and a 'mistake', from a 'blunder'. As an extension of the classical theory, a model of fallible play is developed. Using this, an additional quantity can in principle be associated with each position, so that we have not only its 'game-theoretic value' but also its 'expected utility'. A function of these two variables can be found which yields explications for many evaluative terms used by chess commentators. The same model can be used as the basis of computer play. It is shown to be easier to justify, and to adjust to realistic situations, than the minimax model on which state of the art chess programs are based.

REQUIREMENTS OF A THEORY

The game tree of chess contains about 10^{46} positions (Good 1968), a substantial proportion of which are terminal. The rules of the game assign a value to every terminal position, $+1$, 0, or -1 according to whether the position is won, drawn, or lost for White. These values can be backed up the game tree using the minimax rule, so that in principle every position can be given a value, including the initial position. This last is known as 'the value of the game', and is widely conjectured to be 0 for chess. If this conjecture is

correct, and if both sides play faultlessly, i.e. only execute value-preserving moves (it follows from the 'back-up' method of assigning values that there is at least one such move available from every non-terminal position), then the game must end in a draw. A fragment of a hypothetical game tree is depicted in Fig. 4.1. In Fig. 4.2 the method of attaching game-theoretic values to

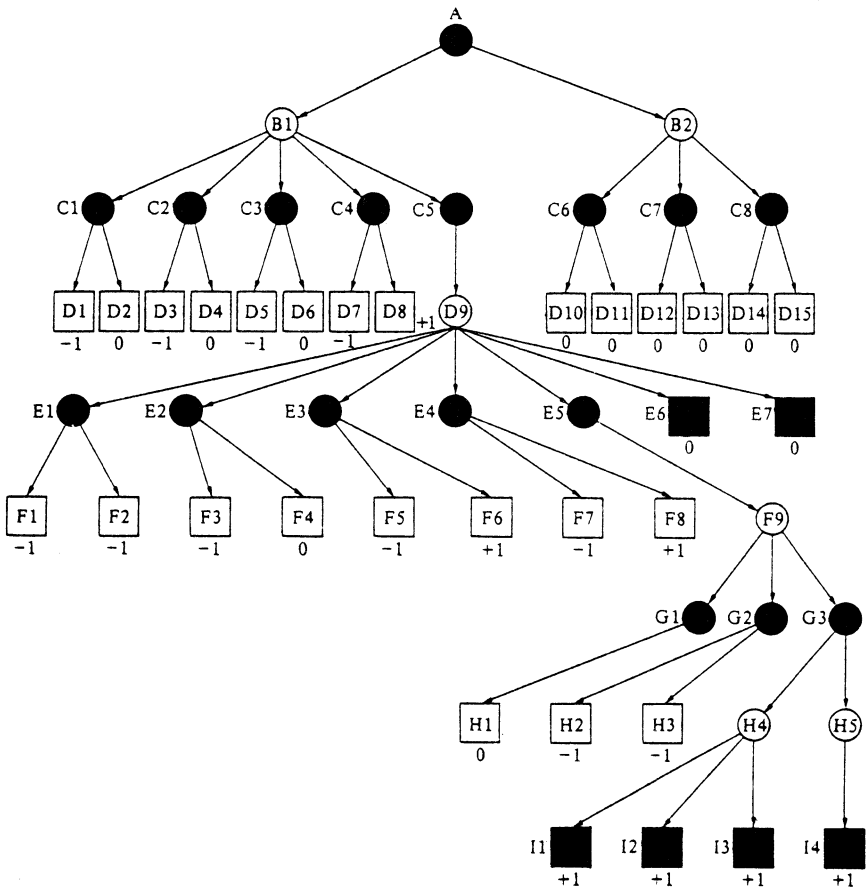


Fig. 4.1 — A game tree with its terminal nodes (shown as squares) labelled with outcome values from the set $\{+1, 0, -1\}$. Shading of the remaining nodes (circles) indicates which player has the move.

positions is illustrated.

An evaluation function could, in principle, map board positions into a larger set of values making it possible to express a distinction between positions which are 'marginally' won and positions which are 'overwhelmingly' or 'obviously' won, or between drawn positions in which White, or Black, 'has the edge' and drawn positions which are 'equally balanced', and

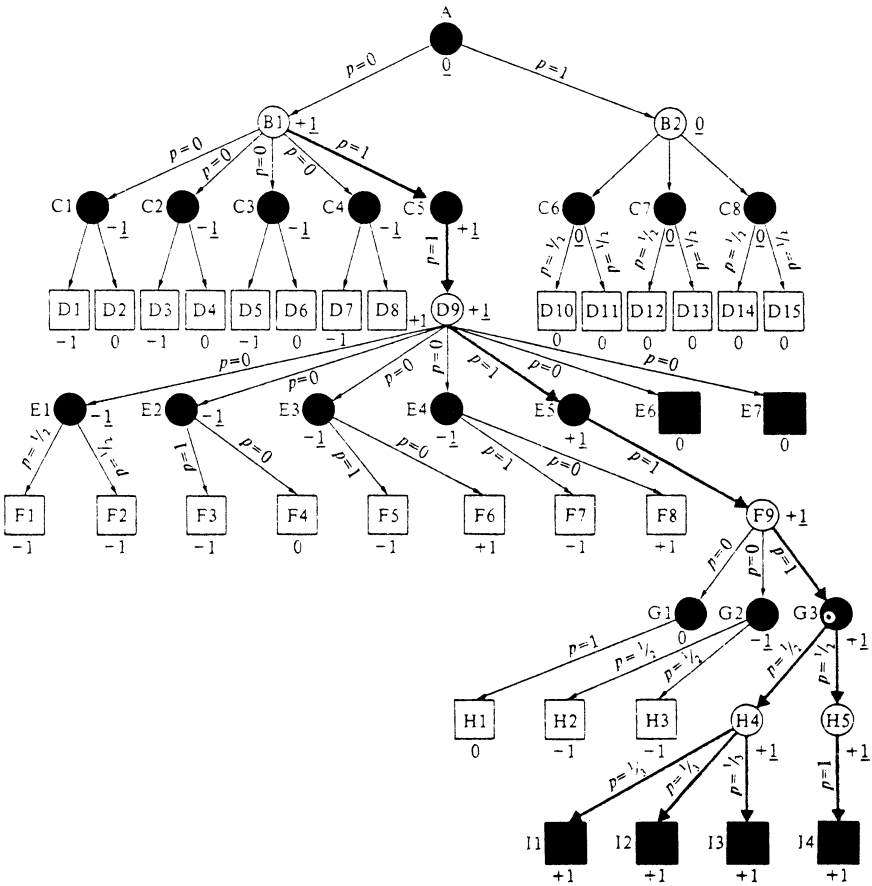


Fig. 4.2 — The game tree of Fig. 4.1 with its non-terminal nodes labelled (underlined values) by minimax back-up. White's best strategy from B1 is drawn with a heavy line. Arcs are marked with the conditional move-probabilities corresponding to perfect play: since the game-theoretic value of B1 is +1, Black chooses with probability 1 to move to B2.

so forth. Two circumstances suggest that a useful purpose might be served by multi-valued functions.

(i) Chess Masters and commentators have developed a rich descriptive language for the expression of such distinctions.

(ii) Computer chess programs employ real-valued functions for evaluating terminal positions, not of the game tree which is too large, but of the look-ahead tree. Values backed up from the look-ahead horizon are used to select the next move. We lack a formal basis for assigning definite interpretations to such values.

There is thus a need for a stronger theory of position-evaluation. This paper discusses chess, but the treatment is general and covers all two-person zero-sum games of perfect information without chance moves.

A good theory should explicate a variety of commentators' concepts. Table 4.1 is a representative list. Where a conventional symbol is available it precedes the verbal comment.

MAIN FEATURES OF THE THEORY

The game-theoretic model presupposes perfect play, whereas in the real-life game of chess (whether human or computer) both sides are susceptible to error. Our theory is based on this distinction, and presents the following main features:

(1) We follow Good (1968) and interpret the values of terminal positions as *utilities* as though the game were played for a unit stake. Values for pre-terminal positions are then calculated as *expected utilities*. In order to avoid confusion we shall refer to these throughout as 'expected utilities' or 'scores', never as 'values', reserving the latter term for game-theoretic values.
(2) A model of imperfect but skilled play is developed. Chess skill appears in this model as an adjustable parameter running from 0 (random play) to ∞ (perfect play).
(3) In the new model the classical game-theoretic treatment appears as a special case.

THE CALCULATION OF EXPECTED UTILITIES

Consider a state, s_0, from which transitions to successor states $s_1, s_2, s_3, \ldots s_n$ can occur with respective probabilities $p_1, p_2, p_3, \ldots p_n$. Let us suppose that these successor states have associated utilities $u_1, u_2, u_3, \ldots u_n$. Then the expected utility associated with s_0 is

$$\sum_{i=1}^{n} p_i u_i$$

It follows trivially that if we interpret as utilities the values attached by the rules of chess to the terminal position, then the values assigned to the non-terminal positions by minimaxing can be interpreted as expected utilities. In this special case the ps associated with those arcs of the game tree which carry a change of game-theoretic value are all 0. Consequently, the evaluation of $\sum_{i=1}^{n} p_i u_i$ at each node reduces to obtaining the 'min' or the 'max' of the successor-values according to whether White or Black has the move. The above specification is ambiguous in the case when two or more of the moves applicable to a given board position are value-preserving. We can either select one of these at random and assign a probabilitty of unity to it and zero probabilties to the rest, or we can divide the unit probability equally among

Table 4.1 — A representative list of commentators' comments

(1)		A dead draw (nothing that either players can do can avert a draw)
(2)		A complicated position
(3)	=	A balanced position
(4)	±	White has a slight advantage
(5)	∓	White has a clear advantage
(6)	+ −	White has a decisive advantage
(7)		A certain win for White
(8)		A difficult posiiton for White
(9)		A losing move
(10)		An inaccurate move: White weakens his position
(11)		White strengthens his position
(12)	?	A mistake
(13)	??	A blunder
(14)	!	A strong move
(15)	!!	A very strong or brilliant move
(16)	!?	A brilliant but unsound move
(17)		Best move
(18)	(!)	Best move in difficult circumstances
(19)		A safe move
(20)		White should press home his advantage
(21)		Black should play for time

them. In the case of error-free play, calculation of expected utilities according to either procedure leads to the same result. As the basis of a model of actual play we shall adopt the second alternative, which is illustrated in Fig. 4.2.

We now relax the game-theoretic condition that at each choice-point on the tree there is a probability of unity that a value-preserving move ('sound' or 'correct' move) is chosen, and we introduce the possibility of error. In constructing a model of error, we express the relative probabilities of making alternative moves from a given position as a monotonic increasing function (decreasing function for Black, since all utilities are expressed from White's standpoint) of the expected utilities of the corresponding successor positions. Thus the move leading to the highest expected utility will be chosen with highest probability (but not with probability 1 as in the game-theoretic error-free model), the move leading to the next highest expected utility with next highest probability and so on. We thus envisage an idealized player whose statistical behaviour reflects the rank-ordering of the expected utilities of chess positions. Using such a model it is again possible to label all the nodes of the tree, working upwards from the terminal nodes, but by a procedure which differs from the minimax method.

THE NOTION OF DISCERNIBILITY

In order to carry out some illustrative computations based on this idea, we now choose an actual monotonic function. No significance is claimed for the particular choice, since the points which we seek to establish are qualitative rather than quantitative. Certain ideas must, however, be reflected in any such function. A central one is that of *discernibility*. We conceive the player as standing upon a given node of the game-tree and looking towards its successors. These are labelled with their expected utilities, but the labels are not fully discernible to him. Discernibility is directly related to the strength of the player (the labels are fully discernible to an infinitely strong player) and inversely related to the number of moves separating the node from the end of the game: next-move mates and stalemates are fully discernible even to the beginner, but next-move expected utilities obtained by backing up are less so. Reflecting these considerations, we shall define the discernibility from a board state S_0 of the expected utility of a given successor state s_j as:

$$d = (M+1)^{[3(r_j+3)/(r_j+\epsilon)]} \tag{4.1}$$

where M is the merit of the player in kilopoints of the US Chess Federation scale, so that $0 \leqslant M$, and r_j is the number of moves that the value associated with s_j has been backed up. The symbol ϵ denotes an arbitrarily small quantity introduced to avoid the expression becoming infinite for $r_j = 0$.

The expected utilities themselves are real numbers lying in the range from -1 through 0 to $+1$. They are interpreted as being in logarithmic measure, to base d. Using this base, we take the antilogarithms of the expected utilities associated with the n successors of a given position as giving the *relative probabilities* with which a player of merit M who has reached s_0 selects the corresponding moves. Thus, for the transition $s_0 \rightarrow S_j$,

$$P_j \propto d^{uj} \tag{4.2}$$

Normalising these so as to obtain actual probabilites, $p_1, p_2, \ldots p_n$, the expected utility of a position is evaluated as $\sum_{i=1}^{n} p_i u_i$, where u_i is the expected utility of the position generated by the ith member of the set of available moves. Starting at the terminal positions, this gives a method for assigning expected utilities to successively higher levels of the game tree until every position has been labelled.

A SAMPLE COMPUTATION

Consider the terminal fragment of game-tree shown in Fig. 4.1. We shall illustrate step by step the calculation of expected utilities so as to label every node in the diagram. First we make assumptions for the playing strengths M_W and M_B of White and Black respectively. If we are to extract examples of the broad range of evaluative concepts from so ultra-simplified a game

tree we must set these strengths very low. Let us set $M_W = 0.2$ and $M_B = 1.4$: White is thus an abject beginner and Black a weak tournament player. In our model $M = 0$ implies random play. The notation $u(s)$ denotes the expected utility of position s.

H4: All successors have the same value, +1: $u(\text{H4}) = +1$.
H5: There is only one successor, so the move-probability is unity: $u(\text{H5}) = +1$.
G1: Unique successor: $u(\text{G1}) = 0$.
G2: Equivalued successors: $u(\text{G2}) = -1$.
G3: Equivalued successors: $u(\text{G3}) = +1$.
F9: From proportionality (4.2) we have
Move to G1: $d^0 = 1$ = relative probability.
Move to G2: $r = 1$, so, from Eqn (4.1), $d = 1.2^{12} = 8.915$.
Relative probability = 1/8.915 = 0.1121.
Move to G3: $r = 2$, so $d = 1.2^{7.5} = 3.925$ = relative probability.
Normalized probabilities: G1, 0.1985; G2, 0.0222; G3, 0.7792.
$u(F9) = (0.1985 \times 0) + (0.0222 \times -1) + (0.7792 \times +1) = +0.757$.
E1: Equivalued successors. $u(\text{E1}) = -1$.
E2: $r = 0$. $u(\text{E2}) = -1$, and similarly for $u(\text{E3})$ and $u(\text{E4})$.
E5: Unique successor. $u(\text{E5}) = 0.757$.
D9: **Move to E1:** $r = 1$. $d = 1.2^{12}$. Relative probability = 1/8.915 = 0.112 and similarly for moves to E2, E3, and E4.
Move to E6: Relative probability = 1, and similarly for move to E7.
Move to E5: $r = 4$. $d = 1.2^{5.25} = 2.604$. Relative probability = 2.0640.
Normalized probabilities: E1, 0.025; E2, 0.025; E3, 0.025; E4, 0.025; E5, 0.457; E6, 0.222; E7, 0.222 (total 1.001).
$u(\text{D9}) = (0.457 \times 0.757) - 0.100 = 0.246$.
C1: $r = 0$. $u(\text{C1}) = -1$, and similarly for $u(\text{C2})$, $u(\text{C3})$ and $u(\text{C4})$.
C5: Unique successor. $u(\text{C5}) = 0.246$.
C6: Equivalued successors. $u(\text{C6}) = 0$, and similarly for $u(\text{C7})$ and $u(\text{C8})$.
B1: **Move to** C1: $r = 1$. $d = 1.2^{12}$. Relative probability = 1/8.915 = 0.112 and similarly for moves to C2, C3 and C4.
Move to C5: $r = 6$. $d = 1.2^{4.5} = 2.272$. Relative probability = 1.2240.
Normalized probabilities: C1, 0.06703; C2, 0.06703; C3, 0.06703; C4, 0.06703; C5, 0.73190 (total 1.00002).
$u(\text{B1}) = (0.7319 \times 0.246) - 0.2681 = -0.088$.
B2: Equivalued successors. $u(\text{B2}) = 0$.
A: **Move to** B1: $r = 7$. $d = 2.4^{4.286}$. Relative probability = 1.391.
Move to B2: Relative probability = $d^0 = 1$.
Normalized probabilities: B1, 0.582; B2, 0.418.
$u(\text{A}) = (0.582 \times -0.088) + (0.418 \times 0) = -0.051$.

In Fig. 4.3 the tree of Fig. 4.1 is shown with expected utilities, calculated as above, attached to the nodes. The expected utility of the root node, A,

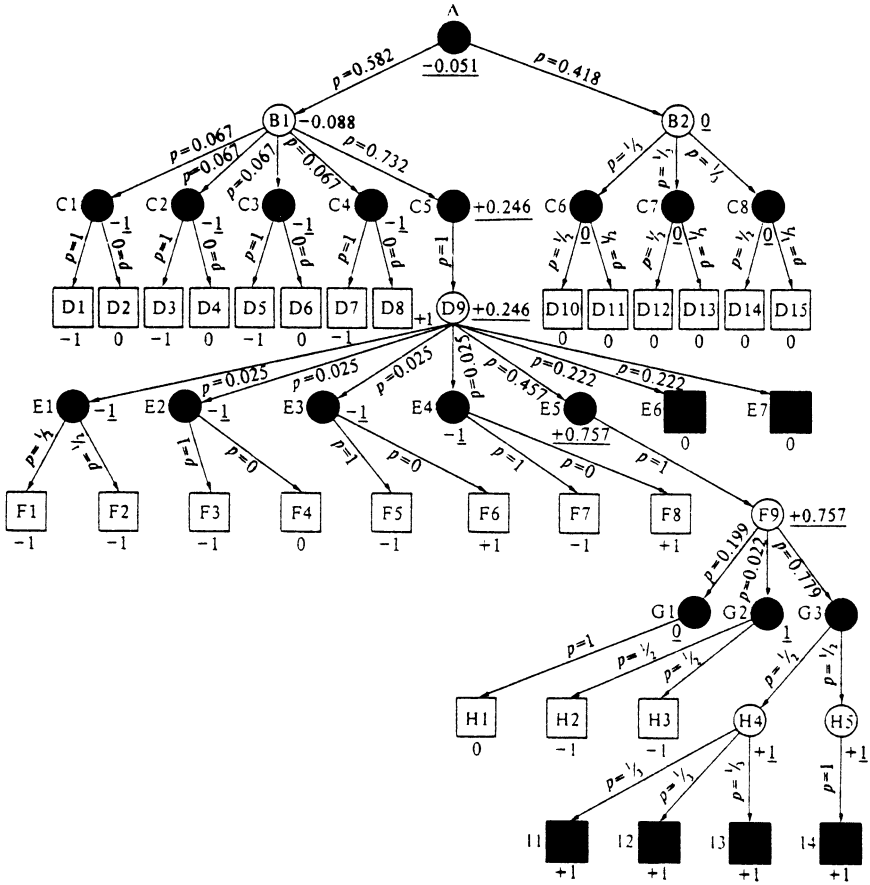


Fig. 4.3 — The game tree of Figs 4.1 and 4.2 labelled with expected utilities calculated from a model of fallible play. White has been credited with playing strength $M_W = 0.2$ and Black has $M_B = 1.4$. Conditional move-probabilities generated by this model are entered against the corresponding arcs and used to 'back-up' expected utilities to successively higher levels. As before, backed up values are underlined.

turns out to be one twentieth of a unit in Black's favour — a 'slight plus' for Black. The analysis of Black's 'plus' is worth pursuing, for it illustrates certain fundamental concepts to which our theory is directed, in particular the idea that a *losing move* (in the game-theoretic sense of a transition for White to value -1 or for Black to value $+1$) can also be the 'best' move against a fallible opponent.

Note that Black can secure a certain draw by moving to B2. Note also that the move to B1 is a losing move in the game-theories sense, for White can then win by the sequence B1 → C5 → D9 → E5 → F9 → G3, as shown by the heavy line in Fig. 4.2. Yet the *expected utility* of the move, -0.088, is marginally better for Black than that of the 'correct' move (expected utility zero), and our model of Black, possessed of a weak tournament player's

discernment, shows a 58% preference for the move. The statistical advantage arises, as can be seen by inspecting the diagram, from the fact that play is switched into a subtree where the error-prone White has numerous opportunities for error presented to him. He has to find the needle of sound play in a haystack of hazards. In such a situation we sometimes say that Black sets 'traps' for his opponent. If the aesthetic features of the move to B1 appeal to the commentator, he may even use the annotation '!?', which we take to mean 'brilliant but unsound'. A sufficient increase in the strength of White could give cause to remove the '!' or even to convert it into a second '?'. To illustrate this point we have recalculated the entire diagram after setting $M_W = M_B = 1.4$, shown in Fig. 4.4. Here the move to B1 does not

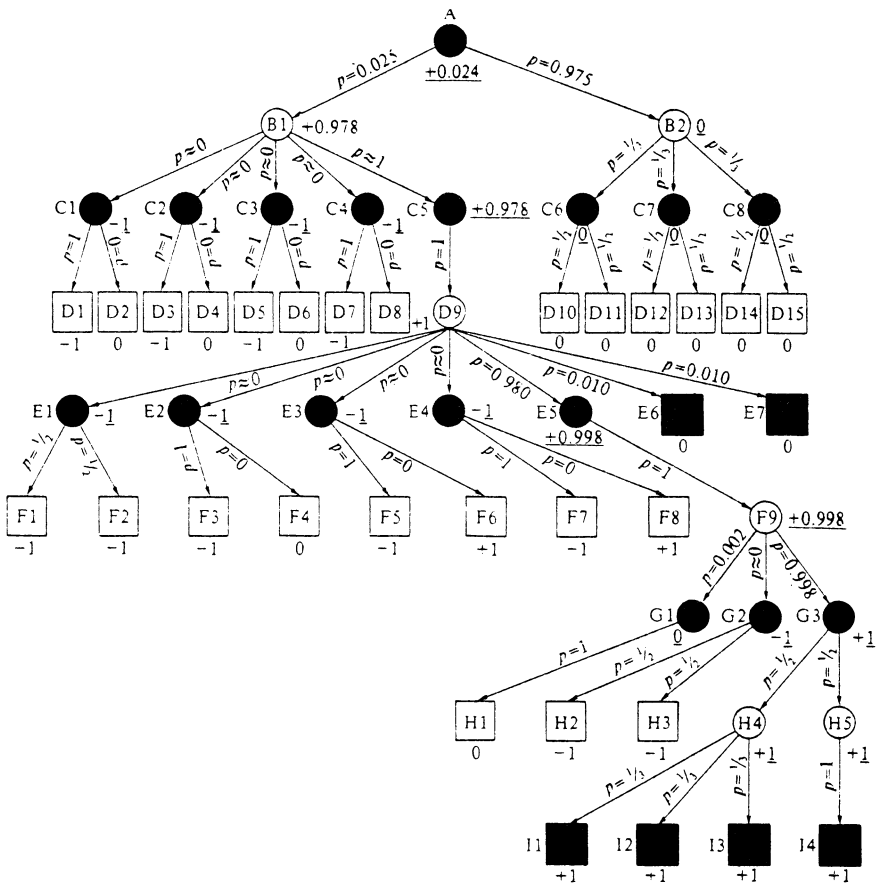


Fig. 4.4 — Expected utilities backed up the game-tree using a different assumption about the strength of the players, namely $M_W = M_B = 1.4$; i.e. both players are of weak club standard. The expected utility associated with the root node now favours White, and the model of Black's play shows a 40:1 preference at this choice-point for the 'safe draw'.

appear as 'best', nor even as a mistake, but as a blunder, and correspondingly our model of Black shows a preference of approximately 40:1 for B2.

Returning to the list of specimen evaluative comments in Table 4.1, we can now derive explications for them (Table 4.2). Wherever possible, an

Table 4.2 — Explication of the evaluative comments of Table 4.1

	Comment	Explication
(1)	A dead draw	$v = 0$ for all terminal descendants of s
(2)	s is complicated	The first few levels of the tree rooted in s have high branching ratios
(3) =,	s is balanced	$v = 0$ and $u \simeq 0$
	Case 1: s is lifeless	var $(v_1) \simeq 0$ } see text
	Case 2: s has high tension	var $(v_1) \gg 0$ } see text
(4) ±,	White has a slight advantage	$v = 0$ and $u > 0$
(5) ±,	White has a clear advantage (good winning chances)	$v = 0$ and $u \gg 0$
(6) + −,	White has a decisive advantage	$u \simeq +1$
	Case 1: White has excellent winning chances	$v = 0$ and $u \simeq +1$
	Case 2: Although White's game is theoretically lost, he is amost bound to win	$v = -1$ and $u \simeq +1$
	Case 3: An easy win for White	$v = +1$ and $u \simeq +1$
(7)	A certain win for White	$v = +1$ and $u = +1$
(8)	s is difficult	$v \gg u$
	Case 1: White needs accuracy to secure the draw	$v = 0$ and $u \ll 0$
	Case 2: White needs accuracy to secure the win	$v = +1$ and $0 < u \ll 1$
	Case 3: Although theoretically won, White's position is so difficult for him that he should offer a draw	$v = +1$ and $u < 0$
(9)	A losing move	$v(s_2) = -1$ and $v(s_1) > -1$
(10)	An inaccuracy: White's move weakens his position	$\Delta v = 0$ and $\Delta u < 0$
(11)	White's move strengthens his position	$\Delta v = 0$ and $\Delta u > 0$
(12) ?,	A mistake	$\Delta v = -1$ and not $(\Delta u \ll 0)$
(13) ??,	A blunder	$\Delta v < 0$ and $\Delta u \ll 0$
(14) !,	A strong move	$\Delta v = 0$ and $\Delta u > 0$ and s_1 is difficult
(15) !!,	A very strong or brilliant move	$\Delta v = 0$ and $\Delta u \gg 0$
(16) !?,	A brilliant but unsound move	$\Delta v < 0$ and $\Delta u \gg 0$
(17)	Best move	Δu is max
(18) (!),	Best move in difficult circumstances	Δu is max and s_1 is difficult
(19)	A safe move	$\Delta v = 0$ and s_2 is lifeless
(20)	'White should press home his advantage.' The rationale for trying to shorten the game when ahead can be understood by noting in Fig. 4.3 how the advantage decays as we move backwards from the terminal positions. In Fig. 4.5 White, in moving from B1, has been given an additional option in the form of a move to C5.1, from which Black is forced to move directly to F9 (*S*-shaped arc in Fig. 4.5). Game-theoretically the choice between moving to C5 and moving to C5.1 is equally balanced since they are both 'won' positions for White. But the expected utilities, +0.246 against +0.757, tell the true story, that if he incurs needless delay in a won position, especially if it is a *complicated* position (high branching ratio of immediately dependent tree), he multiplies his chances of error. Our model selects the move to C5.1 with 1.7 times the frequency of C5, with a corresponding increase of u(B1) (see Fig. 4.5).	
(21)	'Black should play for time' is the complementary advice one should give to the *other* player in the foregoing situation. If our hypothetical node C5.1 had a second branch leading to D9 (shown as a broken line in Fig. 4.5), then Black should prefer it to F9.	

explication is expressed in terms of two functions of a board position, namely its game-theoretic value v and its expected utility u. Where a move, rather than a position, is described, we use the notation $\triangle v$ and $\triangle u$ to denote the changes in the corresponding quantities affected by the move. We denote by s_1 the position from which the move is made and by s_2 the position which it generates. Some items of the original list have for completeness been differentiated into sub-concepts. Some of these would never appear in a chess book although under assumptions of very low playing strength they are generated by our model. Case 2 of (6) is an example of this: a 'decisive advantage' of this kind would characterise, for example, the initial position if Bobby Fischer gave Queen odds to a beginner.

We exhibit systematically in Table 4.3 various combinations of u and v,

Table 4.3 — Evaluative comments on positions (comments on moves are now shown here) corresponding to various combinations of expected utility, u, and game theoretic value, v

	$v=-1$	$v=0$	$v=+1$
1. $u=0$	s is virtually impossible (because of the unlikelihood that u should be identically zero).	s is a certain draw ('dead draw').	s is virtually impossible (because of the unlikelihood that u should be identically zero).
2. $u=-1$	s is a certain win for Black.	s is impossible.	s is impossible.
3. $u=+1$	s is impossible.	s is impossible.	s is a certain win for White.
4. $u \approx 0$	White has excellent drawing chances. Black needs accuracy to ensure his win.	s is a balanced position.	Black has excellent drawing chances. White needs accuracy to ensure his win.
5. $u \approx -1$	An easy win for Black (decisive advantage).	Black has excellent winning chances. White needs accuracy to make sure of the draw.	White has a theoretical win but is almost bound to lose.
6. $u \approx +1$	Black has a theoretical win but is almost bound to lose.	White has excellent winning chances. Black needs great accuracy to make sure of the draw.	An easy win for White (decisive advantage).
7. $-1 \ll u < 0$	Black has a mildly difficult win.	Black has a slight advantage. White needs care to make sure of the draw.	White needs extreme accuracy to make sure of his win (a very difficult win for White).
8. $+1 \gg u > 0$	Black needs extreme accuracy to make sure of his win (a very difficult win for Black).	White has a slight advantage. Black needs care to make sure of the draw.	White has a mildly difficult win.
9. $-1 < u \ll 0$	Black has a clear advantage.	Black has good winning chances. White needs accuracy to make sure of the draw.	White has a theoretical win but is likely to lose.
10. $+1 > u \gg 0$	Black has a theoretical win but is likely to lose.	White has good winning chances. Black needs accuracy to make sure of the draw.	White has a clear advantage.

entering in each case the evaluative comment which seems most appropriate.

'TENSION'

The minimax value of s can be regarded as in some sense summarizing the values of the terminal nodes of the tree rooted in s. More obviously, the expected utility of s, which has the form of a weighted mean, constitutes a

summary of a different kind of this same set of quantities. It seems natural to proceed to statistics of higher order, i.e. from representative values and means to variances. Might such second-moment statistics also possess recognizable meaning in terms of the chess commentator's vocabulary?

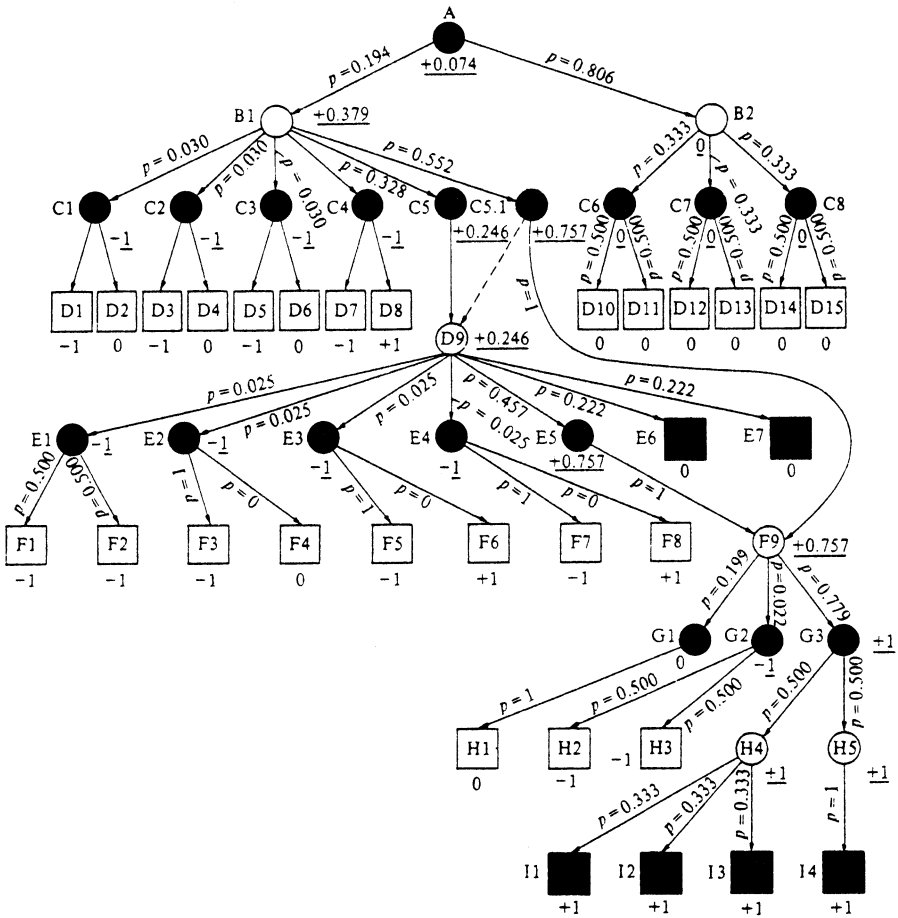
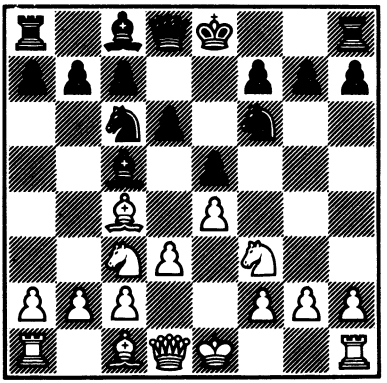


Fig. 4.5 — A modified version of Fig. 4.3 in which a new node, C5.1, has been added leading to F9 (the broken line represents a hypothetical delaying move for Black, see text). Although without effect on the game-theoretic values of nodes lying above it in the tree, interpolation of this short-cut option tips the balance of expected utilities, so that at the root the move to B2 becomes 'best'.

Good (1968) discusses a property of chess positions which he calls 'agitation'. He defines it by considering how sharply the estimated utility of a position is changed by investing a further unit of work in deepening the forward analysis. This quantity will necessarily be positively related to the variance of the distribution of u values over the dependent sub-tree, and

hence to the measure which we develop below for the 'tension' of a position. The former British Champion, Alexander, uses this term in an introductory chapter to *Fischer v. Spassky, Reykjavik* 1972. Alexander (1972) writes (see Fig. 4.6),

Giuoco Pianissimo



Gruenfeld defence

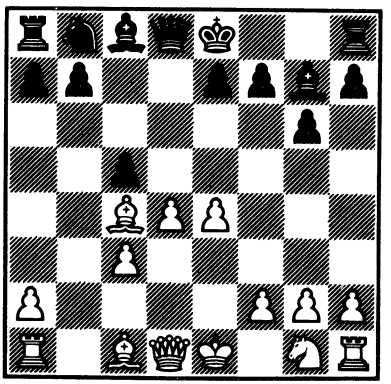


Fig. 4.6 — Positions of low and high 'tension' (from Alexander 1972).

> 'Let me illustrate (a little crudely) this question of tension by comparing two openings:
> A. (Giuoco Pianissimo) 1. P–K4, P–K6; 2. Kt–KB3, Kt–QB3; 3. B–B4, B–B4; P–Q3, P–Q3; 5. Kt–B3, Kt–B3.
> B. (Gruenfeld Defence: see the Siegen game Spassky v. Fischer) 1. P–Q4, Kt–KB3; 2. P–QB4, P–KKt3; 3. Kt–QB3, P–Q4; 4. P × P, Kt × P; 5. P–K4, Kt × Kt; 6. P × Kt, B–Kt2; 7. B–QB4, P–QB4.
> The moves in example A are perfectly correct — but after five moves the game is as dead as mutton; it is too simple, too balanced, and is almost certain to lead to an early and dull draw. The moves in example B are objectively no better — but the position is full of tension; White has a powerful Pawn centre but Black can exert pressure on it and, if he survives the middle game, may stand better in the ending — the players are already committed to a difficult and complex struggle in which a draw is not very likely.'

A simple way of capturing the spirit of Alexander's definition within the framework of our theory is to use the *weighted mean square* of the terminal values of the tree rooted in s, i.e.

$$\text{var}(v_t) = \sum_{t \in T} p_t v_t^2$$

where T is the set of terminal positions and p_t is the probability of arriving at the tth member of this set starting at s. A value of unity corresponds to maximal tension and a zero value to minimal tension (the latter can only be attained by a 'dead draw'). The tension of the root node of Fig. 4.3 is estimated by this method at 0.559. Referring to comment No. (3) above we assign this root node to Case 2 rather than to Case 1 of the category 'balanced'. Note that although 'tension' is calculated from game-theoretic values, v_t, use is made of the u_ts in the calculation of the probabilities, p_t, and hence the measure is affected by variation of the merit parameters M_W and M_B. As soon as we postulate greater playing strength on the part of White some of the tension of the position is reduced. The tension of node A in Fig. 4.4 is only 0.024, reflecting the fact that the Black is almost certain to steer play into the 'dead draw' sub-tree.

Note that $\sum_{t \in T} p_t v_t^2$ is equal simply to the probability of a non-drawn outcome. But we have preferred to formulate the expression explicitly as a variance, since in realistic cases game-theoretic values are not likely to be available, or calculable in practice. The approximating formula $\sum_{t \in U} p_t y_t^2$ may then prove useful, where the y_ts have been assigned by some evaluation function (or by human intuition) to the members of U, the set of states on the lookahead horizon.

SUMMARY OF IDEAS SO FAR

We have extended the strict game-theoretic model of chess, which assigns to board positions only three values: $+1$, 0 and -1. A good model should do justice to the profusion of chess commentators' evaluations. Specimen evaluative comments have been displayed as benchmarks against which to assess the extended theory. We have illustrated with worked examples a simple model based on the notions of utility and statistical expectation. Our model finds no particular difficulty in explicating the specimen evaluative comments. It also reduces to the game-theoretic model in the special case of error-free play.

APPLICATION TO COMPUTER CHESS

A worthwhile study would be to explore parts of a non-trivial sub-game of chess of which complete game-theoretic knowledge exists, as in K + N versus K + R (Bratko & Michie 1980, Kopec & Niblett 1980). The program's own comment on sample end-game play could be compared with the intuitions of experienced players.

A more satisfying use of the model would be for generating computer

play. The procedure exhibited earlier for calculating scores by backwards iteration from the terminal nodes of the game-tree was derived from classical decision theory. State of the art tournament programs also use 'backed-up' scores and they base move-selection on them. But they follow the minimax model. Might not such programs benefit from using expected utilities rather than minimax? After all, the near-universal adoption of the minimax rule in computer game-playing rests on no demonstrated theoretical foundation†.

When look-ahead is conducted to the end of the game, the validity of minimaxing rests on its built-in guarantee against selecting a game-theoretically 'losing move'. The reader can remind himself of this by inspecting Fig. 4.2: the constant-value sub-tree rooted in a given node defines a value-preserving strategy for all play ensuing from that node, provided that we have some rule for tie-breaking among a node's equivalued successors. But Fig. 4.3 shows that against a *fallible* opponent, this concept of validity is harmful, for here a 'losing move' is Black's decision-theoretically best choice.

A further difficulty arises when computational resources do not permit complete look-ahead . For this Shannon and Turing independently prescribed that the program should look ahead to some limited depth, and then assign to the terminal nodes of the look-ahead tree estimates of their game-theoretic values supplied by an 'evaluation function' — typically a linear combination of terms corresponding to measurable features of the position (piece advantage, mobility etc.). These scores are then backed up by the minimax rule to the current position's immediate successors, in place of the desired but inaccessible game-theoretic values. The rule of play selects the successor with the most favourable backed-up score (move B in Fig. 4.7).

Except in the (unrealistic and uninteresting) case that the evaluation function approximates the game-theoretic value so closely that the decisions given by the rule are invariant with respect to the depth of lookahead, this rule has lacked formal justification. We are thus free to attribute its empirical success to the fact that it can be regarded as an approximation to a decision-theoretically correct rule of the kind developed earlier. Note that the larger are the values of M_W and M_B, the closer is the approximation; in the limit the two models coincide.

The new model raises a point of particular relevance to the present situation in computer chess. Fast, partly parallel, special-purpose chess machines have recently been developed and interfaced to powerful computers (see for example Moussouris *et al.* 1979). Chess programs of conventional type interfaced to such machines become capable of searching to an average depth in excess of 9-ply, almost twice that attained by chess masters (see de Groot 1965; note that we are speaking of the average length of the longest branch of the look-ahead tree). To give such a machine the best chances it should be endowed with a 'hunger for complexity'. The idea must be continually to drive for high-tension positons avoiding simplifying

† Beal and Bratko have, however, recently established a sufficient condition (in *Advances in Computer Chess*, Vol. 3, Pergamon).

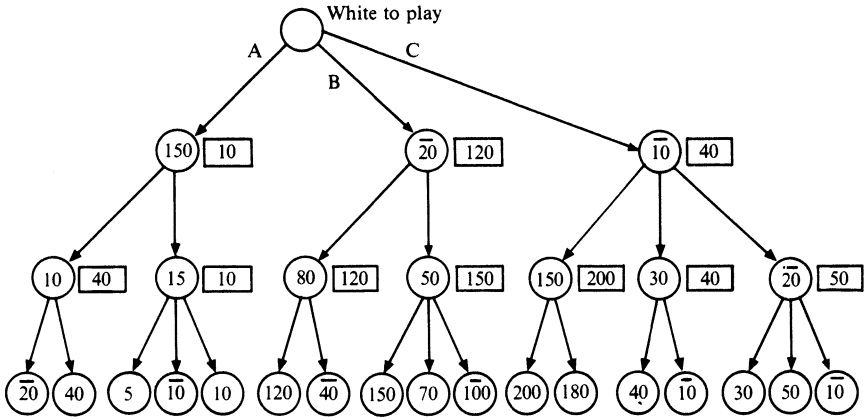


Fig. 4.7 — Positions are shown as circles in this look-ahead tree, in which the nodes are marked with 'face scores' (bars over negative). Boxed figures are values backed up from the look-ahead horizon. If move-selection were decided by face scores then move A would be chosen, but if backed-up scores then move B. What is the rationale for B?

exchanges where possible. In this way cognitive strain on the human player is intensified by the need for vigilance against tactical traps which may lie $\geq$9-ply deep. Such, a policy calls for a model incorporating opponent fallibility.

CONCLUDING REMARKS

An objection to the theory here developed is that the opponent model is arbitrary. Two comments are in order.

(1) It is of no theoretical consequence what particular opponent model is used for illustration, provided only that it has the right overall properties. The reader is free to use the theory with any opponent model he pleases.

(2) No choice of opponent model is as arbitrary, or as inflexible, as minimax. Moreover, even on the basis of complete look-ahead to the end of the game, minimax back-up does not yield the best strategy against a fallible opponent.

REFERENCES

Alexander, C. H. O'D. (1972) *Fischer v. Spassky, Reykjavik* 1972. Penguin, Harmondsworth.

Bratko, I. and Michie, D. (1980). A representation for pattern-knowledge in chess endgames, in *Advances in Computer Chess*, edited by M. R. B. Clarke, Vol. 2, pp. 31–54. Edinburgh University Press, Edinburgh.

Good, I. J. (1968). A five-year plan of automatic chess, in *Machine*

Intelligence, edited by E. Dale and D. Michie, Vol. 2, pp. 89–118. Edinburgh University Press, Edinburgh.

Good, I. J. (1977). Dynamic probability, computers and the measurement of knowledge, *Machine Intelligence*, edited by E. W. Elcock and D. Michie, Vol. 8, pp. 139–150. Edinburgh University Press, Edinburgh.

Groot, A. de (1965). *Thought and Choice in Chess*, edited by G. W. Baylor, Mouton, The Hague. [English translation with additions of the Dutch 1946 version.]

Kopec, D. and Niblett, T. (1980). How hard is the play of the King-Rook-King-Knight ending?, in *Advances in Computer Chess*, edited by M. R. B. Clarke, Vol. 2, pp. 57–81. Edinburgh University Press, Edinburgh.

Moussouris, J., Holloway, J. and Greenblatt, J. (1979). CHEOPS: A chess-oriented processing system, in *Machine Intelligence*, edited by J. E. Hayes, D. Michie and L. I. Mikulich, Vol. 9, pp. 351–360. Horwood, Chichester; Halsted, New York.

Shannon, C. E. (1950). Programming a computer for playing chess, *Philosophical Magazine*, 41, pp. 256–275.

Turing, A. M. (1953). Digital computers applied to games, in *Faster than Thought*, edited by B. W. Bowden, pp. 286–310. Pitman, London.

5

Computable sub-games of chess

A well-known argument, set out in the preceding chapter, demonstrates a sense in which chess—and other finite two-person games in which both players are allowed to see what is going on—is a foregone conclusion. The same imagined computation for assigning a won–drawn–lost value to the starting position (and any other position which we wish to evaluate) also defines correct strategies for the players.

A strategy which is no more than correct is unsatisfactory in that it lacks a 'sense of direction'. In a won position an ideal strategy presses forward to victory, preferably by the shortest route. In a lost position a Fabian tactic of delay is indicated: in case the opponent were fallible, we would want to give him as many opportunities to slip as possible. These ideas can be formalized by an appropriate modification of the minimax rule described in the earlier chapter by which the won–drawn–lost values of terminal nodes are backed up the tree of the game. The trick is to modify backed-up values according to their distance from the end and then to proceed as before. The effect is to pick out from the correct-strategy tree found by the unmodified procedure an optimal-strategy sub-tree defining the behaviour of ideally motivated players.

Note that a strategy which is 'optimal' is always 'correct', but the converse does not hold. Correct but non-optimal moves could perhaps be described as 'inaccuracies'; they are not 'mistakes'. There is a sense in which an inaccuracy, or even a mistake, might be a 'good move' relative to the limitations of an opponent. Such a sense, extensively explored in the previous chapter, is not considered further here.

Considerable interest would attach to the computation by this method of an optimal, or even a merely correct, strategy for the complete game of chess. But the game is too large. Claude Shannon estimated that to perform the required calculation, working back from all the terminal positions to the start, a machine operating at the rate of one variation per micro-micro-second would require over 10 to the power 90 years. He assumed an average

move-choice of 30 from each position and a duration of 40 moves for the typical game.

At first sight, then, the intersection of chess and practical computing seems too small to be of interest. Actually this is far from the case.

At a lower level of aspiration than exhaustive computation of the complete game, two different avenues are open, both inviting. First, one can accept a degree of approximation or error in the results of chess computations, and decide simply to develop practical playing programs able to hold their own with human masters. The second avenue, to which the present chapter is devoted, exists by virtue of the fact that exhaustive computation can fully solve sub-games of chess which are not fully understood by Grandmasters, or even by life-time students of the end-game. Where this has been done, results of interest have emerged.

The Grandmaster's secret weapon (typically secret even from himself) is his voluminous chess knowledge. Computers have now made it possible to probe such knowledge, and to investigate how it can be acquired, applied, refined, corrected, compacted, measured, and validated. Chess invites such machine-based study, not least because of its vast accumulations of codified lore. It has been something of a shock that the first machine analyses have indicated that this corpus may be so deficient as to be almost valueless. A Grandmaster is made by a combination of innate talent, inspiration from mentors, and sustained study and practice. None of these, it seems, endows him with the ability to articulate what he knows. It is in this sense that his acquired knowledge is secret even from himself, as we shall see.

A first step of machine analysis is to chop a small fragment at a time from the total game for detailed examination. Perhaps the smallest and most elementary fragment is the ending King and Rook against King. Reuben Fine's *Basic Chess Endings* devotes one page to it, including a diagram of a position said to require sixteen moves to mate. A computer tabulation by Clarke of the legal configurations of pieces (28056 after allowing for symmetries) reveals that with optimal play only fifteen moves are needed for this position, and that the longest mating path for any position is 16, not 17 as stated by Fine. Even the smallest sub-games, then, raise questions on which Grandmaster knowledge errs. Levels of increasing complexity can be arranged in the following progression.

Level 1. Correct play is trivial for a Grandmaster, such as the King–Rook–King (KRK) case cited, and other standard endings such as KQK, KPK, KBBK, KBNK. Note that we distinguish between correct and optimal play.

Level 2. A Grandmaster finds the problem of correct play sufficiently soluble for practical play against human opponents, but finds serious difficulty against an optimal machine opponent. Examples are the King–Rook–King–Knight (KRKN), KQKR, and KPKP end-games. Optimal-play tabulations have been computed not only for these but for all pawnless four-piece endings, notably by Kenneth Thompson of Bell Labor-

atories. Such a database can be used to detect departures from optimal play, as also from correct play.

Level 3. Correct play is beyond the capability of the Grandmaster when facing optimal opposition. Even if he has specially prepared for the given end-game, sooner or later errors will rob him of the theoretically anticipated outcome. Yet the problem space is still small enough for a complete look-up database to be machine-constructed. Examples are KQPKQ, KRPKR, KRBKR, KNNKP, and KBBKN. All the interesting pawnless five-piece endings have now been tabulated by Thompson. Each such table comprises upwards of a hundred million entries. With present technology complete exhaustion in this style of the space of possibilities is probably feasible (although not yet attempted) for end-games with as many as seven pieces on the board in all, including the Kings.

Level 4. Endings are too complex to be fully penetrated by exhaustive computation, let alone by the skill of Grandmasters or end-game specialists. Possibilities remain of constructing by man–machine cooperation complete strategies even for endings such as these, and of proving them correct. But in this case proof must of necessity be helped by formal reasoning, and cannot rely on total exhaustion of the problem space. Even then exhaustive databases can offer useful tests for gaps or mistakes in the proof, or can support the main proof by brute-force verification of key lemmas.

LEVEL 1 END-GAMES

In the beginners' manuals the King–Rook–King (KRK) ending is usually followed by exposition of the much harder King–Bishop–Bishop–King (KBBK) and of the very hard King–Bishop–Knight–King (KBNK). Another level 1 end-game, namely King–Pawn–King, although easier to play correctly than KBNK, has properties which make it a serious programming challenge. Programs can be checked for correctness against a complete look-up table computed by M. B. Clarke which gives the won–drawn value for each position with the minimax-optimal path-length to pawn-promotion in the case of won positions. The longest such path is 19 moves.

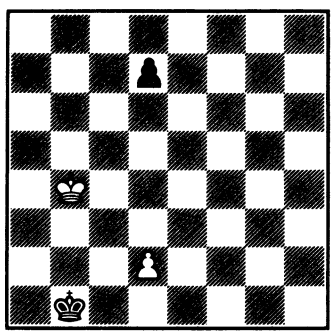
A program by M. Bramer uses a list of goal-patterns ranked in order of desirability, and selects whichever move attains the highest-ranked of those attainable in a single move. It has a modular structure which allows modification of its play by incremental injection of additional goal-patterns. At the cost of increasing the length of its goal-list from 20 patterns to 38, adequate (correct) play was converted into play describable as 'locally optimal', i.e. optimal with respect to the goal of safe pawn-promotion, rather than going on to calculate the subsequent moves of the resulting queen. A move which minimizes the number of steps to promotion is not necessarily that which minimizes the number of steps to mate.

This result, incidentally, suggests at least a rough-and-ready basis for the quantitative measurement of the *difficulty* of a task. One can say that

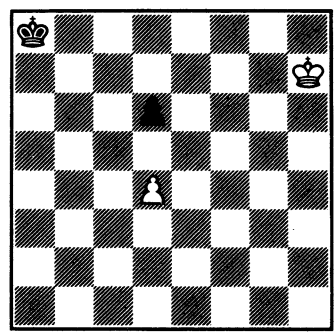
optimal KPK has about twice the difficulty of correct KPK, judging by the number of patterns that must be memorized in order to perform the respective tasks in a calculation-sparing manner.

LEVEL 2 END-GAMES

Addition to the KPK game of an enemy pawn, even under the restriction that it be on the same file as the friendly pawn, is sufficient to introduce the possibility of the machine's finding positions which a human master or end-game scholar would recognize as 'studies'. A study, or composition, is a position with a unique winning (or in appropriate cases drawing) line of play: for every variation introduced by the opponent there must be one and only one way of bringing about the desired result. Also the composition should exhibit certain properties of elegance, surprise, didactic value, and even wit, which are not easy to define. Clarke has endowed his program with criteria for searching through an exhaustive database to retrieve 'study-like' positions. Fig. 5.1 shows two of its discoveries for KPKP not previously in the



(a) White to win



(b) White to win

Fig. 5.1 — Two computer-assisted study compositions in the KPKP domain. The natural-looking Pd4 for (a) and Kg6 for (b) fail. Correct are Kc3 and Pd5 respectively.

chess literature.

Clarke's computations showed that, allowing for symmetries, there are 245 760 White-to-move positions, comprising 60 271 wins, 29 804 losses, and 155 685 draws and illegal positions. The longest path to a won KPKP position is 23 moves.

The King–Rook–King–Knight end-game is comparable in playing difficulty with that of the general KPKP game when this is not restricted to the case where the pawns are on the same file. There are 1 347 906 legal positions with the stronger side to play (White, let us say). There are 1 567 222 legal positions with Black to play. In 1970 Thomas Strohlein performed the earliest recorded exhaustive chess computations, for which he chose the

KQKR, KRKB, and KRKN end-games. The latter has been further studied in our laboratory. White can force the win in 696 414 of the 1 347 906 White-to-play positions and in 1 364 561 of the 1 567 222 Black-to-play positions. Two worst-case won-for-White positions exist with no fewer than 27 moves (53 'plies', counting Black's replies) until capture of the knight. These are shown in Fig. 5.2.

Position; WK:d1, WR:h1, BK:b1, BN:g4

1	Rh4	Ne5	2	Re4	Nf7	3	Rb4 + Ka2
4	Kc2	Ka3	5	Kc3	Nd6	6	Rb6 Ne4+
7	Kd3	Nf2	8	Kc4	Nd1	9	Rb3 + Ka4
10	Rf3	Nb2+	11	Kc3	Ka3	12	Rg3 Na4+
13	kc4	Ka2	14	Kb4	Nb2	15	Rg4 Nd3+
16	Kc3	Nc5	17	Rc4	Ne6	18	Ra4 + Kb1
19	Ra5	Ng7	20	Re5	Ka2	21	Kd4 Kb3
22	Kd5	Kc3	23	Kc6	Kd4	24	Kd6 Kd3
25	Ke7	Kd4	26	Rg5	etc.		

Position: WK:c1, WR:f8, BK:a3, BN:e2

1	Kd2	Nd4	2	Kc3	Nb5+	3	Kc4 Nd6
4	Kc5	Nb7	5	Kb6	Nd6	6	Rf4 Kb3
7	Kc5	Nb7+	8	Kc6	Nd8+	9	Kb5 Ne6
10	Rf3+	Kc2	11	Kc4	Kd2	12	Rf5 Kc2
13	Rf2+	Kd1	14	Kd3	Nc5+	15	Kd4 Nb3+
16	Kc3	Ke1	17	Rb2	Nc5	18	Kd4 Ne6
19	Ke3	Kf1	20	Rb6	Nc7	21	Ke4 Kf2
22	Ke5	Ke3	23	Rb7	Na6	24	Kd6 Kd4
25	Rb6	Ne5	26	Rb4	etc.		

Fig. 5.2 — Optimal move sequences for the two longest wins in the King–Rook–King–Knight ending (White to move), from D. Kopec and T. Niblett, 1980. Many positions, for either side, have more than one optimal continuation. The above two lines should therefore be regarded as specimen paths excerpted arbitrarily from two optimal-strategy trees.

To execute the winning manoeuvres reliably, or to conduct a good rearguard action against them with the knight's side, lies beyond the powers of the chess-master, although the leading end-game specialist John Roycroft was able by special study to acquire complete optimal-play mastery of play of the Rook's side from won-for-White positions (contrast Level 3 endings, later).

A small step beyond KRKN in complexity brings us to territory in which a new and unexpected effect compounds the problems of the human opponent.

This phenomenon made a public appearance when International Masters Hans Berliner and Lawrence Day undertook at the 1977 meeting at Toronto of the International Federation of Information Processing to demonstrate winning play of king and queen against king and rook. They played against a minimax-optimal database of Kenneth Thompson. Although every position with which they were faced at every stage was theoretically won, they found themselves unable to make progress. They complained that the machine's style of defence was counter-intuitive and even bizarre. This event dramatized the human's dependence on economy of mental representation, a need which is not shared by the machine. Simplified rules of thumb must be adopted by the human player, yet these will often be sub-optimal and sometimes erroneous. Masters self-trained to play against masters only have experience of strategies which are compactly describable and memorizable, and can flounder when faced with a strategy which cannot be defined in humanly comprehensible terms.

LEVEL 3 END-GAMES

Working in Moscow on a British ICL System 4/70 computer, Vladimir Arlazarov and Aaron Futer tabulated minimax-optimal strategies for KQPKQ and KRPKR, storing the results in each case on 8 magnetic tapes. Fast tape-search routines enabled the machine to demonstrate play at tournament speeds. These were used for an adjudication by Grandmaster Averbakh of a wager by International Master David Levy, which he lost, to the effect that a correct KRPKR strategy could not be implemented on a computer within a stated time limit.

In KRPKR the greatest length of any optimal path to pawn-promotion is 60 moves by each side. There are just two essentially distinct types of starting position, each with two instances. The four positions are the following, all with Black to move.

1. W: Kc3 Rc4 Pb2 B: Ke4 Rd1.
2. W: Kc3 Rc4 Pb2 B: Kf4 Rd1.
3. W: Kd1 Rd6 Pb2 B: Kh6 Ra8.
4. W: Kd1 Rd6 Pb2 B: Kg7 Ra3.

The KQPKQ database computed by the same laboratory has the distinction of being the first ever to be consulted during Grandmaster tournament play. Bronstein, playing at Kiev, was left with a KQPKQ position at weekend adjournment, leaving time for his seconds to arrange for a suitable excerpt of computer printout to be sent by train from Moscow. After resumption Bronstein played on to win. This database has also shown that the worst-case optimal path to pawn promotion from a pawn-on-7th-rank position is 58 moves long, so that the win would be forfeit under the present tournament

50-move rule. More extreme discoveries of this type had previously been made without machine aids. Notably the King–Knight–Knight–King–Pawn ending had been studied intensively by Troitzky in the early 1930s, following a treatment by Chapais in 1780. There is a position from which Troitzky states that 85 moves must elapse before the next advance of the pawn. Experience with computable end-games indicates that all quantitative assertions of this nature should be machine-checked. The same qualification, *a fortiori*, attaches to the introductory pages of Troitzky's celebrated 60-page treatise, which contains the words:

'This end-game contains no more secrets'.

Returning to the computer era, Kenneth Thompson's recent discoveries of significant facts new to chess include the status of King–Bishop–Bishop–King–Knight (KBBKN) previously believed to be drawable provided that the Knight's side can establish a 'Kling–Horwitz' position as in Fig. 5.3(a) (either side to move).

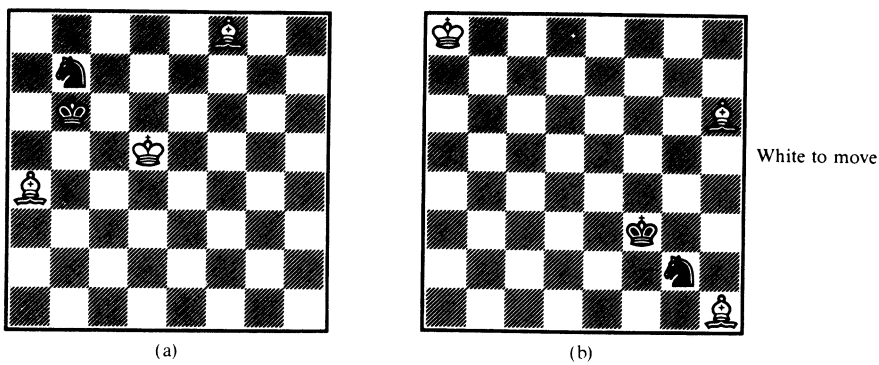


Fig. 5.3 — Two positions from the KBBKN ending (see text).

The position dates from 1851. The verdict that the defending side can draw positions like this (based on the placing of the two black men and largely ignoring the white placement) is repeated in all relevant chess end-game textbooks. In 1983 Thompson demonstrated the general win of this end-game in at most 66 moves, and the particular win in Kling-Horwitz positions in about 40–45 moves. Cases of this kind have pushed the World Chess Federation into an increasing number of *ad hoc* modifications of the 50-move drawing rule.

Not only does the Thompson database, comprising some two hundred million legal positions, show that the Bishops' side can win from all but a few

freak starting positions, but the manner of winning passes master comprehension. The critical fourth stage of a 5-stage win from the starting position shown in Fig. 3(b) involves a procedure 'not to be found in any book and characterized by excruciating slowness and mystery' (Roycroft). Moreover, following more than a year's study by the leading end-game specialist A. J. Roycroft and others, with access to a variety of computer aids, it seems possible that human mastery (as opposed to machine mastery) of this ending may never be attained.

REASONING IN PLACE OF EXHAUSTION

Proving properties of sub-games of chess (e.g. that mate can or cannot be forced from some or all starting positions in a given ending) introduces a new perspective on feasibility in chess computations. Shannon's earlier-cited arguments were conducted entirely within the constraints of a self-imposed assumption, namely that complete solution of a position or of a sub-game is to be performed entirely by exhaustive deductions, i.e. in a space of individual positions and moves. There is, however, another way to exhaust large spaces, namely by reasoning about generalized categories: indeed by this method infinite spaces can just as readily be subdued. The author has used computer-assisted reasoning to validate a long-standing conjecture by the Hungarian chess analyst Jeno Ban. Place the White king and rook in the sole corner of an infinite board. Place the Black king anywhere else at all, as in Fig. 5.4. Ban conjectured that White can force mate in a finite number of steps.

Plainly, exhaustive computations are powerless against such a problem. It succumbs, however, to pattern-directed case-analysis together with a proof of properties of an optimal solution strategy. A formula for the minimal number of solution steps from any given starting configuration was also derived.

Although Troitzky's treatment of the KNNKP game was not formal, the spirit of this approach breathes through his classic analysis. He writes of 'the discovery of regularities' and continues: 'In this way I found it possible to concentrate attention not on separate moves but on sequences of moves, to find manoeuvres and combinations of manoeuvres, to devise tactics for both sides and finally to construct entire strategical plans'.

For machine implementation, the Troitzky approach needs to be raised to the level of full logical rigour and completeness. Although we may think it unlikely, there is nothing in the present state of knowledge to tell us that the space of some 10 to the power 46 legal positions could not be reduced to a manageable number of theorems, and the game of chess brought within the reach of complete analysis. As a corrective, however, to naive expectations it would be prudent if an excursion or two in the foothills were first attempted, such as rigorous symbolic proof of Troitzky's KNNKP theories or of Thompson's empirical discovery that KBBKN is a won game. Suppose

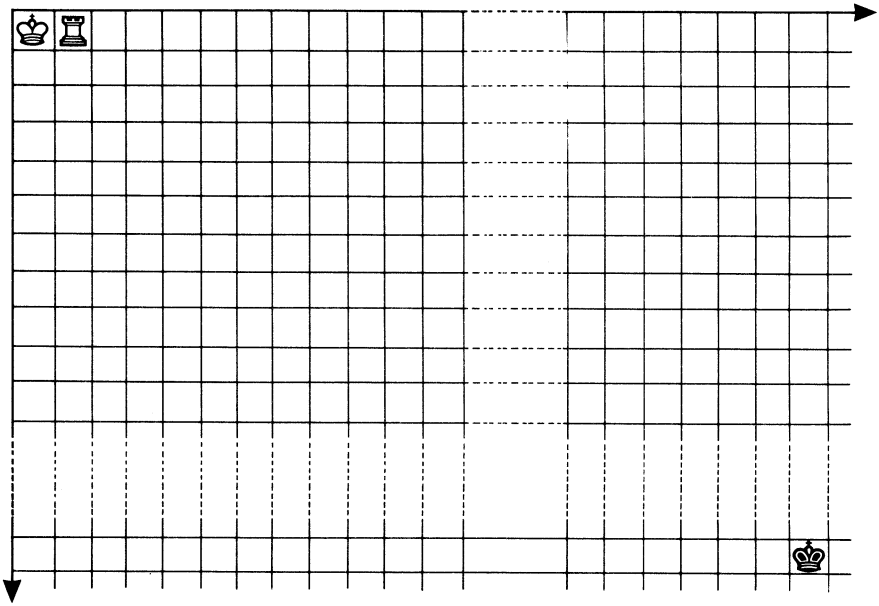


Fig. 5.4 — A problem of KRK on an infinite board with one corner (top left). Intuition says (correctly) that the White king must get to the south-east of the Black king so as to drive him back to the corner. Intuition also says (incorrectly) that the Black king can flee as fast as the White king can pursue him.

that such attempts met with failure, or just rough going. The main game would then stand revealed as unconquerable — certainly without aid of locally exhaustive computations and possibly altogether. Be that as it may, the use of symbolic reasoning, especially where it can be partly mechanized, puts a new complexion on the notion of practical computability.

Let us illustrate by means of a toy example. 'Everybody knows' that King–Knight–Knight–King (KNNK) is impossible for the knights' side to win. But how to prove it? An exhaustive approach might compute out the space of a few hundred thousand legal positions by forward repetition-free search and note failure to arrive at any forced mates. Could the same generalization be arrived at more directly and economically?

The answer is 'Yes'. The general idea of such reasoning will be familiar to everyone who has ever pondered about what the knights can and cannot do against the lone king. His argument will have gone something like this.

First, no checkmate position is possible with the opponent's king away from the edge of the board, since the king's square and the eight squares surrounding it must all be attacked for the king to be in checkmate. It can quickly be seen that no arrangement of the three White pieces exists which controls a total of more than seven altogether of this block of nine squares, since each knight can take care of at most two, and their king of only three.

Berliner, H. J. (1978) Computer chess. *Nature,* **274**, 745–48.

Bramer, M. A. (1980) An optimal algorithm for king and pawn against king using pattern knowledge. *Advances in Computer Chess 2* (ed. M. R. B. Clarke), 82–96. Edinburgh: Edinburgh University Press.

Bratko, I. (1978) Proving correctness of strategies in the AL1 assertional language. *Information Processing Letters,* **7**, 223–30.

Bratko, I., Mozetic, I., & Lavrac, N. (in press) Automatic synthesis and compression of cardiological knowledge. *Machine Intelligence 11* (eds. J. E. Hayes, D. Michie, & J. Richards) Oxford University Press.

Clarke, M. R. B. (1977) A quantitative study of king and pawn against king. *Advances in Computer Chess 1* (ed. M. R. B. Clarke), 108–118. Edinburgh: Edinburgh University Press.

Fine, R. (1941) *Basic Chess Endings.* New York: David Mackay; London: G. Bell.

Kling, J. & Horwitz, B. (1851) *Chess Studies, or Endings of Games.* London: Skeet.

Kopec, D. & Niblett, T. (1980) How hard is the play of the King–Rook–King–Knight ending? *Advances in Computer Chess 2* (ed. M. R. B. Clarke), 57–81. Edinburgh: Edinburgh University Press.

Michie, D. (1977) King and rook against king: historical background and a problem on the infinite board. *Advances in Computer Chess 1* (ed. M. R. B. Clarke), 30–59. Edinburgh: Edinburgh University Press.

Roycroft, A. J. & Thompson, K. Documentation of recent exhaustive endgame analyses is to be found in articles in the January 1986 issue of the British magazine *EG*, vol. 83.

Shannon, C. E. (1950) Programming a computer for playing chess. *Philosophical Magazine,* **41**, 356–75.

Shapiro, A. D. & Michie, D. (1986). A self-commenting facility for inductively synthesised endgame expertise. In *Advances in Computer Chess 4* (ed. D. Beal), Oxford: Pergamon Press.

Troitzky, A. A. (1937) *Chess Studies.* Leeds: Whitehead and Miller. Translation from the Russian original. Leningrad, 1935.

6

Computer chess and the humanization of technology (1982)

Chess provides the opportunity for studying the representation of human knowledge in machines. But it took more than a century since its conception by Charles Babbage for chess playing by machines to become a reality. The World Computer Chess Championship and other computer chess tournaments where program is matched against program occur regularly. But can the less clever but more intelligent human Master use the computer's brute force technology as a source of new chess knowledge?

The first serious proposal to have a machine play chess was made by Babbage [1], the British pioneer of digital computing, but was never executed. In the early years of this century the Spanish engineer Quevedo demonstrated an electromechanical device for mating with king and rook versus king [2]. But it was not until the late 1940s that serious experiments with the complete game were conducted. The British logician and computation theorist Turing [3], in collaboration with Champernowne and others, constructed and tested various 'paper machines' embodying mechanized strategies for chess [4]. Play was poor. In 1950 the American founder of information theory Claude Shannon [5] published the classic paper for the theoretical ideas. During the same period Groot's [6] study of the thought processes of chess masters revealed that their special ability does not derive from 'computer-like' qualities of memory or of accurate, fast, and sustained calculation, but from powers of conceptualization. This result had been foreshadowed by Binet's [7] investigation in 1900 of the ability of chess masters to play many games of blindfold chess simultaneously. He concluded that in addition to *la mémoire* this accomplishment rested on *l'érudition* (the use of accumulated chess knowledge to form meaningful descriptions of positions) and *l'imagination* (ability mentally to reconstruct a position from a description). Progress has been made in mechanizing the 'computer-like' mental attributes, but little in respect of *l'érudition* and *l'imagination*.

DEVELOPMENTS

The earliest chess programs, developed in the 1950s, have been reveiwed by Samuel [8]. The modern era dates from the 1967 entry into human tournaments of the Greenblatt–Eastlake–Crocker [9] program under the playing name MacHack. This program played at the level of a weak-to-middling club player — Category III, USCF rating 1400–1600 (Candidate Master level begins at 2000, National Master at 2200, International Master at 2400, Grandmaster at 2500).

Computer chess tournaments, in which all games are program-against-program, are now organized annually in the United States by the Association for Computing Machinery (ACM). The first took place in 1970 in New York. In addition, every three years a World Computer Chess Championship sponsored by the International Federation of Information Processing Societies is held. The first [10], in Stockholm in 1974, resulted in a victory for the program KAISSA developed in Moscow by V. L. Arlazarov, G. G. Adelson-Velskiy, A. R. Bitman, and M. V. Donskoy. CHESS 4.0, entered by L. Atkins and D. Slate of Northwestern University, United States, came second. Standards of play corresponded approximately to USCF 1600–1650. In 1975 the ACM tournament evoked play at the same general level, but produced one game, between CHESS 4.4 and CHAOS, of a more distinguished standard.

In the late 1970s most progress in computer play resulted from a combination of improvements in the efficiency of deep tree-searching methods with faster speeds of available computing hardware. CHESS 4.6 won the Second World Computer Chess Championship at Toronto in 1977, and domination of the ACM by updated versions of CHESS continued until BELLE (Bell Labs) won in 1979. The Third World Computer Chess Championship at Linz, Austria in 1980 was also won by BELLE. The program [11] ran on an LSI 11/23 micro-computer linked to a hard-wired chess machine and could 'see' of the order of 100,000 board positions per second.

In 1981 the four strongest programs BELLE, CHESS 4.9, NUCHESS, and CRAY BLITZ, were all in the Candidate Master (2000–2199) range of play. The same year saw the emergence of several commercially available portable chess machines claimed by their manufacturers to rate at least 1900 (Category I).

Computer chess has been described as the *Drosophila melanogaster* of machine intelligence. Just as Thomas Hunt Morgan and his colleagues were able to exploit the special limitations and conveniences of the *Drosophila* fruit fly to develop a methodology of genetic mapping, so the game of chess holds special interest for the study of the representation of human knowledge in machines. Its chief advantages are: (1) chess constitutes a fully defined and well-formalized domain; (2) the game challenges the highest levels of human intellectual capacity; (3) the challenge extends over the full range of cognitive functions such as logical calculation, rote learning, concept-formation. analogical thinking, imagination, deductive and induc-

tive reasoning; (4) a massive and detailed corpus of chess knowledge has accumulated over the centuries in the form of chess instructional works and commentaries; (5) a generally accepted numerical scale of performance is available in the form of the US Chess Federation and International ELO rating system.

COMPUTATIONAL AND COGNITIVE MECHANISMS

The fundamental procedure, proposed independently by Turing and by Shannon, involves *look-ahead* from the current position along a branching tree of possibilities. Of course, chess-masters also look ahead (concrete analysis) but on a severely restricted scale. According to de Groot, 30 positions is around the limit of what is normally held in look-ahead memory. By contrast chess programs commonly grow look-ahead trees comprising millions of nodes.

Branching of the look-ahead tree ranges from around 25 branches per node (if no pruning is applied) down to less than two branches per node for Masters. The number, variety, and severity of pruning rules vary from one program to another. In one or two of the stronger programs all but the alpha-beta rule are effectively absent. In such a case the program seeks to make up in brute-force calculation what it lacks in selectivity. All programs apply termination rules to halt growth of the tree beyond certain limits. The main factor in termination is the occurrence of quiescent positions (Turing's 'dead' positions) in which no capture or other violent changes are in immediate prospect. At the deeper levels of the look-ahead tree quiescence is taken as a sufficient indication to discontinue forward analysis.

In the Turing–Shannon scheme on completion of the look-ahead tree the program applies an *evaluation function* to the terminal positions, labelling them with computed estimates of their degree of strategic strength or weakness. The labels are then *backed up* the tree by the minimax rule: that is, a node for which it is white's turn to play is credited with the maximum-valued of the labels attached to its successors, while a black-to-play node receives its label from the minimum-valued of its successors. The wave of labelling thus spreads through the tree from the terminal nodes towards the root node (representing the current position) until all its successors are labelled. The program then selects the move leading to the highest-valued of these successors, if it is white's turn to move; otherwise the move leading to the lowest valued successor.

The functioning of this basic mechanism can be improved by various devices aimed at eliminating redundant calculations and redundant storage. In favourable conditions the alpha-beta rule [12] for pruning almost doubles the realized depth of look-ahead without altering the final result of the computation: the spirit animating this rule is that once the search has found that a particular line falls short of being selected there is no point in exploring its further ramifications to determine by just how far it falls short. Look-ahead depths attained in modern computer chess lie in the range 6–15,

somewhat in excess of the typical depths to which Masters and Grandmasters look (average 6–7 as found by de Groot). In spite of this the standard of computer chess remains far below Grandmaster levels. Some of the reasons are as follows:

(1) Horizon effect. The computer scientist and former World Correspondence Chess Champion, Hans Berliner [13], has pointed out that reliance on the unaided Turing–Shannon procedure renders a program oblivious to all events which may occur beyond its look-ahead horizon. Even though a post-horizon loss, or a post-horizon gain, may appear inevitable and obvious to the human onlooker, the program plans from hand to mouth, foolishly sacrificing material to delay a loss which cannot indefinitely be averted; alternatively it may forfeit an eventual large expectation by grabbing at a small gain.

(2) Lack of long-range ideas. A Master plans at the conceptual level, linking the main milestones with detailed steps as a separate operation. Contemporary programs have no corresponding capability. In the end-game in particular, where long-range reasoning of this kind is at a premium, programs can flounder aimlessly, promoting small disconnected goals with no unifying strategic thread.

For these reasons, computer programs make a poorer showing in the end-game than in the opening and mid-game, performing like club players rather than candidate masters. Advances in sheer computer power, even micro-microsecond processors or mega-megabyte memories, are not expected in themselves materially to improve this situation. Remedies must take their departure from an appreciation of the ability of the chess-master to utilize very large bodies of highly conceptualized and cross-referenced chess knowledge. But in programs of the Turing-Shannon type the only significant repository of chess knowledge is in the evaluation function, typically a linear combination of terms measuring such features as material advantage (conventional scores: 9 for Q, 5 for R, 3 for B, 3 for N, 1 for P), king safety, piece mobility, pawn structure, rook control of files, and so on. Typically the number of features contributing terms to the evaluation function in state-of-the-art tournament programs lies in the range 30–50.

So simple a scheme is too weak to represent the fine structure of human knowledge marshalled in the standard expository works such as Reuben Fine's *Basic Chess Endings*. Contemporary research is directed towards buttressing the Turing–Shannon paradigm along a line sometimes described as the 'knowledge approach'. Essential to this approach is the extension of studies like Binet's and de Groot's to the discovery of the basic concepts (properties and relations of pieces, files, ranks, and so on) which the Master uses as the bricks and mortar of his mental descriptions. Chase and Simon [14] found that the relations of defence of one piece by another, proximity of pieces, and being of the same denomination or colour were all used as mental building-blocks, and that a particularly important relation for binding together clusters of pieces held as a unit in memory was combination of pieces of the same colour to converge on the opponent's king position.

Tan [15] formalized the process of conceptualization for the special case of pawn structures. His computer program was able to describe pawns-only positions in terms of 'islands' and 'fronts' forming pawn-relations graphs, from which 'attack-defense-diagrams' are automatically constructed. The dynamic potentialities of the position are thus summarized. More recently a computer induction algorithm [16] derived from Hunt's "Concept Learning System" [17] has been used to synthesize complete machine-executable theories for the endings king and pawn versus king [18] and king and pawn (on a7) versus king and rook [19].

Chess is a two-person finite game with perfect information which satisfies the zero-sum condition — an outcome which is good for one player is bad in equal measure for the other. For any such game it is theoretically possible exhaustively to calculate backwards from the terminal (checkmate or drawn) positions in such a way as to determine for every position whether it is drawn, won, or lost, and in the latter two cases what continuations correspond to best strategy. In practice such computations, even if performed on the fastest conceivable computers, are infeasible except for end-games simple enough to contain less than a thousand million or so legal positions. Such computations were first done [20] for elementary ending such as king and rook versus king (KRK) and king and pawn versus king (KPK) which consist respectively of 50 015 and 179 656 legal positions. They have been extended to all the non-trivial four-piece endings, such as KQKR, KRKN, and so on and to a subset of KPKP (M.R.B. Clarke, personal communication). The most complex enumerations to have been performed in this way are KQPKQ [21] and KRPKR [22], of which the first is notable for having been consulted with good effect by Bronstein during adjournment of a master tournament in Kiev. Significant findings have been made with the use of these end-game 'databases', including the previously unsuspected prevalence of serious error in master texts on the end-game. Thus Fig. 6.1 shows a derivative of the celebrated position given by al-Adli in the ninth century, rediscovered and (incorrectly) analysed in *The Chess-players' Chronicle* in 1859, and repeatedly (and incorrectly) re-analysed since then. Among errors of Grandmaster Fine's analysis in *Basic Chess Endings* is his classification of the position in Fig. 6.1 as a draw. Computer analysis shows that knight-capture or mate can be forced in 12 further moves [23].

BRUTE-FORCE COMPUTING IN TECHNOLOGY

The available power of computation advances at the rate of almost tenfold every five years. For today's large machines one hundred million calculations per second is not abnormal. Measured on a comparable scale, the human brain musters perhaps five per second. The brain, being able to deploy a large associative store of pattern-based concepts, is not normally used in this restricted sequential way. Otherwise feats such as recognizing a person from a photograph, or understanding his speech, would be impossible for a device with such weak calculational powers. On the other hand

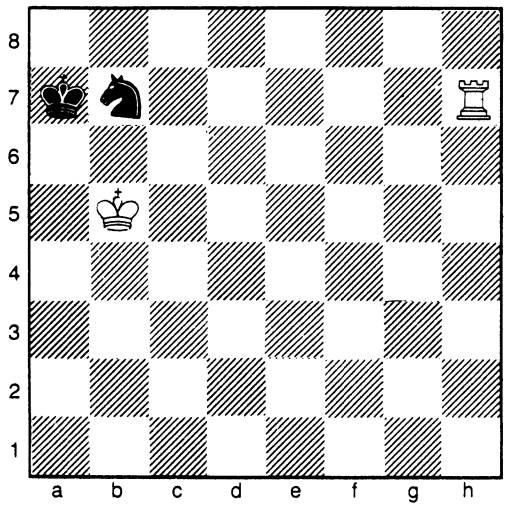


Fig. 6.1 — One of two positions generated in Fine's analysis of the al-Adli position (see text) to which the wrong won/drawn status is assigned. Fine states that, after black's Kb8, white can only draw. Computer analysis reveals that by Kc6 white can then win in 12 moves. As shown in ref. 23, lesser errors abound.

computing systems still rely primarily on brute force. So long as the present rate of advance in hardware technology continues, they can afford to. But can we, their less clever but more intelligent masters, afford to let them?

Several recent incidents have involved complex computer control systems. The suggestion is that reliance on the escalating power of brute force may be heading towards danger. However effective and reliable such systems may be in normal conditions, use of brute force may not be worth the price paid during the rare episodes when a computer-controlled power station or military installation or air-traffic control system malfunctions. On these occasions a new factor becomes paramount: the human operator or supervisor needs to follow what the computing system 'thinks it is doing'.

The computer's processes are measured in millions of steps per second. The human's proceed very slowly — but in a richly furnished space of descriptive concepts. These concepts are not mirrored in any way in the machine's relatively small memory. So when operating conditions stray from the norm, useful dialogue between the two breaks down. In its report on the Three Mile Island accident the Kemeny Commission concluded that the main failures were operator failures, and that the main cause of operator failure was bewilderment by the stream of messages, warning displays and the like from the control computer [24].

If such unsettling phenomena deserve laboratory analysis, then we could hardly find better material than the game of chess. The current world computer chess champion examines a tree of more than ten million possibilities in look-ahead analysis before selecting a move, and is able on this basis to stand up to Grandmasters in purely tactical play. The Grandmasters, by

virtue of their associative stores of conceptualized chess knowledge, have the edge in strategic, or positional play. But as earlier stated a Grandmaster's mental investigation of look-ahead positions averages at most 30. So the kind of mismatch that was noted by the Kemeny Commission, namely between the calculation-rich but concept-poor computer and the calculation-poor but concept-rich human, should be reproducible in the computer chess laboratory. This is indeed the case, as has already been shown through an analysis of the mechanisms employed in state-of-the-art computer chess and its theoretical basis [25].

MACHINE PENETRATION

Machine penetration into complex positions began to reach beyond the human horizon as early as 1977. In the Second World Computer Chess Championship held that year in Toronto, the position shown in Fig. 6.2

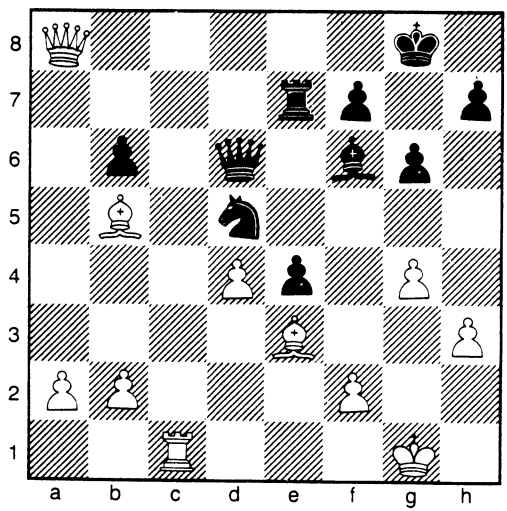


Fig. 6.2 — Position in the Toronto game DUCHESS against KAISSA after white had played Qa8+. Black's play of the apparently meaningless rook sacrifice Re8 was seen by an audience which included several strong chess-masters as a grotesque blunder. Overnight analysis by KAISSA's programmers showed otherwise.

arose in a game between the then reigning champion KAISSA, a Moscow program running on an IBM 168 computer, and the North Carolina program DUCHESS. DUCHESS had just given check with the queen. To the several hundred computer scientists and chess players in the auditorium the only reasonable option was to move the king out of check. KAISSA instead interposed a rook, promptly losing to Q×R check. With a whole rook advantage DUCHESS had no difficulty in crushing KAISSA in another 15

or so moves, and the Russian team handed in KAISSA's resignation in the conviction that they had been robbed by an unsuspected program error.

Next morning Arlazarov and Donskoy announced the result of a retrace operation which had occupied them half the night and had revealed no evidence of error. On the contrary, deep-going scrutiny showed KAISSA's apparent blunder to have been a brilliancy which purchased an extended lease of life for a program which had already arrived in a hopelessly lost position. The rook sacrifice had cleverly averted a mating combination which both KAISSA's and DUCHESS's look-ahead were deep enough to spot, but which eluded onlookers who included former world champion Grandmaster Mikhail Botvinnik.

EXPERT VERSUS MARTIAN SYSTEMS

Now consider chess as a laboratory model of real-life decision-taking. Imagine KAISSA's brute-force computations to be those of an automated control system for a nuclear power station. Let Grandmaster Botvinnik be the engineering supervisor, highly knowledgeable about the domain, assigned to monitor the system's decisions and to intervene with manual over-ride when he judges malfunction to have occurred. The machine makes an unexpected decision. Does the supervisor intervene? Lacking the calculational power fully to probe the system's martian mentality, let us suppose that he does. Disaster follows. Yet had he been able to interrogate the system he would have realized that the seemingly aberrant action was really geared to buying vital time — time in which staff could be evacuated, population warned, ambulances and fire engines summoned, and so forth. Yet he has to decide on the basis of his knowledge and best judgement. Not being a martian, he decides wrongly.

This problem cannot be brushed aside by improvements to the program's surface features, such as better trace and diagnostics and more 'user-friendly' command languages. For the particular case, chosen from 1977 vintage computer chess, this might suffice. But as the depth of calculation increases, a point is reached at which mere surface modifications will not do. Radical reconstruction of the program becomes necessary, using as building blocks machine representations for the very same concepts as those which the knowledgeable human brings to bear on the problem.

This approach leads to a new form of computing system, known as the 'expert system', which is deliberately built in the human rather than martian mental mould. The use of such systems to act as interpretative buffers between the two mentalities was first demonstrated at the Rand Corporation in RITA, a computer program knowledgeable both about the intricacies of the ARPA transcontinental computer network and about the limitations of non-programming personnel desirous of using the network [26].

Brute-force computing is pushing information technology towards regions of complexity which only machines will be able to penetrate. To make it possible for them to report back what they see, RITA-like developments in human-interface software will be required. An eight-year programme of

research and development in advanced computing technology recently announced by the Japan Information Processing Development Centre is based in part on this conception, namely on the design and construction of intelligent knowledge-based systems [27]. Their role will be to mediate betwen the world of machines and the world of people.

FUTURE WORK AND PROSPECTS

Large-scale transfer of human knowledge from books (or brains) to computers has not been achieved in any human intellectual domain. Computer chess is at the leading edge of experimental attempts to achieve it. Endeavours centre round three foci:

(1) The design of data-structures in forms suitable for representing conceptualized knowledge (descriptions, patterns, and theories) which are also convenient for the human user to modify and increment interactively.

(2) Improved facilities for inductive inference, so that programs can acquire new knowledge both from illustrative examples supplied by human tutors, and also from the results of their own internal generation of examples for self-administration.

(3) The engineering of conceptual interfaces between program and human expert, making it easier for the latter to 'teach' the machine.

Advances under all of the above headings are required before the goal of Grandmaster play by machine can be seriously envisaged. By the same token, few investigators doubt the ultimate attainment during the present decade of Grandmaster levels. Apart from benefits to the arts of programming, such an extension of technique also has a bearing on the study of cognition.

I thank Senior Master Danny Kopec for helpful comments.

REFERENCES

[1] Babbage, C. *Passages from the Life of a Philosopher* (Longman, Green, Longman, Roberts & Green, London 1864). Reprinted in *Babbage and his Calculating Engines* (Dover, New York, 1961).

[2] Vigneron, H. *La Nature* 56–61 (1914).

[3] Turing, A. M. *Faster than Thought* (ed. Bowden, B. V.) 286–310 (Pitman, London, 1953).

[4] Maynard Smith, J. & Michie, D. *New Scient.* **12**, 367–369 (1961).

[5] Shannon, C. E. *Phil. Mag.* **41**, 256–275 (1950).

[6] de Groot, A. *Thought and Choice in Chess* (ed. Baylor, G. W.) (Mouton, Paris, 1965).

[7] Binet, A. *Psychologie des Grands Calculators et Joueurs d'Echecs* (Hachette, Paris, 1894).

[8] Samuel, A. *IBM J. Res. Devl.* **3**, 210–229 (1959).

[9] Greenblatt, R. D., Eastlake, D. E. & Crocker, S. D. *Proc. FJCC.* 801–810 (1967).

[10] Hayes, J. E. & Levy, D. N. L. *The World Computer Chess Championship* (Edinburgh University Press, 1976).

[11] Condon, J. H. & Thompson, K. *Advances in Computer Chess* Vol. 3 (ed. Clarke, M. R. B.) 45–54 (Pergamon, Oxford, 1982).
[12] Knuth, D. E. & Moore, R. W. *Artificial Intelligence* **6**, 293–326 (1975).
[13] Berliner, H. Ph.D. thesis, Carnegie-Mellon University, Pittsburgh (1974).
[14] Chase, W. G. & Simon, H. A. *Cognit. Psychol.* **4**, 55–81 (1973).
[15] Tan, S. T. *Advances in Computer Chess* Vol. 1 (ed. Clarke, M. R. B.) 74–88 (Edinburgh University Press, 1977).
[16] Quinlan, J.R. *Machine Intelligence* Vol. 10 (eds Hayes, J. E., Michie, D. & Pao, Y-H) 159–172 (Horwood, Chichester, 1982).
[17] Hunt, E. B., Marin, J. & Stone, P. *Experiments in Induction* (Academic, New York, 1966).
[18] Shapiro, A. & Niblett, T. B. *Advances in Computer Chess 3* (ed. Clarke, M. R. B.) 73–92 (Pergamon, Oxford, 1982).
[19] Shapiro, A. Ph.D. thesis, Machine Intelligence Research Unit. University of Edinburgh.
[20] Clarke, M. R. B. *Advances in Computer Chess* Vol. 1 (ed. Clarke, M. R. B.) 108–115; *Appendix: King and rook against king*, 116–118 (Edinburgh University Press, 1977).
[21] Komissarchik, E. A. & Futer, A. L. *Problemy Kybernet* **29**, 211–220 (1974).
[22] Arlazarov, V. L. & Futer, A. L. *Machine Intelligence* Vol. 9 (eds Hayes, J. E., Michie, D. & Mikulich, L. I.) 361–371 (Horwood, Chichester, 1979).
[23] Kopec, D. & Niblett, T. *Advances in Computer Chess* 2 (ed. Clarke, M. R. B.) 57–81 (Edinburgh University Press, 1982.)
[24] Kemeny, J. G. *et al. Report of the President's Commission on the Accident At Three Mile Island.* Washington D.C. (1979).
[25] Michie, D. *SIGART Newslett. No. 80.* 64–70 (1982).
[26] Waterman, D. A. *Pattern-Directed Inference Systems* (eds Waterman, D. A. & Hayes-Roth, F.) (Academic, New York, 1978).
[27] JIPDEC Fifth Generation Computer Committee *Preliminary Report on Study and Research on Fifth-generation Computers 1979–1980* (Japan Information Processing Development Centre, Tokyo, 1981).

Section 2 Intelligent robots

INTRODUCTORY NOTE TO SECTION 2

It has been noted by moralists that on every issue of significance human society takes two distinct views. It is not that a question is perceived as having two sides, although this is sometimes asserted. On the contrary, social man selects one side, the 'right' side, to which he then proceeds to give two contradictory expressions. Contradiction vanishes when it is realized that the expressions are intended for two different modes of being, concerned respectively with doing and with acting.

The word 'acting' is here used in the theatrical sense, and corresponds, one may say, to the ritual view of some matter, developed to shape and guide mutually supportive feelings within a group. The other view is the operational view, directed towards shaping and guiding reality. Interviewed in Dr Jonathan Miller's television series 'States of Mind', the Princeton anthropologist Clifford Geertz spoke of the accumulated technical lore by which certain Pacific islanders guide the fashioning of their canoes. Along with this lore is another valued accumulation, namely traditional incantations for projecting safe and prosperous futures onto the vessel under construction. The interviewer seemed taken aback: the second approach to boat-building must surely be at best ineffective, at worst vacuous. How could one and the same person practise and believe in both? That misses the point, insisted Professor Geertz, since the domains of application of the two procedures are entirely different and do not overlap. The incantations are not instrumentally motivated in the sense of impacting causally on the external world. Rather they are aimed at regulating a different world, an internal world around which communal consciousness is assembled. It is important that everybody should feel right, and feel in harmony, about the boat-building operation. This is the role of the acting mode — to save the doing mode from its own kind of vacuity.

I want now to introduce a sacrilegious fantasy. Suppose that the tribal boat chants include imaginings of what a superperfect canoe should be —

how it should look, how it should handle in the water and progress at magical speeds, a boat for the gods. Suppose that these passages are very detailed and could in principle be put together and interpreted as a functional specification. There is little danger that anyone will disrupt the practical building work by actually trying to do this. For one thing the technology of the islanders, optimized over centuries, contains its own detailed performance specs suited to the constraints of the traditional methodology. In addition it may be supposed that trained craftsmen can in any case distinguish between dreamings and blue-prints, between religious art and technology, between acting and doing.

Suppose, however, that some canoe-builder, perhaps with the aid of some of his peers, begins to attempt just this. In the process of straining for forms and features not attainable by the old ways, his breakaway group is forced along paths of exotic innovation. How will his fellows in the community feel?

I think that this person is likely to be banned from the boat-yard. He may fail to appreciate this as communal wisdom. He may imagine that it is the fact of innovation itself which has offended. If so, then he will fail, as socially insensitive innovators do, to understand what in the minds of others is at stake. The stake is nothing less than the integrity of accumulated cultural wealth. To revise the dreamings in order to restore them to harmony with changed blue-prints must, to be sure, on occasion be contemplated. For this, care and deliberation must be used as Cardinal Bellarmine repeatedly counselled Galileo. Corruption of the culture *in the reverse direction*, i.e. redirection of technical practice into conformity with literal interpretations of mythology, has less sense and perhaps more danger: so much so that no sane society will encourage it without deep prior reflection.

In Chapter 3 a scientific project to build a robot of a new type was mentioned. Chapters 7–13 record the conception, design objectives, and development of FREDDY. In retrospect I see these chapters as a series of stills from an anthropological disaster movie, eventually resolving happily in Chapter 14's note added in proof.

Robots permeate the myths and legends of all ages. Homer's god of crafts, Hephaestus, built a team of gold androids (more properly 'gynaecoids', since they were female) to labour for him at his forge. Judaeic lore portrays a robotic creature, the Golem, of which we have a modern rabbinical appraisal from the American AI pioneer Azrael Rosenfeld. In the childhood of Mary Shelley, who was later to write the tale of Dr Frankenstein, the Swiss clockmakers were already animating life-size simulations of people. These performed feats of piano-playing and hand-writing on demand, with melodies and messages respectively supplied by the user. The level of human likeness and naturalness of movement then achieved has not subsequently been equalled.

Here then is the acting-mode view of the robot: human-like to the closest attainable degree of simulation, including human-like knowledge and intelligence, if that too can be implemented. The burden of the FREDDY project, and more broadly of this book, is that predictable developments of

automation make it imperative to attempt precisely what I earlier said no sane society will lightly encourage: namely without detriment to operational effectiveness to restructure the technology of man to the forms of his inner world.

As for doing-mode models, these were well established in the 1960s, not only in concept but as practical factory devices: a blind, dumb, insentient race of pick-and-place manipulators. When Richard Gregory and I, meeting in New York in 1966, conceived the FREDDY project we wanted *not* to dedicate yet one more laboratory to the stepwise refinement of what we regarded as a cul-de-sac technology. As is made clear in the 'Tokyo–Edinburgh Dialogue' (Chapter 8), we sought a pattern-oriented approach to cognitive modelling, as a means, among other goals, of promoting change in the methodology of programming. The penultimate paragraph of Chapter 10 seems worth picking out for relevance unaltered over the years:

> Our aim is to contribute to restructuring the arts of programming so that much of what is today done by programmers can ultimately be done by machines. The science of programming needs, as does any other science, such as physics, to develop both theoretical and experimental sides. Robot work is an example of the use of *experimental programming* to validate theoretical conjectures and results and to suggest new hypotheses.

This passage outlining technological objectives was published in 1973, along with a separate statement of scientific intent 'the development of a systematic theory of intelligent processes, wherever they may be found' (Chapter 11). It was a time of great vulnerability for AI robotics. The task was to balance pressures towards theoretical psychology against even less welcome pressures from Government sponsors towards blind pick-and-place automation. My own aspiration lay with neither, but with a new approach to software which I christened 'Knowledge Engineering' to symbolize the aimed-for confluence of programming methodology with machine cognition (Chapter 12).

Chapter 13 was written to rebut ignorant charges that AI scientists had been making overblown promises. It was also an opportunity to give wider exposure to FREDDY's technical triumph reported to the 1973 International Joint Conference on Artificial Intelligence and subsequently in the specialist literature (Pat Ambler, Harry Barrow, Christopher Brown, Rod Burstall, and Robin Popplestone, Vol. 6 of *Artificial Intelligence*, 1975).

At that moment the scene received an incursion from an onlooker of great eminence. His complaint was that it was not easy to distinguish knowledge engineers from people naively trying to apply Mary Shelley's Frankenstein romance and other anthropomorphic myths. In my Pacific island parable, collective wisdom prescribed a measure of discouragement for aberrant canoe builders. To quell AI's impious roboticists the instrument of discouragement was the 'Lighthill Report' entitled *Artificial Intelligence: a general survey,* published by the Science Research Council. In computing

circles this interpolation from a distinguished outsider — Sir James Lighthill is a theoretical physicist — was seen as inappropriate.

It is possible that objections couched in these terms may rest on a category confusion, as though Lighthill expected computer scientists to sit down with solemn faces and read his report as a technical review document. It is clear even from a cursory glance that it was not intended as anything of the kind. It does not seek to assess technical or operational merits, but chooses rather to question from a higher level: is it a respectable activity to take the realities of automation and seek to wire them up literalistically to creatures of folk-lore and science fantasy? Eminence is normally sufficient entitlement to advise at the level of tribal attitudes. If anything was lacking, therefore, it was not so much appropriateness as soundness of the advice.

Lighthill partitioned the domain of discourse with a celebrated ABC:

A — Advanced Automation
B — Bridge, or Building Robots
C — Cognition, or Central Nervous System

with the rider that B was not respectable and should cease.

The FREDDY project has the ABC:

A — Automation
B — Better software (via use of C as a functional spec for A)
C — Cognition

with the rider that B is all-important.

The last Chapter of this Section, number 14, completes the continuing theme. Early in 1985 the project was relaunched in incarnation no. 3 at the Turing Institute in Glasgow. My personal assessment is that, fourteen years on, it is time to take a look at the FREDDY project's own ABC and rider. The eight chapters which follow supply a means for doing exactly this.

7

Integrated Cognitive Systems (1970)

Work is in progress in several laboratories [1–6] directed towards the construction of an integrated cognitive system (ICS). I avoid the phrase 'intelligent robot' because of its science fiction connotation of humanoid appearances and other attributes. The research is concerned with intellectual attributes, involving sensorimotor and reflex capabilities only to the extent that these form a necessary substratum for the acquisition or display by computing systems of purely intellectual skills.

At this early stage the 'intellectual' skills which research aspires to emulate may seem to the onlooker so primitive as scarcely to deserve the name. Let him, however, reflect on the struggles of small children with the simplest tasks of deduction, generalization, and description, and their dogged attempts to construct and refine world-models adequate for their growing needs, representing a succession through which every developed human intellect has passed. Even these first exploits of the infant mind are beyond the abilities of any computing system yet devised. Computers equipped with optical input and manipulative devices are available in at least two laboratories, but understanding of machine perception and cognition has not advanced so far that they could be programmed to compete with human infants, for example on tasks such as the following, which is taken from Stanford-Binet IQ tests [7]. The task involves obeying simple commands, and is designed for 2½ year old infants. With a brick, a button, a dog, a box, and a pair of scissors laid in order on a table, the child is told (a) 'give me the dog'; (b) 'put the button in the box', and (c) 'put the scissors beside the brick'. A machine passing tests of this sort would be disqualified if it had merely been pre-programmed *ad hoc* for each individual test. An artificial intelligence worth the name must show some degree of generality.

Problems of abstracting from the world of crude sensations and of planning and physically doing things in space and time are dominant in intellectual activity at this early stage; possibly they also form an indispensable springboard for the flights of abstract thinking attained later. Emula-

tion by machine therefore demands, as a matter of technical necessity, the design of non-standard computer peripherals to serve as 'eyes', 'hands', and the like. The alternative course would be to simulate entire real-world problem situations inside the machine, a desperate measure of unimaginable scale and cost. R. L. Gregory (private communication) has truly remarked that the cheapest store of information about the real world is the real world, and this indeed is the rationale of the recent emphasis by artificial intelligence projects on 'hand-eye' and 'robot' devices.

How long is it likely to be before a machine can be developed approximating to adult human standards of intellectual performance? In a recent poll [8], thirty-five out of forty-two people engaged in this sort of research gave estimates between ten and one hundred years. There is also fair agreement that the chief obstacles are not hardware limitations. The speed of light imposes theoretical bounds on rates of information transfer, so that it was once reasonable to wonder whether these limits, in conjunction with physical limits to microminiaturization of switching and conducting elements, might give the biological system an irreducible advantage. But recent estimates [9, 10], which are summarized in Tables 7.1 and 7.2, indicate that

Table 7.1 — A comparison of information-handling powers of brain and computer.

	Brain	Computer
Speed	1000 bits traverses 1 neurone in 1 s	1000 bits transferred in or out of core memory in 1 μs
Store	10^{12}–10^{15} bits	10^{12} bits, retrieval 50 ms

This table is based in part on data from ref. [9]. The upper comparison, which appears to give the computer the advantage in speed, is compensated by the brain operating in a highly parallel fashion, as opposed to the sequential processing characteristic of the computer.

this is not so, and that the balance of advantage in terms of sheer information-handling power may eventually lie with the computer rather than the brain. It seems a reasonable guess that the bottleneck will never again lie in hardware speeds and storage capacities, as opposed to purely logical and programming problems.

Granted that an ICS can be developed, is now the right time to mount the attempt? Is it possible that effort should instead be put into some abstract field of philosophy, linguistics, or pure mathematics? Perhaps only by postponing rash attempts to construct actual systems can a sufficiently deep understanding be gained to enable artificial intelligence problems to be tackled in the right way.

Theoretical studies are certainly central. But it is not clear that they

Table 7.2 — The supply position of mechanical energy and mechanical information processing capacity.

(a) Ratio of energy available from mechanical sources and from human muscle power

Year	1500	1700	1800	1900	1945	1965
ER	10^{-4}	10^{-3}	10^{-1}	1	5	10

Total human muscle power potentially available is assumed to be of the order of $0{\cdot}25\ 10^5 P$ Wh/annum (P = world population). For 1965 it has been assumed that mechanical energy supply was of the order of 4×10^{16} kcalories or $0{\cdot}75\times10^{16}$ Wh at 15–20 per cent conversion efficiency.

(b) Ratio of mechanical and human information processing capacity

Year	1955	1965	1970	1975
CR	5×10^{-4}	10^{-2}	2	50

This comparison is based on those tasks where the human channel capacity of *c.* 20 bits/s is a major rate-determining factor, as may be the case in many routine clerical operations and computations.

As in (a), (b) attempts to compare facilities available world wide. Both assessments, and in particular (b), are obviously very tentative only.

would be aided by abstention from experimental work. Indeed the lessons of history point in the opposite direction, as with the relationship between classical thermodynamics and the development of the steam engine. Typically engineering artefacts come first, and provide the theoreticians not only with the needed spur to rationalize what is being done, but also with test gear on which to check their formulations. There are elements of a similar relationship between the robot building now in progress in various laboratories and the recent spate of activity by theoreticians in the same laboratories in two areas in particular: visual scene analysis by machine [1, 3, 11], and construction and formal description of abstract models of the world for use as the basis of plan-formation [12–14].

Yet the principle of 'unripe time', distilled by F. M. Cornford [15] more than half a century ago from the changeless stream of Cambridge academic life, has provided the epitaph of more than one premature technology. The aeroplane industry cannot now redeem Daedalus nor can the computer industry recover the money spent by the British Admiralty more than a hundred years ago in support of Charles Babbage and his calculating machine. Although Babbage was one of Britain's great innovative geniuses, support of his work was wasted money in terms of tangible return on investment. It is now appreciated that of the factors needed to make the stored-program digital computer a technological reality only one was missing: the means to construct fast switching elements. The greater part of a century had to elapse before the vacuum tube arrived on the scene.

It can reasonably be said that time was unripe for digital computing as an industrial technology. But it is by no means obvious that it was unripe for Babbage's research and development effort, if only it had been conceived in terms of a more severely delimited objective: the construction of a working model. Such a device would not have been aimed at the then unattainable goal of economic viability; but its successful demonstration might, just conceivably, have greatly accelerated matters when the time was finally ripe. Vacuum tube technology was first exploited for high-speed digital computing in Britain during the Second World War [16]. But it was left to Eckert and Mauchly [16] several years later to rediscover and implement the conceptions of stored program and conditional jumps, which had already been present in Babbage's analytical engine [17]. Only then could the new technology claim to have drawn level with Babbage's design ideas of a hundred years earlier.

How is ripeness of time to be diagnosed for projects to build intelligent machinery? The only absolute safe method is to wait until someone does it, just as the only error-free pregnancy test is to wait for the birth of a demonstrable infant. We would prefer, I think, to detect what is still in the womb, and accept a risk of being wrong. Table 7.3 shows four possible

Table 7.3 — Check list of signs and symptoms for the early stages of five technological projects.

	A	B	C	D	E
(1) Multiplicity of effort (i.e. how many laboratories?)	√	√	×	√	√
(2) Availability or feasibility of all essential instrumentation	×	√	×	√	√
(3) Demonstration of a working model	×	√	×	√	(?)
(4) Theoretical proof of overall feasibility	×	×	×	×	×
Fate: S succeeded F failed	F	S	F	S	?

(A) Transmutation of elements in the time of alchemy; (B) steam engines in the time of Watt; (C) stored-program digital computing in the time of Babbage; (D) heavier-than-air flight in 1900; (E) intelligent machinery in 1970. The symbol '(?)' is used to mean 'possibly imminent' in distinction from '?' which means 'undecided'.

criteria of ripeness which might be applied, from the outside as it were, to budding technological enterprises.

The four criteria are listed in decreasing order of superficiality. Criterion 1 is of a kind which can be applied by a policy-maker 'off the cuff',

without delving into technical considerations. It says: 'This is not a bee in an isolated bonnet. Laboratories all over the world are in the race. Can we afford not to join?'. The fact that it is so widely persuasive stems chiefly from the fact that it is superficial, and hence cheap and easy to apply: also it acts on the 'keeping up with the Joneses' reflex. For what it is worth it is favourable to the proposal to construct an integrated cognitive system, which is being studied actively in America, Britain, and Japan. But we must remember that the highest recorded score for this particular criterion could probably be claimed for the alchemists.

Jumping to the other end of the range, criterion 4 is in general of little assistance because of the 'pregnancy test' argument. Overall theoretical analysis is usually only achieved on the morrow of success. An interesting counter-example is Lovell's project to build a giant radio telescope at Jodrell Bank [18]. Here criterion 4 was satisfied in advance. But of course there was never any problem concerning unripeness of time in the feasibility sense. All that was in question was the balance of expected benefits against expected costs. This issue was finally settled when the first Russian sputnik was successfully tracked by an instrument designed for other purposes, and the world found itself dependent on Jodrell Bank for accurate data on satellite orbits. This decisively ended the era of uncertainties of funding for the project. One may wonder whether some element of international 'keeping up with the Joneses' was at work here, to the great good fortune of astronomical science.

An even more interesting case is radio broadcasting. Feasibility was shown theoretically by Maxwell in 1865, and verified experimentally by Hertz about twenty years later. Shortly afterwards Popov and Marconi independently achieved the first 'working models'. An equally clear example is the modern (junction) transistor, the basic action of which was predicted by W. Shockley in 1949. A further application of criterion 4 arises if theoretical infeasibility is demonstrated, as in the case of the perpetual motion machine. Anti-gravity is another example, in spite of rumours of continued multiplicity of effort behind the security curtain. But it is well to look on such negative proofs with caution. The possibility of broadcasting radio waves across the Atlantic was convincingly excluded by theoretical analysis. This did not deter Marconi from the attempt, even though he was as unaware of the existence of the Heaviside layer as everyone else.

INSTRUMENTATION

To summarize the uses of the four criteria, no. 1 is so weak that it is better left alone, while no. 4 is so strong that it is usually not available in real cases of doubt. We are therefore thrown back on criteria 2 and 3. It will be interesting to relate these, if possible, to the present scene in machine intelligence research. First, I shall consider availability of instrumentation. Four categories are involved: (1) computing hardware; (2) programming systems; (3) utility packages (such as deduction routines, parsing routines,

learning routines, search routines and so on), and (4) 'robot' input–output devices.

These can be interpreted as stages in a development programme. First, get your computer. Then develop a software and programming language base adequate for the needs of machine intelligence research. Only then is it feasible to build a library of useful packages and to construct special peripherals such as 'hand–eye' attachments. The next step, at least in aspiration, is the construction of a working model of an intelligent machine. Most workers who partake of this aspiration would, I think, agree that categories 1–4 are beginning to be in reasonable shape to provide the tools for the job.

WORKING MODEL

A working model is almost a necessary condition of confidence in the feasibility of any proposed technological innovation. It is by no means a sufficient condition. New and possibly prohibitive difficulties may be brought into being by the scaling-up process. We would do right to be impressed by a power-driven model aeroplane. But suppose that an inventor proposed to develop a man-sized jumping machine able to clear the top of St Paul's Cathedral. A flea-sized model jumping a similar multiple of its own height would scarcely be convincing. So if the objective which I shall discuss were to be attained within the next few years, this would by no means imply that an intelligent machine was round the corner. But it might indicate that significant success at the scaled-up level was perhaps only a decade or two away.

What is meant by a working model of an intelligent machine? The best approach is to map out roughly the principal constituents of such a machine's 'mental world', and then say that a working model is constructed on something like the same overall plan, differing only in the relative poverty of the individual subsystems which are linked together. Also it must be 'working' in the sense of displaying the various constituents and their collective operation in interacting with real-world problems; for example problems of the type illustrated by the Stanford–Binet tests for infants which I cited earlier. An ICS able to operate at this level would bear the same relation to the intelligent machines of the future as a powered toy aeroplane to passenger-carrying airliners. But even such a primitive ICS has two rather interesting features: first, its achievement lies, in the opinion of some workers, only a few years in the future; and second, such an ICS could almost certainly be made the basis of an industrial development programme to produce before the end of the 1970s a range of commercially useful devices.

APPLICATIONS

At Edinburgh we recently commissioned the consultant firm Scicon Ltd to do a study addressed to the question: 'Assuming solution of the technical problems, what industrial applications can be envisaged for the late 1970s?'

The possibilities included in their report were, first, anchored devices for luggage-handling at airports, crane-controlled assembly, and automatic control and inspection of machine tool output; and second, free-roving devices for exploratory vehicles for the space programme, ocean bed exploration, laying pipelines in deserts, and tree-felling in remote forests.

Activities such as these in inhospitable or inherently unstructured environments are difficult and expensive when conducted by conventional means, so that any prospect of delegating them to cognitive machines will have attractions. At Edinburgh we are assisting in a feasibility study of automatic parcel-handling for the Post Office Telecommunications Headquarters, and a Japanese group [5] are interested in applications of ICS work to assembly line operations. The robot project at Stanford Research Institute envisages exploratory vehicles of various kinds as a major industrial payoff.

PROGRAMS AND PLANS

Leaving industrial implications, I shall now consider the quintessential activity of an integrated cognitive system, in the sense that locomotion is the quintessence of a motor car. This, it can be argued, is planning, for it is by the relative absence of this activity that we recognize that existing automatic systems of prediction and control, however sophisticated and 'clever', are not true examples of intelligent behaviour. Further, when we speak of machines able to form plans and to reason about the adequacy of a plan to a given task we can be quite precise about what we mean, by pointing out that a plan of action can be usefully treated as formally equivalent to a computer program ('plan of computation'). Forming a plan is then seen as having the same logical status as writing a program, and validating a plan as utilizing the same mathematico-logical apparatus which programming theorists have developed in recent years for proving things about programs. This insight, elaborated recently by C. C. Green [12], is of profound importance for the future development of artificial intelligence, and may well be destined to occupy a place as central as, say, the equivalence of the corpuscular and wave models in the theory of optics.

An integrative cognitive system, then, can be conceived as a plan constructor and plan implementer. At any moment it is either in interactive mode or in planning mode. 'Planning mode' is interpreted in a broad sense to include all processes of inference involved in the formation of new plans, including purely internal reorganizations or extensions of stored descriptions.

In interactive mode it is executing a plan. No reasoning in the sense used here occurs in this mode, until an interrupt generated by an input device causes reversion to planning mode. Input (for example 'eye') and output (for example 'hand') devices interact directly with an external world in such a way that the next sense-datum received is dependent on the past input–output sequence. The precise form of this dependency is governed by the

laws of nature. It is part of the business of an ICS to form an approximate picture of these by processes of abstraction.

What else is in memory, apart from plans, and to what top-level control are the contents of the memory subject? Four major categories are envisaged (Fig. 7.1): 'plans' (in the form of programs); 'images' (in the form of

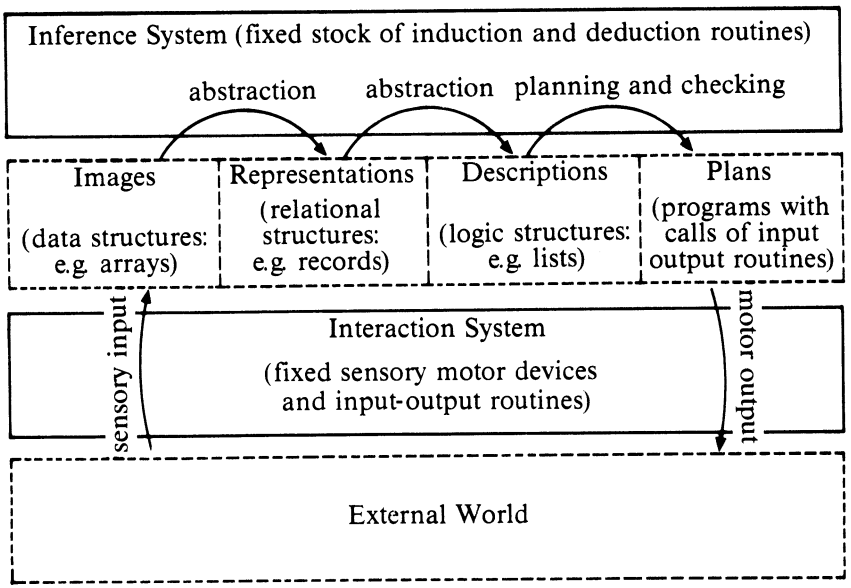


Fig. 7.1 — Schematic representation of relations between an integrated cognitive system and its external and internal worlds. The fixed systems are shown with solid lines, while those which are subject to change in the interaction process are drawn with broken lines.

data structures); 'representations' (for example in the form of relational structures), and 'descriptions' (in the form of logic sentences).

At the top level a common inference system operates on: (*a*) plans, not only to construct them, but to verify in advance that they will work, using in the processs (*b*) images, which are direct point-to-point projections of objects in the external world (for example a map is an image of a particular geographical area); (*c*) representations, which model objects in the external world by abstracted features, and (*d*) descriptions, usually called 'theories', which make general statements about objects in the external world and their relations to each other in space and time.

It is worth commenting on the extraordinary faithfulness with which the brain can store images: the extreme case is 'eidetic imagery' in which a visual pattern can be stored for periods in considerable detail. It is possible, and has been argued by Richard Gregory [19], that this type of direct modelling

is biologically more primitive, and hence has had time to evolve to a higher pitch, than the storage and sequential processing of symbolic information, as in natural language and in formal reasoning.

AUTOMATIC PLAN-FORMATION

The idea of getting a computer to write its own programs has appeared and disappeared several times in the past two decades. Early attempts, inspired by the example of biological evolution, were based on generating program symbols randomly, conserving the more successful sequences [20,21]. Such an approach is now considered naive, and nature tends to be thought a poor model for cost-conscious designers. Present ideas centre round the systematic construction of a program, either as a side product of mechanically proving from given axioms that the task which the program is to accomplish is theoretically capable of accomplishment, or alternatively as an end-product of a process of heuristic search. Keeping in mind the formal analogy between programs and plans, it may help fix ideas to consider an example from an unpublished study by Popplestone in which elements of both approaches are used.

The theorem-proving approach, developed by C. C. Green, uses the apparatus of formal logic to form plans of action. There are difficulties in the approach. One is the 'frame problem': it is necessary to say not only what things are changed by an action, but also what remains unchanged. Not only are the frame axioms tedious to write, but they also tend to lead the theorem-proving process astray. This raises the second principal difficulty, the Achilles heel of present-day mechanized proof procedures, that they very easily stray into unprofitable inference paths through lack of any adequate formulations of the notion of relevance.

An alternative approach to plan-construction, suggested by the work of Floyd [22], is to start at the goal and work backwards to the present situation. The goal is represented by a sentence, which is conceived as having been deduced from the conjunction of a preceding situation (presented by a sentence) and an antecedent action. A backwards search tree can be grown until a state-description is produced which is a logical consequence of what is known about the initial pre-planning situation. This process, which is currently being developed by R. J. Popplestone, is illustrated in Fig. 7.2 for a housekeeping task in a world furnished with a cupboard, a table, and a chair. Initially the cupboard contains exclusively forks and the robot's hand is at the chair. A plan is required to create a situation in which at least one fork is at the table. The example is exhibited here for its didactic value rather than for its originality of approach. Novel features do, however, exist (*a*) in the way in which general statements, rather than detailed specifications, are handled in the search, and (*b*) in Popplestone's method for guiding the search heuristically. For this he uses a notion of approximation to the desired condition of being logically implied by the initial situation. The degree of approximation is estimated from the

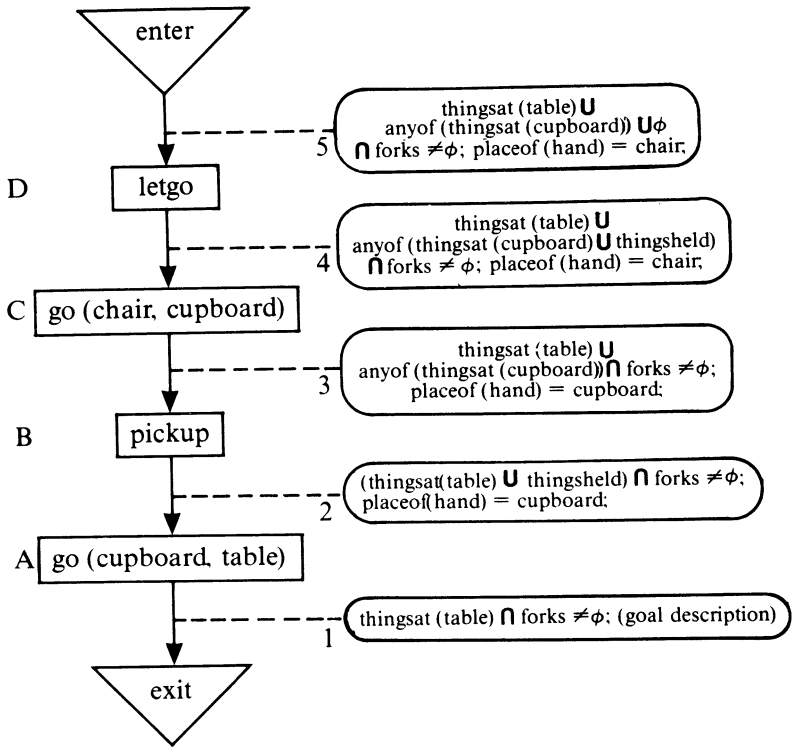


Fig. 7.2 — Successful action-chain constructed by backwards search from the goal situation, using logical inference to associate descriptions with successively earlier situations. The process terminates when a description is produced which is a direct logical consequence of the initial state-description, in this case giving the sequence D, C, B, A as the answer. A plain-language transcription of comments 1–5 might read as follows: (1) there is at least one fork at the table; (2) the things at the table together with the things held in the hand include at least one fork, and the hand is at the cupboard; (3) the things at the table together with a random selection from the things in the cupboard include at least one fork, and the hand is at the cupboard; (4) the things at the table together with a random selection from the things distributed between the cupboard and the hand include at least one fork, and the hand is at the chair; (5) the things at the table together with a random selection from the things in the cupboard include at least one fork, and the hand is at the chair. This last statement can be obtained as a logical deduction from the initial state-description: thingsat (cupboard) ≠ø; thingsat (cupboard) ⊂ forks; placeof (hand) = chair. There is also a goal description: thingsat (table) ∩ forks ≠ø. The method by which description (2) is obtained from description (1) and action A, (3) is obtained from (2) and B, and so on is due to Floyd [22].

number of interpretations of the description of the initial situation which are inconsistent with the current one. The smaller this number the more 'promising' the given situation as a point of departure for extending the backwards search.

'HAND-EYE' PROBLEMS

The provision of a suitable formal basis for reasoning about even such simple systems is by no means trivial, and the properties which must be possesed by a satisfactory calculus of situations and actions have been re-examined by McCarthy and Hayes [13] and by Burstall [14]. In several laboratories, including our own in Edinburgh, experiments are in progress with various 'hand-eye' and 'robot' attachments to computers in order to provide instrumentation and a software base adequate to put such calculi to the test of practice. Each laboratory doubtless has its own graded repertoire of tasks with which to challenge its local evolving ICS. Our schedule [6], designed to fill the next two to three years, is concerned with six types of task: first, as far as vision is concerned we wish to develop a machine that will identify single objects (definitions provided by the programmer) placed within the field of vision, and learn to identify single objects (by generalizing from examples); second, manipulating a hand to move to any accessible prescribed position and pick up an isolated object, and also an object from a group. The third task is that of world modelling to enable the machine to 'know' relative locations of objects and 'self', and to update the internal model on the basis of sensory input; to integrate several views of an object and tactile information. The fourth is simple planning by simulation (graph traversing) [23], and is concerned with planning a route for 'self' from one location to another, avoiding contact with obstacles and planning movements of the hand to pick up an object from a set and move it without disturbing other objects, also executing plans and reforming them if they fail, or are about to fail. The fifth task is that of higher-level planning (theorem proving) [12] whereby, given a set of world axioms, means of achieving simple states are designed, for example 'Go to a cube', 'Put a ball into a cup'. The sixth task we have set ourselves comes under the heading of generalization (induction): to learn general statements about simple events from specific occurrences, e.g. '"Put cup onto ball" implies failure', '"Put ball onto cup" implies success'; to generalize from several similar specified observations, for example '"Put anything onto ball" implies failure'; and eventually to generalize to qualified sets, e.g. '"Put anything flat onto cube" implies success'.

EDUCATIONAL AIDS FOR YOUNG CHILDREN

There are similarities between some elements of our schedule and the Stanford–Binet tests for infants. This circumstance justifies a speculative postscript to the earlier list of industrial uses. It is generally agreed that an important application for advanced computer systems will be in educational technology. It is also already apparent that very young children, for example of primary school age, are in some ways the most rewarding subjects, because the teacher to child ratio is too low fully to satisfy the young child's appetite for continual responsiveness. Anyone who has watched 6 year olds

wrestle absorbedly, through the complexities of the teletype, with computer-supplied arithmetic homework cannot help being struck by the motivating power of the interactive terminal. What about even younger age-groups or mentally handicapped older children? Something can be done using the cathode ray display and voice output, permitting communication between child and machine in pictures and words. This possibility is being investigated in our laboratory among others. But just as the human teacher supplements pictures and words with direct demonstrations, by manipulation, for example, of cuisenaire rods for arithmetic, of buttons and beads for sets, of cups and sand and liquid for conservation laws and so on, so it may turn out that when computer terminals can be equipped with adequate 'hand-eye' capability these too will be pressed into service as teachers' aids.

The possibility of such a development deserves serious attention. It is particularly attractive for the research worker who likes to have some specific application in mind, because the subject matter of infant teaching has a certain relevance to the intellectual content of artificial intelligence research: namely the explication of real-world phenomena in terms of basic logical and mathematical concepts. To the lay onlooker, however, there may seem to be something de-humanizing, even psychologically dangerous, in the exposure of the very young to interaction with machines. I shall not trespass on the province of the educational psychologist, beyond saying that in our own work with small children at Edinburgh [24] this criticism has indeed been encountered in some quarters; but it has been conspicuously lacking from three specific categories of person, (1) the teachers, (2) the children's parents, and (3) the children themselves.

REFERENCES

[1] J. Gresser (1968) *MIT Project MAC Art. Intell. Mem. 165*; A. Guzman (1968) thesis *MAC-TR-59 AD-692-200*; A. Guzman (1968) *Amer. Fed. Information Processing Soc.*, **33**, 291; S. Nelson & M. Levitt (1969) *MIT Project MAC Art. Intell. Mem. 172*; H. N. Mahabala (1969) *MIT Project MAC Art. Intell. Mem. 117*; B. K. P. Horn (1969) *MIT Project MAC Art. Intell. Mem. 178.*

[2] J. McCarthy, L. D. Ernest, D. R. Reddy & P. J. Vicens (1968) *Amer. Fed. Information Processing Soc.*, **33**, part 1, 329; J. A. Feldman, G. Falk, G. Grape, J. Pearlman, I. Sobel, & J. M. Tenebaum (1969) *Proc. Intern. Joint Conf. Art. Intell.* (eds D. E. Walker & L. M. Norton), 521; M. Heuckel (1969) *SAI Project Publication AIM-105*; R. Paul, G. Faulk, & J. A. Feldman (1969) *SAI Project Publication AIM-101*; G. Falk (1969) *SAI Project Publication AIM-107*; C. C. Green (1969) *SAI Project Publication AIM-96.*

[3] N. J. Nilsson, G. Forsen, C. A. Rosen, L. Chaitens, B. Raphael, & S. Wahlstron (December 1968) *Report for Rome Air Development Center, Griffiss Air Force Base, New York 13440*; P. E. Hart & R. O. Duda (1969) *SRI Project 7494*, AIG Tec. Note 3; C. L. Fennema & R. Brice (1969) *SRI Project 7494*, AIG Tec. Note 7; N. J. Nilsson (Febru-

ary 1969) *SRI Project 7494, Report for Rome Air Development Center, Griffiss Air Force Base, New York 13440*; N. J. Nilsson (1969), *Proc. Intern. Joint Conf. Art. Intell.* (eds D. E. Walker & L. M. Norton), 509; C. C. Green (1969) *Proc. Intern. Joint Conf. Art. Intell.* (eds D. E. Walker & L. M. Norton), 219; B. Raphael (1969) *SRI Project 7494*, AIG Tec. Note 13.

[4] L. L. Sutro & W. L. Klimer (1969) *Proc. AFIPS Spring Joint Computer Conf.*, 113.

[5] H. G. Barrow, D. Michie, R. J. Popplestone, & S. H. Salter, *Computer J.*, **14**, no. 1, 91. Also below, Ch. 8, pp. 000–000.

[6] H. G. Barrow (1970) *Research Memorandum MIP-R-72*. University of Edinburgh: Department of Machine Intelligence and Perception.

[7] L. M. Terman & Maud A. Merrill (1961) *Stanford-Binet Intelligence Scale Manual for the third revision, Form L-M*, 93, 97. London: George G. Harrap & Co. Ltd.

[8] *European AISB Newsletter*, no. 9, 4 (1969).

[9] H. C. Davis (27 October 1969) *Scientific Research*, 21.

[10] P. L. Bargellini (1965) *Adv. Comput.* **6**, 195. Cited by H. F. Damners (1968) *Inform. Storage Retrieval*, **4**, 113.

[11] M. Kelly (1970) thesis. Stanford University: Department of Computer Science.

[12] C. C. Green (1969) in *Machine Intelligence 4* (eds B. Meltzer & D. Michie), 183. Edinburgh: Edinburgh University Press.

[13] J. McCarthy & P. J. Hayes (1969) in *Machine Intelligence 4* (eds B. Meltzer & D. Michie), 463. Edinburgh: Edinburgh Universtiy Press.

[14] R. M. Burstall (1970) *Research Memorandum MIP-R-73*. University of Edinburgh: Department of Machine Intelligence and Perception.

[15] F. M. Cornford (1953) *Microcosmographia Academica*. Cambridge: Bowes and Bowes. (Recent American edition published by Barnes and Noble, New York, 1966).

[16] J. McCarthy (1966) *Sci. Amer.*, **3**, no. 215, 65.

[17] L. F. Menabrea (1963) in *Faster than Thought* (ed B. V. Bowden), appendix 1, 341–408. London: Pitman. (From Bibliotheque Universelle de Geneve, no. 82, October 1842.)

[18] B. Lovell (1968) *The Story of Jodrell Bank*. London: Oxford University Press.

[19] R. L. Gregory (1968) *Bionics Research Reps.*, no. 1, 18. Edinburgh.

[20] R. M. Freidberg (1958) *IBM J. Res. Dev. 2*, 2; R. M. Freidberg, B. Dunham, & J. H. North (1959) *IBM J. Res. Dev. 3*, 282.

[21] L. J. Fogel, A. J. Owens, & M. J. Walsh (1966) *Artificial Intelligence through Simulated Evolution*. London: Wiley.

[22] R. W. Floyd (1967) in *Mathematical Aspects of Computer Science*, 19. Amer. Math. Soc., Providence.

[23] D. Michie (1970) *Computer J.*, **14**, no. 1, 96.

[24] D. Michie & J. A. M. Howe (1970) *Scottish Educational J.*, **53**, no. 9, 200.

8

Tokyo–Edinburgh dialogue on robots in artificial intelligence research (1971)†

At the Conference of the International Federation of Information Processing Societies, which was held in Edinburgh in 1968, E. A. Feigenbaum of Stanford University, USA, delivered a paper entitled 'Artificial Intelligence: themes in the second decade'[1]. In it he said:

'History will record that in 1968, in three major laboratories for AI research, an integrated robot consisted of the following:
(a) a complex receptor (typically a television camera of some sort) sending afferent signals to...
(b) a computer of considerable power; a large core memory; a variety of programs for analysing the afferent video signals and making decisions relating to the effectual movement of...
(c) a mechanical arm-and-hand manipulator or a motor-driven cart.

The intensive effort being invested in the development of computer controlled hand-eye and eye-cart devices is for me the most unexpected occurrence in AI research in the 1963–68 period.

†This chapter was written with H. G. Barrow, R. J. Popplestone, and S. H. Salter.

Since then research on computer-controlled robots, as a major aid to artificial intelligence research, has proceeded apace, for example in the three laboratories mentioned by Feigenbaum, directed respectively by M. Minsky at MIT, J. McCarthy at Stanford University, and C. Rosen at Stanford Research Institute.

Recently, Japanese groups have been entering the field in strength, notably the Electro-technical Laboratory in Tokyo. This laboratory was represented by S. Tsuji on a survey team of robot engineering recently sent on a world tour by the Japan Electronic Industry Association under the leadership of Professor Y. Ukita. The team paid a visit, among other ports of call, to the Department of Machine Intelligence and Perception, University of Edinburgh, and submitted a list of thirty-five questions concerning the project in progress here. We found it an extremely useful and clarifying exercise to answer these questions, which seem to us wide-ranging and shrewd.

Since the aims and content of artificial intelligence research, and of experimentation with robot devices in particular, are not yet widely known outside a very few specialist groups, there may also be benefit in making the dialogue available to a wider scientific readership. We produced the text of the exchange below:

GENERAL

(1) Q What is the purpose of your research on intelligent robots?

A To investigate theoretical principles concerning the design of cognitive systems and to relate these to the theory of programming. To devise adequate methods for the formal description of planning, reasoning, learning and recognition, and for integrating these processes into a functioning whole. In terms of application (long-range) we can envisage a possible use of an intelligent robot as a teaching machine for young children. But our project is a research project, not an application project. Robots for us play the role of test gear for the adequacy of the formal descriptions referred to above.

(2) Q Which do you think most important in your research — scene analysis, problem-solving, dexterous manipulation, voice recognition or something else?

A Problem-solving.

(3) Q Do you have a plan for developing any new hardware for manipulators, locomotion machines or special processors for vision?

A We plan to use equipment already developed by ourselves and others, and we prefer to simulate locomotion by movement of the robot's world as first suggested to us by Mr Derek Healy. The present 'world' is a 3 feet diameter sandwich of hardboard and polystyrene which is light and rigid. It rests on three steel balls and is moved by wheels, driven by small

stepping motors, mounted on the anchored robot. A pair of bumpers, one in front, one behind, operate two microswitches to determine contact with obstacles. Our next piece of equipment is a platform 5 feet square which may be moved anywhere in a 10 feet square by flexible drive wires from two servo-motors. The platform can carry weights of 200 lbs and will move at up to 10 inches per second with accelerations of 1/10 *g*. Various types of hand–eye systems may be hung from a bridge above the platform.

(4) Q We assume that the speed of available digital computers is still too slow for real-time processing of complex artificial intelligence problems. Is this true? If so, do you have any ideas for solving the difficulty?

A We agree that the speed of available computers is still too slow, especially for sophisticated peripheral processing such as vision. Dedication of satellite processors to sub-tasks (e.g. pre-processing the video signal) is one approach. Special-purpose hardware could of course increase the speed of processing, but it seems doubtful whether it can exhibit behaviour of great logical complexity which a digital computer is capable of doing. An improved instruction set, or more parallel computation (multi-processor) may yield significant improvements. But the immediate obstacles lie in fundamental problems of software design, rather than in hardware limitations.

(5) Q Which language do you use in robot research, FORTRAN, ALGOL, PL/1, ASSEMBLER, LISP or other list processing language? What would be the features of robot-oriented languages?

A We use POP-2 [2], [3]. The nearer a programming language is to a fully general mathematical notation, the more open-ended its structure, and the more flexibly adapted to conversational use, then the better the language for robot research. We feel that an ideal robot-oriented language would be one that dealt in relations as well as functions, and would have deductive and inductive capabilities.

(6) Q Can you describe the software hierarchy structure in your robot system?

A The mechanism of hierarchy is simply that of function call and a typical hierarchy might be (example taken from the vision hierarchy)
top — program for guiding object recognition.
middle — region-finding program and program for matching relational structures.
bottom — eye control program.

(7) Q What performance capability do you predict for intelligent robots in 1975?

A We expect demonstrations of feasibility before 1975 in the child teaching machine application; that is a system able to recognize and manipulate materials used in teaching children the elements of arithmetic, sets, properties and relations, conservation laws etc.

(8) Q Will there be any chance of applying the newly developed techniques

in research on intelligent robots to some industry (for example assembly line) in the near future?

A We see possible industrial applications in the late 1970s including assembly line. Other conceivable applications are luggage handling at airports, parcel handling and packing, machine tool control and repair, and various exploratory vehicles, e.g. for pipe-laying in deserts, forest clearing in remote areas, ocean-bed work and planetary exploration. Applications for cognitive vehicles will probably remain restricted to work in environments which are essentially intractable.

(9) Q What do you think of the control of many industrial robots by a mini-computer? What level of 'intelligence' would such a computer-robot system have?

A We would certainly expect to see the control of many 'fixed program' robots by a mini-computer. Such a system would not show much intelligence.

(10) Q May we know the budget and manpower available for your project?

A We have £500 per annum from the Science Research Council for 'construction of models for on-line control experiments' supplemented by small sums earned as revenue through consultancy and rental of computer time. In additon the GPO Telecommunications Headquarters have awarded a contract for £10 000 over two years specifically for the robot research.

The mechanical engineering for our Mark 1 robot, costing about £1000 to construct, was largely the work of Mr Steve Salter of the Bionics Research Laboratory of this Department, at that time directed by Professor R. L. Gregory and supported by the Nuffield Foundation. The electronics, interfacing and software have been mainly done in the Experimental Programming Unit by one grant-supported research scientist working part-time on the robot work (Dr Harry Barrow) and one University Lecturer (Mr Robin Popplestone). But the work is being carried out in the general context of a large-scale study of machine simulation of learning, cognition and perception, financed on a generous scale by the Science Research Council (£260 000 over five years) and by the University of Edinburgh. The POP-2 software and conversational computing system has received support also from the Medical Research Council to the amount of about £70 000 over five years. About a dozen research scientists are employed in the general project. Seven of these constitute a 'Robot working party' which meets fortnightly under the chairmanship of Professor Donald Michie, and plans the robot work, but this is a side-line activity for them with the exception of the workers mentioned above.

EYE

(1) Q What are the aims and targets of your research in the context of vision?

A Picture-processing performance should be sufficient for forming

plausible recognition hypotheses concerning members of a limited repertoire of simple objects (e.g. ball, pencil, cylinder, wedge, doughnut, cup, spectacles, hammer) as a basis for experimental verification or modification of such hypotheses by the robot through action (changing angle of view or interfering with objects manually).

(2) Q Which input device do you use: vidicons, image dissector tubes, or other special devices?

A We use vidicons but are investigating image dissectors.

(3) Q What is the performance of the input devices in areas such as resolution, dynamic range, sampling rate of A to D converters? In such areas are there any possibilities of improving the input devices?

A Present resolution of TV sampling system is 64×64 points and 16 brightness levels. Speed of conversion of A to D converter is approximately 100 kHz. This system is to be improved to 256×256 points and 64, or more, levels. A to D conversion should be about the same rate.

Sampling time for a picture point is largely determined by the time taken for the TV scan to reach the point (up to 20 ms maximum). We are considering image dissectors, which have negligible settling time.

(4) Q Do the eyes of your robot move (electronic or mechanical movement)? What are the merits of eye movement?

A The eye does not move relative to the main frame. We are considering relative movement of two eyes for depth perception. Also, we are considering using one camera for wide angle views and a second camera with a long-focus lens for investigation of details. Merits, obvious; demerits, complication.

(5) Q Is there any processor for visual input? Is it special hardware? What is the role of the preprocessor?

A We have installed a small processor for pre-processing visual input and thus reducing the load on the multi-access system. Later on we may build special hardware, for instance for doing ranging by stereoscopic or focusing methods. In the case of the stereoscopic method we would probably use hardware correlators. We might also build hardware contour followers for the region analysis approach, if it could be shown that a very significant saving in processing time would result.

(6) Q Do you use linguistic methods to recognize the picture input? Is there any trouble when the line drawing of the solids suffers noise? How do you solve the shadow and hidden line problems? What is the most complex solid which your robot can recognize?

A We are experimenting with a method which involves describing pictures in terms of properties of regions and the relationship between regions [4, 5]. We believe that the system will be moderately immune to

noise. The shadow problem will be solved initially by allowing the combination of regions of different intensity level to form a new region and trying recognition again. Later we might attempt to decide whether something was a shadow or not by measuring differences in texture or distance on each side of boundaries between areas of different light intensity.

At present the robot is capable of recognizing the simple objects described under heading (1) of this section, under controlled lighting conditions and viewing them from a roughly standard position.

(7) Q Does your robot have colour sensing? What are the merits of this?

A No. Colour sensing would, however, undoubtedly aid region analysis and also facilitate communication with the human user concerning a given visual scene. It would be easy to have a single colour-sensitive spot in a moving eye system.

(8) Q How do you solve the difficulties of texture?

A At present we have no method of coping with texture. In the future we will think of dealing with it by ideas like spatial frequency and spatial correlation, e.g. for distinguishing between textures like wood grain and textures like sand.

(9) Q Which do you think best for range measurements, stereoscopic cameras, range finders as with SRI's robot or sound echo method?

A Possible methods of range measurement that we are considering are: stereoscopic cameras, focusing adjustment with a monocular camera, and a touch-sensitive probe.

Focusing has the advantage over stereoscopy in that it cannot be deceived by vertical stripes. However it is probably less accurate. We did a little investigation of sound echo ranging techniques but rejected them. The wave-lengths of practical generators are too long for good resolution on our scale of equipment.

(10) Q How does your robot measure a parameter such as size or position of the objects? Are the accuracy and speed of measurement satisfactory for real-time manipulation?

A At present it does not make such measurements. We are prepared to be satisfied with errors of approximately 5%. Speed limitations are likely to be more severe for vision than for manipulation.

ARM AND HAND

(1) Q Describe the hardware specifications of the manipulators such as degrees of freedom or sensors.

A A manipulator has been designed and is under construction. Two opposed vertical 'palms' can move independently towards and away from each other over a range of about 18 inches and can move together vertically

through about 12 inches. Objects may thus be gripped between the palms, lifted and moved a small distance laterally, in a linear cartesian frame of reference.

Absolute accuracy of positioning will be about 0.2% of full range of movement, but backlash, rigidity and repeatability should all be only a few thousandths of an inch.

Later, it is intended to add rotation of the manipulator about a vertical axis, and rotation of the palms to turn objects over.

Strain gauges at suitable points will give indications of the forces exerted by the arms and the strength of grip.

(2) Q How dexterous will manipulation be and will it be successful?
A Too early to say.

(3) Q How do you design the control loop of the manipulators?
A The controlling computer will output positional information as 10-bit digital words. These will be converted to an analogue voltage to control a DC servo motor. Potentiometers will be used to measure position and tachogenerators to measure velocity.

(4) Q Do you have any suggestions for a system with two hands which would co-operate in a job with human beings?
A Not at this stage in terms of implementation. As an application area we have already mentioned teaching aids for children.

(5) Q Do the manipulators have any reflex actions? Is there any need of a small computer for the exclusive use of the manipulators.
A A peripheral loop will stop movement if an unexpected force is sensed by the strain gauges.

Exclusive use of a satellite computer is not necessary. We shall, however, be using such a machine to pre-process visual information and we will make use of it in controlling reflex movements.

LOCOMOTION

(1) Q Is there any great need to use legs instead of wheels?
A No.

(2) Q How does the robot direct its position in the real world?
A Combinations of dead-reckoning with landmark-recognition are possible, and have been examined by simulations.

(3) Q Does your robot have balance-detecting and controlling equipment?

A No.

(4) Q What are the application fields of robot-like machines with locomotive ability in the near future?

A Mowing lawns! If by 'near future' is meant the next two or three years we do not see commercial applications above a rather trivial level.

COMMUNICATION

(1) Q How does your robot communicate with the digital computer?

A The robot communicates with the computer as a peripheral of the Multi-POP time-sharing system, running on an ICL 4130 computer. Communication is via transfers of single 8-bit bytes. The output byte is decoded as a command to sample the picture or drive the motors. The input byte contains the state of the bump detectors and brightness of the picture point. When the satellite is installed, communication will be *via* a high-speed link with the ICL 4130. The robot will be interfaced to the satellite, essentially as it is now to the ICL 4130.

BRAIN

(1) Q What performance and abilities does the brain of your robot have? Does it have self-learning ability?

A We have engaged in the past in experiments involving developing various abilities in isolation and have not yet finished building an integrated system using these abilities.

For instance there is the Graph Traverser program for problem solving (Doran & Michie [6]; see Michie & Ross for an adaptive version [7]). BOXES and memo functions for rote-learning [8–10], programs for deduction and question-answering [11], and the Induction Engine [12]. Full learning ability requires what is learnt to be expressed in a language more powerful than simply a sequence of weights, as in Perceptrons or Samuel's Checkers learning program.

(2) Q What can the question-answering system in your robot do?

A We have implemented a number of approaches to question-answering. We have theorem-proving programs, which, as Cordell Green [13] has shown, can be modified for question-answering. We also have a program called QUAC based on relational combinators [11].

(3) Q What would be the best interface between robots and human beings?

A The best interface from the human's point of view would be spoken and written natural language, together with the ability to point at things with the robot watching through its television camera. In the immediate future,

for research purposes, typewriter and visual display using a flexible command language: e.g. 'imperative mode' POP-2.

(4) Q What is the most difficult problem in future artificial intelligence research?

A Possibly the internal representation of the robot's world, which will certainly involve automatic methods for inductive reasoning from a very large mass of (mostly irrelevant) data. It seems to us that, to be usable by the robot for serious planning, internal models must involve both *direct* representations in the form of appropriate data structures, as when a map is used to model a terrain, and *indirect* representations in the form of axiom systems and sentences in a formal language such as predicate calculus. Facts are retrieved from the former by look-up and from the latter by reasoning procedures. What is lacking at present is any general theory concerning the relative economics of these two forms of representation, or any principles for automatic transfers of knowledge from one to the other. We are inclined to think that present work on automation of induction will help in the required direction.

On the deductive side, we would mention the problem of discovering the relationship between solving a problem by logical inference and solving it by an algorithm (i.e. no redundant inferences made), so that opportunities for reducing an inference process to an algorithm may be automatically detected and exploited.

A certain confluence is now apparent between work on robot cognition and the field known as theory of programming. This is because formal equivalences can be set up betwen proving that a *plan* will be adequate to bring about a given result in the real world and reasoning as to whether a *program* will compute a given function [14]. We attach importance in this connection to recent advances in the theory of formal proofs about programs [15–17].

In terms of implementing systems capable of operating within reasonable time constraints, methods for handling highly parallel processes will be crucial, and these are still in their infancy.

ACKNOWLEDGEMENTS

The Edinburgh work has on the hardware side mainly been conducted in the Bionics Research Laboratory (of the Department of Machine Intelligence and Perception) founded by Professor R. L. Gregory. The basic software and interfacing work has been done in the Experimental Programming Unit of the same Department. The authors have been greatly assisted by their colleagues on the Robot Working Party: Dr R. M. Burstall, Reader in Machine Inteligence, Dr J. A. M. Howe, Director of the Bionics Research Laboratory and Dr H. R. A. Townsend, Senior Consultant Neurologist in the Western General Hospital and part-time Senior Lecturer in the above Department.

A special debt is acknowledged to an overall philosophy contributed by Professor R. L. Gregory. His dictum: 'The cheapest store of information about the real world is the real world itself' was a major part of the original motivation, and the emphasis laid upon the role of internal models in Gregory's analysis of perception [18] continues to be central to our work.

REFERENCES

[1] E. A. Feigenbaum (1968) Artificial intelligence: themes in the second decade. *Proc. IFIP Congress 1968,* 10–24.

[2] R. M. Burstall & J. S. Collins (1971) A primer of POP-2 programming. *Programming in POP-2* (ed R. M. Burstall). Edinburgh: Edinburgh University Press.

[3] R. M. Burstall & R. J. Popplestone (1971) POP-2 Reference Manual. *Programming in POP-2* (ed R. M. Burstall). Edinburgh: Edinburgh University Press. Also in *POP-2 Papers.* Edinburgh: Edinburgh University Press (1968).

[4] C. L. Fennema & C. R. Brice (1969) A region oriented data structure. *Technical Note no. 7, SRI project 7494.* Stanford: Stanford Research Institute.

[5] C. L. Fennema & C. R. Brice (1969) Scene analysis of pictures using regions. *Technical Note no. 17, SRI project 7494.* Stanford: Stanford Research Institute.

[6] J. E. Doran & D. Michie (1966) Experiments with the Graph Traverser program. *Proc. Roy. Soc. A*, **294**, 235–59.

[7] D. Michie & R. Ross (1970) Experiments with the adaptive Graph Traverser. *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 301–18. Edinburgh: Edinburgh University Press.

[8] D. Michie & R. A. Chambers (1968) BOXES: an experiment in adaptive control. *Machine Intelligence 2* (eds E. Dale & D. Michie), 137–52. Edinburgh: Oliver & Boyd.

[9] D. Michie (1968) 'Memo' functions and machine learning. *Nature,* **218**, 19–22.

[10] D. L. Marsh (1970) Memo functions, the Graph Traverser and a simple control situation. *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 281–300. Edinburgh: Edinburgh University Press.

[11] A. P. Ambler & R. M. Burstall (1969) Question-answering and syntax analysis. *Experimental Programming Report no. 18,* University of Edinburgh: Department of Machine Intelligence and Perception.

[12] R. J. Popplestone (1970) An experiment in automatic induction, *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 203–15. Edinburgh: Edinburgh University Press.

[13] C. C. Green (1969) Theorem proving by resolution as a basis for question-answering systems. *Machine Intelligence 4* (eds B. Meltzer & D. Michie), 183–205. Edinburgh: Edinburgh University Press.

[14] C. C. Green (1969) Applications of theorem-proving to problem-

solving. *Proc. Intern. Joint Conf. on Artificial Intelligence* (eds D. Walker & L. M. Morton), 219–39. Washington DC.

[15] R. W. Floyd (1967) Assigning meanings to programs. *Mathematical aspects of computer science,* 19–32. Providence, Rhode Island: Amer. Math. Soc.

[16] Z. Manna & J. McCarthy (1970) Properties of programs and partial function logic. *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 27–37. Edinburgh: Edinburgh University Press.

[17] R. M. Burstall (1970) Formal description of program structure and semantics in first order logic. *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 79–98. Edinburgh: Edinburgh University Press.

[18] R. L. Gregory (1966) *Eye and Brain.* Weidenfeld and Nicolson.

9

Artificial intelligence (1971)

A number of laboratories around the world are investigating how to program computers to be a little more 'intelligent' than they are. Such studies soon come up against a fundamental problem concerned with 'understanding'. We have to discover, in rather precise terms, what is meant by 'understanding' a topic, or a problem. Otherwise, attempts to get computers to do things normally requiring human intelligence, are likely to remain superficial and, in the long run, unproductive.

A classic example is the rosy dream about the possibilities of 'machine translation'. In the 1950s and 1960s, millions of dollars were spent in the United States on research-and-development projects aimed at this. The techniques of machines breaking up texts grammatically and looking up meanings in a computer dictionary proved too shallow to crack the machine translation problem unaided. Fundamental progress had to wait for the development of an adequate theory of what is involved in 'understanding' a passage of English-language text. The needed theory is only just beginning to emerge.

The syntactic and semantic problems presented by natural language are at present under study by the theoretical section of the Department of Machine Intelligence and Perception in Edinburgh, headed by Professor H. C. Longuet-Higgins. But this is only one of many areas in which we can attempt to achieve 'computer understanding'. An obvious and particularly challenging area is that of understanding the ordinary material world around us. Consider a very simple world, consisting of a play-pen with a few commonplace objects. On first tackling this world, a child has to come to terms with, and mentally organise, not only the visual and mechanical properties of material objects, but also the basic laws of nature, such as gravity, which determine their interaction.

To program knowledge like this into a computer, so that it can inspect such a world through a TV camera and then carry out a particular plan of action (for example, piling bricks, sorting buttons), is a task of at least

comparable difficulty to the language-understanding task I have referred to. In order to do research in this area we have to equip computers with cameras, movable platforms, mechanical 'hands', and other 'robot' devices. But this should not distract attention from the real point. The robot itself is not the object of the exercise. In the robot project in our department, we are concerned with the use we can make of a robot in order to develop *theories* of 'computer understanding' of the real world. The acid test of such theories is provided by the robot itself. If it can be got to *understand* what is involved in operations such as 'find', 'fetch', 'build', 'tidy', and so on, then, and only then, can we program it to do these things.

In fact, a new technology, even more recent than software engineering (the mechanics of computer instructions), is coming into existence. This rapidly growing art might appropriately be called 'cognitive engineering'.

To 'engineer' means to design and construct devices which actually work. In case it seems premature to talk of man-made devices, even software devices, which possess cognitive ability, let me first give an example of something a cognitive software device can do. Take the integrated arrangement of computer programs of which the program developed at Massachusetts Institute of Technology and called PLANNER forms a part.

Fig. 9.1 depicts the simulated 'world' in which problem-solving is carried

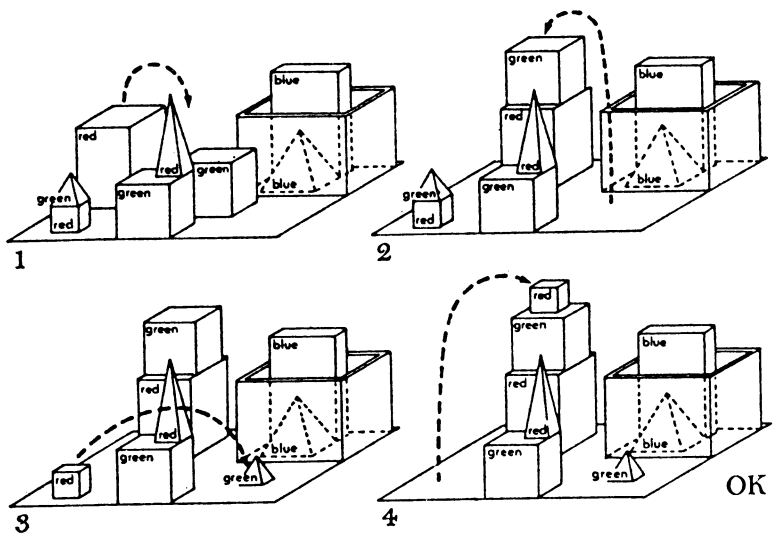


Fig. 9.1 — Problem-solving carried out by machine. Man's command given via the computer's typewriter. Machine finally responds with OK.

out under this system, in response to a command given in English, by means of the computer's typewriter. The figures show successive steps taken by the system in order to carry out the command, and the following is a brief excerpt from the accompanying dialogue between man and machine:

MAN: Will you please stack up both of the red blocks and either a green cube or a pyramid?
MACHINE: OK.
MAN: Which cube is sitting on the table?
MACHINE: THE LARGE GREEN ONE WHICH SUPPORTS THE RED PYRAMID.
MAN: Is there a large block behind a pyramid?
MACHINE: YES, THREE OF THEM: A LARGE RED ONE, A LARGE GREEN CUBE AND THE BLUE ONE.
MAN: Put a small one onto the green cube which supports a pyramid.
MACHINE: OK.

The arrangement of programs which produced this behaviour consists of about a dozen interlocking units, which, together with a dictionary of two hundred English language words, give about 2.5 million separate units of information, to comprise the computer memory requirements of experimental artificial intelligence work today. But problems involving real interaction with the outside world through TV cameras make even heavier demands. People at the Massachusetts Institute of Technology laboratory, in common with the other American computer centres regard a total of a quarter of a million words of fast memory as necessary for a research machine.

The construction of *integrated* intellectual or cognitive systems is among the more ambitious aims pursued by workers in the field of artificial intelligence. As a quick sketch of what I mean by the term 'integrated', consider two hypothetical chess machines.

Machine 1 is capable of beating International Masters. They communicate their moves to it in a standardized format through the typewriter. Machine 2 plays bad amateur chess. But it inspects the board position visually through a TV camera, makes its own moves for itself with a computer-driven hand, can describe its own plans and explain its moves in passable English, improves its play with practice, and can accept strategic hints and advice from a tutor.

Which machine is the more intelligent? This is not a particularly meaningful question; 'intelligence' on any reasonable definition is related to a particular activity rather than being an absolute term.

Which machine would make the more ambitious goal for a research project? Quite impossible to say: both goals would be very ambitious.

Which machine more properly belongs to the category of an integrated cognitive system? Without any doubt at all, machine 2, where the interest is not in the depth of any one skill but rather in the effective knitting together of many skills.

Both machine 1 and machine 2 are figments of the imagination. The nearest to machine 1 which has yet been achieved is probably the Atkins–Slate chess program, which recently defeated a strong amateur player. Possibly the system with strongest superficial resemblance to machine 2 is not a chess program at all, but a program for making a robot play the game of 'Instant Insanity'. The robot is the computer-controlled 'hand–eye' device

developed in Professor John McCarthy's Artificial Intelligence Project at Stanford University. The game of Instant Insanity is played with four large, specially constructed dice, which have colours (red, white, blue, green) rather than numbers on their faces.

If the four dice are pressed together side by side, the 'left–right' faces are all hidden. We are interested here solely in the four colour sequences which we can see as we look along the row; first at the four 'top' faces; then at the four 'near' faces; at the four 'bottom' faces; and at the four 'far' faces. The aim is to arrange things, by rotations of individual dice, so that no colour is duplicated in any one of these four rows of faces.

My investigations lead me to believe that there are three and only three essentially distinct solutions, but I have not proved it. A cube can be given 24 different orientations in space. So we can calculate $24\times24\times24\times24$ as the upper limit of the arrangments we have to check. Allowing for symmetries and redundancies, this comes down to about 2000 essentially distinct states. A brute-force method of solution would have to examine all these individually.

The Stanford program does not rise even as high as brute force. At present the program is set up so that it knows the winning configuration in advance. It concentrates its problem-solving efforts on inspecting and identifying the four blocks and carrying out their final assembly. But like machine 2, in my fanciful chess example, it does do everything for itself. It inspects the cubes through a colour TV camera, and it performs all the manipulations with its computer-controlled hand. One can thus say that a coordination of 'hand', 'eye', and 'brain' exists, even though the individual performance of each member of the trio may leave much to be desired.

The aim of the laboratories working with integrated systems is to master real-world problems that are more and more challenging intellectually.

Now what would be a suitable task, intellectually more challenging than Instant Insanity puzzles? A classic problem in artificial intelligence is known as the 'monkey and bananas' problem, posed by John McCarthy almost ten years ago. A monkey is in a room where a bunch of bananas is hanging from the ceiling just out of the monkey's reach. Somewhere in the room there is a chair. Can the monkey manage to get the bananas?

At first sight, my example may arouse a sense of bewilderment. Why should so trivial a problem be solemnly discussed as a matter of intellectual depth? After all, real monkeys are capable of solving it, though they find it difficult, whereas no one suggests that a monkey could solve Instant Insanity. But the triviality is relative, rather than absolute; i.e. it is relative to the amount of relevant knowledge a monkey or a person or a machine has previously amassed and organized in its memory. Before a machine can be even as intelligent as a monkey in real-world problem-solving, a great deal of this kind of knowledge must somehow be got into it. Only then will these problems begin to be trivial in the sense that they are trivial to a human being.

The background knowledge required for problem-solving in some particular domain constitutes what has been called a 'micro-theory'. The 'micro-

theory' needed for problems of the monkey-and-bananas type must deal with the fundamental logic of the location in space of physical objects ('bananas', 'monkey', 'chair', are all instances of such objects) and their translation through space by the operation upon them of actions (of which 'go to', 'climb' are instances). Such a micro-theory, if it is to mirror the world we know, should, for example, assert that if object X is at position A, and A is not equal to position B, then X is not at B (i.e. one thing can't be at two places at the same time). It should assert that if X is at A and X is not equal to Y, then Y is not at A (i.e. two things can't be at one place at the same time).

I shall come back to the formidable complexities of building an adequate amount (and arrangement) of such general facts into a machine. But, to begin with, let me describe the first recorded solution by a computer-controlled robot of a monkey-and-bananas problem. This was done by Stanford Research Institute's Artificial Intelligence Group and reported by Stephen Coles.

Fig. 9.2 shows Stanford Research Institute's robot 'Shakey'. It stands the

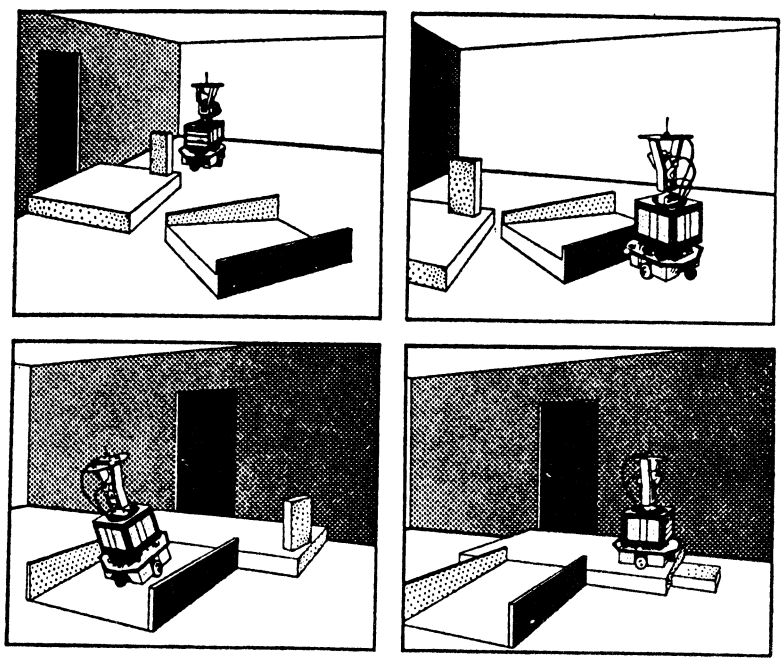


Fig. 9.2 — 'Shakey' in action. The robot is remotely controlled over a radio link.

height of a man, and the computer controls it remotely by radio. Shakey has no hand and cannot climb. The monkey-and-bananas problem was accordingly translated into terms appropriate to the robot's own input-output devices. The reformulation, known as the 'robot and box' problem, is as follows:

The robot is in a room in which a box rests on a platform too high for the robot's wheels to mount. Somewhere in the room is a ramp. The robot's task is to cause the box to be on the floor.

The successful solution of the problem is illustrated in Fig. 9.2. I can here only sketch the way in which the capability to reason out the solution from first principles is programmed into the system. The key techniques are derived from mathematical logic. The trick is to dress up the formation of a plan of action so that it looks exactly like the task of proving a theorem in some logical calculus. The kind of theorem the machine tries to prove is one which asserts that 'a possible state of the world exists in which the box is on the floor'. It is possible to arrange that, as a side-effect of a successful proof, a chain of actions is produced for bringing about the desired state of the world.

There are all sorts of technical difficulties related to mechanical theorem-proving even in such simple situation-and-action problems as this. One in particular, is called the 'frame problem'. For example, though you and I know that, after the monkey pushes the chair, the bananas are still where they were, a mechanical reasoning system must have such facts explicitly represented in its knowledge base. In some other world, it might be the case that chairs exert a repulsive force on bananas.

Coles sets out the stream of 'thoughts', if I can call them that, which go through the robot's 'brain' (by which I mean the program running in Stanford's SDS 940 computer) in the form of the following informal English translation:

'My first subtask is... to move the ramp over to the platform and align it properly. To do this, I must first discover where the ramp is. To do this, I must first see it. To do this, I must first go to the place where, if I looked in the right direction, I might see it. This sets up the subsubtask of computing the coordinates of a desirable vantage point in the room, based on my approximate knowledge of where the ramp is.

'Next, I have the problem of getting to the vantage point. Can I go directly, or will I have to plan a journey around obstacles? Will I be required to travel through unknown territory to get there if I go by an optimal trajectory; and, if so, what weight should I give to avoiding this unknown territory? When I get there, I will have to turn myself, and tilt the television camera to an appropriate angle, then take a picture in. Will I see a ramp? The whole ramp? Nothing but the ramp? Do I need to make a correction for depth perception?

And so on. And so on. The reasoning part takes about twenty minutes, and the vision and pushing activities another fifteen minutes. So the whole operation takes over half an hour. Great speed-ups of the different functions are likely to be achieved over the next few years of robot engineering. (We have one or two ideas of our own at Edinburgh, where we are just beginning to experiment with a reasonably advanced robot device.) But before I leave Shakey, I would like to mention one concept which Coles raises, that is likely to be important in the future.

This is the number of subgoals that are necessary to solve the problem;

how direct or indirect the solution is. The robot-and-box problem has one level of indirectness (associated with the sub-problem of how to get the ramp to the platform). But, as Coles points out, any system that is complete logically could, in principle, solve problems with as many levels as you like. Yet problems possessing merely half-a-dozen levels of indirectness begin to overtax human intellectual capability. We may think that, at the moment, the human brain has much more sheer computing power at its disposal than even the biggest computer in the world. But the time may be approaching when such thoughts will require careful qualification.

10

Machine intelligence at Edinburgh (1973)

Programming a computer to control an experimental robot (TV 'eye', mechanical 'hand' and steerable viewing platform) seems a far cry from management science. A point of relevance, however, can be found in current plans, under active study in America and Japan, to bring into existence the fully automatic factory.

The US Defense Department's Advanced Research Projects Agency met recently to discuss a report which it had commissioned two years ago from the Rand Corporation concerning the feasibility of an automatic factory project. These findings indicated feasibility in about ten years, given a massive R & D programme to create the technical preconditions. A Japanese plan with a similar time-scale, but on a larger scale (including among its aims an entire computer-controlled city), has been described by Yoneji Masuda, Director of Japan's Computer Usage Development Institute. The total cost will be £25 000 million.

It goes without saying that the administrative processes of a factory must be entirely computerized if the aim of total automation is to be realized. Less obvious is the fact that a diverse range of mechanical handling operations must also be coordinated and that these necessarily include operations of 'eye' and 'hand' which require some degree of intelligence when performed by humans. Such tasks might include sorting out components from a disorderly heap and fitting them together to construct a finished article, in accordance with written and pictorial descriptions. To program a computer to do this using children's construction kits of the 'Meccano' type is an important sub-goal of our project at Edinburgh.

Our general aim is to develop an integrated robot system capable of

interesting behaviour in response to requests and 'hints' supplied by an interactive user. Such work should lead ultimately to knowledge of how to program what Hitachi calls 'intelligent robots', and I have called (*Nature*, Nov. 21, 1970) 'integrated cognitive systems'. This phrase is not ideal: it appears to irritate, by its mental association with psychology, and hence to distract from the technical content. The opening passage of my article is reproduced here (Chapter 7 in this book):

> 'Work is in progress in several laboratories directed towards the construction of an integrated cognitive system (ICS). I avoid the phrase "intelligent robot" because of its science fiction connotation of humanoid appearances and other attributes. The research is concerned with intellectual attributes, involving sensorimotor and reflex capabilities only to the extent that these form a necessary substratum for the acquisition or display by computing systems of purely intellectual skills.'
>
> 'At this early stage the "intellectual" skills which research aspires to emulate may seem to the onlooker so primitive as scarcely to deserve the name. Let him, however, reflect on the struggles of small children with the simplest tasks of deduction, generalization and description, and their dogged attempts to construct and refine world-models adequate for their growing needs, representing a succession through which every developed human intellect has passed. Even these first exploits of the infant mind are beyond the abilities of any computing system yet devised. Computers equipped with optical input and manipulative devices are available in at least two laboratories, but understanding of machine perception and cognition has not advanced so far that they could be programmed to compete with human infants, for example on such tasks as the following, which is taken from Stanford–Binet IQ tests. The task involves obeying simple commands, and is designed for 2½ year old infants. With a brick, button, a dog, a box and a pair of scissors laid in order on a table, the child is told (a) "give me the dog"; (b) "put the button in the box". and (c) "put the scissors beside the brick". A machine passing tests of this sort would be disqualified if it had merely been pre-programmed *ad hoc* for each individual test. An artificial intelligence worth the name must show some degree of generality.'

Performance goals of the type indicated already seem, in terms of the present state of programming technique, too 'easy'. In our current specification for a 'working model' of an integrated robot system we envisage facilities for 'teaching' the system elementary tasks of assembly initially presented by the user as 'unseens' (i.e. no pre-programmed knowledge of each given task or of the materials provided for it). Although there may be eventual applications for such work in automating factory assembly-line operations (the Japanese Government, by voting expenditure of five million

pounds per annum for research on 'Pattern Information Processing' including robot work indicate that they believe this), our own selection of such tasks has been guided by theoretical questions concerning the impact upon current programming language concepts. At the same time, in order to equip ourselves at all to use such novel input–output devices as TV cameras and motor-driven 'hands' we have had to do a certain amount of baseline work in areas such as machine perception and the design of software for controlling the physical manipulations of perceived objects.

THE VISION PROBLEM

Before television cameras had been attached to computers, there was a tendency to regard computer vision as a technological problem and not an integral part of the field of machine intelligence. However, the consensus of opinion is now that the problem of making a computer see what is going on around it is inextricably linked with such problems as dealing with unreliable information, making hypotheses and testing them, making plans of action, using knowledge about the state of the world and its laws, integrating fragments of information to produce a coherent whole and learning complex relationships.

We require to produce a visual system for an intelligent machine. Our first step has been to design a system for extracting and matching descriptions of the retinal image adequate to identify with fair reliability a repertoire of ordinary objects (hammer, cup, doughnut, ball etc., see Barrow and Popplestone [1]; Barrow, Ambler and Burstall [2]). A complete picture is read and stored in the computer. The program first tries to divide the picture into areas which have strong contrast across their boundaries. It does this by finding areas of approximately uniform brightness and then merging together to form larger areas those which are adjacent and have little contrast across their common boundary.

The picture is then described in terms of properties of the regions, e.g. COMPACTNESS (measured as 4π.Area/Perimeter^2) and the relations between them, e.g. ADJACENT, BIGGER THAN.

Finally, the program matches the picture description against stored descriptions of views of objects. These have been formed by the program from examples presented to it during a teaching session. The best match identifies the object.

Identifications are currently made with about 95 per cent accuracy (Turner [3]) at the expense of several minutes' processing time for each identification. Present work is directed towards improving the accuracy of identification by various means, including the extraction of 'depth information'.

The above system was developed without initial reference to *operational* criteria, i.e. to the robot's use of tests and actions to perform tasks. We are currently engaged upon setting up a visual system for our Mark 1.5 robot which will enable it to perform a variety of simple tasks, especially such

operations as picking up objects and placing them in or on other objects. Until recently, work in the field of robot vision had been concentrated upon the problem of making the computer produce a description of a picture. For any single picture there are infinitely many ways in which that pattern of light and shading could have been induced to fall upon the image plane of the camera (e.g. a small thing close up, or a big thing a long way away). In order to produce a likely interpretation the program must have built into it various types of knowledge about the world, e.g. the objects are standing on a table top, all surfaces of objects are planar, objects must be supported. Current work at other laboratories is aimed at making the assumptions made by the program more explicit, at examining the implications of certain assumptions in considerable detail and allowing hypotheses to be made and retracted. So far, however, research has been concentrated on the relationship between polyhedra and their images.

We attack the problem of robot vision from a slightly different direction. In real situations a robot is likely to be looking at a scene that it has seen many times before and therefore about which it already knows a great deal, e.g. positions of objects, their orientation and type. In such circumstances it is unnecessary and even detrimental to performance to process every picture as though it were being seen for the first time. A much more flexible approach is required; sufficient processing should only be carried out to confirm that one's present world model is not radically incorrect. For example, if a blob is seen in the top left of the picture and the robot knows that there is a hammer at a corresponding point in its world, then it does not need to analyse the blob further but it can assume that it represents the hammer. Attention can also be directed to specific parts of the picture if we are only concerned with a localized change in the world. We have already found that it can be extremely cheap in terms of computer time to check visually that there is not an object at a particular location, e.g. when looking for a clear space to put something down, or making sure that a particular object has been picked up. A few years ago the view was usually taken that picture processing was always expensive and therefore should be used as sparingly as possible. However, if the program knows what it is looking for it can carry out highly specific tests which can be computationally cheap.

A working program written by one of our diploma students, Bill Dallas, can be asked to sort the objects which are on the viewing platform, putting objects of one type in one area and those of a second type in a second area. Since we then had no tactile feed-back from the hand, the only way the program can tell whether it has picked up an object is by looking at the place where the object was and making sure it is no longer there. If it is still there the program will make repeated attempts to pick it up. A visual check is also made before putting the object down that the space into which the object will be put is in fact empty. If it is not, a new destination can be calculated. The program has a data-base of information about its world and will first try to sort objects that it knows about; for each object that it knows about, if that object is to be moved, a visual check is made that it is still at its expected position and then it is picked up and placed in a clear space in the

appropriate area. When the program has run out of known objects, it searches the platform. Having taken a picture, it tries to interpret blobs in its field of view as known objects.

THE ASSEMBLY PROBLEM

We aim to get the robot to put objects together. The repertoire of components for building objects is to be extended beyond the plane polyhedra of other projects, and must include such things as slotted rods, shafts and bearings. Optical methods are required for measuring complex surfaces, and the 'shadowgraph' method has been developed for 3-dimensional perception. This uses a projector casting a stripe of light on a scene together with the TV camera to build up a 'depth map'. The principles involved are similar to those used by the independently developed system of Shirai and Suwa [4].

When a shadow is cast by a horizontal edge onto a surface, the irregular path which the shadow appears to follow can be used to reconstruct the relative altitude of every point on the surface lying on the path. Thus, if a computer-controlled camera is looking vertically, and if the height and position of the edge casting the shadow are known and also the position of the light source, then a suitable program can reconstruct the entire 3-dimensional contour of the surface by moving the shadow-casting edge to successive positions.

A program has been developed which performs this reconstruction. An angle of 45° for the light seems to give the best compromise between having the light too oblique (when the edge of the shadow cannot easily be determined) and having the light too vertical (when the shadow is so short as to reduce the accuracy of the method). Light and edge are together moved across the object so as to get height readings from most points on the surface. It will be appreciated that there will be areas beside objects, of width less than the height of the object, for which no height estimation is obtained. Most of these lacunae can be filled by traversing the light from the opposite direction leaving only a few unfathomed depths.

Some care is needed in distinguishing shadow boundaries from naturally occurring changes in the object such a as white label on a dark parcel. Various devices can be used to resolve such confusions: for instance, moving the light source under computer control to exploit the fact that only boundaries that move when the light moves can be shadow boundaries.

As a first step towards assembly by computer we are working on automatic packing of parcels into boxes. This problem is of interest to the GPO. The current version of the program analyses the outline of an object into line segments. The outlines of the holes in the container are also analysed into line segments. The program places parcels by trying 'in its head' to put the parcel in a hole, with one corner of the parcel in a corner of the hole. Studies are in hand to use look-ahead techniques to optimize placing. In this connection a system is under trial for translating statements about desired relationships of rigid bodies ('make the rod fit into a hole in

each of the blocks so that the blocks are against the wall') into equalities and inequalities as vectors, scalars and rotations, and then automatically translating these into program for iteratively solving them.

THE PROBLEM-SOLVING PROBLEM

Consider a sliding-block puzzle, such as the 'Passalong', shown in Fig. 10.1.

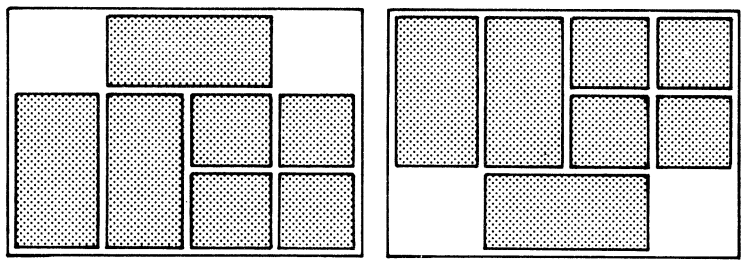


Fig. 10.1

By a succession of sliding movements of the blocks in the tray we are required to transform (a) into (b). This is plainly a problem. It belongs to a class of *games* characterized by the properties:

(1) One or two-person;
(2) Perfect information;
(3) No chance moves.

This class extends from simple puzzles to the mechanization of mathematical manipulations. Thus the three restrictions listed do not make such problems necessarily trivial. They do, however, render them accessible to a range of techniques which can be broadly described as 'look-ahead'. Looking ahead along a branching tree of possibilities is an activity familiar to anyone who has played a game such as chess.

Now consider another problem. A blind, insentient robot must operate in the world shown in Fig. 10.2. The robot has a 'hand', able to execute the

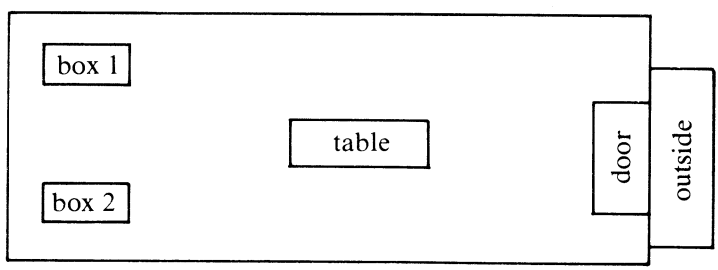


Fig. 10.2

actions GOBOX1, GOBOX2, GOTABLE, GODOOR, GOOUTSIDE, PICKUP, LETGO. The laws governing this world, and the effects of the actions, can be axiomatised in first-order predicate calculus, of which the following is an informal translation. We have attempted a consistent use of upper case to label constants. The reader is not asked to slog through this in detail, but to try to get the general idea.

If a thing, *t*, is held in a situation, *s, and* the HAND is at a place, *p*, in *s, then* the thing, *t*, is at *p* in *s*.

If t is at *p* in *s* and the HAND is at *p* in *s then* a thing taken in *s* is held after doing a PICKUP in *s*.

Nothing is held after doing a LETGO in *s*.

If t is at *p* in *s then t* is at *p* after doing a PICKUP in *s*.

If a thing, *t*, is held in *s then t* is held after doing a GO to *p* in *s*.

If a thing, *t*, is at *p* in *s then* the thing, *t*, is at *p* after doing a GO to *p* in *s*.

If a thing, *t*, is not at *p* in *s* and *t* is not held in *s then t* is not at *p* after doing a GO to *p* in *s*.

If t is at *p* in *s then t* is not at *p* after doing a LETGO in *s*.

If t is not at *p* in *s then t* is not at *p* after doing a LETGO in *s*.

t is at BOX1 or BOX2 or the TABLE or the DOOR or OUTSIDE in *s*.

We can then define the axioms [5] which specify a particular problem.

If a thing, *x*, is at BOX1 NOW and a thing, *y*, is at BOX2 NOW *then* either *x* or *y* is a handle.

A thing, A, is at BOX1 NOW.

A thing, B, is at BOX2 NOW.

A thing, C, is at the DOOR NOW.

Nothing is on the TABLE NOW.

If a thing, *t*, is at the DOOR NOW *then t* is red.

BOX1 is in the room.

BOX2 is in the room.

The TABLE is in the room.

The DOOR is in the room.

If p is in the room *then* the hand is at *p* after doing a GO to *p* in *s*.

If a thing, *t*, is at the DOOR in *s* and *t* is a handle *then* the HAND is at OUTSIDE after doing a GOOUTSIDE in *s*.

Nothing is held NOW.

We now pose a problem, as follows:

If a thing, *t*, is OUTSIDE in *s* and *t* is red *then s* is an 'answer' situation.

Is there an action-sequence guaranteed to bring about an 'answer' situation?

Can a robot be so designed as to be capable of generating valid plans of action? The nub of this qestion can be re-expressed: 'Can an algorithm be specified which will generate and validate such plans?' The execution of plans in the real world belongs to the realm of (difficult and interesting) engineering. We are not concerned with it here.

The first thing to notice is that unaided lookahead techniques are put out of court by *imperfect information* (the initial state is not fully specified: we do not know where the hand is), and by the intrusion of *chance moves* (the PICKUP action transfers a randomly selected object from the place where the

hand is into the hand). Is there an alternative approach sufficiently powerful to do the job?

The answer is 'Yes, in principle, but not in practice'. As sketched by McCarthy & Hayes [6] and implemented by Green [7], the problem of finding a valid plan can be distinguished as a problem of *deduction* from the axioms which describe the problem. In the present case, such a deduction might lead to (informally):

'The situation resulting from the sequence
GOTABLE LETGO
GODOOR PICKUP GOTABLE LETGO
GOBOX1 PICKUP GODOOR LETGO
GOBOX2 PICKUP GODOOR LETGO
GOTABLE PICKUP GOOUTSIDE LETGO
GOTABLE PICKUP GOOUTSIDE
is an "answer" situation.'

The trouble is that the best of contemporary theorem-proving strategies are nothing like adequate to performing such a deduction with an acceptable expenditure of computing time, and it is arguable that unaided deduction is inherently inadequate to such a task.

An approach under investigation in collaboration with J. A. Robinson combines the *generation* of plans with their *logical validation*. The two systems are harnessed to work side by side within the same master program. An automatic rote memory is incorporated which gradually builds up a dictionary of conjectured and proved solutions to problems and sub-problems within the task domain. This dictionary represents the system's accumulated *operational knowledge* about the domain and can be regarded as a growing store of miniprograms for performing tasks in it. Proposals for enabling the system to generalize over this knowledge, using 'relational description matching' as developed by Ambler, Barrow, Burstall, Popplestone, and others, are now being considered.

At the moment the problem-solving work is not being used to control the actual robot apparatus. Automatic creation of plans is a longer term enterprise and in the immediate future we intend to program the robot more directly. But we hope to try out the planning techniques using the hardware when they have been further developed.

THE PROGRAMMING PROBLEM

Given a computer with a TV camera, a hand and a moveable table as peripherals with certain manipulative tasks to be performed, one might well ask 'Why not simply *program* the computer to perform these tasks, just as one programs a computer with a card-reader and a line-printer to perform a payroll task?' We believe that even familiar and apparently simple manipulation taks are *very difficult* to program in the conventional sense of the term; at best one could devise inflexible programs for a few specific tasks. The reasons are:

(1) Imperfect and ambiguous information from the TV camera and other sensors;
(2) the tedium of weaving into each part of the program our detailed and largely subconscious knowledge of physical objects, manipulations and the laws which govern them.

The robot problem directs our attention to techniques beyond the scope of classical programming. Two such techniques have been intensively developed in Artificial Intelligence work: *search techniques and logical inference techniques*. We are now beginning to find out how to incorporate them in programming languages in a really smooth and unified way (for example the PLANNER language developed at MIT by C. Hewitt [8]).

To incorporate search techniques we discard the notion that there is a unique next instruction to be obeyed and say execute this instruction *or* that instruction (i.e. do the first one but be prepared to back-track and do the second if things don't work out). We can use sophisticated heuristic control to guide the search if we like.

To incorporate inference we realize that evaluating expressions like **if** *on* (x,y) **then**... can be quite different from evaluating **if** $x>y$ **then**..., since it can make references to a *data-base of facts* and uses *inference rules*, either standard ones such as resolution or *ad hoc* ones specific to the task. It is most important that we can now improve our programs piece-meal by adding new facts and inference rules, instead of trying to program the whole task in one gargantuan effort.

The theme that has emerged in the last year or so is that search and inference techniques alone are too weak to perform interesting tasks unless intimately combined with the full power of a programming langauge.

The language enables us to tell the machine what to do, the search and inference mechanisms enable us

(1) to avoid spelling out each step in explicit detail (if I say 'Shut the door' I don't have to tell you to walk to it first), and
(2) to have the machine do a little more than we tell it by piecing together new plans from given components.

Thus we aim ultimately to develop a *teachable* system as opposed to one which has to be programmed monolithically.

EXPERIMENTAL PROGRAMMING

Our aim is to contribute to re-structuring the arts of programming so that much of what is today done by programmers can ultimately be done by machines. The science of programming needs, as does any other science, such as physics, to develop both theoretical and experimental sides. Robot work is an example of the use of *experimental programming* to validate theoretical conjectures and results and to suggest new hypotheses.

Insights acquired in this way are not, of course, to be developed for ultimate application within the limited domain of laboratory hand–eye

problems, but hopefully to the design and implementation of real-world interaction systems which have economic significance, whether factory assembly, navigational guidance, traffic control, air-line booking, chemical engineering or other complex commercial systems. It is, however, often desirable to learn to walk before attempting to run. Approached in this spirit, the study of laboratory 'hand–eye' problems may help lay foundations on which others can build more ambitious, and more economically applicable, software systems.

ACKNOWLEDGEMENTS

I would like to thank my colleagues, Harry Barrow, Rod Burstall and Robin Popplestone for assistance in writing this paper. The work of our group at Edinburgh described here has been supported by the Science Research Council and the General Post Office.

REFERENCES

[1] H. G. Barrow & R. J. Popplestone (1971) Relational descriptions in picture processing. *Machine Intelligence 6* (eds B. Meltzer & D. Michie), 377–96. Edinburgh: Edinburgh University Press.

[2] H. G. Barrow, A. P. Ambler & R. M. Burstall (1972) Some techniques for recognising structures in pictures. *Proc. Intern. Conf. on Frontiers of Pattern Recognition,* Honolulu, Hawaii-New York: Academic Press.

[3] K. J. Turner (1971) Object recognition tests on the Mark 1.5 robot. *Research Memorandum MIP-R-92.* University of Edinburgh: Department of Machine Intelligence, School of Artificial Intelligence.

[4] Y. Shirai & M. Suwa (1971) Recognition of polyhedrons with a range-finder. *Proc. Second Intern. Joint Conf. on Artificial Intelligence,* 80–7. London: The British Computer Society.

[5] We also need axioms to do with the consequences of adding to, and taking away from, finite sets.

[6] J. McCarthy & P. J. Hayes (1969) Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence 4* (eds B. Meltzer & D. Michie), 463–502. Edinburgh: Edinburgh University Press.

[7] C. Green (1969) Application of theorem proving to problem solving. *Proc. Intern. Joint Conf. on Artificial Intelligence* (eds D. E. Walker & L. M. Norton), 219–39. Washington DC.

[8] C. Hewitt (1970) PLANNER: A language for manipulating models and proving theorems in a robot. *MIT Project MAC AI Memo* 168.
See also:
R. M. Burstall (1969) Formal description of program structure and semantics in first order logic. *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 78–98. Edinburgh: Edinburgh University Press.
R. M. Burstall, J. S. Collins & R. J. Popplestone (1971) *Programming in POP-2.* Edinburgh: Edinburgh University Press.

J. E. Doran & D. Michie (1966) Experiments with the Graph Traverser program. *Proc. R. Soc. A,* **294**, 235–95.

G. W. Ernst & A. Newell (1969) *GPS: A Case Study in Generality and Problem Solving.* New York and London: Academic Press.

R. E. Fikes & N. J. Nilsson (1971) STRIPS: A new approach to the application of theorem proving to problem solving. *Proc. Second Intern. Joint Conf. on Artificial Intelligence,* 608–20. London: The British Computer Society.

D. Michie (1970) Future for integrated cognitive systems. *Nature*, **228**, 717–22, and above, 000–000.

D. Michie (1971) Notes on G-deduction. *Research Memorandum MIP-R-93.* University of Edinburgh: Department of Machine Intelligence, School of Artificial Intelligence.

D. Michie & R. Ross (1969) Experiments with the adaptive Graph Traverser. *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 301–18. Edinburgh: Edinburgh University Press.

A. Newell, J. C. Shaw & H. A. Simon (1957) Preliminary description of a general problem solving program — I (GPS-1), *CIP Working Paper no. 7.* Pittsburgh: Carnegie Institute of Technology.

J. A. Robinson (1965) A machine-oriented logic based on the resolution principle. *J. Ass. comput. mach.,* **12**, 23–41.

11

Machines and the theory of intelligence (1973)

The birth of the subject generally referred to as 'artificial intelligence' has been dated [1] from Turing's paper [2] *Intelligent Machinery* written in 1947. After twenty-five years of fitful growth it is becoming evident that the new subject is here to stay.

The scientific goal of research work in artificial intelligence is the development of a systematic theory of intelligent processes, wherever they may be found; thus the term 'artificial intelligence' is not an entirely happy one. The bias towards artefacts is reminiscent of aerodynamics, which most people associate with aeroplanes rather than with birds (yet fruitful ornithological application has been achieved) [3]. Here I shall review briefly some of the experimental knowledge systems which have been developed, and indicate how pieces of theory abstracted from these might fit together.

SOME PERFORMANCE SYSTEMS

Game playing was an early domain of interest, and Shannon [4], Turing [5], and Newell, Shaw, & Simon [6] contributed classic analyses of how machines might be programmed to play chess. The first significant performance system was Samuel's program [7] for checkers (draughts), which eventually learned to play at the level of a good county player, far higher than that of Samuel himself. This last circumstance played a valuable part in discrediting the cruder manifestations of the doctrine that 'you only get out what you put in'.

The fundamental mechanism underlying all this work has been a cycle of processes: look-ahead, evaluation and mini-maxing. These derive ultimately from a method used to establish a 'foregone conclusion theorem' for such games (two person, zero sum, perfect information, no chance moves) which states that the outcome value can be computed on the assumption that both players follow a (computable) best strategy. For a trivial game, such as

that schematized in Fig. 11.1(a), the computation can actually be performed: all terminal board positions are assigned values by the rules of the game, and these are 'backed up' by the minimax assumption that White will always choose the immediately accessible position which has the maximum value and that Black will select the one with the minimum value. Clearly the procedure not only demonstrates a theorem but also defines a strategy.

But what is to be done when, as in any serious game, it is not practicable to look ahead to the end? Turing and Shannon independently suggested looking ahead as far as practicable, to what may be termed the 'look-ahead horizon', assigning some approximate values to the positions on the horizon by an evaluation function, and backing these up by the same minimax rule. The corresponding strategy says 'choose that immediate successor which has the highest backed-up value'.

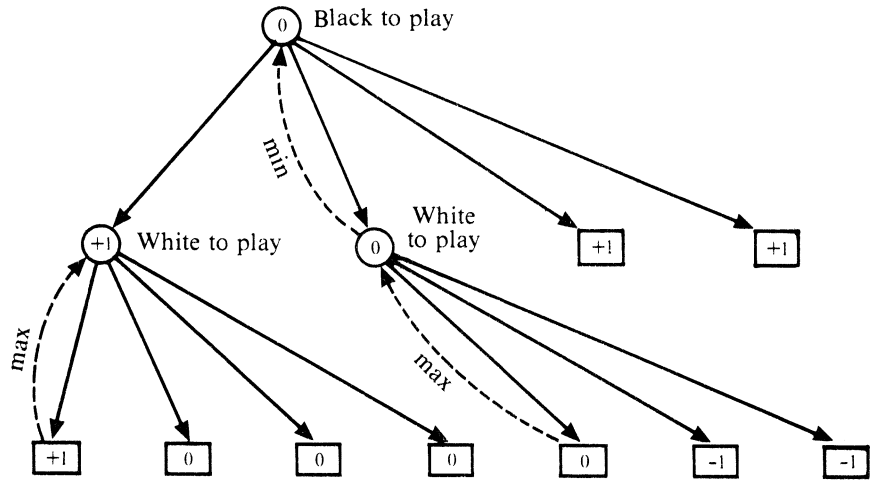
This rule has been proved empirically in numerous game-playing programs, but in spite of its intuitive appeal it has never been formally justified†. The question is posed diagrammatically in Fig. 11.1(b).

Search procedures form part of the armoury of the operations-research man and the computer professional. Stemming from such work as Samuel's, people concerned with game playing and problem solving have implemented mechanisms for guiding the search, first, by forming sub-problems [8] or, second, by making heuristic estimates of distance-to-goal [9]. Various theorems have established conditions under which such techniques can be used without sacrificing the certainty of termination or the optimality of the solution found [10, 12, 13].

The use of an 'evaluation function' to guide the search is a way of smuggling human *ad hoc* knowledge of a problem in through the back door. There is no cause to disdain such a route; it is after all one of the principal channels through which natural intelligences improve their understanding of the world. At the same time automatic methods have been developed for improving the reliability with which problem states are evaluated [11].

Samuel's early work on game-learning [7] indicated that seemingly pedestrian mechanisms for the storage and recall of previously computed results can have powerful effects on performance. Recently the combination of rote learning schemes with heuristic search has been shown to have applications to plan formation in robots [13, 14]. To exploit the full power of this combination, whether in game-playing, in robotics, or in other applications, one would like the rote dictionary to contain generalized descriptions of 'concepts' (for example, of classes of game-positions 'essentially similar' from a strategic point of view) to be looked up by processes of recognition, rather than by point-by-point matching. Such a dictionary is to be used in the style: 'If the situation is of type A, then perform action x, if of type B, then action y', and so on. One is then in effect processing a 'decision table' which is formally equivalent to a computer program. There is thus a direct link between work on the automatic synthesis of strategies in game playing and robotics, and work directed towards automatic program-writing in general.

†Beal and Bratko have recently proved sufficient conditions (in *Advances in Computer Chess*, Vol. 3, Pergamon, 1986).



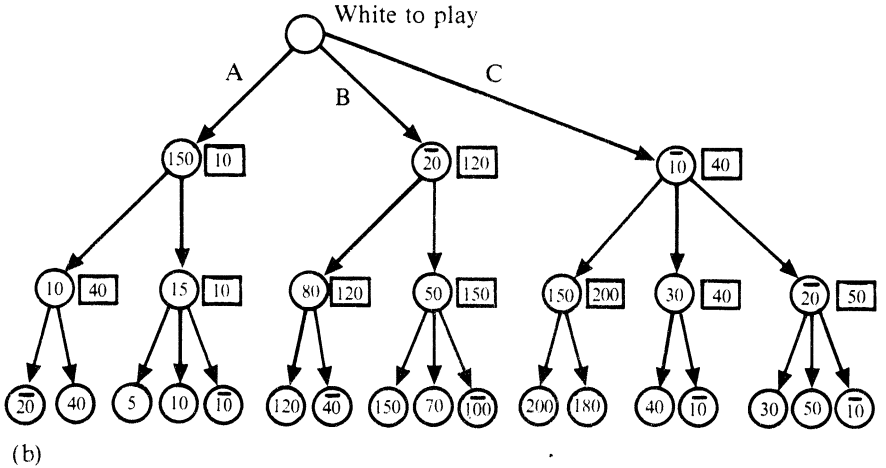


Fig. 11.1 — (a) The root of this two-level look-ahead tree acquires a value by alternate application of the 'max' and 'min' functions. If alternation is extended backwards from all terminal positions of the game tree, the initial position of the entire game will ultimately be assigned a value. Terminal positions are shown as boxes. (b) Look-ahead tree in which the nodes are marked with 'face values' (bars over negative values). Boxed figures are values backed up from the look-ahead horizon. If move-selection were decided by face values, then move A would be chosen, but if backed-up values then move B. What is the rationale for B?

Recognition usually involves the matching of descriptions synthesized from sensory input with stored 'canonical' descriptions of named objects, board positions, scenes, situations and so on. Choice of representation is crucial. At one extreme, predicate calculus [15] has the merit of generality, and the demerit of intractability for updating and matching descriptions of objects, positions, scenes or situations; at the other extreme lie simple 'state

vector' representations, which fall down through awkwardness for handling complex inter-relationships. Somewhere in the middle lies the use of directed labelled graphs ('relational structures', 'semantic nets') in which nodes stand for elements and arcs for relations. Impressive use of these structures has been made in a study of concept formation in the context of machine vision [16].

Language interpretation has been the graveyard of many well-financed projects for 'machine translation'. The trouble proved to be the assumption that it is not necessary for the machine to 'understand' the domain of discourse. One of the first demonstrations of the power of the semantic approach in this area was Bobrow's STUDENT program [17] for answering school algebra problems posed in English. A program by Woods, Kaplan and Nash–Webber [18] for the interrogation in English of a database with a fixed format has been used by NASA scientists to answer questions about Moon rocks. An essay by Winograd [19] on computer handling of English language dialogue, again making intensive use of an internal model of the dialogue's subject matter, has left no doubt that machine translation can only be solved by knowledge-based systems. The knowledge base required to render arbitrary texts non-ambiguous is now recognized to be bounded only by the knowledge possessed by their authors. Winograd compares the following two sentences:

> The city councilmen refused to give the women a permit for a demonstration because they feared violence.
> The city councilmen refused to give the women a permit for a demonstration because they advocated revolution.

The decision to refer 'they' to 'councilmen' in the first case and to 'women' in the second implies a network of knowledge reaching into almost every corner of social and political life.

Mass spectrogram analysis was proposed by Lederberg as a suitable task for machine intelligence methods. The heuristic DENDRAL [20] program developed by him and Feigenbaum now outperforms post-doctoral chemists in the identification of certain classes of organic compounds. The program is a rich quarrying-ground for fundamental mechanisms of intelligence, including the systematic conjecture of hypotheses, heuristic search, rote learning, and deductive and inductive reasoning. I shall refer back to this work later in connexion with the use made by intelligent systems of stored knowledge.

Of all the knowledge systems which have been attempted, robotics is perhaps the most simple in appearance. In reality, however, it is the most complex. The chess amateur can appreciate that Grandmaster chess has depth and subtlety. But there is no such thing as a human amateur at tasks of navigation and 'hand–eye' assembly. Every man is a Grandmaster at these tasks, having spent most of his waking life in unwitting but continual practice. Not having been informed that he is a Gramdmaster, and having long since stored most of his skill at a subliminal level, he thinks that what

seems subjectively simple is objectively so. Experience of research in robotics is a swift and certain cure. Something of the depth of analysis which is required can be gleaned from the discussion by McCarthy & Hayes [21] of the properties which should be possessed by a calculus of situations, actions and causal laws.

The crux of any such calculus is how to represent in a formal language what the robot knows about its world. McCarthy & Hayes distinguish 'epistemologically adequate' and 'heuristically adequate' representations. (In an earlier generation Ryle [22] contrasted 'knowing that' and 'knowing how'.) 'The epistemological part is the representation of the world in such a form that the solution of problems follows from the facts expressed in the representation. The heuristic part is the mechanism that, on the basis of the information, solves the problem and decides what to do.'

I shall consider now what is probably the simplest world to be seriously discussed, that of Popplestone's 'blind hand' problem (internal report, Department of Machine Intelligence, Edinburgh), with the object of indicating that there is more to robot reasoning than meets the eye, and expanding a little the epistemological–heuristic distinction.

A blind, insentient, robot shares with one or more 'things' a world consisting of only two places, 'here' and 'there', and has available to it the actions 'pickup', 'letgo' and 'go'. 'Pickup' is non-deterministic and causes (if the hand is empty when the action is applied) a 'thing' selected at random from the place where the robot is, to acquire the property 'held'. An initial situation called 'now' is defined, in which it is asserted that everything at 'here' (and there is at least one such) has the property 'red'. A goal situation is defined as one in which at least one red thing is at 'there'.

INVARIANT FACTS AND LAWS

The kinds of facts which the robot needs to know include that the robot and anything held by it must be in the same place, and that something cannot be in both places at once. Using a prescription of Green [23], a formalization of this apparently trivial problem in first order logic might start along the following lines. (The variables *t, p* and *s* are to be interpreted as standing for objects, places and situations respectively.)

for all t,p,s: held(thing(*t*),*s*) *and* at(thing(*t*),*p*,*s*) *implies* at(robot,*p*,*s*),
for all t,p,s: held (thing(*t*),*s*) *and* at(robot,*p*,*s*) *implies* at(thing(*t*),*p*,*s*),
for all p,s: at(robot,*p*,*s*) *implies* at(thing(taken(*s*)),*p*,*s*),
for all t,s: at(*t*, here,*s*) *implies not* at(*t*, there, *s*).

The conjunction of these statements describes some of the physics of this world. The last statement, for example, asserts that an object cannot be both at 'here' and at 'there' in one and the same situation.

The initial situation, 'now', is described in like manner:

for all t: at(*t*, here, now) *implies* red(*t*), at(thing (a), here, now).

The latter statement merely asserts that at least one thing (represented by the constant (a) is at 'here' in situation 'now'. The function 'thing' is a convenience for distinguishing other objects from the robot, whom we may wish to exclude from some otherwise universal statements — like one implying that the robot is 'held', for instance.

How can the machine be enabled to reason about the chains of possible consequences derivable from 'now' and so to construct an action chain leading to a goal situation? The goal may be defined, using Green's 'answer' predicate [24], as:

for all t, s: at(*t*, there, s) *and* red(*t*) *implies* answer(*s*).

But how do we handle the actions? The contrast between epistemological and heuristic criteria becomes very sharp at this point. Consider two approaches.

One can go the whole way and stick to formal logic, defining the transition laws of our world under the various actions. For example, the first of the following three 'letgo' axioms translates freely 'in the situation produced by doing a "letgo", nothing is held':

for all t, s: *not* held (thing(*t*), do(letgo, *s*))
for all t,p,s: at(*t,p,s*) *implies* at(*t, p,* do(letgo, *s*))
for all t,p,s: not at(*t,p,s*) *implies not* at(*t, p,* do(letgo, *s*))

and similarly for the other actions.

Now the problem of plan construction is reduced to one of logical deduction, in fact deduction of the statement 'answer (do(go(there),do-(pickup,do(go(here),do(letgo,now))))))'. This says, in English, that 'the goal situation is the one resulting from doing a "go there" in the situation resulting from doing a "pickup" in the situation resulting from doing a "go here" in the situation resulting from doing a "letgo" in the situation "now"', and it is clear how this can be reinterpreted as an algorithm.

This deduction can in principle be mechanized, but there are two severe snags. First, the need to incorporate 'frame axioms' [24,25] (which spell out all the facts which remain unchanged after the performance of given actions, as in the last logic statement above) escalates for nontrivial problems and renders the automatic deduction process intractable even in the present toy problem. Second, the logic representation is not heuristically adequate.

On the other hand, one can go to the other extreme, and express the whole problem as a computer simulation couched in a suitable programming language, matching situations with data structures and actions with procedures. But this approach encounters difficulties with the epistemological criterion, for the structure of the problem world can be readily complicated so that it can no longer easily be described by the use of simple representations of the 'state vector' type. Various attacks are being made on the representation problem in an attempt to make the best of both worlds, the

epistemological and the heuristic. Some good early suggestions were made by Popplestone, using essentially the same blind hand problem, and were reviewed in *Nature* [27] two years ago. Since then powerful new programming aids, such as the PLANNER [28], QA4 [29] and CONNIVER [30] languages have come into play. In addition particular mention should be made of the Stanford Research Institute's study of autonomous plan formation [14, 15], in which many of the matters discussed above have been under experimental investigation.

The key ideas on which much work centres is that plan construction should be conceived as a search through a space of states of knowledge to generate a path connecting the initial knowledge state to one which satisfies the goal definition. Everything turns on finding ways of representing knowledge states so that the transformation of one into another can be neatly computed from the definition of the corresponding action ('What will I know about the state of affairs after doing A?').

EXPERIMENTAL ROBOTICS

The STRIPS system [14, 15] at Stanford Research Institute combines reasoning in first-order predicate calculus with heuristic search. In the situation depicted in Fig. 11.2 the robot must devise a plan for pushing objects around

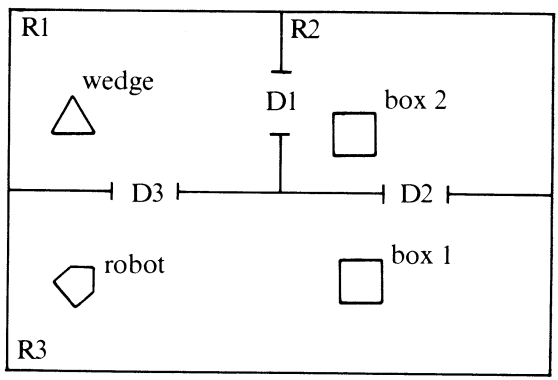


Fig. 11.2 — Robot environment for a constant problem [31].

so that one of the boxes ends up in room R1, subject to the constraint that at no time must the wedge be in the same room as a box. If the plan goes wrong, the system must be capable of recovering from error state and, if possible, 'mending' the failed plan appropriately. Facilities are incorporated whereby successful plans are automatically 'remembered' and their elements recombined for use in appropriate future situations [14].

Following simultaneous development of the idea of optical ranging in

Japan [32], Britain (R. J. Popplestone, personal communication) and America [33], Stanford University's robot project uses a laser optical ranging system for mapping the three-dimensional surfaces of 'seen' objects. Another branch of the same project is currently able to assemble an automobile water pump comprising two pieces, a gasket and six screws (J. Feldman, personal communication). This is done blind, using mechanical feedback.

At Edinburgh automatic assembly is also under study. Programs exist for packing simple objects onto a confined surface, identifying a limited set of objects by visual appearance, and solving problems of stacking rings on pegs [34].

In industrial laboratories, notably in America (for example, the Charles Stark Draper Laboratory of MIT) and Japan [35], automatic assembly studies are multiplying.

IDEA OF A THEORY

I have already mentioned the abstracting of pieces of theory from performance systems such as those listed above. What is meant by 'theory' in this context? I have just considered a fragment of simple robot world theory, and one can, of course, speak of a piece of chess end-game theory (for example, that expressed by Tan's program [36] for the two-kings-and-one-pawn end-game) or of the theory of mass spectrometry embedded in the heuristic DENDRAL program. One can even legitimately speak of Winograd's program as constituting a linguistic theory, or at least as containing or implying one. But these theories are descriptive of specific domains, not of intelligence itself.

It would be naive to pretend that the search for a meta-theory is something new, or even that it is anything but old philosophy in new dress. An early name suggested for what is now 'artificial intelligence' was 'epistemological engineering' (P. M. Woodward, personal communication). The new epistemology, however, has a trick which the old philosophers lacked, namely to express any given theory (of knowledge, reasoning, abstraction, learning and the like) in a sufficiently formal style to program and test it on the machine.

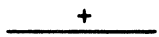
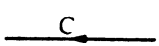
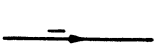
Hence there is no longer a meaningful distinction to be drawn between a theory of some given intelligent function, and an algorithm for carrying it out (which could in turn be converted into a program for some particular machine) together with any useful theorems for describing the algorithm's action. Algorithms, then, are theories, and this has been true for a long time. But there have been no reasonable mechanisms available for handling them. Mathematics, on the other hand, has had the necessary mechanisms for manipulating the formalisms which it uses for describing physical systems. Hence closed-form mathematics has been the 'typical' embodiment of theory in the physical sciences. By contrast, the 'typical' embodiment of theory in cognitive engineering is algorithmic.

WHAT USE IS KNOWLEDGE?

The value of stored knowledge to a problem-solving program again divides into epistemological and heuristic parts. In the first place sufficient knowledge must be present for solutions to be in principle deducible. But that is only the start. Heuristically, the value of knowledge is that it offers ways of avoiding, or greatly reducing, processes of search. The natural enemy of the worker in the field of artificial intelligence is the 'combinatorial explosion', and almost his entire craft is concerned with ways of combating it. The following three examples illustrate the use of stored knowledge to damp off combinatorial explosions.

First, Tables 11.1 and 11.2 show the number of combinatorially possible

Table 11.1 — A labelling scheme [43]

	1	Convex edge
	2	Obscuring edges — obscuring body lies to right of arrow's direction.
	3	
	4	Cracks — obscuring body lies to right of arrow's direction
	5	
	6	Shadows — arrows point to shadowed region
	7	
	8	Concave edge
	9	Separable concave edges — obscuring body lies to right of arrow's direction — double arrow indicates that three bodies meet along the line.
	10	
	11	

ways in picture-processing of labelling various patterns of intersecting lines, contrasted with the number that are physically possible on the assumption that they arise in retinal projections of three-dimensional scenes composed of plane polyhedral bodies, such as that shown in Fig. 11.3(a). The computer program achieves this order of reduction by the use of an appropriate theory. Here I shall review briefly a subset of the theory, adequate for interpreting line drawings of plane-surfaced polyhedra, with trihedral ver-

Table 11.2 — Comparison of number of combinatorially possible labellings with the number that are physically possible [43]

	Approximate number of combinatorially possible labellings	Approximate number of physically possible labellings
	2500	80
	125000	70
	125000	500
	125000	500
	6×10^6	10
	6×10^6	300
	6×10^6	100
	6×10^6	100
	6×10^6	100
	6×10^8	30

tices only and without shadows. In this way the flavour can be imparted of the kind of reasoning involved in more complex cases.

Each line in such a drawing can be assigned to one or another of various possible causes: it corresponds to a convex edge, a concave edge, or to an edge formed by two surfaces, only one of which is visible. A corresponding label can be attached to each line, as has been done in Fig. 11.3(b) using Huffman's conventions [37]. The remarkable fact emerges from Huffman's analysis that only a few of the combinatorially possible ways of labelling such drawings correspond to physically possible structures in the outside world: only twelve distinct configurations of lines around vertices are possible. A computer program can use the theoretical constraints to process the picture, by searching through the space of possible labellings for those which are

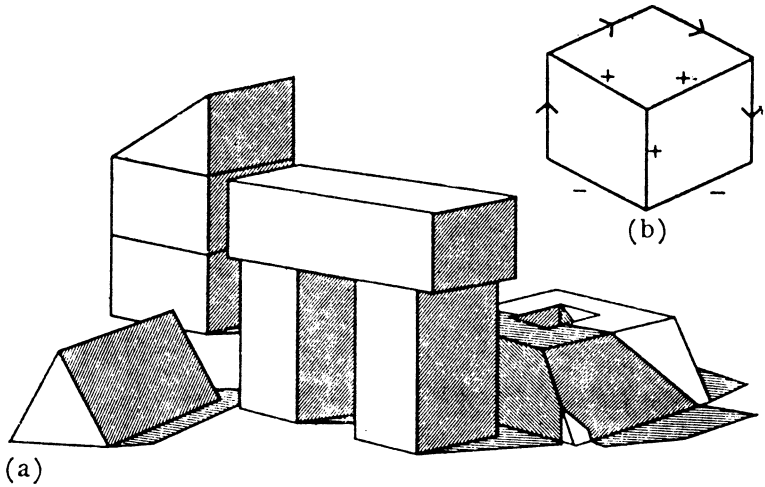


Fig. 11.3 — (a) A complex three-dimensional scene. (b) Huffman labels for a cube. Plus implies a convex edge, minus implies concave, and an arrow implies that only one of the edge-forming surfaces is visible The cube is assumed to rest on a plane surface.

legal (i.e. do not entail that any line should receive two different labels) under the constraints.

Second, Table 11.3 contrasts the number of topologically possible molecular graphs corresponding to given empirical formulae with the number of candidate interpretations remaining after the heuristic DENDRAL program has applied its stored theory of chemical stability. The program constructs, using evidence of various kinds, a GOODLIST of substructures which must appear in any structure hypothesized by the program and a BADLIST of substructures which must not appear. As a simple example, at a given stage down a search tree might be the partial hypothesis —CH_2—O—CH_2— and a possible next move for the structure-generator procedure might be to attach a terminal carbon, forming —CH_2—O—CH_2—CH_3. But unless the data contains peaks at mass 59 and at the molecular weight minus 15 this continuation is forbidden. Again, the structure-generator can be made to handle as a 'super-atom' a fragment indicated by the mass spectrum. Additional opportunities to do this arise when the presence of methyl super-atoms can be inferred from nuclear magnetic resonance data, when available.

Third, McCarthy's problem of the mutilated checkerboard [38] is quintessential to the point here discussed. The squares at opposite corners of an 8×8 checkerboard are removed, leaving sixty-two squares. Thirty-one dominoes are available, each of such a size and shape as to cover exactly two adjacent squares of the checkerboard. Can all the sixty-two squares be exactly covered by some tessellation of the thirty-one dominoes?

However sophisticated the search procedure which a heuristic program

Table 11.3 — Comparison of the number of topologically possible molecular graphs corresponding to given empirical formulae with the number of candidate interpretations remaining after the heuristic DENDRAL program has applied its stored theory of chemical stability

		Number of isomers	Number of inferred isomers A	B
Thiol	1-nonyl	405	89	1
	n-decyl	989	211	1
	n-dodecyl	6045	1238	1
Thioether	di-n-pentyl	989	12	1
	di-n-hexyl	6045	36	1
	di-n-heptyl	38322	153	1
Alcohol	n-tetradecyl	38322	7639	1
	3-tetradecyl	38322	1238	1
	n-hexadecyl	151375	48865	1
Ether	di-n-octyl	151375	780	1
	bis-2-ethylhexyl	151375	780	21
	di-n-decyl	11428365	22366	1
Amine	n-octadecyl	2156010	48865	1
	N-methyl-n-octyl-n-nonyl	2156010	15978	1
	N,N-dimethyl-n-octadecyl	14715813	1284792	1

A, Inferred isomers when only mass spectrometry is used; B, Inferred isomers when the number of methyl radicals is known from nuclear magnetic resonance data [20].

might use to attack this problem by trial and error, the combinatorics of the problem will defeat it. If the reader is unsure of this, let him mentally enlarge the board to say, 80×80, or $10^8\times10^8$. But so long as the dimensions of the board are both of even or both of odd length (such boards are called 'even' boards) then the problem stays the same for any solver armed with certain crucial pieces of knowledge, namely: that the two squares which are removed from opposite corners of an even board must be of the same colour, and that each domino must cover exactly one white and one black square. The problem now falls apart. The mutilated checkerboard cannot be covered.

To discover formal schemes within which such key facts can automatically be mobilized and their relevance exploited in an immediate and natural fashion is closely bound up with what was earlier referred to as 'the representation problem'. A familiar example is that certain representations of the game of Nim trivialize the calculation of a winning strategy; but the program capable of inventing such representations is yet to be devised.

PROGRESS TOWARDS AN ICS

Two years ago I discussed in *Nature* [27] the possibility of implementing in software an Integrated Cognitive System (ICS). The attainment on a laboratory scale of a 'working model', it was suggested, could be used as an

indicator of ultimate feasibility. A working model of an ICS, as a minimal set of requirements, should be able: to form an internal representation of its task environment, summarizing the operationally relevant features; to use the representation to form plans of action, to be executed in the task environment; to perform directed perceptual sampling of the environment to switch execution along conditional branches of the plan; to recover from error state when execution fails; to cope with complex and ill-structured environments; to be told new goals and to work out its own approaches to them; and to use the record of past failures and successes to revise and extend the representation inductively.

A computer program which was not able to do most of the above, however excellent a feat of software technology it might be, would not count as an artificial intelligence program. The guidance software for the Apollo on-board computer, written for NASA by Draper Laboratories (J. Moore, privately circulated report, Department of Computational Logic, University of Edinburgh) and charged with the task of getting the spacecraft to the Moon and back, is disqualified on this criterion. On the one hand, it is an acknowledged masterpiece, and on the other, in common with other and lesser automatic control systems, it scores a significant mark only for the third item in the above list.

The on-board computer does not need to plan because hand-coded routines have been provided for all probable situations — analogous, perhaps, to the elaborate, but essentially reflex, nervous system of an insect. The reason for regarding the Apollo on-board system as sub-intelligent is thus concerned with the nature of the internal model which it has of its environment. More than a quarter of a century ago Craik [39] first called attention to the crucial role in thought and perception of internal models. The world of the Apollo computer is so simple and determinate that its behaviour can be completely characterized by computationally simple equations. These equations, which comprise the system's 'internal model' in Craik's sense, capture the dynamics of all possible configurations of the objects of its world, and supply all information needed about their interactions and properties.

But consider the mission: not to go to the Moon and back, but the much harder one of going down to the tobacconist and back. By contrast with the space mission, the task environment is exceedingly complex and 'messy' and the unexpected lurks at every point of the route (the stairs may be swept, unswept, blocked ..., the front door may be open, shut, locked ..., the weather may be bright, dull, wet, windy ... and so on). Alternatively, and only a little less taxing (at least the environment does not contain other autonomous beings to worry about), consider the mission of a Mars Rover vehicle, such as that already envisaged by NASA [40] and by the space section of the USSR Academy of Sciences (N. Zagoruiko, personal communication). Arising from the fact that it is not possible to pre-program solutions to all problems which might arise while exploring an unknown terrain, a specific ten-year programme of machine intelligence research is regarded as a necessary preliminary condition for putting such operational

vehicles into commission. Note that if such a vehicle is to handle all the tasks of autonomous exploration, and assembly and use of instruments, which will be demanded of it, then it must score seven out of seven on the criteria posed earlier.

That achievement lies in the future. How do matters stand today with regard to 'working models'? Each of the seven capabilities listed can now be found in one or another experimental system, and there are some systems which exhibit many, or even most, of them. Unfortunately the most interesting capability of all, central to the phenomenon of intelligence, is the one which is still the least well understood, namely inductive generalization. Yet significant progress has been made [11,31].

In summary, incomplete systems are becoming commonplace and complete 'working models', at the most primitive level, now seem not very far off. The likely technological lag before such systems might be upgraded to near-human intellectual performance is a topic for separate consideration.

IMPLICATIONS AND FORECASTING

It would plainly be desirable to find some objective basis for predicting the rate of development and social impact of machine intelligence. An objective basis is lacking at present and it is only possible to record samples of subjective opinion and to categorize lines of enquiry which more objective studies might follow. Fig. 11.4 summarizes some of the results of an opinion

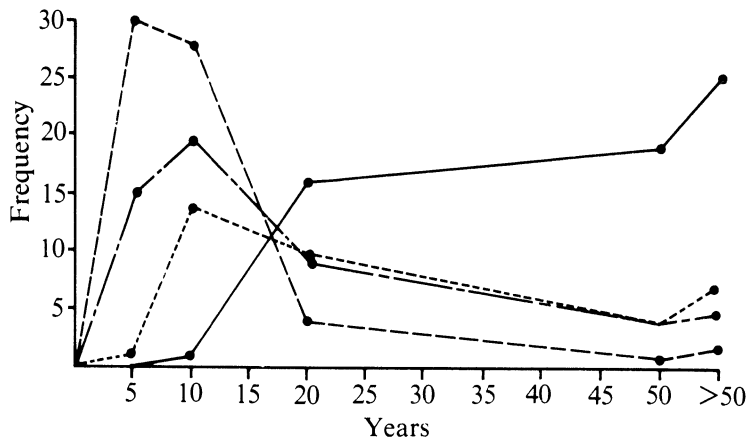


Fig. 11.4 — Opinion poll on machine intelligence. Estimated number of years before: ———, computing system exhibiting intelligence at adult human level; – – – –, significant industrial spin-off; — · — · —, contributions to brain studies; - - - - -, contributions from brain studies to machine intelligence.

poll taken last year among sixty-seven British and American computer scientists working in, or close to, the machine intelligence field.

In answer to a question not shown in Fig. 11.4, most considered that attainment of the goals of machine intelligence would cause human intellec-

tual and cultural processes to be enhanced rather than to atrophy. Of those replying to a question on the risk of ultimate 'takeover' of human affairs by intelligent machines, about half regarded it as 'negligible', and most of the remainder as 'substantial' with a few voting for 'overwhelming'.

A working party recently convened under the auspices of the Rockefeller Foundation at Villa Serbelloni, Lake Como, on June 11 to 15, 1972, considered the gradations through which complex information systems might evolve in the future, ranging from contemporary industrial control systems, and 'data look-up' retrieval, to autonomous computer networks developed for controlling urban functions (telephones, electricity distribution, sewage, traffic, police, banking, credit systems, insurance, schools, hospitals, and so on). The backbone of such systems will develop anyway, by straightforward elaboration of conventional computing technology, including the integration of the various computational networks into total systems. It seems likely that such systems will also ultimately incorporate autonomous planning and decision-taking capabilities, derived as 'spin-off' from developments based on artificial intelligence in, for example, space and oceanographic robotics. A danger could then arise of city dwellers becoming dependent on systems which could no longer be fully understood or controlled. Counter-measures to such dangers might include the introduction of auditing procedures for computer programs, research on program-understanding programs, and system-understanding systems generally, and, finally, the advent of programs to teach the users of intelligent systems.

On the other side of the balance sheet, the working party took preliminary note of several anticipated benefits. The mechanization of industrial production has been associated in the past with the imposition of a deadening uniformity of design. Automated intelligence in the factory could offer the possibility of restoring the diversity and the 'one-off' capability originally associated with human craftmanship. Related to this is the introduction of computer aids for the artist, composer, writer, architect and mathematician. Even the ordinary hobbyist might be enabled to perform feats which would today seem daunting or bizarre — building his own house, publishing his own writings, for example. The possible effects on computer-aided education have been stressed by others [42]. Advances in this area will be of value not only to the young but also to older people as a means of acquiring new skills.

The formulation of an outline scheme of topics, and the compilation of relevant documents, represents an early stage of a study expected to occupy a number of years. Technical developments which occur in the intervening period will doubtless give such studies a firmer basis.

REFERENCES

[1] M. J. Lighthill (1973) *Artificial Intelligence: a general survey*. London: Science Research Council.

[2] A. M. Turing, in *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 3. Edinburgh: Edinburgh University Press (1969).

[3] J. Maynard Smith (1952) *Evolution,* **6**, 127; reprinted in *On Evolution*, 29. Edinburgh: Edinburgh University Press (1972).
[4] C. E. Shannon (1950) *Phil. Mag.*, **41**, 356.
[5] A. M. Turing (1953) in *Faster than Thought* (ed B. V. Bowden), 288. London: Pitman.
[6] A. Newell, J. C. Shaw & H. A. Simon (1958) *IBM J. Res. Dev.*, **2**, 320.
[7] A. L. Samuel (1959) *IBM J. Res. Dev.*, **3**, 210.
[8] D. Michie, R. Ross & G. J. Shannan (1972) in *Machine Intelligence 7* (eds B. Meltzer & D. Michie), 141. Edinburgh: Edinburgh University Press.
[9] J. E. Doran & D. Michie (1966) *Proc. Roy. Soc. A,* **294**, 235.
[10] P. Hart, N. J. Nilsson & B. Raphael (1968) *IEEE Trans. on Sys. Sci. and Cybernetics,* SSC-4, 100.
[11] D. Michie & R. Ross (1969) in *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 301. Edinburgh: Edinburgh University Press.
[12] I. Pohl (1970) *Artificial Intelligence,* **1**, 193.
[13] D. Michie (1971) in *Artificial Intelligence and Heuristic Programming* (eds N. V. Findler & B. Meltzer), 101. Edinburgh: Edinburgh University Press.
[14] R. E. Fikes, P. E. Hart & N. J. Nilsson (1972) *Artificial Intelligence,* **3**, 251.
[15] R. E. Fikes & N. J. Nilsson (1971) *Artificial Intelligence,* **2**, 189.
[16] P. H. Winston (1970) MIT thesis, reprinted as *MAC-TR-76* MIT: Project MAC (1970).
[17] D. G. Bobrow (1964) MIT thesis, reprinted in *Semantic Information Processing* (eds M. Minsky). Cambridge, Mass.: The MIT Press (1968).
[18] W. A. Woods, R. M. Kaplan & B. Nash-Webber (1972) *BBN Report no. 2378.* Cambridge, Mass.: Bolt, Beranek & Newman.
[19] T. Winograd (1970) MIT thesis, reprinted in revised form as *MAC-TR-84.* MIT: Project MAC (1971); also available as *Understanding Natural Language.* Edinburgh: Edinburgh University Press (1972).
[20] E. A. Feigenbaum, B. G. Buchanan & J. Lederberg (1971) in *Machine Intelligence 6* (eds B. Meltzer & D. Michie), 165. Edinburgh: Edinburgh University Press.
[21] J. McCarthy & P. J. Hayes (1969) in *Machine Intelligence 4* (eds. B. Meltzer & D. Michie), 463. Edinburgh: Edinburgh University Press.
[22] G. Ryle (1949) *The Concept of Mind.* London: Hutchinson.
[23] C. C. Green (1969) *Proc. Intern. Joint Conf. Art. Intell.* (eds D. E. Walker & L. M. Norton), 219. Washington DC.
[24] B. Raphael (1971) in *Artificial Intelligence and Heuristic Programming* (eds N. V. Findler & B. Meltzer), 159. Edinburgh: Edinburgh University Press.
[25] P. J. Hayes (1971) in *Machine Intelligence 6* (eds B. Meltzer & D. Michie), 495. Edinburgh: Edinburgh University Press.
[26] R. M. Burstall, J. S. Collins & R. J. Popplestone (1971) *Programming in POP-2.* Edinburgh: Edinburgh University Press.

[27] D. Michie (1970) *Nature,* **228**, 717.
[28] C. Hewitt (1971) MIT thesis, reprinted as *Art. Intell. Memo 258.* MIT: Artificial Intelligence Laboratory (1972).
[29] J. F. Rulifson, R. J. Waldinger & J. A. Derksen (1970) *Technical Note 48.* Stanford Research Institute: Artificial Intelligence Group.
[30] D. V. McDermott & G. J. Sussman (1972) *Art. Intell. Memo 259.* MIT: Artificial Intelligence Laboratory.
[31] R. E. Fikes, P. E. Hart & N. J. Nilsson (1972) in *Machine Intelligence 7* (eds B. Meltzer & D. Michie), 405. Edinburgh: Edinburgh University Press.
[32] Y. Shirai & M. Suwa (1971) *Proc. Second Intern. Joint Conf. Art. Intell.*, 80. London: British Computer Society.
[33] P. M. Will & K. S. Pennington (1971) *Artificial Intelligence,* **2**, 319.
[34] D. Michie, A. P. Ambler, H. G. Barrow, R. M. Burstall, R. J. Popplestone & K. J. Turner (1973) Vision and manipulation as a programming problem. *Proc. First Conf. on Industrial Robot Technology,* 185–90. University of Nottingham.
[35] M. Ejiri, T. Unon, H. Yoda, T. Goto & K. Takeyasu (1971) in *Proc. Second Intern. Joint Conf. on Art. Intell.*, 350. London: British Computer Society.
[36] S. T. Tan (1972) *Research Memorandum MIP-R-98.* University of Edinburgh: School of Artificial Intelligence.
[37] D. A. Huffman (1971) in *Machine Intelligence 6* (eds B. Meltzer & D. Michie), 295. Edinburgh: Edinburgh University Press.
[38] J. McCarthy (1964) *Memo no. 16.* Stanford University: Stanford Artificial Intelligence Project.
[39] K. J. W. Craik (1952) *The Nature of Explanation.* Cambridge University Press.
[40] R. Choate & L. D. Jaffe (1973) Science aspects of a remotely controlled Mars surface roving vehicle. *Proc. First National Conf. on Remotely Manned Systems* (RMS), *Exploration and Operation in Space* (ed. E. Heer), 133–48. California Institute of Technology.
[41] G. Plotkin (1969) in *Machine Intelligence 5* (eds B. Meltzer & D. Michie), 153; and (1971) in *Machine Intelligence 6* (eds B. Meltzer & D. Michie), 101. Edinburgh: Edinburgh University Press.
[42] S. Papert (1971) *Art. Intell. Memo 247.* MIT: Artificial Intelligence Laboratory.
[43] P. H. Winston (1972) in *Machine Intelligence 7* (eds B. Meltzer & D. Michie), 431. Edinburgh: Edinburgh University Press.

12

Knowledge engineering (1973)

A widespread theme in artificial intelligence is an interest in problem-solving mechanisms. One can relate this to automatic program-writing: will computers ever be able to write their own programs to a significant extent? The topic is beginning to enjoy a considerable vogue in America, where there tend to be two directions of approach. On the one hand there is the approach through computation theory, and on the other hand there is the artificial intelligence approach via study of how knowledge can be represented and used in the machine. Signs of merging of approaches are already apparent. A recent advance by Boyer & Moore, [1] working in Professor Meltzer's Department of Computational Logic at Edinburgh, demonstrates automatic methods for proving LISP programs. Their program, as well as writing new programs on its own account, uses generalization and generates its own induction hypotheses — true elements of 'knowledge engineering'.

THE KNOWLEDGE APPROACH

In the knowledge approach we distinguish three levels, as indicated in Table 12.1.

The lowest level is the one at which AI programmers are still struggling today. The highest level of all, a long way from attainment, is a 'knowledge machine' able to find out how to do things by reading books — how to play better chess by reading books, how to build model aeroplanes by reading the

Table 12.1 — Three levels at which task-specific knowledge can in principle be got into the machine. The arrow indicates the desired extension of technique for 'hand–eye' vision and assembly tasks

	Chess	Assembly	Chemistry
1. Transfer (by programmer) of algorithmic knowledge from book+programmer into program	Program executes standard end-games strategies (Huberman [2], Tan [3])	Program takes kit of parts and makes model car (e.g. Michie *et al.* [4])	Program interprets mass spectrograms (Feigenbaum *et al.* [5])
2. Generation (by machine) of descriptions and generation (by machine) of action-scheme sequences (plans), e.g. to bridge gaps in book knowledge	Program uses given knowledge plus simulated playing experience to extend end-game theory (i.e. generate end-game strategies *de novo*)	↓	Program extends theory of molecular bond stability in light of example identifications (Buchanan *et al.* [6])
3. Acquisition (by machine) of algorithmic knowledge by reading books	Program improves its play by reading chess books	Program uses instructions and diagrams to make model car	Program copes with new families of compound by looking up chemistry texts

written instructions and diagrams, how to fashion furniture by reading cabinet-making manuals, how to interpret mass spectrograms by reading chemistry textbooks, and so forth. The obstacles to their doing this are not purely linguistic, but are to do with the need for machine 'understanding' of the book's subject-matter.

Level 3 clearly demands mastery of computational linguistics (and much else besides). Terry Winograd's success [7] at MIT with his 'blocks world', and his computer program which discourses convincingly about this world in plausible English, encourages the belief that 'doing a Winograd' for more complex worlds, such as those of chess, hand–eye assembly, or chemistry, will come within reach. What is very clear from his work is that linguistic success can only be built on a thoroughly engineered knowledge system for the domain concerned.

KNOWLEDGE ABOUT CHESS

To get the flavour of level-1 programming in chess, Dr Soei Tan in our Department has recently transferred into program the knowledge about king and pawn *vs* king endings which is contained in a few pages of text in the classical chess books. It turns out that there is much more to transferring such material into program than meets the eye. The main reason is that most of the knowledge which ought to be in the book looks at first sight as though it's there, but turns out on closer scrutiny to be largely missing. The book, of course, is written not for a machine but for an intelligent human reader, who can be guaranteed to 'bridge the gaps' by referring to his own understanding

of the problem. In the process of bridging the gaps for the computer, Tan found that he had to extend the theory of King and pawn end-games, and that level-2 problems, connected with abstraction and learning from examples, cannot be postponed if efficient level-1 processes are to be devised.

One of the first tasks given to the chess beginner is to master certain elementary strategies for checkmate. Chapter 1 of Reuben Fine's *Basic Chess Endings* deals with (1) Q and K *vs* K, (2) R and K *vs* K, (3) 2 B's and K *vs* K, (4) B, Kt and K *vs* K. This entire chapter occupies only six pages of explanatory text and diagrams. Yet the problem of transferring just (2), (3), and (4) — four pages of the book — into program sustained a three-year PhD study by Barbara Huberman [2]. The difficulty is connected with the fact that the human reader of Fine's book brings to it a considerable prior body of knowledge, while Huberman had to write her programs for a system containing no pre-existing knowledge of any aspects of chess at all. The connection between ease of transfer of knowledge from a book and the possession of prior knowledge by the target system can be illustrated in an extreme fashion by asking the reader to imagine trying to learn R and K *vs* K without even knowing the rules of chess armed only with Fig. 12.1 and the

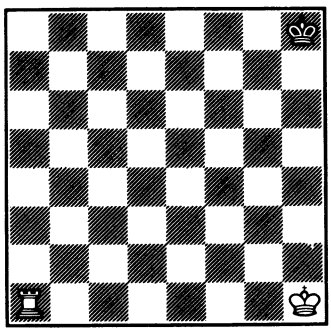


Fig. 12.1 — The ending Rook and King against King.

following text from Capablanca): The principle is to drive the opposing King to the last line on any side of the board. In this position the power of the Rook is demonstrated by the first move, R–R7, which immediately confines the Black King to the last rank, and the mate is quickly accomplished by: 1 R–R7, K–K1; 2 K–Kt2 ...

(The combined action of King and Rook is needed to arrive at a position in which mate can be forced. The general principle for a beginner to follow is to keep his King as much as possible on the same rank, or, as in this case, file, as the opposing King. When, in this case, the King has been brought to the sixth rank, it is better to place it, not on the same file, but on the one next to it towards the centre.)

2 . . . K–B1; 3 K–B3, K–K1; 4 K–K4, K–Q1; 5 K–Q5, K–B1; 6 K–Q6 ...

(Not K–B6, because then the Black King will go back to Q1 and it will take much longer to mate. If now the King moves back to Q1, R–R8 mates at once.)
6 . . . K–Kt1; 7 R–QR7, K–R1; 8 K–B6, K–K1; 9 K–Kt6, K–R1; 10 R–B8 mate.

It has taken exactly ten moves to mate from the original position. On move 5 Black could have played K–K1, and, according to principle, White would have continued 6 K–Q6, K–B1 (the Black King will ultimately be forced to move in front of the White King and be mated by R–R8), 7 K–K6, K–Kt1; 8 K–B6, K–R1; 9 K–Kt6, K–Kt1; 10 R–R8 mate.

The problem of developing a super-teachable programming system in which Huberman's accomplishment would be, say, a three-day instead of a three-year task shades into the problem of endowing a program with so much prior understanding of chess that it would be capable of doing the whole Huberman's job for her — synthesizing strategies (2), (3), and (4) *de novo*. R. Ross and I are starting to look at what might be involved in such a feat, using a 'search-memorize-generalize' scheme described elsewhere [8].

KNOWLEDGE ABOUT 'HAND–EYE' ASSEMBLY

Similar issues are raised by work with 'hand–eye' robots, such as Edinburgh's FREDDY. This project, under Dr R. M. Burstall's supervision, has reached the stage where the user can in a few hours transfer to the machine his knowledge about how to recognize the parts of a construction kit and how to assemble them to make, say, a toy car. At the end of this 'teaching' phase the robot is able to perform the desired assembly, using its TV 'eye' and mechanical 'hand', with fair reliability. Once again, further streamlining of the man-machine process demands methods by which the machine can fill in the gaps in what its instructor tells it. Mechanizing this gap-filling process is of course a particular instance of the automatic programming problem. R. J. Popplestone [9] is developing just such a system with reference to fairly complex robot movements, for example fitting a rod into a socket, subject to various constraints.

Experimental robotics, to which Robin Popplestone's study belongs, involves the wider international scene of AI in its relationship to technology. The broad spectrum of this relationship was reviewed in the USA at the recent 'First National Conference on Remotely Manned Systems (RMS)' which included an AI forum to consider such questions as: what kind of advice can an AI researcher provide to the RMS designer? When should a system designer expect to be able to use AI results? Are there likely to be any software packages for other users coming out of AI laboratories? What is the adaptability potential to other workers of high-level software such as LISP or PLANNER? Should there be a field of applied AI that bridges the gap between the AI laboratory and other engineering laboratories? And finally, what relationship should there be between AI and other engineering disciplines, for example control theory, material science, etc.?

Martin Levine (McGill) and Meir Weinstein [10] (Cal Tech) have written the following summing up of the forum's outcome:

> 'It seems that the field of artificial intelligence could be a rather large source of support to other disciplines. It can support with know-how particularly with regard to robotics, manipulators and sensors. AI has developed and is developing new concepts in software which could also be extended and used in other disciplines.'

MACHINE-ORIENTED NOTATIONS

Associated with this new technology are certain scientific and philosophical issues. I believe that we are seeing, not only in AI but in Computer Science more generally, the emergence of new techniques for handling our internal intellectual models of the world in symbolic forms. Among past revolutions of this kind one might instance the invention of writing and the introduction of algebra. Machine-oriented notations for describing our messy and complex surroundings are now arising from pressures exerted by AI research for more flexible programming languages: LISP, POP-2, PLANNER, SAIL, QA4, CONNIVER, and others. Ultimately, perhaps in radically modified form, these innovations will reach the market place, as has already occurred in the case of POP-2. But the point of origination has been in almost every case academic.

REFERENCES

[1] R. S. Boyer & J. S. Moore (1973) *Proving theorems about LISP functions Memo 60.* University of Edinburgh: Department of Computational Logic.

[2] B. J. Huberman (1968) A program to play chess and games. *Technical Report* no. CS 106. Stanford University: Computer Science Department.

[3] S. T. Tan (1972) Representation of knowledge for very simple pawn endings in chess. *Research memorandum MIP-R-98.* University of Edinburgh: Department of Machine Intelligence.

[4] D. Michie, A. P. Ambler, H. G. Barrow, R. M. Burstall, R. J. Popplestone & K. J. Turner (1973) Vision and manipulation as a programming problem. *Proc. First Conf. in Industrial Robot Technology,* 185–9. Nottingham: University of Nottingham.

[5] E. A. Feigenbaum, B. G. Buchanan & J. Lederberg (1971) On generality and problem solving: a case study using the DENDRAL program. *Machine Intelligence 6* (eds B. Meltzer & D. Michie), 165–90. Edinburgh: Edinburgh University Press.

[6] B. G. Buchanan, E. A. Feigenbaum & N. S. Sridharan (1972) Heuristic theory formation: data interpretation and rule formation. *Machine*

Intelligence 7 (eds B. Meltzer & D. Michie), 167–90. Edinburgh: Edinburgh University Press.

[7] T. Winograd (1971) Procedures as a representation for data in a computer program for understanding natural langauge. MIT thesis, reprinted in revised form as MAC-TR-84, MIT Project MAC; also available as *Understanding Natural Language,* Edinburgh: Edinburgh University Press (1972).

[8] D. Michie (1971) Formation and execution of plans by machine. *Artificial Intelligence and Heuristic Programming* (eds N. V. Findler & B. Meltzer), 101–24. Edinburgh: Edinburgh University Press.

[9] R. J. Popplestone (1973) Solving equations involving rotations. *Research Memorandum MIP-R-99.* University of Edinburgh: Department of Machine Intelligence.

[10] M. D. Levine & M. Weinstein (1973) A review of the First National Conference on Remotely Manned Systems (RMS), Exploration and Operation in Space. *Firbush News 3,* 54-63. University of Edinburgh: Department of Machine Intelligence.

13

Machine intelligence as technology (1973)

From time to time the Hitachi Corporation and others in Japan proclaim the goal of building intelligent robots. Their research, and similar projects elsewhere, will contribute in the long run to a development of great industrial novelty — the fully automated factory. In the USA the Rand Corporation is even of the opinion that this is attainable in less than ten years.

If the prevailing social and moral climate were like that of 1873, when all new technology was regarded as an unqualified Good Thing, then predictions of this kind would arouse optimistic excitement, coupled perhaps with some rather jingoist reflections. After all, the United Kingdom could have excellent chances of cornering a share of the world robotics market. But British attitudes have changed since the days when Tennyson [1] wrote:

> 'Men, my brothers, men the workers,
> ever reaping something new:
> That which they have done but earnest
> of the things that they shall do.'

For Tennyson and his contemporaries the common-sense view prevailed that technology is our living, and this view was coupled with idealistic beliefs about inevitable progress towards general well-being:

> '... Forward, forward let us range,

Let the great world spin for ever,
down the ringing grooves of change.'

(Tennyson's poetic imagination outran his grasp of railway technology which is based, of course, on raised tracks, not grooves!)

The twentieth century has seen these beliefs severely shaken. Congested cities, polluted air, contaminated rivers, military devastation of large rural areas — what has happened to the Victorian dream? The shock to those with responsibility for science and technology has been substantial. In the United Kingdom there are now those who argue first, that Britain's best chance is to identify an appropriate national life-style and live it to the exclusion of all else; and second, that the appropriate role for a people rich in history but poor in resources is to act as a cultural oasis for the world's tourists.

Extremists might maintain that in order to sustain this role we should be prepared even to relapse into a rustic economy and population size. I would prefer to argue that national revival depends on grasping rather than surrendering world leadership in one particular sector — the art of instructing computing systems how to do difficult things. Is the communication, computation and control network of the future going to occupy itself at the 1973 level of programming technique? Alternatively will today's laboratory systems of machine learning and perception be built into the public facilities available to the automation specialist?

Four computer scientists [2], from Stanford Research Institute and from Lockheed, recently examined likely industrial consequences of machine intelligence research, with results which make the second alternative look at least plausible. They used the *Delphi* technique of technological forecasting. A large and carefully designed battery of questions concerning future trends was put to an international panel of experts. The replies were analysed and summaries fed back to the experts, each of whom was asked for comments on, and revision of, his earlier estimates. After the third re-cycling, an additional questionnaire was employed to calibrate each individual's degree and direction of error in unrelated tasks of numerical estimation, and a self-rating scheme was employed to assess his professional expertise over the various sub-topics involved.

Table 13.1 summarizes some of the results. it will at once be noted that one of the earliest products to reach prototype stage is expected to be what the authors term 'industrial robot'. They are using the phrase sloppily, for industrial robots have been around for many years. What the authors had in mind is rather vaguely indicated by the following passage taken from their report:

> 'The addition of simple visual and tactile sensors would significantly broaden the application. For example, the General Motors Research Lab successfully demonstrated a system which could mount wheels on a hub, using visual techniques to align the wheel with the studs.'

If the authors' survey results can be criticized, it is on grounds of

Table 13.1 — Summary of Delphi results, reproduced from Firschein, Fischler, Coles, & Tenebaum (1973). This tabulation is given for its broad-brush indication only. For explanation of the various concepts the original paper should be consulted.

Products	Median prototype date	Median commercial date
High potential significance		
Automatic identification system	1976	1980
Automatic diagnostician	1977	1982
Industrial robot	1977	1980
Automated inquiry system	1978	1985
Personal biological model	1980	1985
Computer-controlled artificial organs	1980	1990
Robot tutor	1983	1988
Insightful economic model	1984	1990
Automated intelligence system	1985	1991
General factotum	2000	2010
Medium potential significance		
Voice response order-taker	1978	1983
Insightful weather analysis system	1980	1985
Talking typewriter	1985	1992
Mobile robot	1985	1995
Automatic language translator	1987	1995
Computer arbiter	1988	1995
Computer psychiatrist	1990	2000
Robot chauffeur	1992	2000
Creation and valuation sysem	1994	2003
Low potential significance		
Universal game player	1980	1985
Animal/machine symbiont	2000	2010

conservatism. The median prototype date given for 'industrial robot' as defined by them is 1977. But Hitachi have already announced in Tokyo a system which must be fairly close to qualifying. A computer–controlled hand–eye device inspects moulds for concrete poles as they pass on the belt, finds the bolts, confirms their location by tactile sensing and then tightens the bolts with an impact wrench. Other protuberances are avoided, and the task is performed with a consistency and efficiency which will make the device cost-effective, it is claimed, relative to human labour. Hitachi also

state that they expect the system 'to find a wide field of application in other industrial fields such as assembly and inspection processes'.

This emphasis on versatility is, of course, the central theme, and indeed the entire justification, of the machine intelligence approach. The thrust of such work is directed at the process of *re-instruction.* Most automatic assembly machines, for example cannot be re-instructed at all, or at best only within very narrow ranges of variation of the task. Consequently wherever there are short runs, large and costly upheavals of re-tooling and write-off occur. A year ago, re-instruction of an experimental programmable hand–eye machine was a matter of days for very simple tasks. By next year we and others believe that this will have shrunk to a few hours. The trick has to do with replacing as much as possible of the conventional step-by-step programming by a process of of instructing by *examples* and *outline sketches,* leaving the machine to fill in the rest. Thus teaching the Edinburgh system [3] to recognize objects seen through the TV camera involves no programming at all; the user merely shows a number of views of each object associating in each case the view with the object's name. This is preliminary to the task of picking physical components out of heap, visually identifying them and using them to assemble a specified object, say a toy car. It is performed by the 'hand–eye' robot under the control of a computer program in the following stages:

Instructional phase

(1) Individual parts are tossed onto the platform and the robot is told, for each possible view of the object, its designation, how to pick it up, and what to do with it (e.g. 'turn it over', or 'put it exactly here in preparation for assembly').

Approximately five of these training views are needed for each designation (e.g. 'car-body on side', 'car-body on back'); of course it only needs to be told once what to do with it.

(2) Starting with the parts laid out in the fixed position, the robot, working blind, is guided through the assembly operation. The instructions developed at this time to guide the robot constitute the assembly program; thenceforth running the assembly program transforms the laid-out parts into the final product.

Execution phase

(1) Someone dumps a pile of parts (perhaps with missing or extra parts) onto the platform and starts the inspection and layout process (Fig. 13.1).

(2) The robot examines the platform and lays out any recognized parts for assembly.

(3) Any unrecognizable pile of parts is pulled apart into its component parts by a set of routines which can deal with arbitrary heaps of arbitrary parts.

(4) If all the parts for the assembly are found, extra parts are dumped in a special location. If some parts are missing, the robot appeals for help.

(5) The assembly program is run (Figs 13.2–13.5). The above described

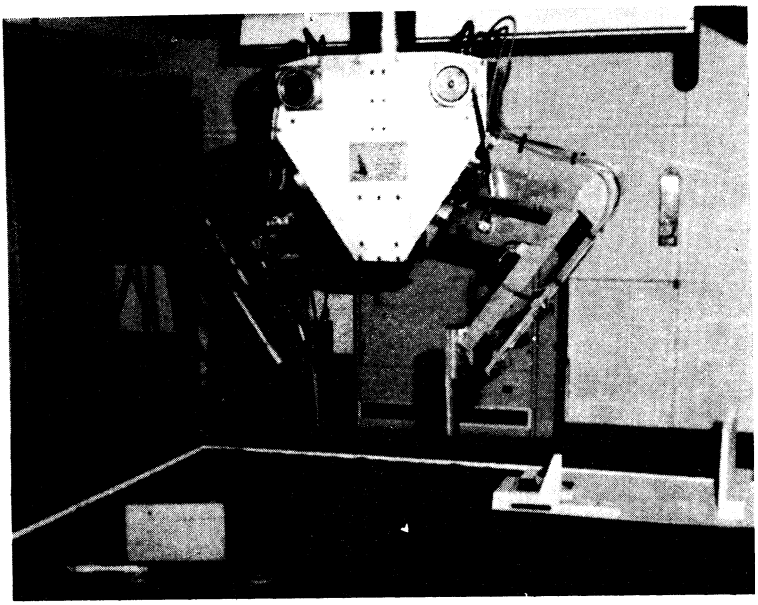


Fig. 13.1 — Viewing platform and computer-controlled mechanical arms. The workbench used during assembly and the components of the toy car are also shown.

performance is based on an elaborate suite of programs which confer a fair degree of versatility on the system — it can be taught a new assemply, say a model ship, at a day's notice. How far we still have to go in incorporating 'teachability' into software can be judged from the fact that a three-year-old child can be taught a new assembly at the same level of complexity in five minutes! The discovery of better design principles for 'teachable' programming systems is a major goal of most research laboratories in the machine intelligence field.

Will self-programming and 'teachable' systems be developed to a degree sufficient to bring quite new behaviours within reach of automation? Will a computer ever be able to make a real, as opposed to a toy, car, or (even more difficult) to drive one?

The *Delphi* report does not envisage the possibility of a robot chauffeur before 1992. On the other hand the introduction of computer-controlled robot assistants on the automobile assembly line, complete with visual sensing, tactile feedback, some higher-level planning of movements, and a limited capability to receive English-language instructions from a human supervisor is certainly not remote in time. It is of interest that at the beginning of 1973 General Motors in America were devoting no R & D at all to this topic. By the end of the year they had more than twenty research roboticists working full time.

It does not take much imagination to predict that a not inconsiderable shake-up and re-tooling of conventional assembly-line methods may ultimately follow. New concepts in manufacture, which at present seem absurd,

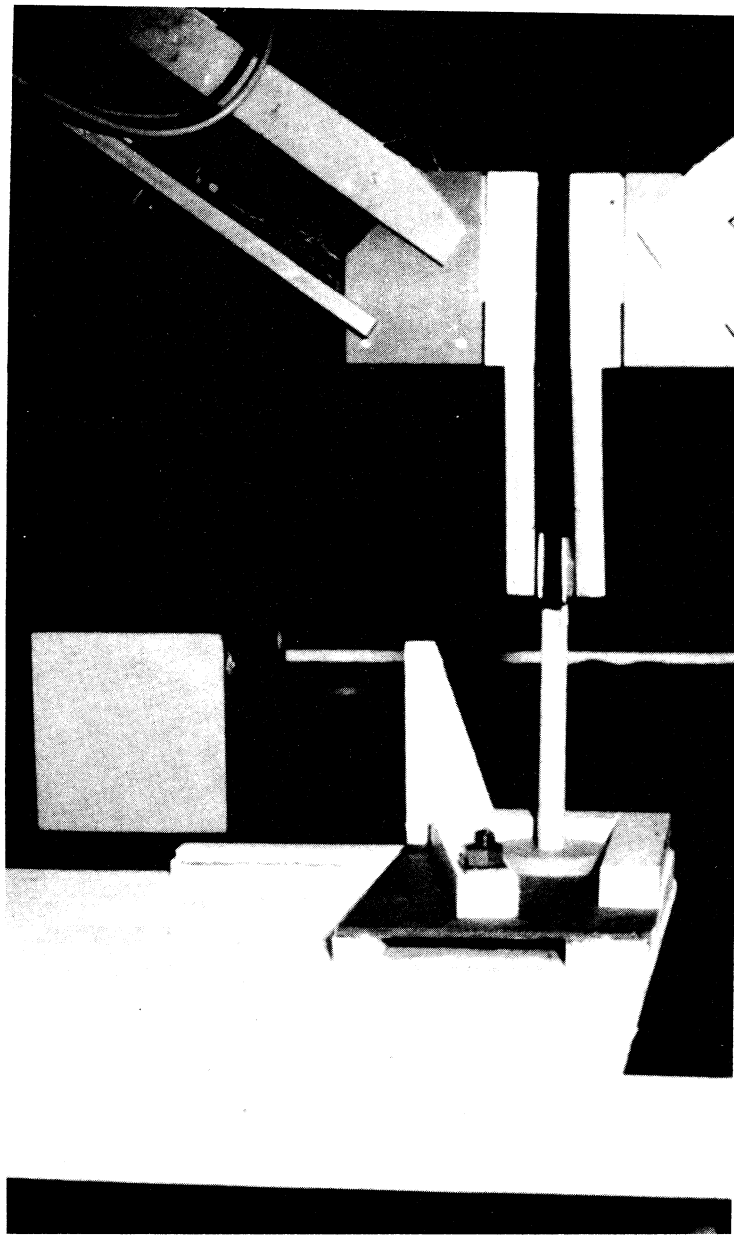


Fig. 13.2 — FREDDY uses a crude vice to clamp a wheel while fitting in the axle. The car body and another wheel can be seen in the background.

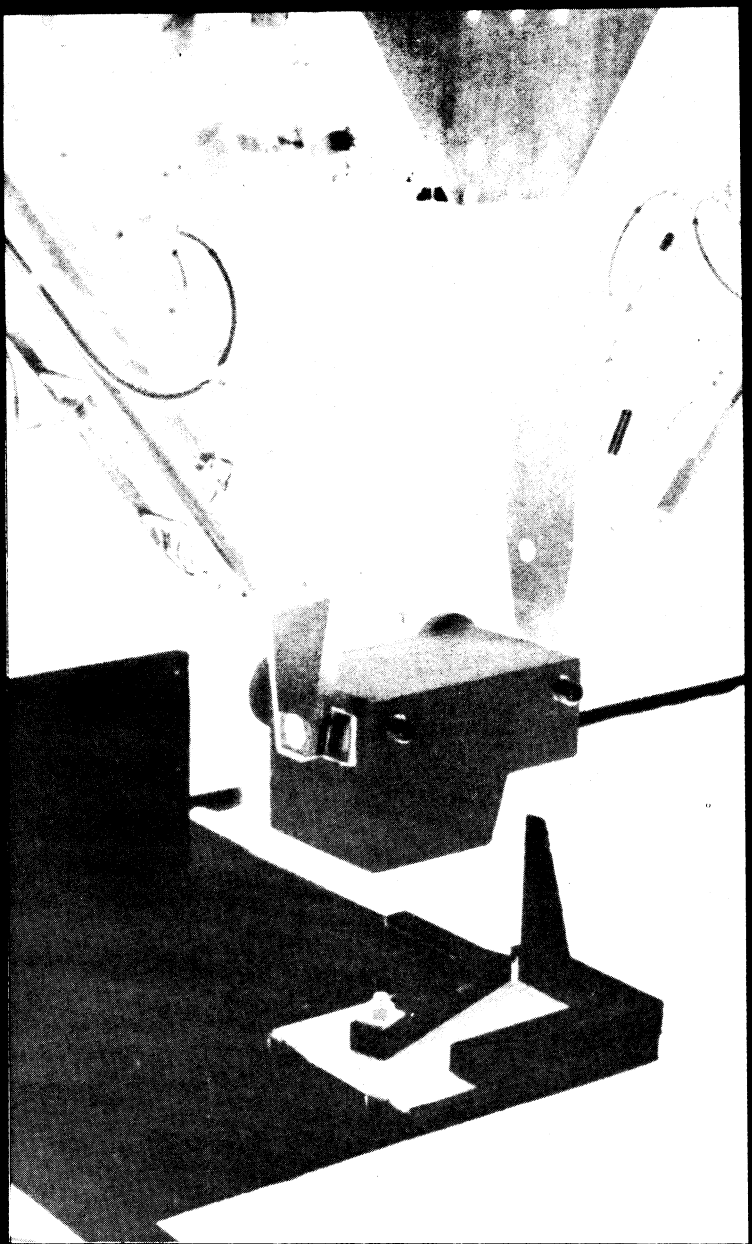


Fig. 13.3 —Two wheels with axles are now in place, with two wheels still to be fitted.

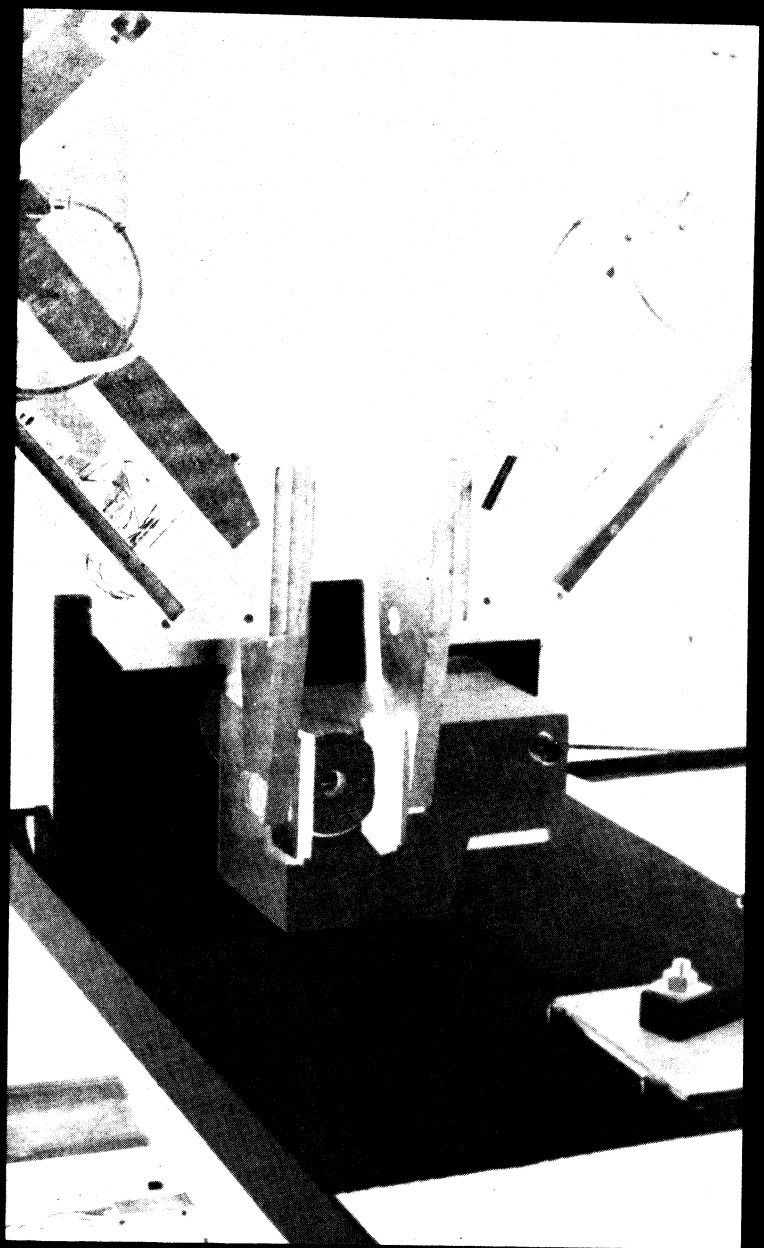


Fig. 13.4 — Having stabilised the incomplete assembly against a vertical surface, FREDDY adjusts the third wheel.

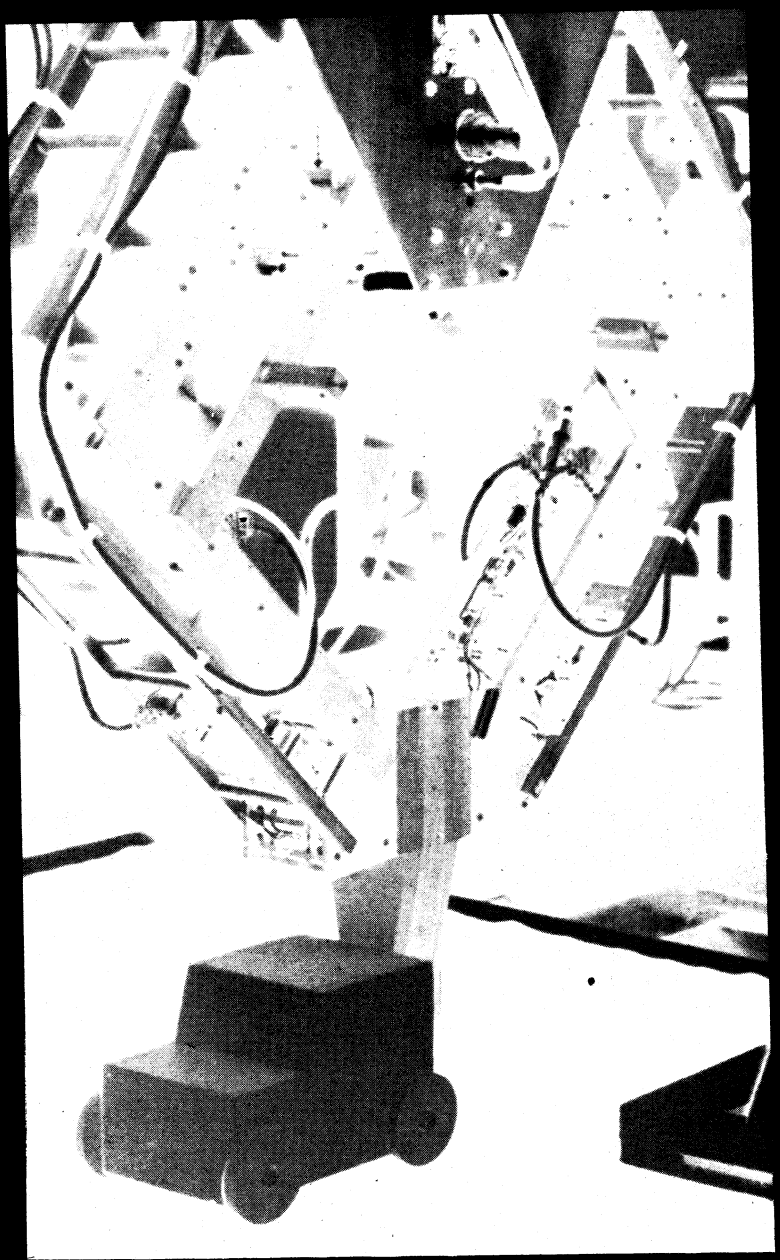


Fig. 13.5 — The assembly is complete.

may become commonplace. One man, controlling a team of computer-driven assembly machines, might be able to assemble whole cars as an act of individual craftsmanship — instead of assembling one-thousandth of a car every few seconds as at present. This image can be enlarged. We can envisage the automobile craftsman being freed of the necessity to travel each day to his robotic workshop. Just as the office worker in the era of the universal computer network, so the factory worker may be able to ply his trade at home *via* high-speed video links and the rest of the apparatus of tele-operator technology. Some expert observers (Professor John McCarthy of Stanford University is one of them [4]) foresee large-scale development along this line in the 1980s.

These speculations, and the overall indications of Table 13.1, may seem somewhat revolutionary, and it is of course fashionable to speak of the computer revolution. Yet if we take as our criterion the idea of sudden discontinuity then what we are witnessing is better described not as revolution, but rather as an extreme acceleration of an evolutionary trend as old as civilization.

Since the earliest times man has been storing and processing information in symbolic form, so as to predict and control his environment. Judging from the extent of Maecenean inventories revealed by deciphered fragments of Linear B, not to mention the vast bureaucratic book-keeping of the Roman Empire, ancient peoples handled information of considerable complexity with no mechanical aids beyond tally and stylus. Civilization is the growth, typically occurring in sudden bursts alternating with phases of consolidation, in the 'machinery' of information processing. On the one hand there is abstract machinery ranging from the Egyptian architect's *vade mecum* of geometrical calculations and the Arabian notation and rules for arithmetic to the whole imposing structure of modern applied mathematics. On the other hand there is physical machinery such as parchment and quill in place of stone and chisel and the development of printing presses, typewriters, cameras, slide rules and calculating machines, culminating in the high-speed digital computer. In this last species of machinery the two evolutionary processes, abstract and concrete, or as we must now say software and hardware, finally join.

But if something happens fast enough, does it matter whether it is described as evolution or revolution, as expansion or explosion? The present development of computer technology is faster by orders of magnitude than anything which has happened before. So if computing should be classified as evolution, let us remember that it is an evolutionary process undergoing very rapid acceleration, and that there is no corner of automation into which it will not penetrate. The pattern of penetration will of course be determined by industry, but academic centres have a part to play, particularly in the training of the new generation of engineers. There is a peculiar belief that the academic mind is of so sensitive a nature that its bloom can be corrupted by injudicious contact with industrial technolgy. Samuel Johnson [5] took a different view. To him the academic cloister was the really bad spot:

'Deign on the passing world to turn thine eyes,
And pause a while from letters to be wise;
There mark what ills the scholar's life assail,
Toil, envy, want, the patron, and the jail.'

Without going all the way with Johnson, the machine intelligence worker need not be averse from seeking a direct coupling between academic research and industrial technology. Indeed, the nature of his subject is such that this coupling will assert itself in the end whether he seeks it or not.

REFERENCES

[1] Tennyson, A. (1842) Locksley Hall. *The Poems of Tennyson* (1969, ed Ricks, C.), 688–99. London: Longmans Green.

[2] Firschein, O., Fischler, M. A., Coles, L. S. & Tenebaum, J. M. (1973) Forecasting and assessing the impact of artificial intelligence on society. *Third International Joint Conference on Artificial Intelligence,* 105–20. Stanford: Stanford Research Institute Publications Department.

[3] Ambler, A. P., Barrow, H. G., Brown, C. M., Burstall, R. M. & Popplestone, R. J. (1973) A versatile computer-controlled assembly system. *Third International Joint Conference on Artificial Intelligence,* 298–307. Stanford: Stanford Research Institute Publications Department.

[4] McCarthy, J. (1972) The home information terminal. *Man and Computer. Proc. int. Conf., Bordeaux 1970*, 48–57. Basel: Karger.

[5] Johnson, S. (1755) The Vanity of Human Wishes. *Dodsley: Collection of Poems, IV,* in *Johnson: prose and poetry* (1950, ed Wilson, M.), 161–70. London: Butler & Tanner.

14

Steps towards robot intelligence (1985)

In 1972, the Science Research Council received a report from a panel set up in the previous year to review long-range computing research. Among a number of constructive and timely proposals the panel urged increased 'use of the robot as an analytical tool'. Although endorsed by Council, this report was overtaken by events which need not concern us here. What will concern us is the meaning of those words.

INSTRUMENTATION FOR MACHINE INTELLIGENCE

The panel was evidently saying that, along with the robot as *technology*, there is a notion of the robot as instrumentation for *scientific enquiry*. But enquiry into what? The answer I shall give is: enquiry into the design principles of cognitive processes. One particular process will be singled out because of its topicality, namely machine learning.

The Royal Institution very often mounts displays and exhibits from outstanding academic and industrial laboratories, all devoted to the technological issue. In our country's precarious situation, this issue gains urgency with every year that passes. But by way of complementation I propose to address another issue, namely the rationale of *experimental* robotics as a branch of machine intelligence, with goals distinct from, although by no means in isolation from, industrial robotics. The aim of such work is to build and test operational theories of what is sometimes called the 'recognise-act cycle'. I draw strength from the knowledge that, in this field at least, today's science has a habit of becoming tomorrow's technology. In particular, robot technology is beginning to knock on the door of machine learning studies. This is because the automation engineer requires of the machine, more than anything else, that it be a *teachable* machine.

NECESSITY FOR SENSORS

To exhibit the undeveloped state of teachability in present-day industrial robots, consider the steps in teaching a Control 6R Research robot the trick of describing a figure in the air.

The method is to use the typical facility of step-by-step key-pad programming, followed by imprinting the trajectory in the machine memory. The path can then be repeated indefinitely, moving (let us say) a paint-spray over its prescribed course for each of a continuous sequence of replicate objects. All is well until the unexpected happens. Perhaps one of the objects is wrongly positioned, or missing, or the line stops. Regrettably, industrial robots of this first generation are blind. They are also deaf, dumb, and devoid of tactile sense. A level of teachability above that of mere rote learning would be desirable. Not only this, sensors are needed — preferably smart sensors. Without them the recognition part of the recognize–act cycle cannot operate. Whether viewed from the stand-point of industrial machinery or scientific equipment, it is in the interests of all concerned for both sensors and effectors to be reasonably cheap.

For those who wish to experiment for themselves, the 6R robot described can be purchased for about £2400. For the home hobbyist, or the hard-up machine intelligence laboratory, the little 6E which can be instructed by voice command can be got for about £800 and driven from a personal micro such as the Apple II. A few hundred pounds more can cure deafness. With a microphone and associated speech-recognition package, voice instruction presents no difficulty. For the ambitious, a further few hundred pounds will secure a TV camera complete with picture-input programs for capturing and pre-processing frames of 256×256 resolution.

MACHINE LEARNING OF RECOGNITION RULES

Many will already know of the pioneering work on image-processing by Dr Michael Duff and his colleagues at University College London. W. K. Taylor's robot vision group is likewise part of this same initiative. At Edinburgh we have been combining the parallel array principle of Duff's CLIP-4 machine with machine learning of recognition rules from examples. To give a quick flavour of what it means to teach a machine strategies rather than trajectories, consider the teaching of an Apple II a decision strategy for the circumstances under which the robot should open its umbrella.† This toy example can help clear up a point of terminology which otherwise will give trouble later. The decision rule which is synthesised can be thought of as a *theory* of some kind, in this case a theory of umbrella management. But it is also a *program*. The Apple can execute it, or output it for another machine or a human to execute. When executed it is like a program. When inspected, analysed, or evaluated it is like a theory. In other words, theory and program are words descriptive of one and the same object, the nuance being derived

† The computer displays on its screen hypothesised strategy rules to account for the example decisions which it has so far been given (Fig. 14.1).

from the purpose in mind. The point will assume importance when I later discuss machine synthesised theories which, unlike our umbrella-management theory, can take forms which cannot be inspected, analysed or evaluated by people.

Weather	Inside	Soaked	Decision Class
wet	no	yes	DON'T-USE
dry	—	no	DON'T-USE
—	yes	no	DON'T-USE
wet	no	no	USE

Decision Tree

```
Weather
    dry: DON'T USE
    blustery: DON'T USE
    wet: Inside
        true: DON'T USE
        false: Soaked
            true: DON'T USE
            false: USE
```

Fig. 14.1 — The upper part shows the first four example-decisions of a training set input to the machine. The symbol '—' means not specified. The lower part shows a machine-generated strategy, which is, in effect a program. Programming by example is a process of inductive learning whereby examples are used to refine incrementally a partial solution to a problem. The particular algorithm described in this Discourse is called ID3 (Iterative Dichotomiser Three) QUINLAN 79 based on Hunt's Concept Learning System HUNT, MARIN & STONE 66. There are two phases: 1. A teaching phase, consisting of supplying examples in order; 2. An execution phase, using the decision tree as a program that informs of the correct action for any situation covered by the new rule.

ARTIFICIAL INTELLIGENCE

Returning to models of cognitive processes, the five questions listed in Fig. 14.2 exemplify foci of intense scientific activity. Any substantial artificial intelligence laboratory today may be expected to have ready-to-use exemplars of most of these categories of experimental program, categories absent from the commercial computing world outside. The top half of Fig. 14.2 concerns a capability now being transferred, under the name 'expert systems', from research laboratories into industrial organisations. To an important extent this work has been found to require the use of inductive learning as a means of speeding the acquisition of expert capability by the machine partner. The lower two items of Fig. 14.2 are concerned with the use of mechanised *deduction*, in some sense complementary to machine

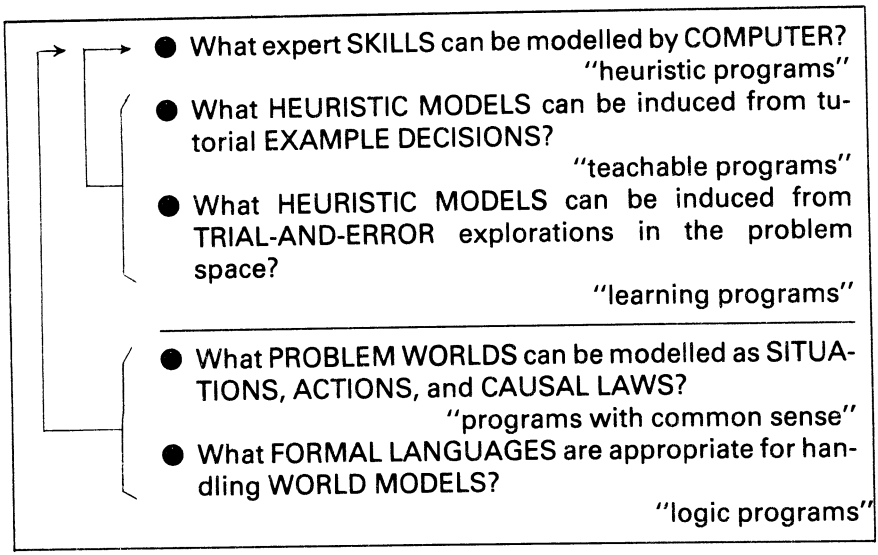


Fig. 14.2 — A classification of the kinds of questions posed by the study of robot intelligence. The upper half is concerned with the acquisition of logical models of decision-taking from data. The lower part is concerned with the software technology for expressing models directly.

learning. Before leaving it, we should realise that mechanised deduction is fundamental to the development of robots able not only to *learn* strategies but also to reason out strategies for themselves. Instead of imperative commands, the computer is told relevant facts about its world together with rules of inference. The requisite new discipline of logic programming has arisen from Robert Kowalski's pioneering work at Edinburgh and was subsequently developed by him and his colleagues and students at Imperial College. Its embodiment in the programming language PROLOG has now been adopted by Japanese authorities as a central plank of their recently revealed national computing plan which has already been started.

ARE PATTERNS NECESSARY?

In the present context, inductive learning is concerned with the acquisition of pattern-based strategies from examples. Perhaps patterns, then, are necessary design ingredients for an effective decision engine?

In the philosophy of the mathematician or the theoretical physicist, the time-cost of performing the calculations necessary to interpret and verify formal descriptions tends to be ignored. In such a philosophy the answer to the above question is given as 'no'. In a classic paper published in 1950 Claude Shannon dramatised the point in regard to the design of a decision engine for two-person zero-sum games, in the following terms:

> The unlimited intellects assumed in the theory of games... never make a mistake and the smallest winning advantage is as good as mate in one. A game between two such mental giants, Mr A and Mr B, would proceed as follows. They sit down at the chessboard, draw for colours, and then survey the pieces for a moment. Then either
> (1) Mr A says 'I resign' or
> (2) Mr B says 'I resign' or
> (3) Mr A says 'I offer a draw' and Mr B says 'I accept'.

According to Shannon's estimate the lookahead rule of calculation followed by the two mental giants would occupy a machine faster than any of today's supercomputers for more than 10^{90} years: sufficient for the physicist or pure mathematician; not sufficient for the engineer.

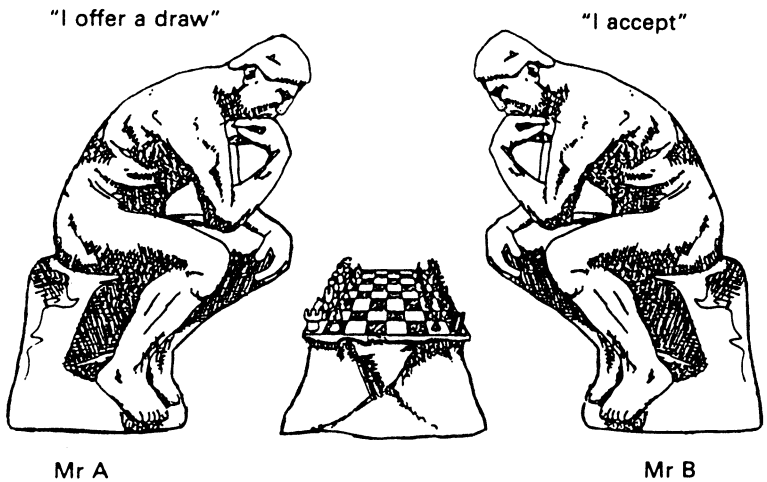


Fig. 14.3 — Unlimited intellects assumed in the theory of games, a branch of mathematics, apply to a class of games exemplified by chess which has the following properties: two-person; perfect information; no chance moves; zero-sum; finite. There are approximately 10^{45} positions and 10^{125} games of chess.

IS CALCULATION NECESSARY?

Is *calculation* then necessary, even if not sufficient, for an effective decision engine? If this time we disregard memory-cost, then the answer is 'no'. Given enough patterns, calculation can virtually be dispensed with, as it must be under acute time-constraints such as bicycling or piano playing, and also of course when a Grandmaster plays lightning chess. Work at Carnegie-Mellon University and at the University of Illinois has shown that the number of patterns stored in a chess-master's long-term memory lies

between 10 000 and 100 000. This is well within the bounds of today's rapid-retrieval storage media. So in contrast to the time-cost for Shannon's mental giants, we can legitimately disregarded memory-cost even for quite complex skills. Cheap computer memories have accordingly permitted the rise of the new programming craft of 'expert systems' or 'pattern directed inference systems' as they are sometimes called. This craft has roots in insights gained from early artificial intelligence work of the 1960s, of which three particularly relevant examples will be given, namely from rule-based adaptive control, chess concept learning by machine, and perception and cooperation in robots.

In all three cases a relatively small number of patterns was sufficient to defuse otherwise intractable combinational explosions. The pole-balancing system, based on 225 patterns, will be discussed. Here, a human subject is required to learn how to move a cart so as to keep a pole balanced. The following points are of importance:

(1) The role of patterns.
(2) Trial-and-error learning by machine.
(3) Sufficiency for the human of a heuristic model. Hiding the underlying physical systems from the user is no impediment to his learning performance.

In parallel with this study, the late Sir Eric Eastwood tackled exactly the same problem for his Faraday lectures using modern adaptive control methods. When we subsequently discovered each other's existence we met, exchanged film showings, and analysed the respective trade-offs involved in the two approaches. The rule-based program BOXES scored in run-time computation cost, resilience of learning to arbitrary changes in physical parameters of the task, and in the existence of a man–machine mode. But it was more costly in the operation of its learning component. I learned more from these discussions than from the project itself, and I value this opportunity to pay tribute to Sir Eric's memory.

PRODUCTION–RULE PROGRAMMING

The control structure underlying the experiment in Fig. 14.4 is extremely simple. It is today commonly called 'production rule' programming. The fundamental idea is to separate the program into, on the one hand, a set of facts descriptive of a modifiable situation and, on the other hand, a set of rules for making modifications. This is illustrated in Fig. 14.5. In the case of the pole and cart the 'database' contains simply the current status of each of the four variables: position, velocity, angle, rate of change of angle. The rules are *if–then* constructions of which the left-hand parts, or antecedents, are conjuncts defined over these variables and the right-hand parts, or consequents, are actions (either 'rightwards' or 'leftwards' for each case), as below:

if cart is far left

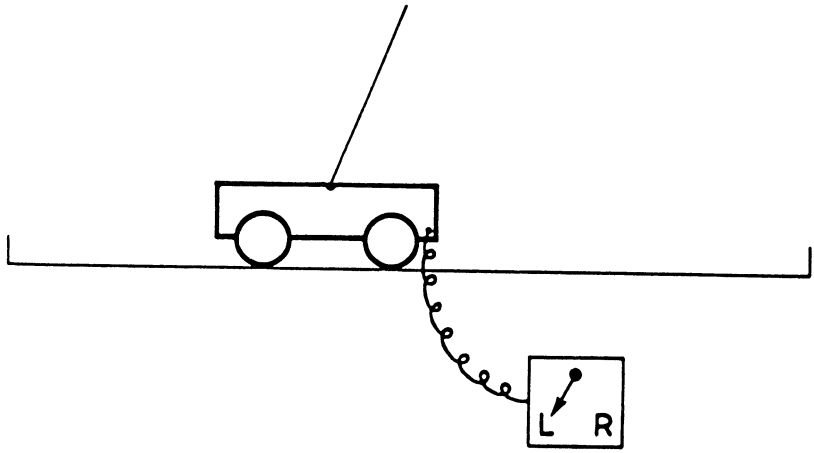


Fig. 14.4 — The pole and cart problem. Chambers and Michie's BOXES program generated a rule-based controller. The controller's task was to keep the pole balanced and the cart on the track.

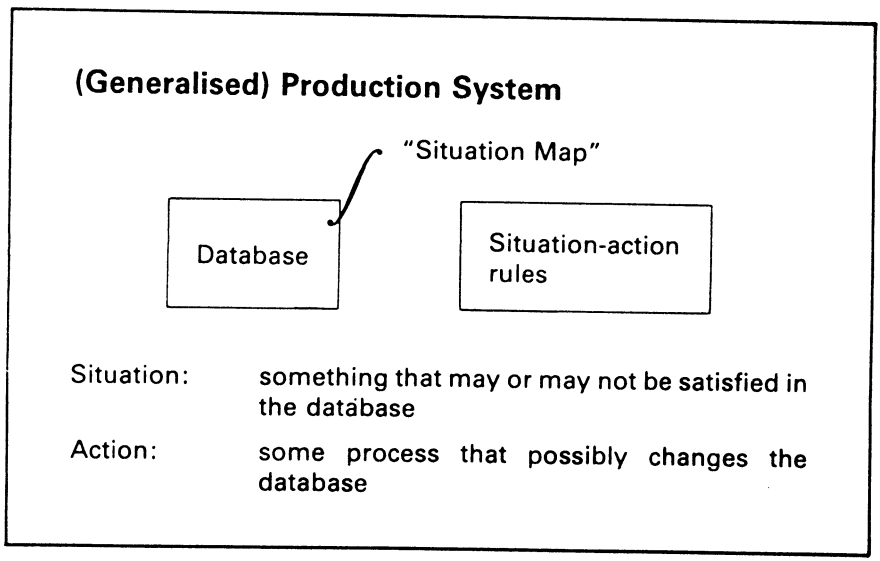


Fig. 14.5 — Production-rule programming: a break with the past. The driver of a rule-based system is the situation map, not the sequence.

and cart is hardly moving
and pole is hardly leaning
and pole is swinging to right
then do
rightwards

In each cycle (20 per second in our case) one of the 225 rules is selected by pattern match and the corresponding action is executed.

ACQUISITION AND MODIFICATION OF PATTERNS

During machine learning, the BOXES rules were continually revised in their right-hand parts, *i.e.* by re-setting the given action in the light of locally accumulated experience — local, that is, to the region of state space specified by the pattern part of the given rule. Modification of left-hand parts, the antecedent patterns themselves, is also possible, and was the basis of the teachable performance of the Edinburgh robot FREDDY. The skill studied with this system was that of identifying by sight and touch the parts of simple kits and manually assembling them. Some of the methods have become classical, in particular the manipulation of internal descriptions in a form nowadays called 'semantic nets'; also the development by Rod Burstall and Harry Barrow of fast algorithms for finding partial matches between one such net and another. But FREDDY's primitive algorithms for the induction itself have been superseded, for example, by decision-tree approaches which we use in Edinburgh today. These stem from Ross Quinlan's ID3 algorithm, in turn developed from Hunt's CLS (Concept Learning System) of the 1960s. Its task can be described as follows:

Given: a collection of positive and negative instances, where each instance is the description of an object in terms of a fixed set of *attributes* or properties

Produce: a decision tree for differentiating positive and negative instances.

It would be interesting now to go back to the pole and cart and compare on this hard task our present learning programs with the crude statistical induction methods we employed then. We have now brought these programs to what we regard as a fairly high pitch, providing options for sampling human as well as machine data-sources, and incorporating facilities for handling attributes which present themselves as real-valued measurements. In fairly complex recognition problems taken from chess they can be made to give performance consistently higher than human. Moreover we confirm Quinlan's finding that, at relatively high levels of complexity, simplistic use of these programs generates decision rules for which high efficiency of machine execution is matched by their total opacity to human comprehension. Recently, however, Shapiro and Niblett at Edinburgh have succeeded in developing a man–machine style which we call 'structured induction' and which results in humanly *transparent* decision rules. The price paid to achieve this is prior expert scrutiny of the problem with the aim

of partitioning it into a few major sub-problems, before the induction program runs are commenced.

Fig. 14.6 shows the top-level rule induced in this structured style for classifying the space of legal king-pawn-king positions into 'won for white' and 'drawn'. This little semi-automated construction and its dependent sub-theories has proved to be of interest to chess-masters as a replacement for the fragmentary and flawed codifications found in the chess books. Not very far up on the complexity scale lie levels beyond human capacity to acquire in entirety, let alone to codify. Shapiro has induced a complete, human-readable, machine-executable theory for a subset of the ending king and rook against king and pawn for which complete human mastery is at best marginal. Rather than illustrate what we mean by 'concept learning' from such esoteric material I have elected to display an engaging tutorial example from my Illinois colleague, Ryszard Michalski, shown in Fig. 14.7.

The same induction program which Michalski used for trains has also successfully constructed diagnostic rules for diseases of the soy-bean, a staple crop in the State of Illinois. Not only do these, when run on the machine, out-perform professional pathologists: they also constitute an improved classification scheme for human use. Michalski's results thus exactly parallel our ID3-based results in chess and those of our colleague Ivan Bratko in the classification of lymphatic cancers. Applicability in robotics seems inviting: in a class project at Illinois my students were able to teach the robot to discriminate good, bad and unacceptable table settings, using white plastic tableware set on dark cloth, just from showing examples of these three categories.

THIRD-PARTY INTERFACE

I want to turn from this homely task to a curious problem. The task of the shopfloor robot supervisor is already a responsible, and sometimes harassing one. I do not know anyone connected with this trade who is not aware that in course of time facilities for setting two robots to work together on given tasks will be part of the industrial scene. Will the robots communicate with each other by electronic signals which the human supervisor cannot hear and could not interpret if he heard them? If so it takes little imagination to see how rapidly the complexity of his task will escalate, assuming that we are now firmly in the era of second-generation machines with smart sensors and improved learning capabilities. The easiest way of providing what may be called a 'third-party interface' is to insist on a literally *audible* signalling convention. One could cite a nursery example to illustrate the idea. When a large robot requires a blue box it emits a high-pitched squeak and for a red box a lower-pitched one. Hearing this, the small robot pushes a box of appropriate colour from the dispenser which lands within its coworker's reach. It could be called a 'recognize-squeak' cycle.

Perhaps the industrial safety rules of the future will lay down that automation devices must talk to each other in this rather literal sense! But as

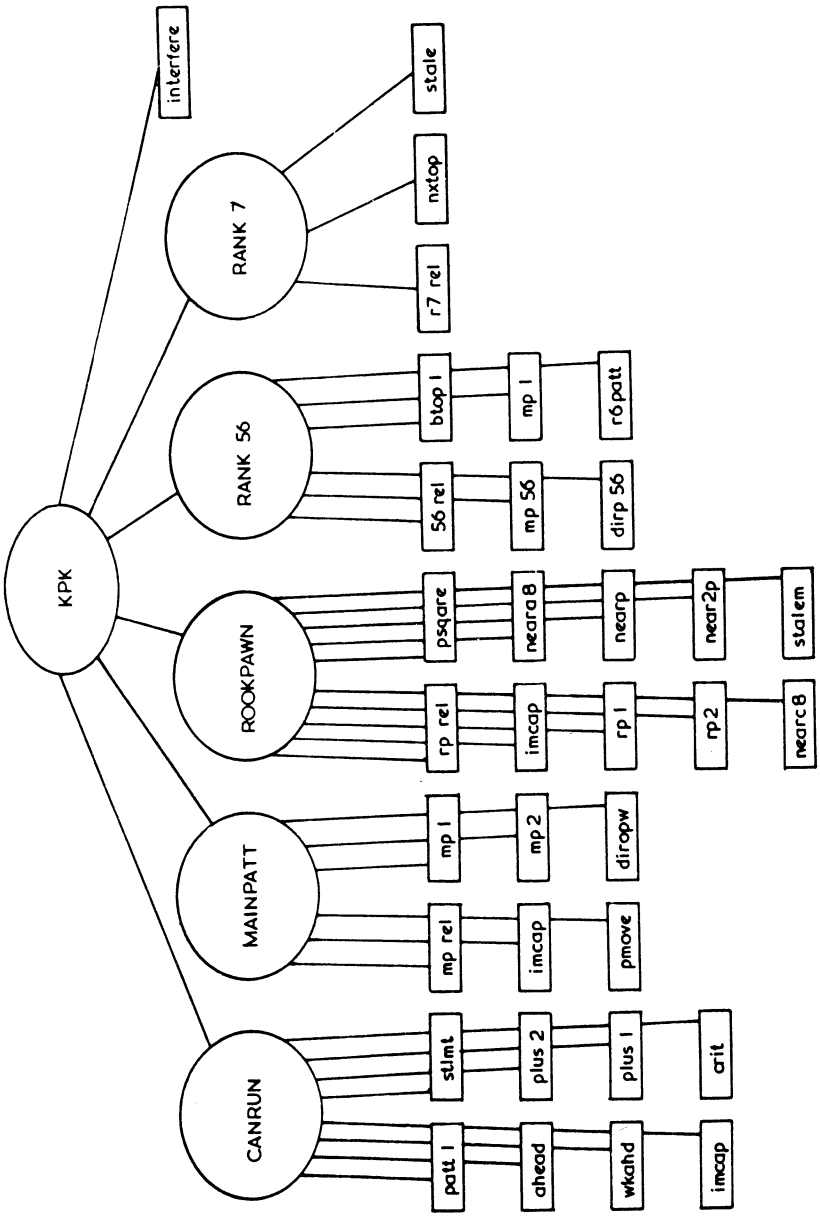


Fig. 14.6 — Two-level decomposition by Shapiro and Niblett of the King-pawn-King problem into six sub-problems, of which one ('interfere') is definable as a primitive pattern, while each of the other five is decomposed further. Each of the five compound concepts at level 1 was induced from files of examples and counter-examples. Finally the top-level concept was induced in the same way in terms of the six previously constructed sub-concepts (five pre-induced, one hand-coded).

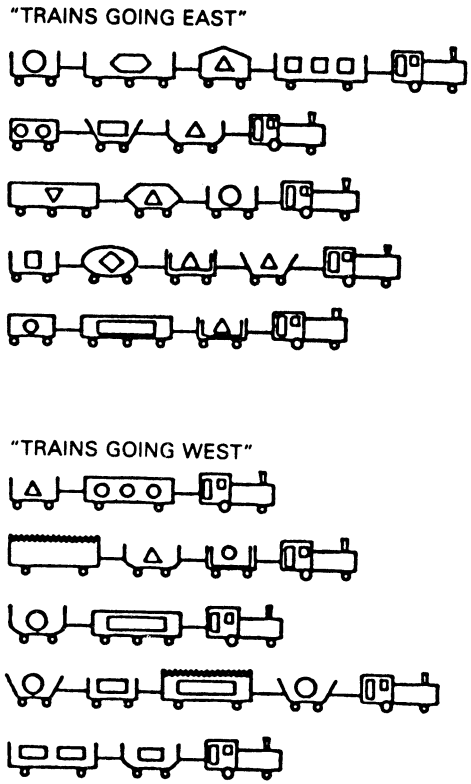


Fig. 14.7 — Larson–Michalski trains with rules of human and machine origin. The following were the three simplest rules found by Larson and Michalski's inductive learning program: (i) if there is a triangle-containing car immediately followed by a polygon-containing car then Eastbound else Westbound (polygons of course include triangles); (ii) if there is a short closed car then Eastbound else Westbound; and (iii) if there are two cars, or if there is a jagged-top one then Westbound else Eastbound. Of 70 human subjects many produced (iii) and some (ii). In addition the following two were produced: if there are more than two kinds of freight, then Eastbound else Westbound. Number of sides in the cargo (circle counts 1, triangle 3, etc) is a factor of 60 if and only if the train is going West!

the content of messages becomes more complex, how are human and machine partners to remain in adequate rapport? A much deeper study will be needed than is yet being conducted of the nature and pre-conditions of what is coming to be called 'cognitive compatibility'.

TOWARDS A TECHNOLOGICAL BLACK HOLE?

In our work and in Quinlan's, preliminary sightings have been made of a potentially disturbing phenomenon already mentioned in the harmless context of laboratory tests with chess. I refer to the possibly inscrutable nature of products of computer induction when these products are control

programs for domains of high complexity. Inscrutability is of no consequence in some task environments. But it is of more than passing concern in situations such as the control of nuclear power plants, air traffic control or

Lost 3-ply Experiments
(CDC Cyber 72 at Sydney University)
49 attributes (Mixture of 35 pattern based
4 low-level predicate
10 high-level predicate)
715 distinct instances
177 — node decision tree found in 34 seconds

Classification method	CPU time (msec)
Minimax search	285.0
Specialised search	17.5
Using decision tree	3.4

Fig. 14.8 — Different ways of constructing a program to recognise the property 'lost in three-ply' in King-Knight-King-Rook chess positions: 1. hand-coding the minimax general search algorithm; 2. hand-construction of an expert system based on pattern-directed search; 3. inductive synthesis of a recogniser from a training set of 715 cases selected by the induction program from the full set of 2½ million cases. Run-time costs are compared.

the operation of military warning systems. The programs which control such installations are today of exclusively human authorship. Yet anxiety is already expressed concerning their lack of transparency to technical personnel at times of suspected malfunction. At some time the developers of these systems will succumb to the lure of inductive and other synthesis techniques at present under development in various laboratories. The evidence is compelling that unless special methods are employed the products of induction can be expected to be opaque to those who will depend upon them at run time.

The economic inducement to prefer such products may sometimes be startling, as suggested by Quinlan's results in the accompanying table. Here a five-fold gain in run-time efficiency was obtainable by use of machine-made rules in preference to hand-made ones. Moreover hand synthesis took man-months in contrast to a few minutes for machine-synthesis. The eventual spread of such methods through the programming industry seems a certainty.

It is important to understand the nature of the technological black hole towards which we are headed. Let me spell it out:

(1) Machines can now be used to generate super-efficient descriptions as programs.
(2) These descriptions, although accurate and efficient when run on machines, make no sense to people.
(3) Even with pre-learning technology, an inscrutable machine can be a dangerous machine.
(4) It seems inescapable that during the coming decade the programming industry will be transformed into a factory operation based partly on automatic synthesis.

NEED FOR A 'BRIDGE' SCIENCE

Apart from the EEC, which has funded a preliminary study of this issue, the centre of awareness appears to be Tokyo. In the 85-page report recently released by the Japan Information Processing Development Centre the diagram reproduced in Fig. 14.9 appears. This picture seems in itself to provide an answer to the question: why has Japan decided to build her 8-year computing plan around artificial intelligence? Stripped of its aura of mysticism, the picture says that without a major AI component (represented in the diagram by the central circle) the computing systems of the 1990s will pass forever beyond human mental ken. The guarantee of cognitive compatibility between man the sorcerer and the new apprentices is to be sought by building the 'bridge' science of artificial intelligence. The task is to design systems which can manipulate on the one hand models of the mentality, requirements and limitations of the human user and on the other hand models of brute force problem solvers of the fourth generation and beyond. There can be little doubt that the same issue will raise its head in factory automation. Hence it would be timely to revive the study of machine intelligence in robotics.

REFERENCES

J. V. Dunworth (Chmn.), *Computing Science Review: policy and programme review of research and training in computing science*, Science Research Council, London, June, 1972.

A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone, *A versatile system for computer-controlled assembly, Artificial Intelligence*, 1975, **6**, 129.

D. Kopec and D. Michie, *Mismatch between machine representations and human concepts; dangers and remedies, FAST series No. 9 Report Eur 8426 EN*, European Community, Brussels, 1983.

D. Michie, *Expert systems and robotics, Handbook of Industrial Robotics*, (S. Noff Ed.), John Wiley, New York, 1985.

T. Moto-oka (Chmn), *Preliminary Report on Study and Research on Fifth-Generation Computers 1979–1980*, Japan Information Processing Development Centre, Tokyo, Autumn, 1981.

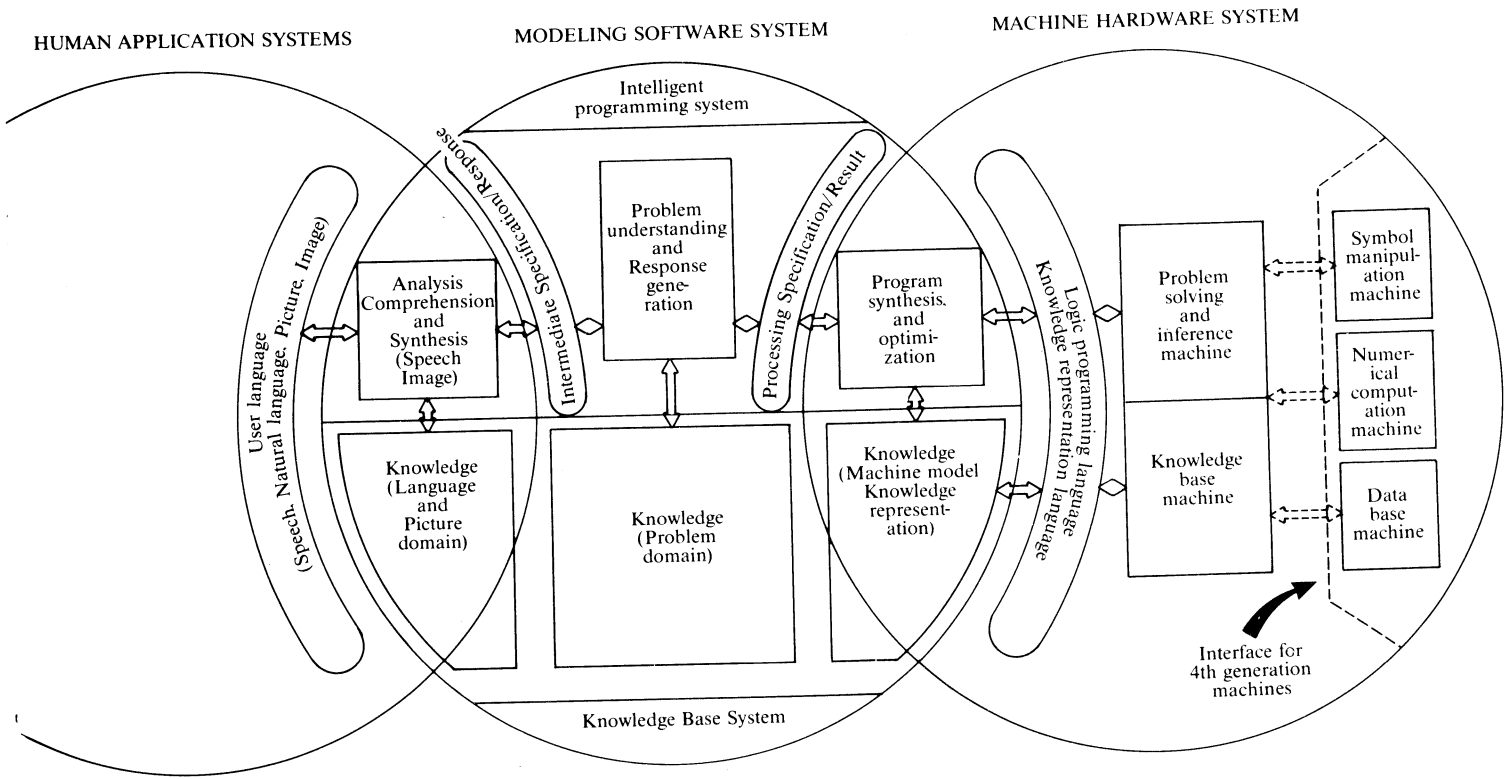


Fig. 14.9 — Conceptual diagram of the fifth-generation computer system, seen by the Japanese Government as a needed bridge between human culture and cognition and the brute-force technology of modern computing.

NOTE ADDED IN PROOF

Following on from the closing sentence of this text, which was delivered as a Royal Institution Friday Evening Discourse on 26th March 1982, a resumption of the FREDDY series of experiments in robot intelligence has since been made possible by the generosity of the Westinghouse Corporation, USA. At the time of writing FREDDY 3 can be taught to build simple blocks-world constructions. Planning and learning capabilities are co-ordinated *via* a PROLOG-oriented software base and include complex assembly-line operations involving two or more co-operating robots.

Section 3 Mechanics of cognition

INTRODUCTORY NOTE TO SECTION 3

Biologists and engineers have always shared a dream that they can solve each other's problems. Under its spell in 1967 the distinguished experimental psychologist Richard Gregory moved to Edinburgh and set up his Bionics Research Laboratory. A re-orientation was thereby imparted to our experimental programming work which was, I believe, far-reaching. The closing passage of the Edinburgh-Tokyo dialogue (Chapter 8) makes the point specifically. Japan's current 'Fifth generation' programme, diagrammatically sketched in Figure 14.9 has recently endorsed a similar re-orientation.

When Gregory himself left the project in 1970 he said to me a little gloomily that he no longer believed in bionics. As a subject, it had disappointed expectations of serving as a two-way street. The contributions of technology to biology, both as instrumentation and as intellectual models, flood in daily. They are pervasive. But what animal devices have ever been adopted by engineers with major industrial success?

I believe that Gregory's over-reaction arose from looking for the contributions of biology in the wrong place. The payoff to the engineer, when it does accrue, lies in copying not the mechanisms of animals but the methods and models of those who study them, not least those of Gregory himself. Not that the humbler forms of life do not make quite direct contributions, as in industries based on fermentation, and more recently in bio-synthesis. But to the general scientific technologist with a problem, an animal is just another, complicating, problem. A biologist on the other hand may through his training have tricks quite new to the traditional engineer — the use of scientific method, for example, to subdue complexity; or the habit of allowing for human factors in the design of instrumentation. More significantly, the biologist may have a factual or philosophical angle on the engineer's problem from which it can be shown that radical departure is necessary from this or that traditional design dogma.

Chapter 15 considers such a case in the interpretation of scenes via

computer-controlled sensors. A need is there discussed to combine map-like representations and symbolic representations, as suggested by phenomena of human visual thinking. Chapter 16 makes a more far-reaching foray with the proposal that the development of information technology has been consistently and almost fatally stunted by a philosophical strait-jacket from the physical scientists. A Spencerian 'bottom up' basis, customary in biological thought, must be substituted. Otherwise progress with machine perception will continue to be slow.

A particularly disabling consequence of ignoring the biological design principles of man's problem-solving brain is addressed in Chapter 17. Essentially a distinction is necessary between the phenomena of 'understanding' and of 'skill'. The former carries high execution costs; this is why the expert brain invokes the latter in its place whenever possible. Chapter 18 sketches a formal framework within which, it is hoped, information technologists and cognitive scientists may be persuaded to live together. In Chapter 19 the section closes by reviewing some indications that cohabitation is overdue.

15

Memory mechanisms and machine learning (1974)†

The authors of this paper are almost wholly ignorant of natural, as opposed to artificial, nervous systems. In machine intelligence our business is to design and experiment with home-made nervous systems: experimental computer programs. Unlike the systems studied by neurobiologists, our artificial ones are transparent. We know and document everything which is inside them, and so in principle we can explain their total behaviour. Perhaps, then, they might be a source of models for the real nervous systems which, however simple, remain distressingly opaque.

Building computer models of biological systems has become the vogue. They come in two kinds, *free-floating* and *anchored.* To make free-floating models, proceed as follows:

(1) Invent an arbitrary theory of how some observed system might work, complete with reverberating circuits, feed-back loops, encoders and decoders and anything else you like.
(2) Program it.
(3) Play with it on the machine, adjusting parameters and patching, until satisfied.
(4) Write up and go to (1).

If the theories which are implemented in this way have sufficient aesthetic appeal then no other justification is required.

Now consider the other kind of model — 'anchored'. Two anchorages are possible:

Type-1 anchorage is a knowledge of the given nervous system's micro-structure and functional connections so detailed as to lead more or less directly to the conjectured model. It may be doubted whether so firm an

† Written with Pat Ambler and Robert Ross, of the Department of Machine Intelligence, University of Edinburgh.

anchorage is possible in the present state of neurobiology, or is likely to be for some time.

Nevertheless, we believe that programmed models can play a useful role in the development of exact theories. In particular, they impose on any candidate theory an important constraint, namely, that it be expressed precisely. Implicit assumptions which might otherwise be concealed are consequently made explicit and therefore susceptible to experimental investigation. To be really useful, however, a programmed model must be capable of making verifiable predictions about the behaviour of the system being modelled. Given a complex theory such predictions might only be calculable if the theory is expressed as a computer program.

An interesting exercise in computer simulation is described in the doctoral thesis of Clymer [1]. This work is based on the 'mnemon' construct suggested by Young [2] as the principal mechanism for discrimination learning in the octopus. Clymer's program can produce, with a reasonable degree of accuracy, learning curves similar to those obtained with real animals. In addition, the program can simulate the effects of operations which interfere with the upper lobe structures. Of greater interest is the fact that the simulation has been used to make predictions about the effect on performance of variations in the training regime. These predictions can be tested experimentally.

Type-2 anchorage is the attempt at efficient engineering implementation of the given behaviour without regard to biology at all. On the gross plane of mechanical engineering, the development of the pump ultimately made it possible to understand the working of the heart. In computer technology, the development of computer vision may, or may not, ultimately help us to understand biological visual systems.

To recapitulate, the three categories are: (1) free-floating; (2) biologically anchored; (3) technologically anchored. Whether or not the time is ripe for successful application of computer models in any of these three categories is something on which only professionally qualified neurobiologists are competent to pronounce. However, we find it easier to be dismissive about (1) than about (2), and about (2) than about (3). Interestingly, the same order of dismissibility has obtained historically in the study of bird flight, where we can consider models of types (1), (2), and (3) as follows:

(1) Balloons: beautiful, but useless as models of bird flight.
(2) Artificial birds: too messy and difficult to be of practical help.
(3) Aeroplanes: highly influential. The interpretation of the fossil record of extinct birds achieved by John Maynard Smith [3] was a direct consequence of his employment during the war as an aircraft designer.

In the end something fit to be called machine intelligence theory will be developed, pushed ahead by the pressures and needs of technologically minded experimentalists. It may then at last become possible for neurobiologists with a foot in this new camp to 'do a John Maynard Smith' with simple nervous systems.

In this paper we exhibit a modest five-finger exercise under type (3). We describe, in particular, a computer program which behaves on a task of learning and discrimination in a way reminiscent of an animal. We explain what is going on inside it to cause this behaviour, and in what ways it could be extended to cause more complex and more 'intelligent' behaviour. The thing to bear in mind is that both the robot device and the computer program driving it were developed in the course of research done with a strictly non-biological motivation. About six hours' programming was subsequently done in order to 'dress it up' in a biological-looking idiom. Thus we are dealing with a genuine, if elementary, type (3) model.

MATERIALS AND METHODS

The experiment was based on a hierarchy of computer programs developed over a number of years to form an integrated software system for tasks of 'hand–eye' assembly. The hardware and software systems have been described elsewhere [4, 5]. A typical task might be the construction by the computer-controlled robot of a toy car from a set of parts, as shown in Fig. 13.1. The programming aids have attained a rather high level of flexibility and ease of use: the user can now instruct the system in a completely new assembly task in the span of a day or two. This is done in part by directly 'teaching' the system the visual appearance of objects by offering examples for inspection through the television camera. Elementary powers of forming, learning and matching descriptions are thus incorporated. The complete system is known locally as FREDDY.

As a result of a reading of J. Z. Young's Croonian Lecture [2], the idea suggested itself of concocting a task of the kind used in studies of animal behaviour. We settled on phenomena of discrimination learning and transfer of learning between sensory modalities. A hypothetical animal, *Freddypus vulgaris*, was defined which has two kinds of object in its environment: inanimate objects, distinguished by being hard to the touch and simple in shape; and food objects, distinguished by being soft to the touch and having complex and irregular outlines. In our experiments hard objects were made from wood and soft objects such as crabs, starfishes and water beetle larvae from foam rubber (see Fig. 15.1). The task was to learn, initially by tactile probing, which objects to put in the discard area and which to carry off into the food area. With experience the system should start performing the appropriate actions without any tactile probing on the basis of visual inspection alone.

Our question then was the following: how quick and easy would it be to set up a model of such simple behaviour by scooping out the necesary software from the full FREDDY system and stitching it together into a working model? One of us (P.A.) took this in hand, and at the expense of only six hours' work had a convincing *Freddypus vulgaris* fully operational. The outline flow-diagram of the resulting program is shown in Fig. 15.2. Figs 15.3–15.6 summarize the behaviour of the program (illustrated in the original presentation by a film) when presented with a number of objects of

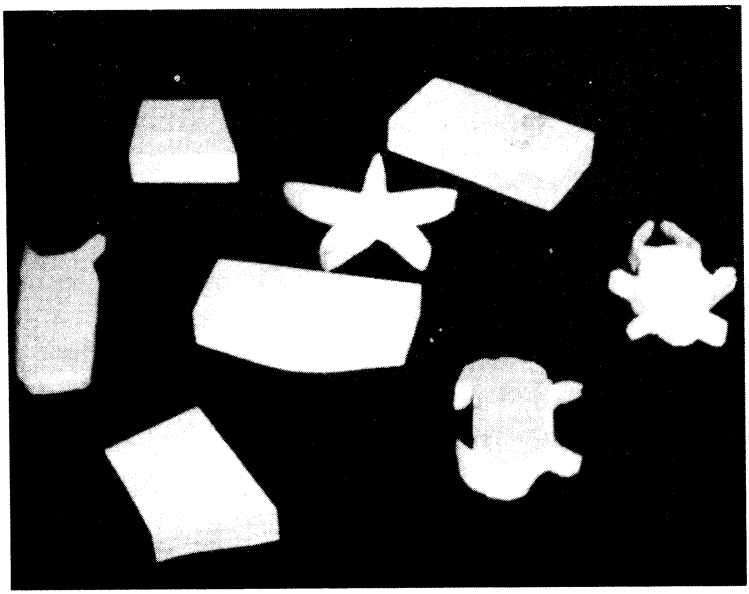


Fig. 15.1 — Examples of hard and soft objects.

the kinds shown in Fig. 15.1. Fig. 15.7 shows the results of fitting curves to the outlines of objects.

WHAT MORALS ARE TO BE DRAWN

It is possible to hope that our precise knowledge of the data-structures and algorithms inside a complex computer program should guide a search for similar structures and processes in nervous systems. We think that this may be naive, and it is probably regarded as naive by most people. At this early stage the best we can hope for is help from computer models in approaching one or two major issues at a rather high level of generality.

One such issue is that of *direct* representation versus *symbolic* representation of external reality, or (as we might rephrase it), *maps* versus *descriptions*. An example of a *map* is a literal map, like those printed in the AA book. An example of a *description* is the triangular table printed at the end of the AA book giving inter-city distances, from which many of the main features of the map of the UK can be reconstructed. Indeed, Gower has developed a technique which achieves precisely this reconstruction more or less successfully [6, 7].

What has this to do with the neural representation of the outside world and with *Freddypus vulgaris*? Consider two questions:

(1) Are *both* categories of representation ('maps' and 'descriptions') used in technological models?

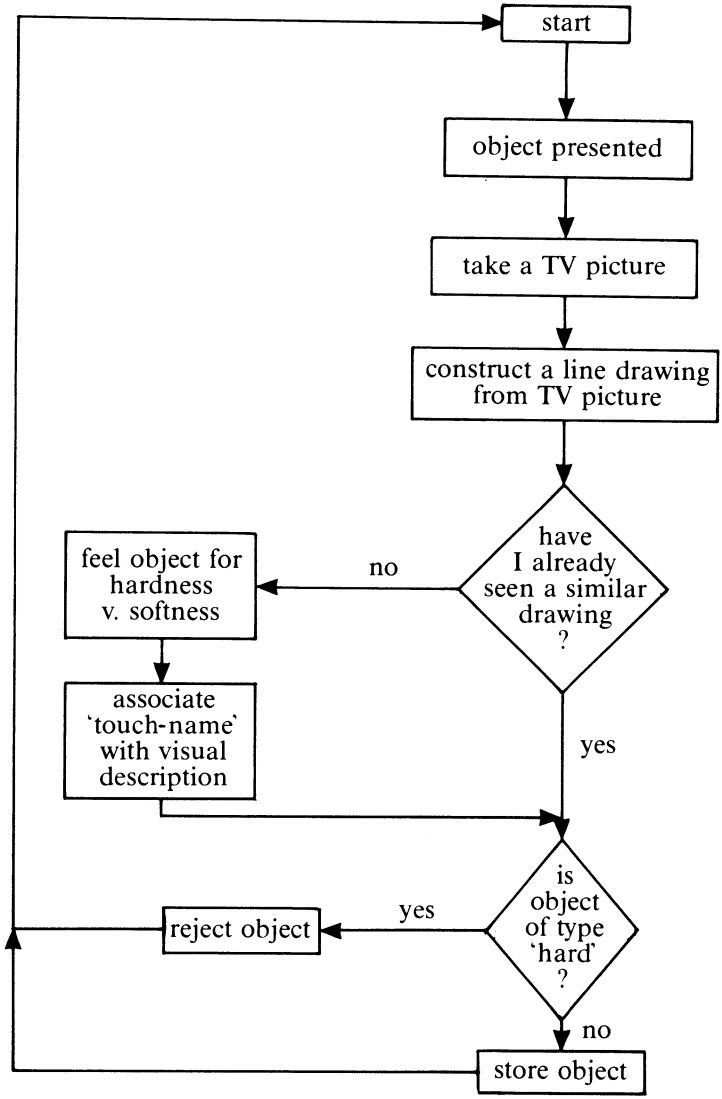


Fig. 15.2 — Outline flow-diagram of the *Freddypus* program.

(2) Are *both* categories of representation used in nervous systems, or only one?

As far as question 1 is concerned, the answer is that both are used. A rather extreme case of map-like representation is provided by recent work of Baker on iterative arrays of sequential logic circuits [8]. Each circuit is in effect a miniature computer, occupying a distinct cell of a 3-dimensional array, and capable of receiving inputs from its neighbouring cells, and in

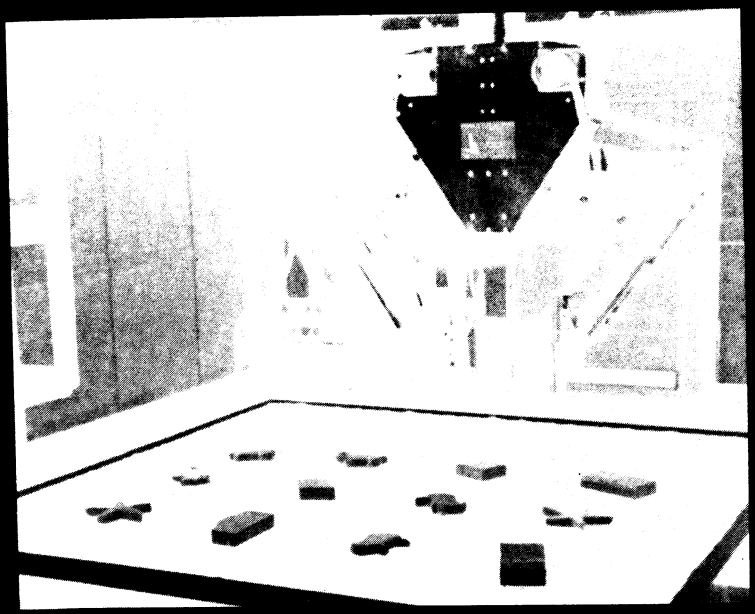


Fig. 15.3 — Layout of objects at the start of the task.

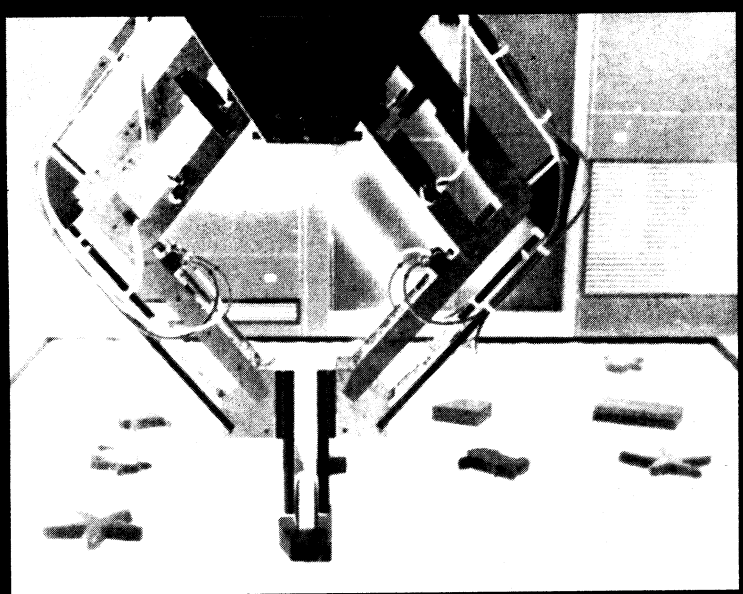


Fig. 15.4 — Feeling an unknown object for hardness v. softness.

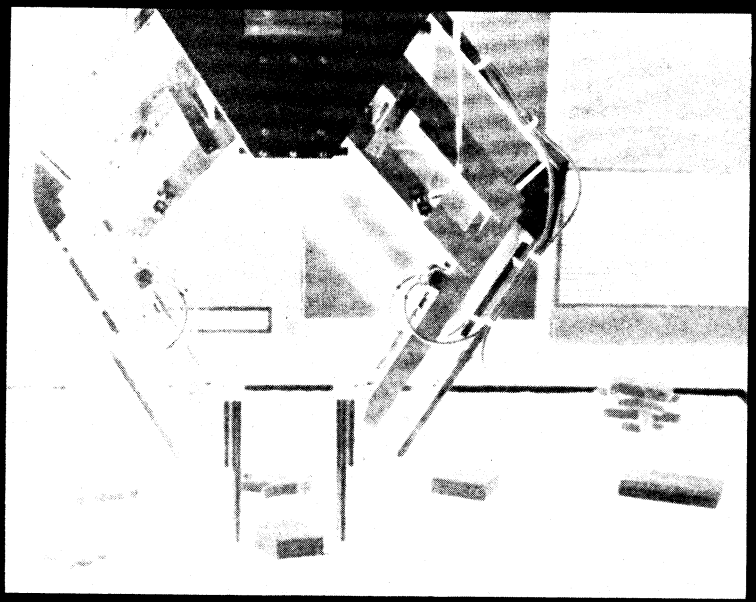


Fig. 15.5 — Picking up an object.

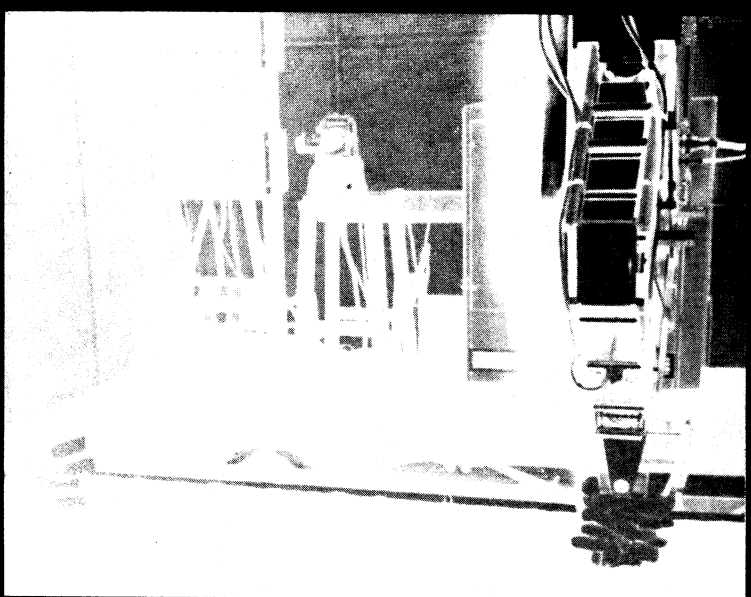


Fig. 15.6 — The completed task.

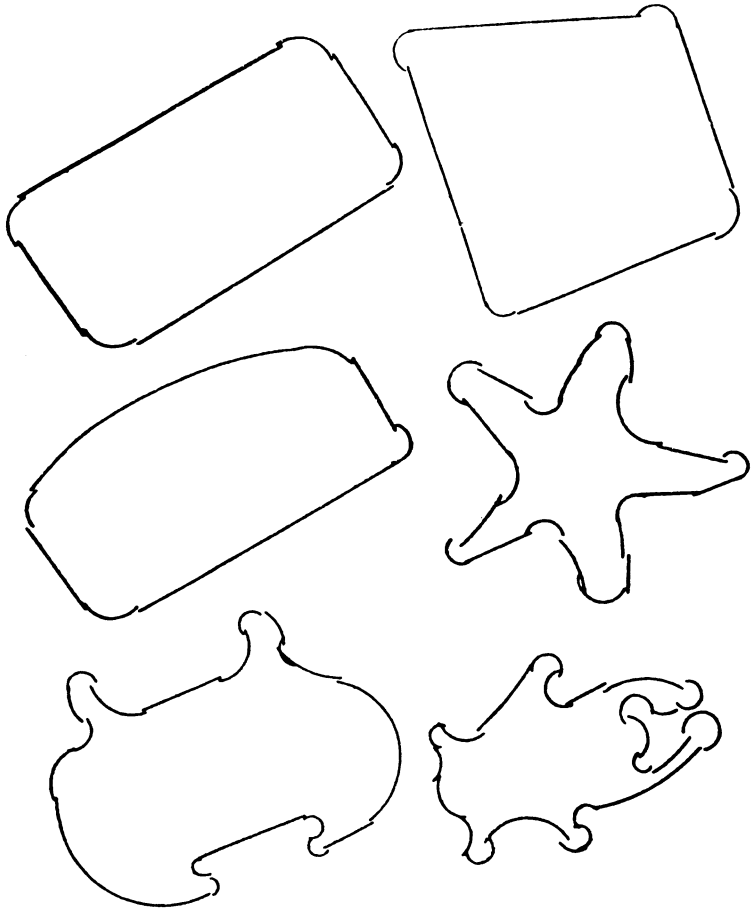


Fig. 15.7 — Examples of line drawings extracted by machine from TV views.

turn delivering to them the results of computations performed on these inputs. Baker developed his computing system with a technological aim in mind, namely, to make it quick and easy to compute predictions of the trajectories of rigid bodies moving through space. He points out the wastefulness of handling such simulations by conventional sequential computing. He also points out that we have a powerful subjective impression that our own predictions are computed map-wise, or as we should say once we introduce the time dimension, by running a sort of cinematograph in the head. Baker's proposal, verified in trial computations on easy cases, is to represent an object in 3-space by assigning 1 to each occupied cell of the array and 0 elsewhere, and to run the cinematograph by appropriately organizing the computation rules of his 3-dimensional array of automata.

Related to this, Sloman [9] gives as an example of what humans sometimes call 'visual thinking' the problem illustrated in Fig. 15.8: what will happen if we pull the A end up?

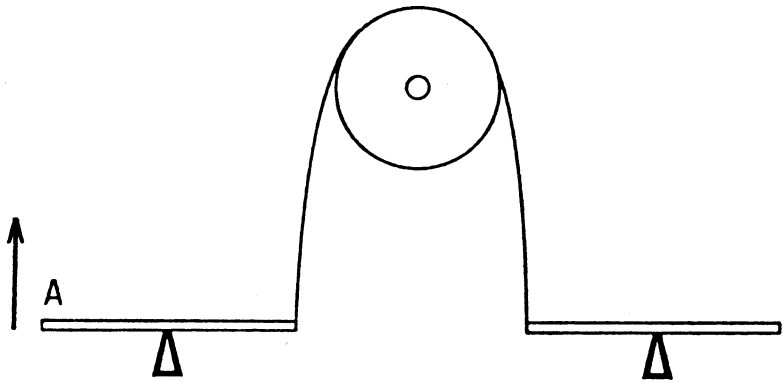


Fig. 15.8 — What will happen if we pull the A end up?

MAP-LIKE REPRESENTATIONS

Clearly, we do not determine this by formulating the problem and the physics of pulley systems as expressions in some formal logical language, from which we then compute the answer by deduction. Is there evidence in the brain of *direct* map-like representation of dynamic events like this? At the present state of knowledge there does not appear to be any such evidence. However, at a more prosaic level Hubel and Wiesel have shown that detectors sensitive to moving lines exist in the visual cortex of the cat [10].

In the case of *Freddypus* there are representations of both kinds interwoven: map-like co-ordinate representation for the viewing platform 'in the large'; and a mixture of the two for constructing the visual descriptions and matching one description with another 'in the head' as it were.

In the table-top world of *Freddypus* the positions of objects are specified by means of a 3-dimensional co-ordinate system whose origin is at the centre of the platform. The X and Y axes are parallel to the sides of the platform, and the Z axis points vertically upwards. For each object various positional data are calculated. Of these the most important is the point on the table corresponding to the centroid of the object, since it is this point that is actually used to specify the position of the object on the table.

DESCRIPTIONS

Fig. 15.9 shows the kind of data-structure used for description of objects 'in the small'. It is termed a 'tree'. The root of the tree consists of a list of the names 'hard' and 'soft', and associated with each name is a list of visual descriptions of the objects so far encountered that have the respective tactile property. A visual description consists of a 'region', which is formed from an 'outline' of the object and a 'holeset'. Associated with each region are two properties of it, namely its area and its compactness. At the next level of

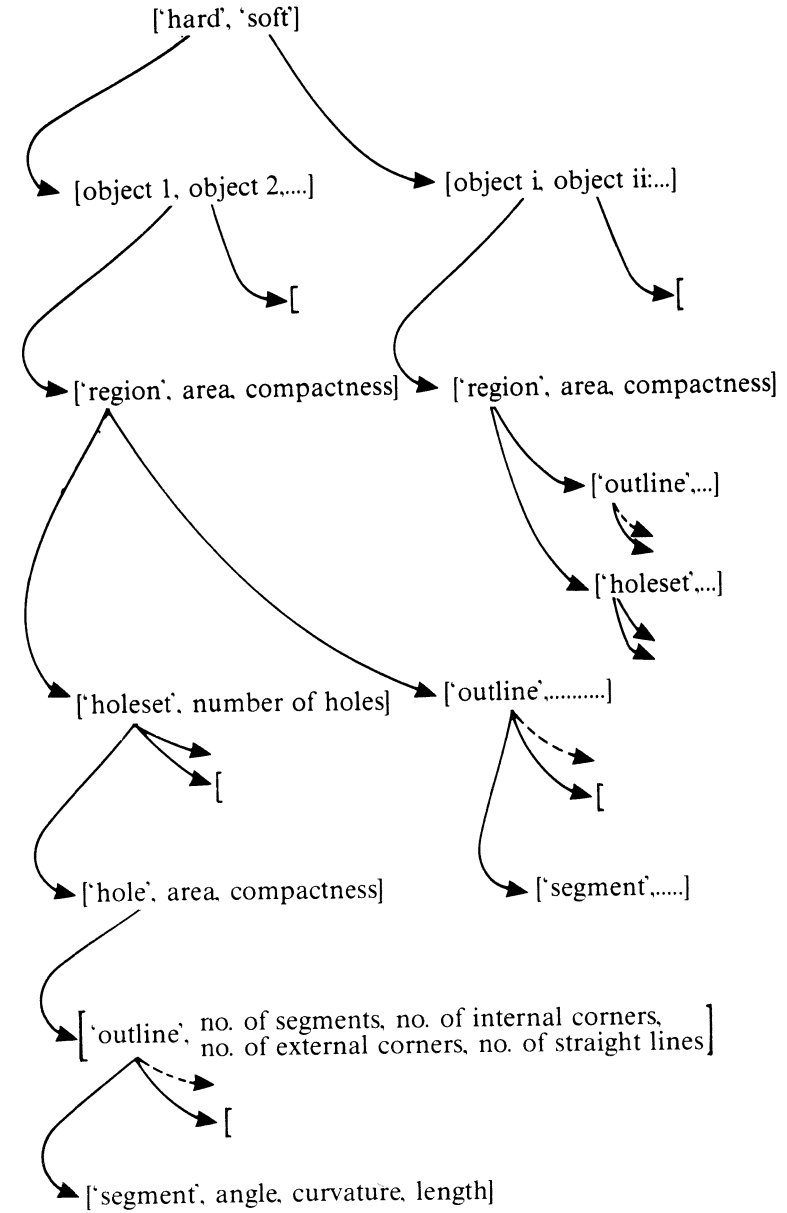


Fig. 15.9 — Example of the type of data structure used in the description of objects.

description we consider 'outline' and 'holeset'. Both these have a number of associated properties and components. The components of 'holeset' are the individual holes contained in the object, if any. Holes are described in terms of line segments and have associated with them their area and compactness. Outlines of a region are also described in terms of line segments.

Since such tree structures do not look in the least like starfishes, crabs, bricks, etc., they belong to the symbolic category. How neurobiologists are going to determine whether such a category exists in the world of neural representations is not for us to say, but the question seems a valid one. The material we have presented may perhaps help to call attention to an issue concerning internal representations which AI people worry about already, and which we suspect will sooner or later have to be tackled at the neurobiological level.

REFERENCES

[1] J. C. Clymer (1973) A computer simulation model of attack learning behavior in the octopus. *Technical Report no. 141.* Ann Arbor: Department of Computer and Communication Science, University of Michigan.

[2] J. Z. Young (1965) The organization of a memory system. *Proc. Roy. Soc. B*, **163**, 285–320.

[3] J. Maynard Smith (1952) The importance of the nervous system in the evolution of animal flight. *Evolution*, **6**, 127–129. Reprinted in *On Evolution* (J. Maynard Smith), 29–34. Edinburgh: Edinburgh University Press, 1972.

[4] H. G. Barrow & G. F. Crawford (1972) The Mark 1.5 Edinburgh Robot facility. *Machine Intelligence 7* (eds. B. Meltzer & D. Michie), 465–80. Edinburgh: Edinburgh University Press.

[5] A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall & R. J. Popplestone (1973). A versatile computer-controlled assembly system. *Proc. Third Intern. Joint Conf. on Artificial Intelligence* 298–307. SRI Publns. Dept.

[6] J. C. Gower (1966) Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, **53**, 325–38.

[7] J. C. Gower (1968) Adding a point to vector diagrams in multivariate analysis. *Biometrika*, **55**, 582–5.

[8] R. Baker (1973) A spatially-oriented information processor which simulates the motions of rigid objects. *Artificial Intelligence*, **4**, 29–40.

[9] A. Sloman (1971) Interactions between philosophy and artificial intelligence: the role of intuition and non-logical reasoning in intelligence. *Artificial Intelligence*, **2**, 209–225.

[10] D. H. Hubel & T. N. Wiesel (1959) Receptive fields of single neurones in the cat's striate cortex. *J. Physiology*, **148**, 574–91.

16

Machine models of perceptual and intellectual skills (1979)

Figure 16.1 shows an unusual result obtained by drawing a square and asking

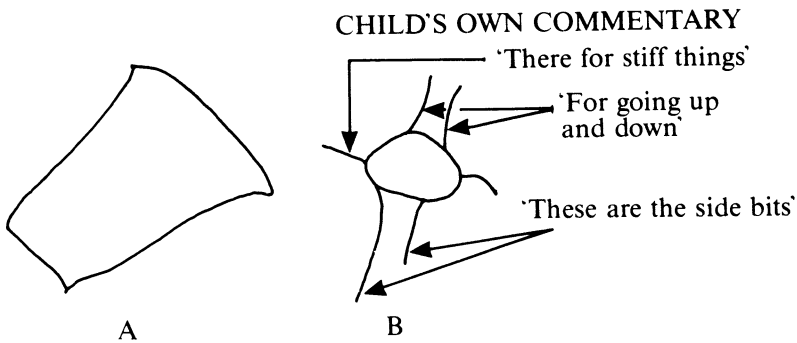


Fig. 16.1 — Copies of a square drawn by a 3½-year-old girl. By asking her to indicate on the original model the 'stiff things', things 'going up and down', and 'side bits', it was ascertained that these phrases denoted the square's corners, uprights, and horizontals respectively.

a 3½-year-old girl to copy it. Her first attempt is on the left. Her second reproduced on the right, departs wildly from the first, and from anything which the ordinary onlooker might have expected her to do. As will be explained later the phenomenon reveals the normally hidden operation of a particular way of compactly encoding percepts of external reality. The species of symbolic description underlying the girl's drawing is today routine in work on computer vision. But without contact with artificial intelligence techniques it is not easy to spot what this strange beetle-like representation is saying about the class 'square', nor to empathize with its neat graphic

encoding of the class's defining properties and relations. Even less obvious, but necessary to complete the insight, is the realization that the more conventional drawing on the left is also not to be interpreted as an effort to trace the retinal image out onto paper. Indeed it is thought to represent the culmination of an even more complex project than the right-hand drawing: more complex, because of the inclusion of an additional stage, namely reconstruction, from the symbolic description, of something which (as the adult would say) 'looks like a square'. I shall return later to this mental skill of drawing a square.

That our most distinctive quality is mental skill was recognized some time ago when the taxonomists gave our species the name *Homo sapiens*. That it is put together like other biological functions, as a large and heterogeneous box of tricks, is a notion slower to gain general acceptance. Explanatory models of man's attributes taken from religion, art, and literature still hold some sway, and the most biologically minded person can on occasion find himself seeking special exemption for the cognitive attribute, picturing himself perhaps as Michelangelo's Adam. Propped on one arm, he stretches the other towards his airborne Creator, to receive... what? Not the spark of life surely, for Adam is already plainly, if somewhat languidly, alive; but the spark of reason to set him above the beasts. Pure Reason, Pure Thought, Pure Faith... all essence, no detail. Such attributes are seen as bestowed from the top down, rather as mathematics — we are told — received the integers. Yet biological attributes invariably come by a different route, and they evidence evolution's *bottom up* mode of construction in the rambling and complex architectures which confront every serious investigator. The medieval scholar's explanation of the effects of chewing poppy seed: '... Quia est in eo Virtus dormitiva!' gives place to the pharmacist's catalogue of soporific compounds and the pharmacologist's detailed maps of molecular structures and their neuronal interactions.

Already in the nineteenth century audacious voices were proposing that attempts to account for the phenomena of cognition must tread just such a path. In this Chapter I hope to indicate that a new and exceedingly forceful model has come amongst us, derived from recent uses of computers in complex problem domains, and that in so far as lessons can yet be read they reinforce the bottom-up philosophy. Indeed those of us who are professionally concerned to emulate advanced mental skills by machine could and should have been quicker to appreciate this essentially biological perspective. Self-regulating systems of high complexity, whether animate or inanimate, are (it will be argued) best viewed in this way.

Priority for this appreciation of the matter seems to belong to the self-taught English social philosopher Herbert Spencer. He was, as William James put it, 'the first to see in evolution an absolutely universal principle' and he boldly applied it to cognitive phenomena among others. The evolutionary principle as he conceived it proceeds by *progressive individuation*. He describe this idea in *Social Statics* published in 1851:

> Between creatures of the lowest type, and creatures of the highest,

> we similarly find the essential difference to be, that in the one the vital actions are carried out by a few simple agents, whilst in the other the vital actions are severally decomposed into their component parts, and each of these parts has an agent to itself.

Spencer indeed intended the principle to be absolutely universal, and to apply to tissues, organisms, species, minds, societies — presumably to computing systems, too, had they existed in his day. Yet each discipline, it seems, must painfully learn it all anew. The history of machine intelligence research over the past ten years has consisted in a progressive abandonment of sweeping uniform procedures for search, deduction, linguistic analysis, and the like, in favour of a more Spencerian approach in which the representation in machine memory of human knowledge in all its *ad hoc* complexity is seen as critical.

It is significant that this methodological revolution was pushed forward by the world's leading institute of technology (at Cambridge, Massachusetts) and not by some centre of pure science. The re-orientation has indeed been painful for many in a field populated largely by those trained in the conspicuously non-Spencerian disciplines — the mathematical and physical sciences. Particularly embarrassing to those who look for progress in the form of a succession of blinding flashes is the realization that the knowledge sustaining a given skill — whether in visual perception, in medical diagnosis, in automated chemistry, in game-playing or in mathematical reasoning — reveals itself as typically vast in total bulk and highly individuated, to use Spencer's term, into a mosaic of interlocking sub-domains and sub-sub-domains. The successful computer program turns out in each case to be the one which, while preserving a degree of overall control, seeks to associate with each separate part of the mosaic its own special package of local advice and useful knowledge. Spencer's scheme can be traced in the microcosm of visual perception built by Waltz (1972) for the restricted but non-trivial task of interpreting line-drawings of polyhedral scenes with shadows, as in Fig. 11.3, which illustrates the task, and Table 11.1 which shows a part of the program's internal catalogue of knowledge useful for that task. Table 11.2 compares the numbers of labellings of different kinds of vertex in a line drawing which are possible before and after application of the constraints derivable from this catalogue. Notice (1) how effectively even a little knowledge can defuse a large combinatorial explosion, and (2) the complete absence of 'top-down' knowledge in the form of conventional theories of optics and mechanics. In macrocosm, the bottom-up philosophy has been expanded by Minsky (1975) into an ambitious proposal as to how we might endow a machine with a usable stock of knowledge about the world at large. At an intermediate level Spencer's characterization applies with wonderful exactitude to Buchanan and Shortliffe's knowledge-based program for clinical bacteriology (Shortliffe 1976), to Buchanan, Feigenbaum, and Lederberg's DENDRAL program for mass spectroscopy (see Buchanan, Smith, White, Gritter, Feigenbaum, Lederberg, and Djerassi 1977 for a

very recent development) or indeed to every one of the as yet small cohort of knowledge-based computer programs which can claim, within their specialized areas of competence, to match human experts in complex domains.

Machine intelligence thus turns out to have the character more of biology than of physics. Although we must always strive for the great unifying and simplifying principles wherever they may be gained, we have to live with the fact that there is such a thing in science as irreducible complexity. Elsewhere I discuss quantitative measures of complexity in the context of intellectual difficulty and of knowledge (Michie 1977).

Let us now try out the Spencerian scheme by interpreting his 'creatures' of higher or lower type as computer programs of the kind created in the laboratory today in machine intelligence work, i.e. machine implementations of human skills. The format of the interpretation goes as in Fig. 16.2.

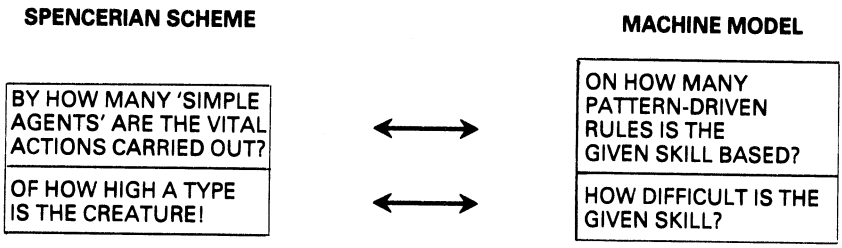


Fig. 16.2 — Interpretation of Spencer's scheme of 'progressive individuation' in terms of experimental computer programs developed to implement complex mental skills.

Among human skills I shall consider visual perception, bicycle riding, clinical bacteriology, mass spectrometry, mental arithmetic, and chess. The first of these, computer vision, has the farthest to go, yet the experimental attempts have yielded some of the more interesting insights into the Spencerian nature of the knowledge problem.

VISION

The existence of internally stored visual patterns is clear to anyone who knows what it is to 'see things' in an apparently randomly patterned surface. The following account is by Leonardo da Vinci:

> If thou wilt look carefully at a wall spotted with stains, or at stones variously mixed, thou may'st see in them similitudes of all sorts of landscapes, or figures in all sorts of actions and infinite things which thou may'st be able to bring into complete and good form.

It would seem that this harmless, and apparently pointless, phenomenon reveals a mechanism of crucial utility when we want to see things which really *are* there, as in Fig. 16.3 from R. L. Gregory (1970). The photograph

Fig. 16.3 — At first sight 'a wall spotted with stains' (see text). This photograph of a dalmation dog is from R. L. Gregory's *The intelligent eye*. (Used with the permission of the McGraw-Hill Book Company.)

might almost be of Leonardo's 'wall spotted with stains', so that the eye's instant discovery of a Dalmatian dog suggests an internal library of dog-part patterns from which the successful identification has been assembled. There is no suggestion that such patterns are like Identikit parts, which take the form of actual templates. On the contrary, Gregory and others tell us that the evidence is for Helmholtz' idea that percepts are reconstructed from stored descriptions of a more generalized nature. The Helmholtz assumption is that perception operates by evocation, a kind of triggering one might say, of appropriate reconstructions from a vast stored treasure-house of knowledge about how the world looks. A brutally direct test of the assumption would be provided if a congenitally blind man were to have the use of his eyes restored in later life. The assumption predicts that, lacking any internal accumulation of perceptual models, he would at first be unable to 'see' in any useful sense.

R. L. Gregory studied just such a case, in which the patient had had sight conferred on him at the age of 51 by an operation of corneal grafting. At first the man could make little of the images received through his eyes, except in the case where he had prior knowledge through tactile experience. He read

upper case block letters immediately on sight, but it took him time to learn lower case letters. It turned out that at the blind school he had been given raised letters on wooden blocks to learn by touch, but only upper case, not lower case. Soon after he left hospital, writes Gregory, 'we showed him a simple lathe and he was very excited. We showed him it first in a glass case, at the science Museum in London, and then we opened the case. With the case closed he was quite unable to say anything about it, except that the nearest part might be a handle... but when he was allowed to touch it he closed his eyes and placed his hand on it, when he immediately said with assurance that it was a handle. He ran his hands eagerly over the rest of the lathe, with his eyes tight shut for a minute or so; then he stood back a little, and staring at it he said: "Now that I've felt it I can see."'

Forty-eight hours after the corneal grafting operation, the patient saw a London bus, a two-decker. Fig. 16.4 gives the 'camera's-eye view' of such a

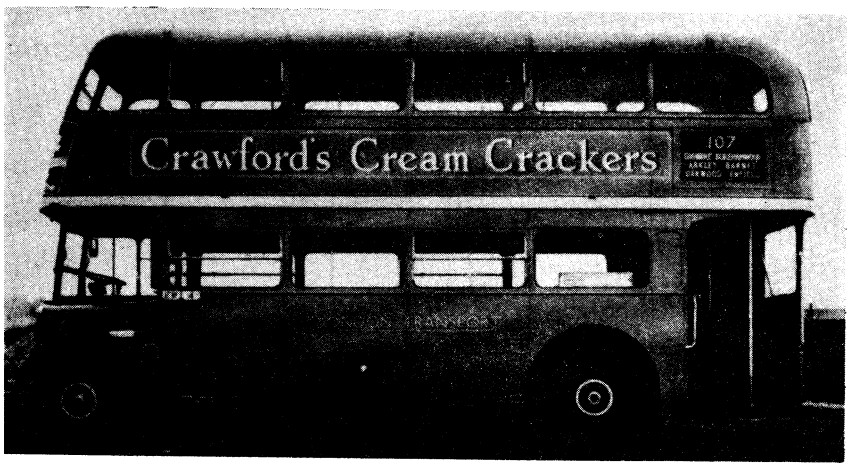


Fig. 16.4 — 'Camera's eye view' of a London two-decker bus (London Transport).

bus. Compare with this the patient's drawing, reproduced in Fig 16.5. Six

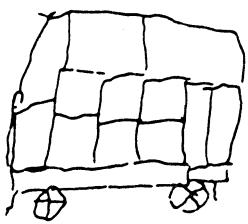


Fig. 16.5 — Patient's drawing two days after the operation.

months later he produced Fig. 16.6, and after a further six months Fig. 16.7.

Fig. 16.6 — Patient's drawing after 6 months.

Fig. 16.7 — Patient's drawing after a year.

Note how experience has filled in the detail of what he is able to see, except for the front of the bus which Gregory conjectures he would not have had opportunities to explore with his hands.

These observations merely support the necessity of internal models without saying anything about their structure.

Piaget's school has accumulated instances of a rare and revealing phenomenon which can be elicited from normal 3–5 year old children asked to copy simple geometrical figures. Returning to Fig. 16.1, which was taken from Hayes (1978), we see the phenomenon in unusually clear-cut and striking form. The beetle-like object on the right does not look in the least like the square which the small girl was asked to copy. But this is not because she cannot make a 'proper' adult-type copy, as evidenced by her first attempt, which is on the left. The child has in this case been induced to tell us in her own words, as shown in the figure, just what she thinks she is doing. It seems as though, as it were without noticing, she has on this occasion omitted the final reconstruction phase and is symbolizing in a graphical

language the descriptive structures with which she represents squares in memory. Hayes introduces her observations in words which can hardly be bettered as a statement of the role of internally stored patterns:

> Work on machine perception in different laboratories has for some time been converging on essentially the same format for representing visual scenes, according to which the scene is decomposed into *primitive elements* which are then characterized by *properties* and *relations* holding among the members of sub-sets of elements (see Barrow and Popplestone 1971, Winston 1970). Representational schemata formed in this way are often referred to as *relational structures*...

This can here be exemplified by a generalized relational structure taken from the Edinburgh robot work, reproduced in Fig. 15.9. She continues:

> ... The possibility suggests itself that some part of the human faculty of visual recognition (which is immensely more powerful than any machine system yet developed) may be based on similar processes of decomposition into primitive features and re-aggregation of these to form internal schemes.

and proposes that 'the child presents a *graphic representation* of his concept of what is perceived rather than attempting to copy the retinal image on to paper.' A bizarre exercise on the theme 'diamond' is shown in Fig. 16.8, accompanied by its interpretation in relational structure form. The more complex structure required to interpret the earlier drawing of a square is shown in Fig. 16.9.

In this chapter the word 'pattern' will appear from time to time. No more is meant by it than a description, in the form of some collection of primitive elements together with properties and relations defined on them. Could some description-handling formalism be developed into an actual 'seeing machine'?

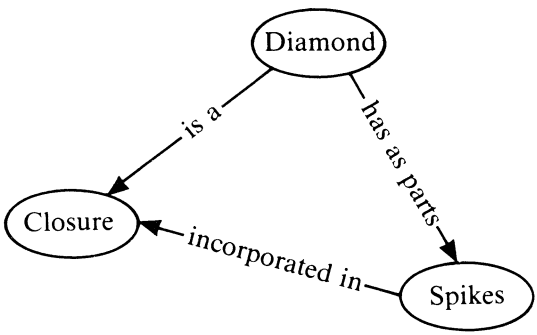
In 1966 R. L. Gregory and I chanced to meet in New York and we planned a project to develop a machine capable of visual perception, interpreting perception as extending to the actions and uses associated with perceived objects. This robot project was ambitious — absurdly ambitious, some felt. But thanks to generous sponsors and to the moral support and hard work of some exceptionally gifted colleagues, it got a surprisingly long way — at first base, while Gregory was still with us, sufficient to demonstrate successful acquisition and matching of descriptive patterns, and eventually far enough to be able to address questions like 'How many stored patterns does it take to "see" a simple world?'

In 1973 the work was discontinued (documentation of the project has been given by Ambler, Barrow, Brown, Burstall and Popplestone, 1973). Subsequently the National Engineering Laboratory, intrigued by the possible industrial uses of a seeing machine, came forward to enable a small-scale recommencement of the work. Today we have a visual command language for instructing a parallel array processor in the extraction and manipulation

(a) A square

(b) A diamond

(c) A diamond



(d)

Fig. 16.8 — (a) a square; (b) a diamond; (c) a diamond; (d) interpretation of (c) as a relational structure.

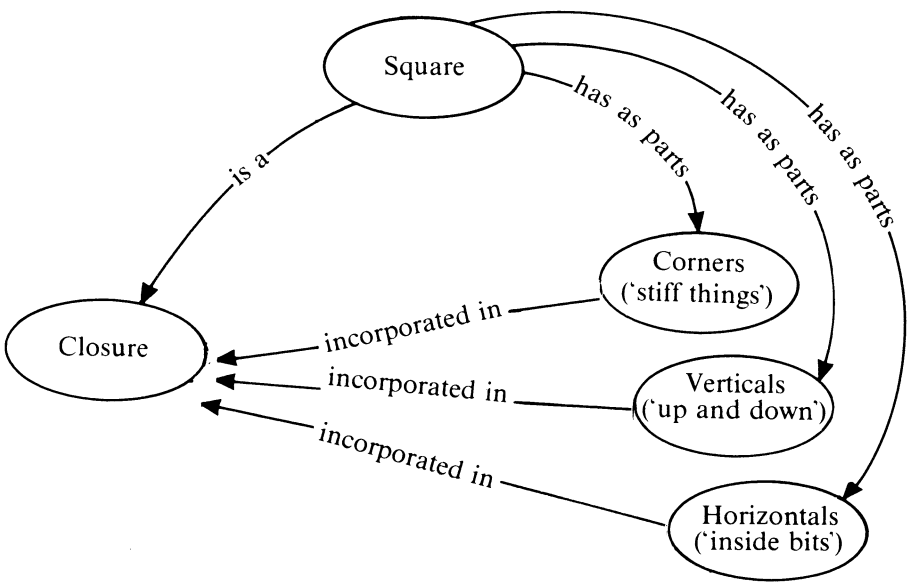


Fig. 16.9 — Symbolic representation of a square as a relational structure, following the child's graphical representation shown in Fig. 16.1.

of descriptive primitives. Execution times of the order of a millisecond seem attainable for the simpler tasks (Armstrong & Jelinek 1977), and solutions have been found for the problem of programming higher-level descriptions in parallel mode.

After this dip into visual perception to set a mental frame, we shall turn to other domains. But first a digression is in order on the subject of top-down *versus* bottom-up theories in science.†

TOP-DOWN AND BOTTOM-UP

The house of science has always had two levels. Upstairs live the great explanatory theories, which proceed top-down from the highest level of abstraction and show with great conciseness how particular consequences can be derived. In the basement are the servants' quarters, the abode of various compilations and catalogues of useful know-how. In general these latter, the bottom-up representations, are expected to keep themselves to themselves, except when called to serve some particular need — to mend or make a chair, to cook a meal, to run a message. Very occasionally some servant-scholar arises — an Aristotle, a Bacon, or a Spencer — and we have an explanatory annotation on the bottom-up trade, arguing that it, too, has its unifying principles. Upstairs people have always smiled indulgently at such quaintness — except in one very exceptional circumstance, which has to do with the introduction of mechanical aids, regarded as being by their very nature downstairs property. In his *Life of Marcellus*, Plutarch tells us of the reactions of Plato, the earliest recorded exponent of the top-down mystique, on learning that two mathematical colleagues had been so engaged:

> Eudoxus and Archytas had been the first originators of this far-famed and highly-prized art of mechanics, which they employed as an elegant illustration of geometrical truths, and as a means of sustaining experimentally, to the satisfaction of the senses, conclusions too intricate for proof by words and diagrams... But what with Plato's indignation at it, and his invectives against it as the mere corruption and annihilation of the one good of geometry — which was thus shamefully turning its back upon the unembodied objects of pure intelligence to recur to sensation, and to ask help... from matter; so it was that mechanics came to be separated from geometry, and, repudiated and neglected by philosophers, took its place as a military art.

Developments in programming the digital computer, so as to invade territory which could not otherwise be easily penetrated by the human

†These terms clash with a similar but distinct use by Arbib (1976). Since he is in print first, I should by rights give way. But as yet I have found no satisfactory alternatives to 'top-down' and 'bottom-up' for the categories which were already revolving in my mind before I came upon his paper: 'intensive' and 'extensive' seem to hit it off in some contexts, and they will also be given some trial runs here.

intellect, have revived the issue rather forcibly and have at the same time removed the last ground for Platonic indignation. A startling recent instance which Plato would have found hard to dismiss is the recent triumph of Appel & Haken (1976) over the celebrated four-colour problem using some 10 000 hours of processor time on a powerful computer. The authors of the computer-aided proof believe that the problem, although simple to express — namely that every map drawn on a plane or a sphere can be coloured using only four colours in such a way that no two same-coloured regions share a boundary — is intrinsically 'bitty' and that no elegant Platonic demonstration ever will be, or ever could be, discovered.

An essential feature of their proof is a catalogue of some 1800 sub-maps which are 'unavoidable' in the sense that any map whatsoever must contain at least one member of this catalogue. The authors do not believe that the size of the catalogue is open to substantial reduction. If they are right, then a truly top-down theory in the form of a *concise* demonstration of the theorem's truth may simply not exist.

The special function of a top-down or 'intensive' representation is to give us the 'Aha!' feeling — the sensation that we have understood something. A bottom-up 'extensive' representation is more like a manual of how to *do* something. Because this distinction was not fully grasped, some of the early computer approaches to complex problem domains (by 'early' I do not necessarily mean anything earlier than about ten years ago) were pointed firmly in the wrong direction. Bitter experience in language-translation, in computer game-playing, and in other areas had to teach us that you cannot implement a skill simply by translating into a computer program the corresponding intensive theory.

Consider bicycle-riding. The intensive theory is all there in Newtonian physics. John Brakewell, however, of the Aeronautics and Astrophysics Laboratory at Stanford University recently abandoned the attempt to program a computer for this particular task. Increasingly we see that in machine intelligence work one glance at biology is worth at least two glances at established computer science and perhaps a hundred and two at mathematical physics. After all, we already knew, or should have done, that skill (as opposed to understanding) is not acquired in this way. Children become language-users without taking a course in theoretical linguistics, they become chess-players without first studying game theory, and they ride bicycles — and circus seals balance poles — in complete innocence of Newtonian statics and dynamics.

A QUESTION OF BALANCE

Pole-balancing was the subject of one of the earliest 'bottom-up' experiments to be done in machine intelligence, little more than ten years ago. It is marginally possible (but not very economical) to control an inverted pendulum by computer program using the classical theory, as shown by Eastwood in the mid-1960s. He illustrated his Faraday lectures with an adaptive computer program able to control a pole balanced on a motor-

driven cart, as in Fig. 14.2. Meanwhile Chambers and I had independently developed a program for the same task based on an elementary bottom-up representation depicted in Fig. 16.10.

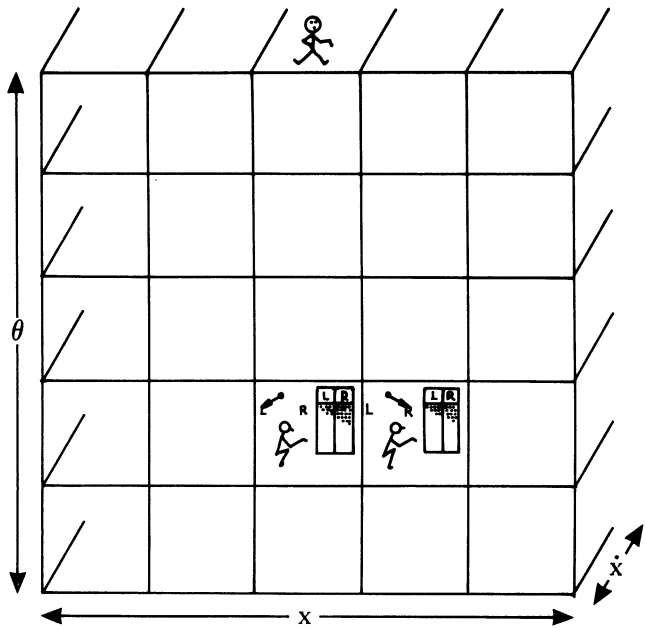


Fig. 16.10 — The state space (for clarity omitting the fourth, i.e. $\dot{\theta}$ dimension) divided into 'boxes' with an independent automation in each box and a 'leader' supervising.

Each local region of the four-dimensional state-space defined by the axes: position, velocity, angle, rate of change of angle, is supervised by a separate computational process (a 'demon', to use the now-fashionable term originally coined by Oliver Selfridge in 1959) which accumulates its own private store of knowledge useful for that particular region. Fig. 16.11 shows a specimen 'learning' run. When it was all over I had the pleasure of illuminating discussions with Eric Eastwood in which we compared the two approaches. The BOXES program, as we called it, exemplified Spencer's scheme with a crudity and literalness redeemed by its demonstrated effectiveness on a hard task.

The program comprised 225 pattern-based rules, which could have been halved by exploiting symmetries. It could be used in a man-machine co-operation option in which at each time-slice the light-pen was interrogated for a control signal from the human partner. The human's signal was adopted whenever there was one: otherwise the needed decision was retrieved from the appropriate 'box' among the 225 boxes covering the state-space. In this way a human 'tutor' could feed his own skill into the machine in

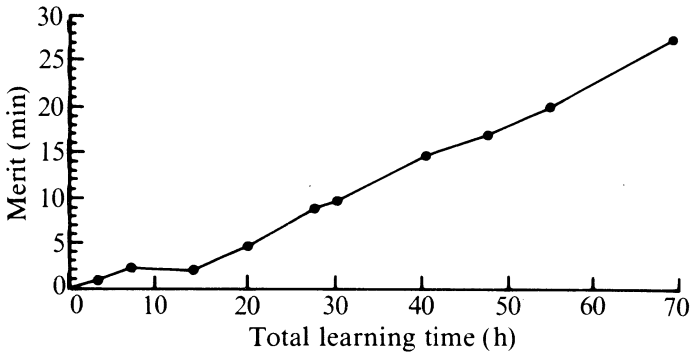


Fig. 16.11 — The pole-and-cart system, set up for pure trial-and-error learning. The lower curve shows a smoothed average ('merit') of the time-until-crash, plotted against the total accumulated learning time.

reinforcement of the program's self-learning. Actually neither partner was carried by the other since the task of skill-acquisition proved to be extremely taxing for the unaided human, and transfer of skill undoubtedly proceeded in both directions. Some regions of the space exhibit markedly counter-intuitive features, as when the cart is wandering dangerously near to the 'precipice' at the end of the track. In a proportion of cases, according to the values of pole angle and angular velocity, the solution is to drive initially *towards* the precipice, so as to impart a swing of the pole away from it. Only then is it safe to direct the motor away from the danger area, 'chasing the pole' with proper control over its angle.

The BOXES program was one of the earliest confirmations of what Allen Newell and his school had already been saying, namely that the most natural machine representation of human skill is the *production system*, as such organized collections of pattern-based rules are known these days. Newell and his colleagues made the further conjecture, documented by a growing experimental corpus, some of it derived from machine intelligence work, that the same skills are similarly implemented in the brain. Table 16.1 gives some figures for a few skills which have been intensively studied in man or implemented to a high level of performance on the machine. The DENDRAL and MYCIN programs, within their restricted domains of scientific know-how, have now attained peformance levels comparable to that of highly trained human professionals. DENDRAL, moreover, manifests at a sophisticated level the capabilities both of taking instruction and of improving its repertoire autonomously. These were both mentioned at a primitive level in the case of pole-balancing. The lesson is already clear that bottom-up representations of knowledge are forced upon the designer if his system is to lend itself to the incremental acquisition of new knowledge. Note that the number of rules underlying these two applied science skills is about 400 in the two cases. To obtain an exact picture of the range of expertise thus implemented, the original literature should be consulted. The expertise is wide enough to be useful in a competitive professional context. I now pass to

Table 16.1 — Some contrasts between the two kinds of theory from the standpoint of machine models of mental skills. The figures indicating the numbers of rules used to implement various skills must not be interpreted as indicating that there is more knowledge in Grandmaster chess than, say, in clinical pathology. DENDRAL and MYCIN each cover only a small fraction of the total problem domain as yet.

Skill	Top-down (intensive) theories (suitable for understanding; lend themselves to computer implementation as *algorithms*).	Bottom-up (extensive) theories (suitable basis for skill; lend themselves to computer implementation as catalogues of pattern-based rules).	No. of pattern based rules in implemented system.
Seeing a scene	Geometry and optics. First-ever vision program, Roberts, early 1960s.	Incremental catalogue of visual patterns. Simple scenes of shadowed polyhedra, Waltz, early 1970s.	10
Balancing a pole	Mechanics, control theory. Eastwood, mid-1960s.	Incremental catalogue of pattern-based rules. Michie and Chambers mid-1960s.	225
Identifying organic compounds from mass spectra	Topology, combinatorics, physics.	Incremental catalogue of pattern-based rules. DENDRAL program of Lederberg, Feigenbaum, and Buchanan.	*c.* 400
Identifying bacteria from laboratory tests on blood and urine	NONE	Incremental catalogue of pattern-based rules. MYCIN program of Buchanan and Shortliffe.	*c.* 400
Calculating-prodigy arithmetic	Peano's axioms with definitions and inference rules.	Alexander Aitken, studied by Hunter, 1962, used pattern-based rules.	?
Grandmaster chess	Zermelo-Borel-von Neumann iterated minimax algorithm. Tournament programs — with heuristic trimmings. Master skill not yet attained.	Chess-masters, studied by Binet, de Groot, Chase and Simon, Nievergelt, use pattern-based rules.	30 000 v. approx.

a skill which in today's world must be judged non-useful in the extreme, namely mental arithmetic.

PRODIGIOUS CALCULATIONS

The world has a false belief about calculating prodigies, namely that they calculate prodigiously.

They would of course *need* to calculate prodigiously if they had built their skill from some intensive theory of arithmetic; in the extreme case let us imagine some theory such as Peano's axioms together with a sufficient set of rules of inference for their use! Such a theory would be, in McCarthy & Hayes's (1969) terminology, 'epistemologically adequate'; even for a prodigy speeded up by a factor so great as to compress the history of the universe into a few seconds it would not be 'heuristically adequate'. If, as is generally but mistakenly supposed, calculating prodigies relied solely on the

more effective and less concise apparatus which we all learned at school, the calculation required for, say, squaring a four-digit number would be formidable enough. But it turns out that this mental skill is no different from the others and indeed constitutes another refutation of the Pure Thought fallacy. The greatest calculating prodigy ever recorded was Alexander Aitken, who held the chair of Mathematics at Edinburgh until his death. He was the subject of a careful study by the psychologist I. M. L. Hunter (1962) who discovered that Aitken's extraordinary capabilities did not include any special aptitude at calculation itself. Aitken's powers turned out to be generated from a vast internal catalogue of pattern-based rules about the natural number system, which he invoked during the first few seconds after each problem was put. He used this phase to set up a calculative plan, and it was in this first phase that all his special ability was concentrated. While executing the plan he proceeded no faster or slower than anyone else — in other words by machine standards at a snail's pace. So much, however, does phase 1 dominate in importance that there is no facetiousness in saying that machine simulation of Aitken's skill would not be easy, even with the aid of the best libraries of numerical routines in the world. The challenge would lie in building a program so 'knowledgeable' as to be able rapidly to react to each new input problem by the marshalling and invocation of just the right combination of special methods. The following illustrations of his virtuosity, with illuminating comments by Hunter, give us an intriguing glimpse of the Aitken production system at work.

Here is Problem 6 of a series in which he was asked to supply an introspective and explanatory account:

> Decimalise 1/851. 'The instant observation was that 851 is 23 times 37. I use this fact as follows. 1/37 is 0.027027027027... and so on repeated. This I divided mentally by 23 [23 into 0.027 is 0.001 with remainder 4]. In a flash I can see that 23 into 4027 is 175 with remainder 2, and into 2027 is 88 with remainder 3, and into 3027 is 131 with remainder 14, and even into 14,027 is 609 with remainder 20. An so on like that. Also before I ever start this, I know how far it is necessary to go in this manner before reaching the end of the recurring period: for 1/37 recurs at 3 places, 1/23 recurs at twenty-two places, the lowest common multiple of 3 and 22 is 66, whence I know that there is a recurring period of 66 places.'

On this and many other such records Hunter comments in terms strikingly reminiscent of those used by de Groot (1965) in his study of the power of the chess-master to apprehend a chess position at a glance:

> A number is apprehended as a multiplicity of numerical attributes and, so to speak, as bristling with signalling properties. This apprehending is immediate, simultaneous, and often autonomous.

Hunter later makes the explicit connection to chess skill:

> ... with this thinker, as with many people, 12 is the immediate

> product of 3 and 4: but unlike most people, the transition from '9 times 12,345' to '111,105' is also immediate for this thinker. Consider also his 'simply seeing in one go' the number 1961 as 37 times 53, and 44 squared plus 5 squared, and 40 squared plus 19 squared. Other leaps concern procedural judgments, that is, diagnosing what method is best to use in calculation. These high-level procedural diagnoses derive from a breadth of past experience which is fully comparable to (and possibly in excess of) that which lies behind the so-called position sense of the chess master...

SKILL IN CHESS

Nowhere is the Pure Thought fallacy more firmly rooted than in popular ideas about chess. Chess-masters are *also* (quite falsely) believed to calculate prodigiously, executing essentially the Zermelo–Borel–von Neumann top-down theory which suggests that one should try to look ahead along all possible paths to the end of the game. A detective story written by Jacques Futrelle around the turn of the century is reviewed by Julian Symons (in *Bloody Murder*, 1974):

> ... Professor Augustus S. F. X. Van Dusen (Futrelle's hero-detective) is introduced to us when he refers contemptuously to chess, saying that a thorough knowledge of the rules of logic is all that is necessary to become a master at the game, and that he could 'take a few hours of competent instruction and defeat a man who has devoted his life to it'. A game is arranged between the Professor and the world champion, Tschaikowsky. After a morning spent with an American chess-master in learning the moves, the Professor plays the game. At the fifth move Tschaikowsky stops smiling, and after the fourteenth, when Van Dusen says 'Mate in fifteen moves', the world champion exclaims: 'Mon Dieu!' (he is not one of those Russians who knows no language but his own) and adds: 'You are not a man: you are a brain — a machine — a thinking machine'.

What is wrong with this story?

Two facts are in combination destructive of the credentials of Professor Augustus Van Dusen.

1. Grandmasters do *not* on the average calculate more than ordinary players. In a classic monograph de Groot (1965) showed that 6–7 half-moves ahead tends to be the limit, with a total of perhaps about 30 positions considered on the lookahead tree. The great Richard Reti dramatized the true pattern-based nature of Grandmasterly skill when he was asked how many moves ahead he looked in tournament play. 'One...' he replied, 'the right one!' This must of course be almost literally applicable when a Grandmaster plays lightning chess, and it is sobering to reflect that when Bobby Fischer plays lightning the quality of play looks substantially better than expert chess: it looks like Master chess.

2. The current ranking U.S. chess program CHESS 4.6 when running on the CDC Cyber 176 computer examines of the order of half a million lookahead positions when choosing each move. By exploiting this stupendous advantage of brute-force calculation it is able to perform at approximately 'expert' level, with Grandmaster play, based as it is on subtle appreciation of positional values, still seemingly unattainable.

To incorporate the Grandmaster's accumulation of chess knowledge an edifice of pattern-based rules will need to be built brick by brick, just as has been done for chemical knowledge in the DENDRAL program over the fifteen years since Joshua Lederberg initiated the work. To know whether such a project for chess is possible (leaving aside whether it is desirable) we require an estimate of how many such patterns form the basis of Grandmaster skill.

Independent estimates (Simon & Gilmartin 1973; Nievergelt 1977) suggest a figure between 10 000 and 100 000. This is about a hundred times the number involved in knowledge systems such as the MYCIN system for clinical bacteriology which took no more than a couple of man-years to construct. The conclusion would seem to be that a Grandmaster chess project, although doubtless strenuous and time-consuming, would not necessarily be resistant to a determined assault. To point up the difference in heuristic adequacy between a brick-by-brick representation — which could ultimately come to occupy 10^{10} bits or more of computer memory — and a 'top-down' representation of great compactness, let it be stated that the second already exists (referred to above as the Zermelo-Borel-von Neumann theory), is well known to constitute a complete theoretical solution to the problem, and need occupy no more than 10^3–10^4 bits of store nor consume more than a few programmer-hours to write. It would, however, as pointed out by Claude Shannon (1950) take of the order of 10^{90} years' continuous running on a super-fast machine to select a move. Contemplation of this beautiful theory certainly gives us the 'Aha!' feeling about finite two-person zero-sum games with perfect information and without chance moves. It tells us little if anything about the nature of Grandmasterly mental skill. Study of brick-by-brick implementations engineered in the spirit of the modern trend of experimental epistemology might, just conceivably, tell us a very great deal.

CONCLUDING REMARKS

Whether the insights obtained from machine models by students of cognition will prove to be sparse or abundant, the process of harvesting them cannot begin until the first large lessons have been truly learned. These are:

1. Compact, algorithmic, intensive 'top-down' theories form the basis of *understanding*; that and that alone constitutes their essential purpose.

2. Their use as the basis of *skill* only makes sense for tasks of low complexity — as, to take an extreme example, the extraction of the square root, for which Newton's *tour-de-force* of concision is also a widely used

machine representation. The fact that *all* tasks attempted by machine were until recent times of low complexity in this sense, blinded the first generation of AI workers to the essential unworkability of such representations for tasks of high complexity.

3. For *complex* tasks the attempt to create skilled programs as transcriptions of intensive theories runs foul of the 'combinatorial explosion'. For such tasks, skill must, for every computing device whether protoplasmic or electronic, be built as a 'bottom-up' creation in which (to recall once more Herbert Spencer's words) 'the vital actions are severally decomposed into their component parts, and each of these parts has an agent to itself'.

REFERENCES

Ambler, A. P., Barrow, H. G., Brown, C. M., Burstall, R. M., and Popplestone, R. J. (1973). A versatile computer-controlled assembly system. In *Proc. III Internat. Joint Conference on Artificial Intelligence,* pp. 298–307. Stanford Research Institute.

Appel, K. and Haken, W. (1976). Every planar map is four colorable. *Bull. Am. math. Soc.,* **83**, 711–12.

Arbib, M. A. (1975). Artificial intelligence and brain theory: unities and diversities. *Ann. Biomed. Eng.,* **3**, 238–74.

Armstrong, J. L. and Jelinek, J. (1977). CHARD user's manual (a FORTRAN CLIP emulator). *Research memorandum MIP-R-115.* Machine Intelligence Research Unit, Edinburgh.

Barrow, H. G. and Popplestone, R. J. (1971). Relational descriptions in picture processing. In *Machine Intelligence 6* (eds. B. Meltzer and D. Michie), pp. 377–96. Edinburgh University Press; John Wiley, New York.

Buchanan, B. G., Smith, D. H., White, W. C., Gritter, R. J., Feigenbaum, E. A., Lederberg, J., and Djerassi, C. (1976) Applications of Artificial Intelligence for chemical inference XXII. Automatic rule formation in mass spectrometry by means of the meta-DENDRAL program. *J. Am. Chem. Soc.,* **98**, 6168–78.

Gregory, R. L. (1966). *Eye and Brain.* Weidenfeld and Nicolson, London.

Gregory, R. L. (1970). *The Intelligent Eye.* Duckworth, London.

Groot, A. De (1965). *Thought and Choice in Chess* (ed. G. W. Baylor), (translation, with additions, of Dutch version of 1946). Mouton, The Hague and Paris.

Hayes, J. E. (1978). Children's visual descriptions. *Cognitive Science* **2**, 1–15.

Hunter, I. M. L. (1962). An exceptional talent for calculative thinking. *Br. J. Psychol.* **53**, 243–58.

McCarthy, J. and Hayes, P. J. (1969). Some problems of philosophy from the standpoint of Artificial Intelligence. In *Machine Intelligence 4* (eds. B. Meltzer and D. Michie) pp. 463–502. Edinburgh University Press, Edinburgh; John Wiley, New York.

Michie, D. (1977). A theory of advice. In *Machine Intelligence 8* (eds. E. W.

Elcock and D. Michie) pp. 131–68. Ellis Horwood, Chichester; John Wiley, New York.
Michie, D. and Chambers, R. A. (1968). BOXES: an experiment in adaptive control. In *Machine Intelligence 2* (eds. E. Dale and D. Michie) pp. 137–52. Oliver and Boyd, Edinburgh.
Minsky, M. (1975). A framework for representing knowledge. In *The Psychology of Computer Vision* (ed. P. H. Winston), pp. 211–77. McGraw Hill, New York.
Nievergelt, J. (1977). The information content of a chess position, and its implication for the chess-specific knowledge of chess players. *SIGART newsletter* **62**, 13–15.
Selfridge, O. G. (1959). Pandemonium: a paradigm for learning. In *Mechanization of Thought Processes,* pp. 511–31, HMSO, London.
Shannon, C. E. (1950). Programming a computer for playing chess. *Phil. Mag. 7th ser.* **41**, 256–75.
Shortliffe, E. (1976). *MYCIN: Computer-based Medical Consultation.* Oxford University Press, New York; Elsevier, Amsterdam.
Simon, H. A. and Gilmartin, K. (1973). A simulation of memory for chess positions. *Cognitive Psychol.* 29–46.
Spencer, H. (1851). *Social Statics* (new edition 1955, with preface by F. Neilson).
Symons, Julian (1972). *Bloody Murder,* Penguin, Harmondsworth.
Waltz, D. L. (1972). Generating semantic descriptions from drawings of scenes with shadows. *Technical report AI TR-271.* MIT AI Laboratory, Cambridge, Mass. See also in *The Psychology of Computer Vision* (ed, P. H. Winston) pp. 19–91. McGraw Hill, New York.
Winston, P. H. (1970). Learning structural descriptions from examples. *Ph.D. Thesis, MAC TR-76.* MIT Project MAC, Cambridge, Mass.

ACKNOWLEDGEMENTS

It is right that an experimental scientist should at intervals be goaded from foot-slogging and factological concerns into presenting some more integrated account of what he is about. For me such a goad was supplied by a contribution to *Artificial intelligence: a paper symposium* published in 1973 by the Science Research Council. The author, a prominent hydrodynamicist, classified AI as a mirage and its practice as a new alchemy. For reasons of space I was then able to reply only on a pragmatic plane. The subjective need to respond in more scientific and philosophical terms found outlet only in 1976, when the Herbert Spencer Lectures Board invited me to deliver a lecture in this general area (published in 1979 by the Clarendon Press, in *Scientific Models and Man*).

I find, therefore, that I have two debts to acknowledge. The first is to the Board itself and to its Chairman, Sir Isaiah Berlin. The second is to Sir James Lighthill for the impetus to analyse the theoretical error underlying his unexpected irruption into my field.

17

High-road and low-road programs (1981)

Consider a class of computing problem for which all sufficiently short programs are too slow and all sufficiently fast programs are too large [1]. Most non-standard problems of this kind were left strictly alone for the first twenty years or so of the computing era. There were two good reasons. First, the above definition rules out both the algorithmic and the database type of solution. Second, in a pinch, a human expert could usually be found who was able at least to compute acceptable approximations — for transport scheduling, job-shop allocation, inventory optimisation, or whatever large combinatorial domain might happen to be involved.

Let us now place problem-solving by machine in the more precise mental context of evaluating two particular kinds of finite function, namely:

$$s: \quad \textit{Situations} \rightarrow \textit{Actions}, \quad \text{and}$$
$$t: \quad \textit{Situations} \times \textit{Actions} \rightarrow \textit{Situations}.$$

These expressions say that s maps from a set of situations (state-descriptions) to a set of actions, and that t maps from a set of situation-action pairs to a set of situations. The function symbol s can be thought of as standing for 'strategy' and t as standing for 'transform'. To evaluate s is to answer the question: 'What to do in this situation?'. To evaluate t corresponds to: 'If in this situation such-and-such were done, what situation would be the immediate result?'.

If the problem-domain were bicycling, we could probably construct a serviceable lookup table of s from a frame-by-frame examination of filmed records of bicyclists in action. But t would certainly be too large for such an approach. The only way to predict the next frame of a filmed sequence would be by numerically computing t using a Newtonian physics model of the bicycle, its rider and the terrain.

Machine representations corresponding to s and t are often called

heuristic and *causal*, respectively. Note that they model different things. The first models a problem-solving skill but says nothing about the problem-domain. The second models the domain including its causality, but in itself says nothing about how to solve problems in it.

The causal model partakes of the essence of the traditional sciences, such as physics. The school physics text has much to say about the tension in a string suspending bananas from the ceiling, about the string's breaking point under stress, the force added if a monkey of stated weight were to hang from a boat-hook of given mass and dimensions having inserted its tip into the bunch, and so forth. *How* the monkey can get the bananas is left as an exercise for the reader, or the monkey.

When it has been possible to couple causal models with various kinds and combinations of search, mathematical programming and analytic methods, then evaluation of t has been taken as the basis for 'high road' procedures for evaluating s. In 'low road' representations s may be represented directly in machine memory as a set of (pattern → advice) rules overseen by some more or less simple control structure. A recent pattern-directed heuristic model used for industrial monitoring and control provides for default fall-back into a (computationally costly) causal-analytic model [2]. The system thus 'understands' the domain in which its skill is exercised. The pattern-based skill itself is, however, sufficiently highly tuned to short-circuit, except in rare situations, the need to refer back to that understanding.

The distinction here spelled out corresponds roughly to that made by Rouse and Hunt between S-rules and T-rules in the context of computer-aided fault-diagnosis in complex machinery [3], for example, in automobiles. Their diagram, reproduced here (Fig. 17.1), is simple but illuminating.

The s versus t distinction has nothing whatsoever to do with the strange but widespread notion that problem-solving representations built from *causal* models are necessarily error-free, proved so by their implementers, and thus in some important sense 'sound', while *heuristic* models are by their nature tainted with unbounded and unquantifiable error. In actuality formal proofs of correctness are no less obtainable for heuristic models [4,5] than for models of other kinds, provided that the domain is such as to sustain precise mathematical reasoning at all. The only problem-solving device yet to achieve a good and versatile record (the expert brain) has been shown to proceed at 'run-time' overwhelmingly by the low road. Moreover, knowledge engineers are beginning to find in one domain after another that almost all the skill comes from the S-rules and almost all the implementational and run-time costs from the T-rules.

Perhaps this discovery should not have taken people by surprise in quite the way it seems to have done. After all it had already been noted that when a Fischer or a Karpov plays lightning chess (S-rules only, no time for anything else) he can still hold his own against an ordinary Master who is allowed all the time in the world for search and reasoning.

In real-world domains no more complex than chess, insistence on 'high road only' has usually led to solutions which are

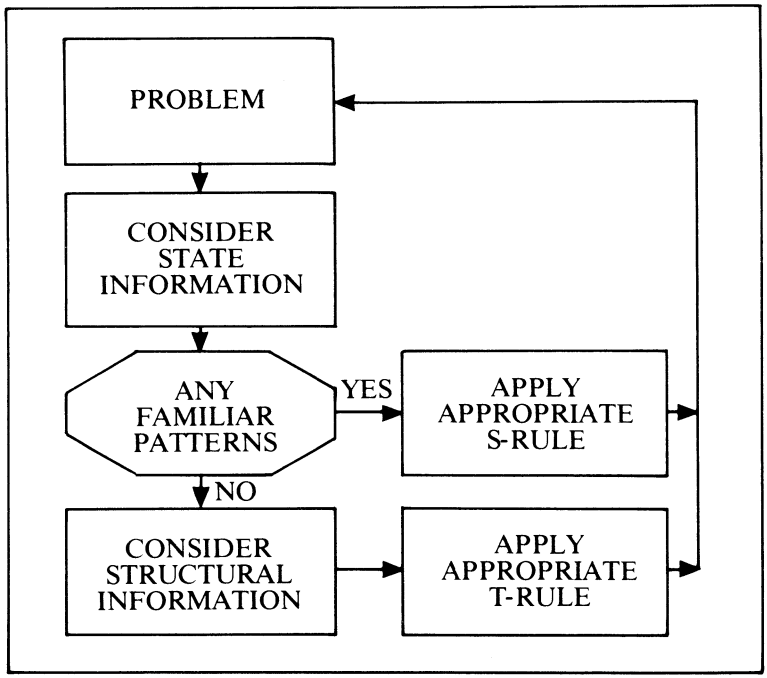


Fig. 17.1 — Overall structure of the model used by Rouse and Hunt. There are really two models, so arranged that (as in the system of Pao *et al.*) the system's 'science' acts as default for its 'craft'. Compare Fig. 4.7 of Chapter 4, where 'interpretative models' and 'predictive models' correspond respectively to the familiar patterns' and 'structural information' of the present Figure.

— opaque to the user, and
— unbelievably costly at run time.

Someone says: 'I need to build an expert problem-solver, but I don't buy heuristic production-rule models. How do I know that they are *correct*, or with proved *error bounds*?'.

He could equally say: 'I need to make an omelet, but I don't buy eggs. How do I know that they are *not addled*?'. The answer can only be: 'Get your eggs certificated; or at the very least buy from a reliable farm. If you don't want to do that, then you'll have to lay them yourself'.

REFERENCES

[1] D. Michie (1977) A theory of advice. In *Machine Intelligence 8*, (eds E. W. Elcock & D. Michie). pp. 151–168. Chichester: Ellis Horwood, and New York: Halsted Press (John Wiley).

[2] Y.-W. Pao, W. L. Schultz, S.-Y. Oh & R. G. Fischer (1980) A knowledge base engineering approach to power systems monitoring and

control. *Technical Report.* Cleveland: Department of Electrical Engineering, Case Western Reserve University.

[3] W. B. Rouse & R. M. Hunt (1980) A fuzzy rule-based model of human problem solving in fault diagnosis tasks. *Working paper.* Urbana: Coordinated Science Laboratory, University of Illinois.

[4] I. Bratko (1978) Proving correctness of strategies in the AL1 assertional language. *Inf. Proc. Letters,* **7**, 223–230.

[5] T. B. Niblett (1981) A provably correct advice strategy for the end-game king and pawn versus king. In *Machine Intelligence 10,* (eds D. Michie & Y.-H. Pao).

18

Measuring the knowledge-content of expert programs (1982)

The theory of what computers cannot ever do, sometimes referred to as intractable problems, can be used to define a class of 'semi-tractable' problems. Such problems are intractable to programs of standard form but accessible to heuristically-based programs. Restriction of the latter to human bounds of storage and calculation yields a subset of semi-tractable problems which we term 'reducible'. This property will be used to explicate an informal notion familiar to problem solvers, namely the subjective difficulty of a problem domain.

INTRACTABLE PROBLEMS

In recent years, certain problems have been show to be intrinsically intractable. Although known to be solvable in finite time, they will never be solved in the life-time of our planet, regardless of how computer technology may develop, how clever people become, or how many resources are committed to the project. Knuth [1] cites an example of Meyer and Stockmeyer where the problem is to decide whether statements expressed in a restricted logical language about the whole numbers are true or false. Sample statements from the easy end of the spectrum are to the effect that every pair of numbers two or more apart has at least one number in between, or that every non-empty set of numbers has a smallest element.

It was already known that the truth-value of any well formed statement of this language can be determined in a finite number of steps. Meyer and Stockmeyer envisaged evaluation of each input statement by an electrical network, this being the design for fastest evaluation. They proved that for input statements of 617 symbols or longer every such network must use at least 10^{125} components. This number is much greater than the number of protons and neutrons in the observable Universe.

SEMI-TRACTABLE PROBLEMS

Translation of this type of result into algorithmic computation on a sequential digital machine yields the conclusion that every space-minimal representation of such an algorithm (as a shortest program) is time-infeasible and every time-minimal representation (as a giant look-up table) is space-infeasible *and every store-time compromise between the two is either time-infeasible or space-infeasible or both*. The sting is in the tail italicised to bring out a distinction between genuinely intractable problems, which will never be solved, and a special kind of problem for which I have elsewhere used the term 'semi-hard' [2] but have subsequently adopted 'semi-tractable'. Algorithmic solution on a sequential machine of problems in this latter class has the property that every space-mimimal representation (as a shortest program) is time-infeasible and every time-minimal representation (as a giant look-up table) is space-infeasible *yet representations exist which are both time-feasible and space-feasible*.

Solutions for semi-tractable problems can thus be feasibly implemented, but only by exploiting the store-time trade-off. Feasible solutions to such problems require additional store for incorporation of heuristics into the program. When the latter take a human-oriented form which we recognise as *domain-specific knowledge* we have a program of a kind generally referred to as knowledge-based, or expert [3]. In knowledge-intensive domains such as medical diagnosis, chess or mathematical reasoning, the human advantage does not rest on superior 'hardware'. Informational measurements on the transactions involved in cognition suggest that although the human achievement is strong, the equipment is relatively weak [4,5]. A semi-tractable problem, feasibly computable by heuristically based programs on fast machines, may or may not be humanly solvable. For human-solvability the problem must possess a further property of *reducibility*. This is illustrated in Fig. 18.1 for three hypothetical functions, one intractable and two semi-tractable. Of these last two, one is reducible and the other not. Semi-tractable sub-domains can be found within intractable domains, for example, mass spectroscopy and chess. Within these semi-tractable sub-domains lie reducible sub-sub-domains.

Gregory Chaitin [6], in an outline of 10 years of mathematical work by Solomonoff, Kolmogorov and himself, uses 'complexity' for a certain property of a sequence of binary digits. His usage is not related in any simple way to the 'complexity-theory' connotation of the same word. We shall accordingly introduce prefixes and rename Chaitin's quantity 'α-complexity'. To understand Chaitin's sense, first fix on some abstract computing machine. Then for each given binary sequence ask the question 'What is the shortest sequence which, when run as a program on the said machine, will reproduce the original sequence?' *The length of this shortest representation is the original sequence's α-complexity*. The ratio of the first to the second length measures the sequence's absolute compressibility. If we restrict ourselves to sufficiently long sequences, as explained in Chaitin's outline, the choice of machine can be disregarded. In essence we are considering a

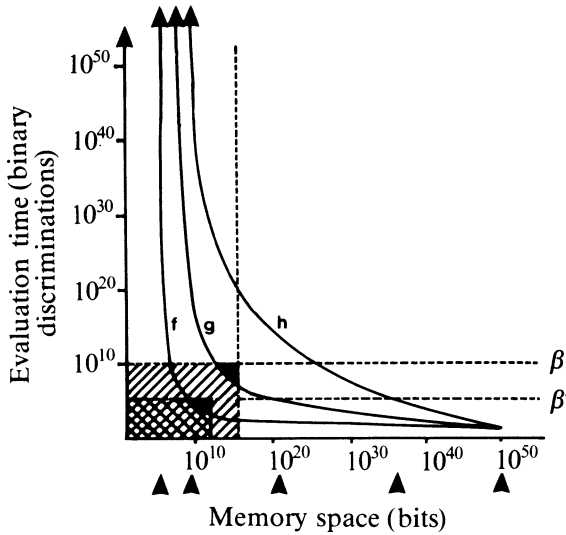


Fig. 18.1 — Store-time trade-off curves for hypothetical finite functions f, g and h. Each has the same information-content (10^{50} bits) and the same α-complexity (10^4 bits). Time-feasibility (β) and space-feasibility limits for machines have somewhat arbitrarily been placed at 10^{11} time-bits and 10^{15} store-bits, respectively. Time-feasibility for human evaluation of the same functions is set at 10^4 time-bits indicated by β′ and space-feasibility at 10^{11} store-bits. The hatched rectangle is the 'zone of feasibility', through which curves f and g pass. Only f passes through the cross-hatched 'zone of human feasibility'. The five upward arrows at the base-line mark, respectively: the α-complexity of f, g and h; the β′-complexity of f; the β′-complexity of g; the β′-complexity of h; the information content of f, g and h. f is semi-tractable; it is also *reducible*, since its β′-complexity is less than the human store-bound. This property offers, without guaranteeing, the possibility of human mastery of f as a problem-solving domain. g is also semi-tractable, but not reducible. h is intractable, and *a fortiori* not reducible. A useful quantity to keep in mind is the maximum bit-rate of human mental calculation, usually taken to be equivalent to about 20 binary discriminations per second. In setting time-feasibility equal to 10^{11} we assume the availability of machines capable of calculating ten million times faster than the brain, say 200 million binary discriminations per second. On this basis it takes the same length of time to wait for a machine to execute 10^{11} binary discriminations as it does for a human solver to execute 10^4.

property intrinsic to the sequences themselves, a property of 'randomness'. A random sequence has an α-complexity not materially less than its own length — no compressed representation exists.

INFORMATION-CONTENT OF A FUNCTION

Suppose that the original sequence is a Shannon-minimal encoding of the extensional form of a finite function f: $X \rightarrow Y$, *i.e.* of a look-up table of pairs (x_1, y_1), (x_2, y_2), (x_3, y_3), ..., (x_N, y_N) where N is the size of f's domain. For sufficiently large X, information theory allows us to equate the length of this sequence in bits to the information-content of f, calculated as:

$$I(f) = -\sum_{i=1}^{N} \log_2 p(y_i)$$

where $p(y_i)$ is the frequency with which the value y_i occurs as a right-hand element in f's function-table. The formula sums over the entire sequence the individual 'surprisals' [8] of successive symbol-occurrences in the sequence, and is equivalent to the more familiar-looking

$$N \times \quad -\sum_{j=1}^{M} p(y_j) \log_2 p(y_j) \quad .$$

The latter first derives the *average* surprisal per symbol-occurrence in the message by summing over the complete alphabet of M symbols, and then multiplies by the sequence-length N to get the total information-content of the sequence.

DIGRESSION ON INFORMATION

The theory of information has classically been concerned with infinite sequences and their steady-state properties. Objection may consequently be raised that the information-content of *finite* messages is not well defined in the theory. Steady-state problems, however, represent a particular specialisation of a formalism which has a more general interpretation. We exploit this generality by using the surprisals associated with successive symbols of a message as the central concept. Of each symbol in turn we ask:

> 'How surprised would a rational receiver be on receipt of this next symbol?'

'Rational' is here used in the sense of the 'rational belief' of Bayesian probability (see reference [9]), and surprisal is defined as $-\log_2(p_i)$ for the ith symbol, where p_i is its prediction-probability. This scheme in effect parameterises relevant properties of source, message and receiver in such a way that any behavioural properties of source and receiver whatsoever, whether prediction be statistically or logically based, can be postulated. It makes no difference whether messages are finite or infinite. The basic information expression can always in principle be computed to yield the expected information-content of the next symbol to be received, as

information (*source*, *symbol-string-received-so-far*, *receiver*).

The last argument-place is reserved for a specification of the receiver's computational capability and degree of access to the values of the other two arguments. The condition for zero expected information is perfect prediction: namely, access is total and computational capability is complete. The

last argument can be 'frozen' in the manner of Schoenfinkel (POP-2 'partial application' [10]) to some given value, to create an information function of two arguments specialised to some particular receiver. Thus, corresponding to receivers 1, 2, 3, ... we create function-procedures *information*$_1$, *information*$_2$, *information*$_3$, ..., all of which may give different answers to any given information problem. Some of these functions will correspond to Shannon's first-order, second-order, etc. information-measures, but others will not, being responsive to logical as well as to statistical regularities of the message. This treatment has the merit of removing obscurity from such questions as: 'How much information is conveyed by receipt of the Nth digit of the decimal expansion of π?'. There is a reminiscence here of Good's [11] notion of 'dynamic probability'.

α-COMPLEXITY OF A FUNCTION

Another interesting property of f is its α-complexity $L_\alpha(f)$ in a natural extension of the Chaitin sense: the length in binary digits of a minimal program for computing f. Solomonoff [7] relates the compression achieved when L_α/I is small to the notion of an explanatory theory in science. He likens the original series of binary digits to a scientist's observations and the program to a theory which enables him to 'understand' them. Using a form of Occam's razor, he says that if different programs p_1, p_2, p_3, ... all reproduce the original sequence, then we should take the shortest as the preferred explanatory theory. The shorter the program, the more comprehensible and elegant the theory. α-complexity can thus be seen as a measure of the extent to which the given sequence can be given a scientific description.

β-COMPLEXITY OF A FUNCTION

But explanation is not the only use of a theory. There is also its use for prediction and control. This brings in the idea of applying it, either by running it 'in the head' or on a computing machine. When measured against the criterion of actual computation a snag appears in Solomonoff's scheme. For *explanation* we do indeed want the shortest program, but operationally we want something quite different, namely the shortest program able in the worst case to evaluate $f(x)$ within a user-acceptable number of steps. In the context of machine computation we denote this bound by the symbol β and define f's β-complexity as the length of the operationally minimal program. To explicate an operational interpretation of Solomonoff's requirement to the effect that the user must be able to make mental application of the theory in addition to simply understanding it or running it on the machine, we replace β with a parameter specialised in its numerical range to the rates of calculation possible for the brain rather than to those of computers. For this we use the symbol β′ and in place of Solomonoff's rule we substitute an ordering of alternative programs based on their β′-complexities rather than

their β-complexities or their α-complexities. We again apply Occam's razor to theories of f, by selecting the shortest of these alternative programs.

Once we leave α-complexity where running time is no object, informational and structural properties of the machine on which the programs are to be run are needed for fully specifying β-complexities. This is so whether we are concerned with abstract machines or with practically implemented electronic hardware or with biologically implemented neuronal 'hardware'. In an adequately general formulation, specification of a finite function's β-complexity is a function of three arguments, thus

$$\beta\text{-}complexity\text{:}\ F \times S \times B \rightarrow N^{+},$$

where F is the class of finite functions, S is the class of machine-specifications and B is the class of possible time-limits for the decoding computation. If we allow unrestricted computation time and restrict F to functions requiring sufficiently long programs, the S argument can be ignored. Then for functions defined over sufficiently large domains we have

$$\alpha\text{-}complexity\text{:}\ F \rightarrow N^{+},$$

as in the Kolmogorov-Chaitin scheme.

If we choose from B a bound to calculation-length appropriate to human solvers working to humanly acceptable waiting times, and specialise S to an s with information-processing properties similar to those of the human brain, then as a convenient explication of the intrinsic difficulty of a problem to a human we have

$$\beta'\text{-}complexity\text{:}\ F \rightarrow N^{+}.$$

To achieve mastery of the evaluation of some f, a human must pack a certain amount of material into his head. The β′-complexity of f sets a lower bound to this amount.

If we choose from B a bound to calculation-length appropriate to a high-speed machine working to a humanly acceptable waiting time, and specialise S to some physically realisable s_i, where s_i might for example be the Cray-1 supercomputer, then we have

$$\beta_i\text{-}complexity\text{:}\ F \rightarrow N^{+}.$$

In general we speak loosely of a given f's β-complexity, having in mind some general class of 'fast machines of the day' for order-of-magnitude decisions concerning tractability.

Returning to Solomonoff's search for models of scientific theories, it seems that different theories, and different parts of the same theory, are developed in science to perform different services.

(1) A purely explanatory theory, uncontaminated with the necessity for its use for prediction, does indeed have its length low-bounded by f's α-

complexity as Solomonoff proposes. The smaller this is, the simpler in general for a human brain to comprehend.

(2) A theory which a scientist may wish to test using high-speed computation has length bounded by f's β-complexity.

(3) A theory which is to be applied to test cases 'in the head' is not to be assessed either on α-complexity or β-complexity, but on f's β'-complexity. The smaller this is, the simpler the theory which can be found for a human brain to apply for prediction.

COMPREHENSION VERSUS PREDICTION

To exemplify these ideas, consider the following series of observations: (19, true), (199, true), (1999, true), (19 999, false), (199 999, true), (1 999 999, true), (19 999 999, true). Here is a theory, of "primeness", to explain them:

> *$f(n)$ is true if n is greater than* 1 *and if for all m greater than* 1 *and less than n, m does not divide n; otherwise $f(n)$ is false.*

The above is clearly satisfactory as an explanation, or 'comprehension theory'. As a 'prediction theory' it is a fiasco. This is immediately discovered if one interprets it as a program and tries some numerically large inputs. Suppose that we try the program on $f(x)$ where $x = 2^{128} - 1$, a 39-digit decimal number. Even at the rate of one million divisions per second it will take about 2000 years to discover the smaller of x's two factors, a 17-digit number. In 1970, however, Brillhart and Morrison performed the factorisation at the expense of only about $1\frac{1}{2}$ hours of computer time using, in Knuth's words, 'a combination of sophisticated methods, representing a culmination of mathematical developments which began about 160 years earlier'.

Now suppose that they had embodied enough of this mathematical knowledge in the routines and data-structures themselves to make the operation fully automatic, rather than the machine-aided paper-and-pencil approach which they in fact followed. Such a program would represent a formidable accomplishment in machine intelligence. It would also constitute a theory of primes quite different in nature from the simple 'explanatory' program given earlier. Essentially the distinction corresponds to the antithesis set up by McCarthy and Hayes [12]: an *epistemologically adequate* representation of a problem-domain contains all the facts logically required for solving all solvable problems of that domain; a *heuristically adequate* representation contains everything required for solving these problems within practical resource-bounds. A natural further step, implicit in our notion of reducibility, is to separate out *cognitively adequate* representations as a special case of those which are heuristically adequate. *A problem-domain is reducible if and only if it has a cognitively adequate representation.* Equivalently it is reducible if and only if its β'-complexity is less than the quantity of human memory which can be loaded with such material in a life-time.

We have considered above one epistemologically and one heuristically adequate representation of the prime-tester function. A similar pair for chess would be the total look-ahead algorithm of Borel and von Neumann on the one hand, and, on the other, the program to play faultlessly which chess programmers would like to write, but will never be able to test if chess turns out to belong to the intractable category. Note that this would correspond to the case that chess, which has high α-compressibility (see earlier), turns out to have low β-compressibility. To give an intuitive flavour of what property of a problem-domain is denoted by its β-compressibility, we can informally equate it to the ratio of its size to the terseness with which a feasibly computable solution-strategy can be specified for the domain. It is convenient to measure the amount of processing not in seconds but in units which are independent of the intrinsic speeds of different devices. For this reason in Fig. 18.1, we express β in terms of the number of binary discriminations performed in the course of evaluating f. In a practical context, the user will choose β as the product of the worst-case acceptable waiting time and the known speed in bits per second of the evaluation device. The idea then is that of a *minimal heuristically adequate* program, *i.e.*, the shortest program capable of β-evaluating f for every argument in its domain. This shortest length is f's β-complexity as earlier defined.

To recapitulate, α-complexity measures how difficult a domain is 'in principle', *i.e.*, how much memory would be required by a solver allotted an arbitrarily large solving time. β-complexity is expressive of a problem's practical machine difficulty, and measures the memory needed if the given device is to complete every evaluation at the expense of no more than β binary discriminations. β'-complexity is expressive of a problem's difficulty to the human solver.

Having fixed on a value for β, we can say that if the β-complexity of f is larger than the bit-content of any feasible store (function h in Fig. 18.1), then the problem is β-intractable: if this is so under any reasonable choice of β then no heuristically adequate representation can exist and the problem is intractable without qualification. As earlier implied, the possibility is open that chess is intractable. We can, however, be sure that chess is at least β'-intractable. A negligibly small proportion of all positions in master games can claim master consensus as to whether they are game-theoretically won, drawn or lost.

KNOWLEDGE AND ADVICE

For a problem such as chess, compact solution algorithms exist, yet heuristic adequacy can only be attained by increasing the occupancy of store by large factors. What does this additional material consist of? The general answer is '*heuristics*', remembering that a theorem, too, can be used as a heuristic. A heuristic is any addition to store which increases the total quantity of realisable information about f. A *useful* heuristic is one for which (assuming

all f-evaluations in X to have equal utilities) the increase of realisable information exceeds the increase of store-occupancy. If it is the other way round, then the heuristic is in this localised context *worse than useless*.

If the heuristic rules, patterns, descriptions, etc. take certain special forms which are humanly recognisable and usable as concepts, then we call these structures *advice* [2]. In this style the store is strictly partitioned (as in Fig. 18.2) into a fixed algorithmic part and an incremental advisory part,

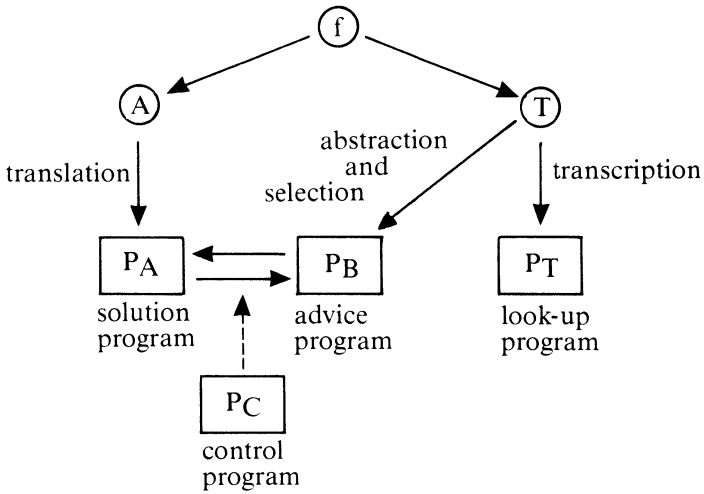


Fig. 18.2 — Relations among various kinds of mathematical and computational objects. Abstract objects are ringed, concrete are boxed. A and T denote two contrasted abstract representations of the function f, namely as an evaluation algorithm and as a function table (ordered set of pairs), respectively.

plus a small part consisting of fixed control routines which make the advice accessible to the algorithmic part. 'Advice' is thus a special category of heuristic.

The role of heuristics is to facilitate the operation of the original naive algorithm in such a way as to increase the system's total realisable information about f, *i.e.*, the information content of that part of f which is β-evaluable by the system. When and only when the facilitating heuristics take the form of advice, then realisable information may be referred to as a program's *knowledge* about f. It corresponds to a special case of the program's ability to answer questions of the form 'What is the value of $f(x)$?'. Note that question-answering ability cannot in general be identified with knowledge. A billion-entry database, or a Martian, or a number-crunching super-computer might all perform impressively as information sources: none of these would qualify as a knowledge source. 'Knowledge' is thus a

special kind of information. Heuristics increase an expert program's realisable information content. Advice increases its knowledge content. How can we measure this?

COST-BENEFIT OF PROGRAM COMPONENTS

The realisable information contained in a given program can be measured in information bits, as follows.

Consider the partial function f_k corresponding to that subset of f which is β-evaluable by the given program running on the given machine. f_k is thus defined for a 'realisable' subset X_k of f's domain. How much information is required to specify f_k? Following our earlier reasoning it is

$$K = -\sum_{x \in X_k} \log_2 p(f(x)) \text{ bits}$$

plus the information associated with the N 'Yes/No' decisions involved in the selection of this particular subset X_k out of X.

Writing L for the store-cost of the program in binary digits we have a benefit-to-cost ratio K/L. We call this the program's *computational advantage*. For some purposes the logarithm of this ratio, $D = \log_{10} K - \log_{10} L$, is convenient. We call it the program's *penetration*, and note that the measure of a good heuristic is that it increases this quantity.

The main use of the theory is not for comparing cost-benefit measurements on whole programs with similar measurements performed on other whole programs. The pay-off comes from detailed examination of the fine structure of a given program's heuristic part — this rule *versus* that, this table *versus* the other, this heuristic *versus* none at all. How do we do this?

A body of advice has its proper values K_B and L_B determined by measuring K and L without the advice loaded and again with it loaded and doing the appropriate subtractions, according to the relations:

$$K = K_A + K_B + K_C$$
$$L = L_A + L_B + L_C.$$

The subscripts A, B and C stand for 'algorithm', 'body of advice' and 'control', respectively. Incremental gains in K_B and L_B associated with additions to B can be measured in the same way. The validity of the assumption of additivity and independence underlying the above-described procedure may be criticised. Thus the goodness of a heuristic may be annulled, or even reversed in sign, by the presence in store of certain other heuristics. The prevalence and magnitude of such interactions will vary from

one domain to another. An approximating assumption of this sort has to be approached empirically, as with the treatment of 'weights of evidence' in the Turing–Good [9] Bayesian calculus of uncertain inference and also the zero-interaction assumption implicit in various procedures in the Fisherian analysis of variance.

LEARNING

Refinements for which there is not space here allow the economics of knowledge-acquisition (learning) also to be monitored. Learning indices are defined in terms of gain in penetration relative to costs of learning. In Watterberg and Segre [13] this finer analysis showed that a particular rote-learning mechanism was ceasing to be cost-effective for larger dictionary sizes. Such warning signals, which could not have been obtained without an appropriate measurement method, can be used to prompt program improvements — for example, by introducing rote-learning into key subroutines as well as into the main program, or augmenting rote-advice with conceptualised knowledge in the form of descriptions, pattern-based rules or useful lemmas.

Advice Theory is intended as a measurement tool for the information engineer. It stands or falls by application to practical knowledge-based programs for domains so complex that human brains hitherto have constituted the only available evaluation mechanisms. Machine representations of chess end-game knowledge are under study for initial validation of this formalism. With its use a comparative cost-benefit accountancy was recently performed on the inductive learning and execution by machine of variant operational theories of the king-pawn-king ending in chess [14].

REFERENCES

[1] Knuth, D. E., *Science*, 1976, **194**, 1235–42.
[2] Michie, D.,'A theory of advice', in Elcock, E. W., and Michie, D. eds, 'Machine Intelligence 8', Chichester, Ellis Horwood: New York, John Wiley, 1977, pp. 151–168.
[3] See, for example, Michie, D., ed., 'Introductory Readings in Expert Systems', London and New York, Gordon and Breach, 1982.
[4] Stroud, J. M., *Ann. of the N. Y. Acad.*, 1966, 623–621.
[5] Miller, G. A., *Psychological Review*, 1956, **63**, 81–97.
[6] Chaitin, G. J., *Scientific American*, 1975, **232**, 47–52.
[7] Solomonoff, R., cited in reference [6].
[8] Sampson. E. W., 'Fundamental natural concepts of information theory', *Report E*5079. Air Force Cambridge Research Station, 1951.
[9] Good, I. J., 'Probability and the Weighing of Evidence', London, Griffin, 1950.
[10] Burstall, R. M., Collins, J. S., and Popplestone, R. J.,'Programming in POP-2', Edinburgh, Edinburgh University Press, 1971.

[11] Good, I. J., 'Dynamic probability, computer chess and the measurement of knowledge', in Elcock, E. W., and Michie, D., eds. 'Machine Intelligence 8', Edinburgh, Edinburgh University Press, 1977, pp. 139–150.

[12] McCarthy, J., and Hayes, P. J., 'Some philosophical problems from the standpoint of artificial intelligence', in Meltzer, B., and Michie, D., eds, 'Machine Intelligence 4', Edinburgh, Edinburgh University Press, 1969, pp. 463–502.

[13] Watterberg, P., and Segre, C., Knowledge measurement, *CS397DM, Group Report* (described in reference [2]), 1976.

[14] Jackson, J., and Michie, D., An elaboration, using advice theory, of the Shapiro–Niblett work referred to in Shapiro, A., and Niblett, T. B., 'Automatic induction of classification rules for a chess end-game', in Clarke, M. R. B., ed. 'Advances in Computer Chess 3', Oxford, Pergamon Press, 1982, pp. 73–92. See ref. [13] at the end of the next chapter.

19

Automating the synthesis of expertise (1984)

Automating the construction of machine-interpretable knowledge-bases is one of the immediate next moves in the emerging technology of information. Feasibility of computer induction of new knowledge from examples has been shown in more than one laboratory. Can we generate knowledge-based programs that are automatically guaranteed analysable and executable by machine and human brain alike?

INDUCTIVE LEARNING

A number of computer programs have been developed which 'learn' by generating rules from examples, *i.e.* by induction (see Michalski & Chilausky [1]; Quinlan [2,3]; Shapiro & Niblett [4]; Paterson & Niblett [5]; and papers in Michalski, Carbonell & Mitchell [6]).

As a result of a long sustained interest in machine learning [7], our Edinburgh group has recently become a practised computer induction shop, along with I. Bratko's Artificial Intelligence division of the Josef Stefan Institute, Yugoslavia, R. S. Michalski's group in the University of Illinois, and J. R. Quinlan's laboratory in the New South Wales Institute for Science and Technology, Sydney. We can now routinely synthesize machine-executable descriptions (in the form of PASCAL, FORTRAN, PROLOG or C programs) for classification tasks too complex for human experts either to program, to verbalize complete strategies for, or even reliably to perform. According to the style adopted, the machine-generated descriptions can be made to take either 'Martian' or human form. In the second case, in spite of its synthetic origin, we call a machine-made description a concept-expression. A recently-developed programming technique [8] has added the following capability: when a machine-made concept is 'run' on a trial case, the system not only classifies the case but also displays an explanatory narrative in intelligible English as to how it arrived at the classification —

something that cannot always be done by the possessor of the equivalent human-made concept. This is a help both to the end-user and to the knowledge engineer.

A brief description of methods and principles used by one of the Edinburgh learning programs, ACLS, is given below. The program is a development from Quinlan's ID3, in turn derived from Hunt, Marin and Stone's CLS (Concept Learning System) [9]. A commercially enhanced version of ACLS has also been described by R. McLaren [10].

ACLS (ANALOGUE CONCEPT LEARNING SYSTEM)

A 'training set' of examples of classifications within a chosen field is presented to ACLS as a list of records. All fields of the record except the last are entered with the values of those attributes considered by the user to be relevant to the classification task. These are known as 'primitive attributes'. The last field is entered with the name of the decision class to which the record is to be assigned (*e.g.* CIRCLE, ELLIPSE, TRIANGLE, SQUARE, POLYGON, OTHER; or MALIGNANT, BENIGN, IMAGINARY; or ALLOWED, DISALLOWED, SPECIAL-CASE; or just TRUE, FALSE).

From these example records ACLS derives a classification rule in the form of a decision tree, branching according to values of the attributes situated at the nodes. ACLS also generates a PASCAL conditional expression logically equivalent to the decision tree, and this PASCAL code can be run to classify new test examples. Whenever a new example is found which refutes the current rule, ACLS can be asked to restructure the rule so as to accommodate the new case, and to display, store or output it as before.

The set of attributes should be chosen by an expert in the given problem domain to be sufficient for classifying the data. In choosing the set of attributes the maxim is: if in doubt, put it in . Any attribute which is in reality redundant will be set to one side by ACLS and not included in the tree (and corresponding PASCAL code) which it generates.

It is not necessary that the expert should be armed with a clear mental picture of the rule which he himself uses when performing the given task.

His role is simply to structure the problem into sub-problems, to supply the list of primitives and to act as an oracle by assigning example data to what he considers to be their correct classes. ACLS observes his behaviour as a classifier, and from this constructs the simplest rule it can, using the primitives supplied, which will assign the same example data to the same classes as he does. As the training set grows with the addition of fresh examples, so the ACLS-synthesized rule grows more sophisticated. As it does so it tends to approximate to a form on which the expert may comment: 'That looks like the way that I think I probably do it'.

The expert is thus enabled to transfer to the machine a judgemental rule which he already had in his head but has not explicitly formulated.

We list below four closely related questions under investigation with the aid of ACLS and similar programs:

(1) the possibility that a learning program, such as the one described above, can be used to extend the mental grasp of the human user;
(2) the use of the machine-synthesized programs as instructional or reference texts;
(3) numerical measurement of the knowledge (see later) contained in machine-synthesized programs;
(4) development of human-intelligible concept description languages as mandatory vehicles for output from induction systems.

For a number of reasons, chess is specially qualified as a test domain for experiments of this kind. However, in order to confirm the generality of observed results, after basic work has been done in this measurable 'test-tube' world, ACLS-type programs can be used to investigate knowledge enhancement in the domain of school algebra, robot vision, industrial engineering, fault diagnosis, chemistry, etc. A general idea of how 'programming by examples' is done is given in a recent paper by Shapiro & Michie [8].

KNOWLEDGE MEASUREMENT

Tests using an information theory approach to measure the knowledge-content of machine-produced (synthetic) programs are under way in Edinburgh [11,12] and Illinois [13]. Relevant theoretical work has also been conducted at Stanford [14].

Use is made of machine-oriented definitions, such as the 'knowledge-content' of a program, 'computational advantage', 'penetration', 'grasp', and rate of knowledge acquisition, 'Difficulty' of a problem can also be expressed within the same numerical calculus. The current objective is to build the measurement system referred to above into rule-synthesis programs. Induction programs will then be enabled not only to produce rules which can be executed by machine but at the same time will be able to give the user a quantitative evaluation of each rule as it is synthesized — in terms both of its information content and of its anticipated suitability as a human-usable concept. This style of proceeding can be exemplified from recent work of Shapiro [15]. Shapiro's synthetic rules combined a reasonable approximation to machine optimality with brain feasibility. No general guarantee can be given that the two goals can be combined in any particular case. Moreover, from the point of view of brain feasibility, the problem of program transparency has been found to have some surprising twists.

SUPER-PROGRAMS

Recent results have shown that programs constructed by systems such as Quinlan's ID3 can be, in one sense, 'super-programs' and at the same time quite incomprehensible to people. By a super-program we mean one that is at least twice as efficient in terms of execution cost as the best which could be written by a programmer. For an example we turn back to Chapter 14.

Fig. 14.8 shows the execution costs on a CDC Cyber 72 for three different programs written to solve the same difficult problem. The first implements a standard algorithmic approach. The second was the fastest pattern-coded solution which J. R. Quinlan, the author of ID3 and also an outstanding programmer, could achieve after many months of work. About two months of these were expended on finding an adequate set of primitive descriptors for the problem. The third was a machine-generated program, produced by ID3 itself, equipped with the file of primitive descriptors and a complete file of example data. ID3 running on the Cyber 72 generated this decision-tree program in 34 seconds. The ID3-synthesized program clearly qualifies as a super-program (see Fig. 14.8). Further and perhaps alarmingly, however hard they try, chess experts cannot understand it. Even though it constitutes a complete and correct description, it does not qualify as a *concept expression.*

Work triggered by this observation has shown the reason, which lies in the way the decision tree generated by ID3 (or the corresponding PASCAL expression) is structured. We have therefore developed a method known as 'structured induction'. This is a hierarchical approach in the sense that the rules, or concept-expressions, generated by the program using the original primitive attributes themselves become the primitives for the next pass through the data (see Shapiro and Niblett [4]). Shapiro subsequently applied the approach to the ending:

king and pawn *versus* king and rook (pawn on a7).

This work generated a complete and self-documenting classifier for a space of over 200 000 positions, using some 350 expert-supplied examples. Since human experts are unfortunately not self-documenting, no codified classification theory of the domain previously existed. The classifier thus constitutes an original if miniscule contribution to end-game theory.

The method described constrains the output of the inductive generator to take the form of decision structures which people can recognize as concepts. The constraints have been embodied in the syntax of a rule language CDL (Concept Description Language 1), for which a compiler is operational on the VAX-750. A new version, is under development which enables the user to define 'fuzzy' as well as strictly logical building blocks for his synthetic concept.

A KNOWLEDGE REFINERY

Expert systems have commercial promise. A system of over 2000 rules called R1 is used at Digital Equipment Corporation for configuring computer systems to customers needs (see McDermott [16]). It already outperforms their best technical salesmen. But in *knowledge refining and synthesis*, the focus is not on the product which expert systems were originally designed to deliver, namely interactive advice in conversation with a client, but on the unexpected by-product, the finished knowledge-base itself. Expert system languages and induction systems can be used:

to get knowledge into the machine;
to test it;
to debug it;
to fill gaps;
to extend it;
to modify it.

Finally the knowledge can be put back into the human world in *unrecognizably improved shape.* This important phenomenon was shown by Bratko to be generalisable across different knowledge domains using one and the same computer induction program. In the differential diagnosis of lymphatic cancers Bratko successfully used an Edinburgh learning program derived from Ross Quinlan's ID3 to generate an improved classificatory scheme for this category of malignant disease. The Edinburgh program had been developed by Shapiro and Niblett [4] using the domain of chess. Strategies for chess end-games were conveyed by examples. The resulting correct and complete theory of king and pawn against king is itself of interest to chess masters. Bratko then used the same program, in collaboration with clinical oncologists, to construct a diagnostic scheme for the lymphatic cancers. In this experiment, with help only in extracting data from medical case histories, Bratko was able to perform as if he were a clinical expert, although knowing nothing about medicine [17].

REQUIREMENTS FOR A REFINERY

What are the necessary ingredients for knowledge refining?

(1) Knowledge-engineering software able to make inferences from data supplied, and to retrace and display the lines of reasoning.
(2) Induction modules able to generate rules from examples.
(3) A good software development environment, *e.g.* UNIX or INTERLISP.
(4) Trained knowledge engineers familiar with the above tools.
(5) One or more experts. Even with inductive knowledge generation automated to the limit of current technique, an expert is still needed to choose the repertoire of primitive measurements or attributes to evaluate as relevant to the given problem domain, and to assist the knowledge engineer in the task of decomposing the domain into sub-problems, sub-sub-problems, etc. This top-down analysis is the heart of the structured method of induction referred to earlier.

Originally designed to deliver interactive advice, it now appears that expert systems have a by-product which is in the long term more important — the finished knowledge-base itself. A concluding example may be useful, taken from a recent study by Mozetic, Bratko, & Lavrac [18].

Using the logic programming language PROLOG, the authors collaborated with senior clinical cardiologists at the Ljubljana University Medical

School. The Yugoslav group applied a generalisation of the 'easy inverse' method [19] to the machine construction of a complete and ultra-reliable diagnostic scheme for multiple arrhythmias and their relation to the ECG wave form. The new knowledge, although small in extent, is sufficient to have a use in teaching and as a reference text for the specialist (see next chapter).

COMPUTERS AS CO-AUTHORS

To recapitulate, expert systems can be used:

to get knowledge into the machine;
to test it;
to debug it;
to fill gaps;
to extend it;
to modify it;

and finally

to put the knowledge back into human hands in an improved form.

Thus, the possibility now exists of superseding (as the automobile has superseded the horse) a craft which has been in existence for thousands of years — namely, the writing of manuals and texts on how to do things.

REFERENCES

[1] Michalski, R. S., & Chilausky, R. L. Knowledge acquisition by encoding expert rules *versus* computer induction from examples: a case study involving soybean pathology. *Int. J. for Man-Machine Studies,* **12**, (1980) 63–87.

[2] Quinlan, J. R. Discovering rules by induction from large collections of examples. In: *Expert systems in the microelectronic age* (ed. D. Michie), Edinburgh: Edinburgh University Press, 1979. pp. 33–46. See also Quinlan's chapter in Michalski, Carbonell & Mitchell, 19083).

[3] Quinlan, J. R. Semi-autonomous acquisitions of pattern-based knowledge. In: *Introductory readings in expert systems* (ed. D. Michie). New York, London and Paris: Gordon and Breach, 1982. pp. 192–207.

[4] Shapiro, A., & Niblett, T. B. Automatic induction of classification rules for a chess end-game. *Advances in computer chess 3* (ed. M. R. B. Clarke). Oxford: Pergamon, 1982, pp. 73–92.

[5] Paterson, A., & Niblett, T. B. *ACLS manual.* Edinburgh, UK, and Champaign, USA: Intelligent Terminals Ltd., 1982.

[6] Michalski, R. S., Carbonell, J., & Mitchell, T. (eds) *Machine learning: an artificial intelligence approach.* Palo Alto: Tioga, 1983.

[7] Michie, D. Experiments on the mechanisation of game learning. 2 rule-

based learning and the human window. *Computer Journal,* **25**, (1982). pp. 105–113.

[8] Shapiro, A., & Michie, D. A self-commenting facility for inductively synthesised end-game expertise. In *Advances in computer chess 4* (ed. D. Beal). Oxford: Pergamon, 1986.

[9] Hunt, E. G., Marin, J., & Stone, *Experiments in induction.* New York: Academic Press, 1966.

[10] McLaren, R. *Expert-Ease user manual.* Edinburgh: Intelligent Terminals Ltd., 1984.

[11] Michie, D. A theory of advice. *Machine Intelligence 8* (eds E. W. Elcock & D. Michie). Chichester: Horwood, and New York: Halstead Press, 1977, pp. 151–158.

[12] Michie, D. Measuring the knowledge-content of expert programs. *Bulletin of the Institute of Mathematics and its Applications.* **18** (11/12), 1982, pp. 216–220 (reproduced as Chapter 18, foregoing).

[13] Jackson, J. A. Economics of automatic generation of rules from examples in a chess end-game. M.Sc. thesis. Urbana: University of Illinois, 1975.

[14] Yao, A. Theory and applications of trapdoor functions. *Proc. 23rd Annual Symposium on Foundations of Computer Science* (Chicago), Los Angeles: IEEE, 1982. pp. 80–91.

[15] Shapiro, A. The role of structured induction in expert systems. Ph.D. thesis. Edinburgh: University of Edinburgh, 1983.

[16] McDermott, J. XSEL: computer salesperson's assistant. *Machine Intelligence 10* (eds. J. E. Hayes, D. Michie, & Y.–H. Pao). Chichester: Horwood, and New York: Halsted Press, 1982, 325–337.

[17] Bratko, I. & Mulec, P. An experiment in automatic learning of diagnostic rules. *Informatica,* **4**, 1980, pp. 18–25.

[18] Mozetic, I., Bratko, I., & Lavrac, N. An experiment in automatic synthesis of expert knowledge through modelling. *Proc. Logic Prog. Workshop.* Albuferia, Portugal. June 26–July 1, 1983.

[19] Michie, D. Game-playing programs and the conceptual interface. In: *Computer game-playing* (ed. M. Bramer). Chichester: Horwood, and New York: Halsted Press, 1982. pp. 11–25.

Section 4 AI and society

INTRODUCTORY NOTE TO SECTION 4

Every student of society is at the same time its creature. Margaret Mead's *Growing up in Samoa*, it is said, can be more easily understood by someone acquainted with Greenwich Village in the 1920's than by a resident of the South Seas.

The essay reproduced here as Chapter 20 has no claim to special dispensation. Yet reading it nearly 20 years later I found little to revise. There is, however, a great deal which can today be amplified. In particular the coming knowledge revolution, indicated with rather vague waves of the hand in this Chapter, is given technological substance in the immediately following 'Towards a knowledge accelerator'.

Many of the accomplishments and tactical approaches introduced into the world by machine intelligence are new. In aspiration and strategic thrust, however, they are as old as mankind's recorded culture. A year or two ago the University of Illinois honoured me with an opportunity to spread myself on the subject of 'Machine Intelligence: the first 2400 years'. I have used my write-up of what I said on that occasion to round off this book.

20

Computer — servant or master (1968)

It used to be possible to sweep the social challenge of computers under the carpet, with the dismissive phrase 'high-speed morons'. Today, however, computers play draughts at a good club standard, solve difficult problems in logic, compose dull but passable music, outperform librarians in the relevant retrieval of certain classes of document, translate Russian into useful dog-English, and perform many other exacting tasks of a non-numerical nature. Clearly if we are to bolster our self-respect as humans in face of the new wave of machine accomplishments we may have to find some other way of doing it than by talking about morons.

INTELLIGENCE

My own research as a scientist is concerned with teaching computers *not* to be morons, and with attempts to find general rules for doing this. I am fairly optimistic, if that is the right word, about the rate of progress in our own and in other similar laboratories elsewhere, in Britain and abroad. I qualify the word 'optimistic' because many people view the objectives of such work with deep unease. If informed that in a decade or two the first intelligent machines will be in our midst, their dominant feeling would *not* be one of optimism, but rather of gloom and anxiety. I shall leave to a later stage the discussion of whether there is justification for these fears — de-humanization of life, mass redundancy among brainworkers, suffocation of man by surplus leisure, computer-aided snoopers and 'thought-police' — and consider first in more detail what is meant by the concept of intelligence in a machine.

TURING'S TEST

The problem of testing a machine to see whether it is intelligent was first discussed by the great British logician and computer pioneer, Alan Turing, who died in the early 1950s. It was from my personal association with Turing

during the war and the early post-war years that I acquired my interest in the possibilities of using digital computers to simulate some of the higher mental functions that we call 'thinking'. Turing proposed the following test. The machine was to be placed on one side of a screen and a human examiner on the other side. Conversation between man and machine was permitted through the medium of a teleprinter. If after an hour or two's type-written conversation the machine had managed to fool the examiner into thinking that he had been conversing with a human being, then according to Turing the machine's claim to intelligence should be conceded.

There was undoubtedly an excellent idea here, and one which foreshadowed the modern trend towards 'conversational computing'. Unfortunately Turing's test, in its original form, has been refuted by an ingenious conversational computer program called DOCTOR written by Joe Weizenbaum of Massachusetts Institute of Technology. I reproduce below a fragment of a conversation with DOCTOR.

HOW DO YOU DO. I AM THE DOCTOR. PLEASE SIT DOWN AT THE TYPEWRITER AND TELL ME YOUR PROBLEM
I wish I had my baggage
CAN YOU ELABORATE ON THAT?
I am not quite clear as to what sort of elaboration is wanted
DO YOU BELIEVE IT NORMAL TO BE NOT QUITE CLEAR AS TO WHAT SORT OF ELABORATION IS WANTED?
Yes, quite normal
I UNDERSTAND
I have lost my baggage
YOU SAY YOU HAVE LOST YOUR BAGGAGE
I feel lost without it
DO YOU OFTEN FEEL LOST WITHOUT IT?
No, this is the first time
WHY NOT?
Because I normally do not lose my things
WHAT OTHER REASONS MIGHT THERE BE?
My mother trained me not to lose things
TELL ME MORE ABOUT YOUR FAMILY
I have lost my baggage
DID YOUR PARENTS EVER LOSE YOU?

At first sight this has an air of (slightly scatty) plausibility. In reality it is a trick — the same trick as most of us employ when keeping a conversation going, perhaps at a cocktail party, while our mind is elsewhere. Instead of attending to the sense of our companion's remarks we merely pick up key words (e.g. 'mother') to which we reply with stock phrases (e.g. 'Tell me more about your family'). If really stuck, we merely play back our companion's last remark with some suitable trimming.

This is essentially how DOCTOR works. Its success as measured by Turing's test is impressive. Patients in Massachussetts General Infirmary

were allowed to converse with the program, after being warned that a computer, not a doctor, was at the other end of the line. 60 per cent of them subsequently rejected this information and insisted that they had been in communication with a flesh-and-blood doctor — 'No machine could understand me that well'.

So Turing's test has to be refined if it is going to be useful in the way intended. Perhaps we should insist that the machine should fool Nobel Prize-winning scientists rather than hospital patients, or alternatively perhaps we should direct attention to whether the examiners feel that they have been having an *intelligent* conversation. To apply these definitions, they do not need to be philosophically watertight. Machine intelligence is not an exercise in philosophy but an engineering project.

One side of this engineering project is concerned with defining and implementing the separate components of mental aptitude — such capabilities as trial-and-error learning, pattern-recognition, generalization from individual instances, deductive and inductive reasoning, problem-solving and linguistic skill. Somehow these different capabilities, each represented in the computer by a different program, have got to be integrated together so that they function as an organized whole. We have some ideas about how this co-ordination of computer programs might be achieved, but these are still rather primitive and will not be discussed here. What I shall do is to take *one* of the constituent capabilities as the subject of a brief digression, before considering some of the social and psychological apprehensions which are voiced concerning the development of intelligence in computers.

LEARNING

The mental capability which I shall single out is trial-and-error learning. This is the simplest and lowest form of learning, in which the learner proceeds entirely *ad hoc*. He says to himself merely: 'Have I been in this situation before? If so, what did I do? What were the consequences of my action? If satisfactory, I shall choose the same action again. Otherwise I shall try something else'.

Note that no *generalization* from experience is involved. Situations are separately assessed in the light of past experience, without attempting to link them together into meaningful categories according to higher-level considerations. The surprising thing about pure trial-and-error learning is how far a computer system can get using this trick alone, without venturing into the realm of generalization. Samuel's famous computer program for playing checkers (draughts) was able to train itself to a passable amateur level with a system of pure trial and error (Samuel called it 'rote-learning'), even before its standard of play was further improved by the addition of a learning-by-generalization component. The program asked itself: 'Have I been in this checkers position before? If so, what move did I make? What were the consequences ...?' etc. Some years ago I extracted much spare-time amusement from constructing a trial-and-error machine out of matchboxes, whose task was to learn to play tic-tac-toe (noughts and crosses). More

recently with the help of my colleague R. A. Chambers I have developed a computer version, and this has been tested on a difficult problem which on the face of it does not look in the least like a game.

POLE AND CART

The task is to learn to control an unstable physical system which I shall call the 'Donaldson system', after the Cambridge physiologist who first used it in studies of machine learning. A motor-driven cart is free to run on a straight track of limited length, and balanced on it is a pole pivoted at the base which is free to fall down either left or rght along the line of the track. The motor is controlled by a single switch which determines at each instant whether the motor's force shall be applied in the left or the right direction. The task is to manipulate the switch so as to keep the cart running backwards and forwards along the track without either running off the end or dropping the pole. This task has obvious similarities to one which most of us attempted, with eventual success, during childhood — namely learning to ride a bicycle. Inevitably the child learns by sheer trial and error to begin with.

Our computer program does in fact learn to master the Donaldson system — without utilizing any special knowledge about it or being 'taught' by any human or mechanical mentor. The program is no more, and no less, designed to tackle a pole and cart than to learn to guide a car round a closed track or to monitor and control some simple industrial process. In this it illustrates a property which is a 'must' for any component of an intelligent computing system — *task-independent capability*. The striking feature of the human brain is not so much any outstanding performance at any particular task but rather its ability to make a useful, even if fumbling attempt at almost *any* task.

COOPERATION

An option in the program allows the human user to intervene and perform the control task himself, and a further option permits program and user to work on problems cooperatively, each benefiting from the other's trials and errors. I believe that this type of cooperative interaction between intelligent user and intelligent machine will come more and more to the forefront, and indeed will set the pattern in the future.

When thinking recently about the subject of particular mental capabilities, of which trial-and-error learning is just one example, I amused myself by copying out the late Ludwig Wittgenstein's list of what he called 'language games' and measuring each item against the present state of the art in machine intelligence. I reproduce his list below.

> Giving orders and obeying them.
> Describing the appearance of an object, or giving its measurements.
> Constructing an object from a description (a drawing).

Reporting an event.
Speculating about an event.
Forming and testing an hypothesis.
Presenting the results of an experiment in tables and diagrams.
Making up a story and reading it.
Play-acting.
Singing catches.
Guessing riddles.
Making a joke — telling it.
Solving a problem in practical arithmetic.
Translating from one language into another.
Asking, thanking, cursing, greeting, praying.

Now let us run through the list again. *Giving orders and obeying them* has been a routine function of computing systems for many years. *Describing the appearance of an object, or giving its measurements*, is a difficult task facing those engaged on 'hand-eye' computer projects. For a machine to inspect an object with a mechanical 'eye' and manipulate it with a mechanical 'hand' the first step must be to form a description from the visual image. *Constructing an object from a description* (e.g. building a tower from a photograph of a tower) is among the most difficult long-term goals of hand-eye projects — such as Marvin Minsky's at MIT and John McCarthy's at Stanford, USA. *Reporting an event* is beyond present technique. Again synthesis of a description from primary sense-data is the first step. The second is use of the synthesized description to generate appropriate language text. *Speculating about an event* is even further beyond present technique. *Forming and testing a hypothesis* is a process under active current study. *Presenting the results of an experiment in tables and diagrams* is a routine operation of contemporary computer programs for survey analysis. *Making up a story* is beyond present technique, although *reading it* from printed text is now marginally feasible. *Play-acting* would require a great extension to the arts of robotics: as for *singing catches*, humming the tunes is easy to program, but singing intelligibly is not. *Guessing riddles* is under active current study, but *making a joke* is very far beyond present technique. *Solving a problem in practical arithmetic* presents no difficulty even to primitive computer systems. *Translating from one language into another* is just attaining marginal feasibility by commercial criteria. *Asking, thanking, cursing, greeting, praying* are activities which express emotions, attitudes, desires, sympathies. It is meaningless to talk of them except on the basis of consciousness and self-consciousness in the intelligent system concerned. Many workers in machine intelligence believe that success on a really significant scale will hinge on the degree to which machine-representations of these phenomena can be devised — at least to the degree of permitting the machine to form some sort of internal logical model not only of the external world but also of itself in relation to that world.

Who is to be master? I am inclined to regard the dilemma 'Computer: servant or master' as a false one. To clear the ground for what I have to say

under this heading, let me first sketch a division of tasks into three categories.

(1) *Tasks suitable for humans alone.* This category is concerned with value, i.e. what sort of result do we want to see? For example, what weather do we want, irrespective of problems of prediction. Or what rate of road deaths relative to motorists' convenience are we prepared to tolerate?

(2) *Tasks suitable for computers alone.* These tasks are those of complicated detail and 'tactical' decisions: for example prediction of weather, or control of a city's traffic light system. The case of traffic lights has a special point of interest in the present context; the citizen seems prepared quite happily to accept this form of computer interference in his life, even though he may express great alarm over other forms. The implication is, I think, that the emotions of doubt and opposition to the computer revolution do *not* in reality hinge on a matter of principle — that control by machine is a bad thing. On the contrary it seems to be a matter of the appropriateness or otherwise of computer control in the given case. As applied to traffic lights, the sheer inhuman equitableness of computer control has a positive appeal. I believe that something similar is involved in the popularity among schoolchildren of computer programming as opposed to Latin. With programming there is no conceivable vulnerability to possible biases or prejudices of the teacher. The entire proof of the pudding is (if I may be allowed to mix a rather sticky metaphor) in the running of it on the machine.

(3) *Tasks suitable for cooperation.* These are tasks which are either too difficult at present for either partner to do alone or are in some way intrinsically suitable for conversational computing. In the second category I would place the use of a console connected to a conversational computing system as a 'home tutor' whereby the user can be steered through courses and subjects of study of his own choosing. It is not always easy, once one has taken the plunge into conversational computing, to distinguish between a program to *help* you do something and one to *teach* you to do it.

In this category of intrinsically conversational uses is the 'question-answering' facility which will one day become available as a service. Not only schools, hospitals and commercial firms but also the ordinary householder will be able to tap information and problem-solving power from a national computing grid with the same ease and immediacy as that with which he now draws on central supplies of gas, water and electricity. Along with question-answering services, which will allow us to enquire about restaurants in our locality or politics in Paraguay, will come the games opponent, the puzzle-setter, the quiz-master. An increasing demand upon computer systems will be for aid in coping in a stimulating way with the growing burden of leisure.

HELPERS AND HOBBIES

For many years only the rich will be able to install terminals in their private homes, but I have no doubt that the coming decade will see public telephone boxes up-graded to include a keyboard terminal connected to the computing

grid, and it is well within the reach of foreseeable software technology to offer services which will tempt ordinary people to place their coins in the slot.

Will the computer 'take over'? In the world of information-handling of course the computer will take over. The questions is will it take over as servant or master? To this one must reply: not as servant nor master, but as tutor, as secretary, as playmate, as research assistant. None of these in their human embodiments is a servant or a master; each is better described as a helper. The lessons of experience with computers do not support the idea that brain workers will be thrown out of employment by the machine. The indications are that as soon as brain workers learn to use the new facilities their work will be enlarged and enriched by the new possibilities which become available to them. The working week will, of course, continue to shorten in advanced countries as productivity rises, but this is a question of technological progress in general, and not specifically a consequence of computers. Whether the increase of leisure time is felt as a burden or a joy will depend on the means available for developing spare-time activities which can exercise and challenge man's varied capabilities.

It is my confident prediction that computer-aided self-instruction in science, history and the arts will have become a consuming hobby of large sectors of the population by the turn of this century. As for fears sometimes expressed that by then Big Brother will be able to watch us over the computational grid, or that our superiors or our neighbours may be able secretly to tap our dossiers kept on the universal electronic file, these fears can be dismissed. It is easier to devise 'unpickable locks' in a computing system than in the world of bank vaults and safes.

THE CONVERSATIONAL TERMINAL

The present fears of computers represent nothing new. When the first passenger-carrying railway services were opened, eminent medical men warned that if the human frame were transported at these speeds, fatal haemorrhages and seizures would be caused. There is a good parallel here. Imagine framing the question 'Railway train: horse or rider'. The answer, of course, is 'Neither horse nor rider but *travel assistant*'. As soon as people discovered this, their fears of rail travel disappeared. When computer terminals can offer a useful coin-in-the-slot service, the citizen will, I believe, cease to regard the computer as an alien monster or a ruthless competitor. Instead, the conversational terminal of the future will be welcomed for what it will do to enlarge daily life — as planning assistant, as budgeting assistant, and above all as a novel and challenging type of conversational companion.

21

Towards a knowledge accelerator (1986)

In the last century it was demonstrated in the numerical domain by Babbage and others that calculating machines make possible the discovery and tabulation of large bodies of factual (as opposed to conceptualized) knowledge. Today it is becoming apparent that computing techniques developed in expert systems work can be harnessed to a similar purpose for non-numerical domains, including those which extend beyond the codifying power of the unaided expert intellect. It is moreover possible semi-automatically to render the new material into conceptualized form. Computer chess studied as a branch of artificial intelligence has a central role to play.

INTRODUCTION

Recent findings in a number of laboratories have shown the feasibility of machine-aided synthesis of bodies of knowledge far exceeding in quality and extent any formulations achievable by unaided specialists.

The foregoing statement implies a rather far-reaching possibility, namely that the construction with automatic aids of new knowledge is destined to become a mass-production or manufacturing industry. In the same way that the manufactured articles of the nineteenth century in the first Industrial Revolution took the form of physical objects of use to consumers, now we have the demonstrated feasibility in the laboratory of automating the construction of a new range of commodities, namely, bodies of knowledge which did not pre-exist and in general case could not have pre-existed.

The power of the human specialist to execute knowledge which he already has is impressive. But his power to codify it is quite extraordinarily limited, far more limited than was expected. As a result (see Figs 21.1, 21.2 and 21.3) the knowledge engineering profession during its first decade went

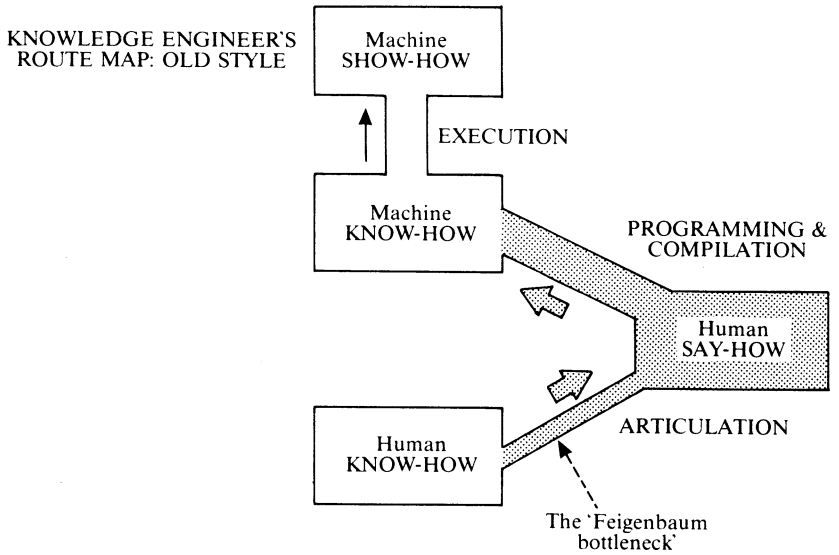


Fig. 21.1 — Old style transfer of know-how from expert brain to machine memory proceeds via the expert's articulation. There is evidence that this channel, corresponding to Feigenbaum's 'bottleneck problem of applied AI', narrows to non-existence with increasing task-complexity.

the wrong way about the knowledge-acquisition task and is only now making a course correction and turning to computer induction of expertise from examples.

We look to the development of the new induction-assisted craft at three levels.

Level 1

At the first level the process is essentially one of extraction and tabulation of expertise which already exists coded in certain human brains. That in itself is novel and commercially promising. A laboratory example is discussed by Alen Shapiro and myself elsewhere (ref. [8] of Chapter 19).

Level 2

Beyond that lies the possibility of automatically constructing new codifications of knowledge which are then accepted and used by the human professional, but which did not pre-exist in human brains and hence constitute genuine de novo synthesis. An example due to Bratko and his co-workers (Mozetic, Bratko, & Lavrac, 1983) is discussed later.

Level 3

Finally, automation can in principle be extended to the synthesis of new

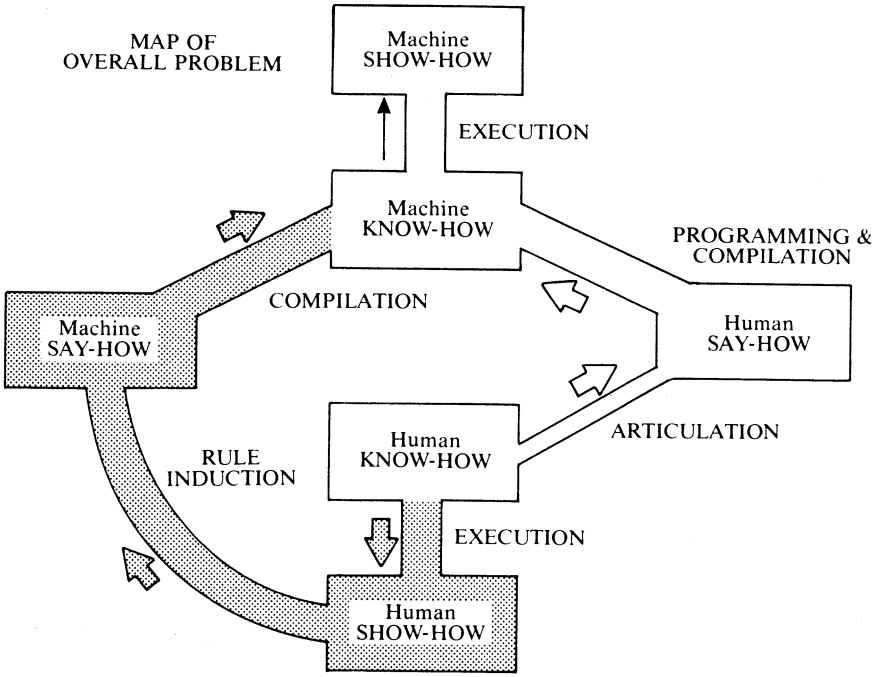


Fig. 21.2 — Map showing a hypothesized 'North-West Passage' through which know-how could in principle be communicated from expert to machine.

knowledge which not only did not pre-exist but could not have pre-existed; that is to say, knowledge which a brain could not possibly synthesize but can assimilate and use if synthesized by some other agency.

AN HISTORICAL PARALLEL

A parallel in the history of technology is the synthesis early in the nineteenth century of organic chemical compounds. Many of the same barriers — some of them mental barriers — as exist now to the automated synthesis of knowledge, existed then to the proposition of automated synthesis of organic compounds. For example the only urea that had ever existed was synthesized in living cells. Most of the chemists of the day were persuaded by the more mystically-minded biologists that this situation was eternal, and that it was unreasonable to expect synthesis by artificial means to be feasible for the carbon compounds. These were the sacred preserve of cellular anabolism, dependent on some vital force.

When, in 1828, urea was synthesized in the laboratory, it was initially synthesized in very small quantities, much too small to be of industrial interest. The amounts were sufficient, however, to upset the principle that organic compounds lay out of reach of technology. Within a fairly short

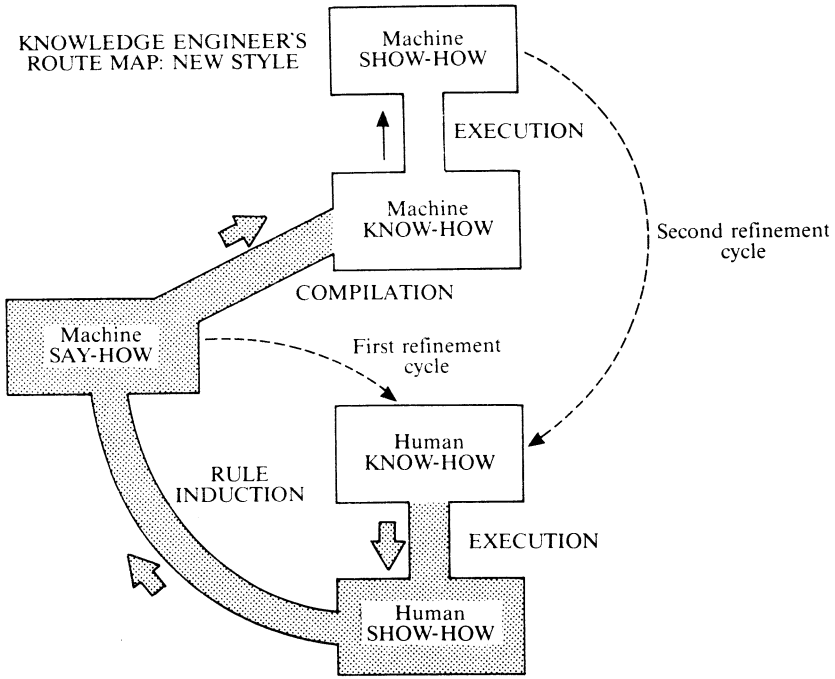


Fig. 21.3 — Confirmation is provided by the routine use in the author's laboratory of inductive expert system generators. Under instruction, the expert operates the induction cycle himself, inspecting on the screen, and correcting by applying remedial examples, successive versions of each induced rule. The outer cycle represents feed-back from tests of the rule-base in the field.

period the synthesis of urea was a mass production industry for fertilizer in agriculture. In the chemical industry, even to the present day, there is still large-scale reliance on extractive methods — insulin until recently came into that category — where it is a matter of extracting a compound already synthesized by some biological agency. This is analogous to extracting knowledge from a domain specialist and recodifying it in machine memory, regardless of whether he is already articulately aware of the knowledge. If he is not aware, then extraction can still be achieved by rule-induction methods, as recently demonstrated by Shapiro (1983). If he is aware, then rule induction may even so offer higher extractive effectiveness.

The chemical industry today relies not only on extractive processes, but also on the synthesis of compounds already synthesized biologically. Moving on to the post-war years of this century we find a third category: the synthesis on an almost overwhelming scale of a vast variety of new organic compounds which have never been synthesized by biological systems, never could be synthesized by any known biological system, but nevertheless are accepted by human tissues (as exploited for example in the pharmaceutical industry) as useful new products. A case can be made that all these phases, difficulties,

and opportunities which technology has passed through in the case of the chemical industry are beginning to be re-enacted in the computer-based knowledge industry.

INVERTING COMPUTATIONS

Provided that a problem domain can be exhausted by a tabulation of, shall we say, a mere billion or ten billion entries, as in the case of the chemical task of mapping mass spectral patterns onto molecular structures, then instead of despairing because the function that we would like to compute is effectively intractable, one can say: I am only interested for the moment in a subdomain of ten million elements and, although the function is adverse to compute, it may have an easy inverse — its inverse may be of low computational complexity.

Such functions have been called trap-door functions, because one can go through one way, but one cannot easily go through the other way. When that is the case, then there is a cure. Imagine that somebody has pointed out a straightforward algorithm to go from mass spectra to molecular structures or, if you like from chess endgame positions to game-theoretic values. Old style (processor intensive), the user inputs his questions into the question-answering machine, and since the algorithm is correct and complete and must terminate, eventually the user gets an answer. The trouble is that it comes after an unacceptable delay.

Instead he can now proceed as follows. The first step is to specify the space of answers: we are going to start from answers and work back to questions. The next requirement is an exhaustive generator, which from that specification will generate an answer and from that generate the next answer and so on until it has enumerated the complete answer space. Now we need a routine which embodies the hypothesised computation. In the case of mass spectroscopy, the inverse computation takes the form of simulating the action of the mass spectrometer, which given a molecular structure, beams electrons at it to break it down into fragments, weighs the fragments and forms a statistical histogram (mass spectrum) of those fragments according to abundancy and ion weight.

Deriving for each answer the linked question, an insertion routine puts the newly-discovered question-answer pair into the giant incremental dictionary, but does its indexing on the question rather than the answer. Hence this manufactured mountain of factual knowledge can be put into the field as an indestructible, everlasting, fast, cheap question-answering machine, to which the user can input questions and immediately get the answers by look-up. That whole process has been routine in a number of laboratories studying aspects of computer chess for at least the last fifteen years. The first person to do it was Thomas Strohlein, who constructed a data-base for the King-Rook-King-Knight end-game for his doctorate in Munich in 1970.

It follows from the prevalence of hard combinatorial problems in industry — scheduling problems, allocation problems, sequencing problems, optimization problems of all kinds — that whereever trap-door

functions exist, the industry will move in with large memory techniques. What do these techniques buy? In terms of artificial intelligence nothing at all. Interest from an AI point of view begins when the need is found to free up some of this mountainous storage space by compacting part of the tabulated material into description form. New facts can then be tabulated and found room for in the memory and generalized in turn. Thus the information-base is by degrees converted into a description-base subject to human-like constraints. Even in terms of sheer brute-force manufacture of knowledge at ground level, the pre-conceptualized level, it is a technology which will become part of our lives. When automated compression from look-up form into conceptual form has been made routine, the technology will become part of our culture.

AN AI EXAMPLE

Is there any way of using the same trick — the easy inverse trick — for proceeding in an AI style from the start? In order to do that, one would be dealing, not with irreducible facts and atomic queries, but with descriptions — concept-expressions in fact. The question space would be a space of concept-expressions and the answer space a space of concept-expressions. Bratko's work at Ljubljana illustrates these notions. The customer requirement is for a program which will take descriptions of electrocardiogram traces and generate diagnostic descriptions of heart disorders. There are automated systems in the market-place which do that after a fashion. I say 'after a fashion' because of limitations in reliability and accuracy of ECG interpretation based on purely statistical, as opposed to concept-based, models. The commercial systems implement statistical decision functions which do not embody anything which could be called cardiological knowledge. In general the first-rate cardiologist can outperform them. Instead of operating at the superficial level of direct statistical association between cardiac arrhythmias and ECG wave form Bratko decided to deal with the cardiac arrhythmias as an enumerable set of descriptions. He and his co-workers generated descriptions on the machine of all the possible arrhythmias, subject to the physiological constraints. These constraints say which arrhythmias can physiologically coexist with which others. Moreover in building this catalogue of physiologically possible arrhythmias Bratko's program goes through a qualitative model of the heart, constructing for each arrhythmia a prediction of what the ECG should look like, expressed qualitatively as a PROLOG description (see Fig. 21.4).

Notice that it is a true example of the easy inverse trick. The adverse function is the one that maps from ECG to arrhythmia diagnosis. However, all the computing goes in the reverse direction. Bratko's program uses a deep model, in some sense a causal model, of the biological system that is generating the ECG traces. Were it not for the physiological constraints, the enumeration task would blow up — the number of mathematical combinations goes up exceedingly fast. Fortunately the constraints dampen the combinatorial explosion, which levels out at approximately 580 different

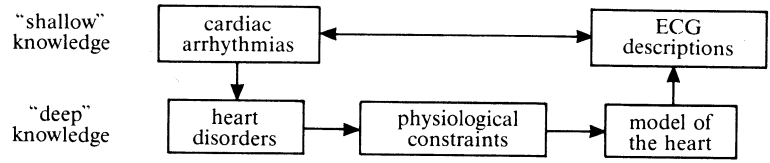


Fig. 21.4 — Program-synthesized rule-base for ECG interpretation exemplifies the difference between modelling an intuitive skill ('shallow' knowledge of ECG-arrhythmia associations) and modelling the causal and physical properties of the corresponding domain ('deep' knowledge of the heart's functioning and of the physiological constraints on the co-occurrence of different arrhythmias). By exhaustive enumeration of the physiologically possible arrhythmias and logical reconstruction from each of a corresponding ECG description, the program was able to generate 'shallow' from 'deep' knowledge.

multiple arrhythmias. The reason why the diagnosis of the multiple arrhythmias is of importance is that even a medical student has little difficulty in relating an ECG to a simple arrhythmia since there is only one specific defect of the heart — some transmission pathway is blocked or some generator which should be generating pulses is not, or is in the wrong site. But in the presence of multiple arrhythmias, the description of the waveforms cannot in any way be obtained by simple-minded summing or averaging or other way of combining the descriptions that belong to the individual constituent arrhythmias.

This machine-synthesized catalogue can give useful service in either direction, either as a diagnostic catalogue to be looked up in the machine (or on paper), or the other way round, going from arrhythmias to ECG characteristics. It is the second mode that is now in use in that particular hospital, namely as a teaching aid.

Here is a path that could lead to a substantial systematic automated manufacture of new knowledge. Not all of that cardiological knowledge catalogue was known to cardiologists. The head cardiologist was initially sceptical. Bratko put him to the test by using the machine knowledge-base to generate unusual and complex cases. In some of these the cardiologist stumbled, mainly through overlooking secondary possibilities. This work is the first recorded case of automated construction of new conceptualizations on other than an extractive basis. It was not done by collecting or inductively extracting diagnostic rules from existing human diagnosticians. The diagnostic rules were synthesized de novo.

The senior author of the work, Dr Ivan Bratko, has for many years been one of the world's leading contributors to the study of computer chess from the artificial intelligence point of view. It has been said that chess is the Drosophila melanogaster of AI. The reference is to the pioneers of the chromosome theory of heredity who bred fruit-flies in the laboratory rather than chickens or cows. It must have been a relief to the early geneticists when the results of the work found practical applications. It is gratifying to record the start of a similar migration of technique today.

ACKNOWLEDGEMENT

This discussion and much of the material reviewed derives from a five-year project funded by ITT (Europe) to explore new methods in the software technology of expert systems. The author is especially indebted to ITT's Director of Research, Dr Charles Herzfeld, for personal encouragement and for a number of illuminating technical discussions.

REFERENCES

Mozetic, I., Bratko, I., & Lavrac, N. An experiment in automatic synthesis of expert knowledge through qualitative modelling, *Proceedings of the Logic Programming Workshop*, Albufeira, Portugal, June 1983.

Shapiro, A. The role of structured induction in expert systems. Ph.D thesis, Machine Intelligence Research Unit, Edinburgh, 1983.

Strohlein, T. Untersuchungen uber Kombinatorische Spiele. Dissertation for Dr. Rev. nat., Technische Hochschule, Munich, 1970.

22

Machine intelligence: the first 2400 years

The field known by its practitioners as AI has attracted attention of a kind usually associated with great novelty. Moreover many people, among whom I include myself, believe that artificial intelligence is about to re-shape our world. So it may seem surprising if I assert that AI is not the youngest but the oldest of mankind's systematic studies.

The modern mind has been conditioned, partly by Dr Kuhn with his revolutions and paradigms and partly by the media. We expect a scientific technology to jump suddenly into our ken, and then to advance in a series of paroxysms called breakthroughs. But the story of machine intelligence has not been at all like this. Over a traceable span of 2400 years AI has been the slow plodder, and possibly owes the world an apology for taking such an unconscionable time arriving. But as will emerge from my narrative there have been retarding circumstances, including ignorant opposition — not from the lowest minds of each age but from the most cultivated.

The story begins with the attitudes expressed by Socrates to the invention of writing. To us writing stands forth as the first necessary device to be placed in AI's box of mind-enhancers, the key to the rest. Socrates saw it differently. Speaking in Plato's *Phaedrus* of one of Egypt's junior gods, by the name of Thoth, he says:

> Among his inventions were number and calculation and geometry and astronomy, not to speak of various draughts (i.e. checkers) and dice, and, above all, writing.

Thoth goes to the arch-god Ammon and declares: 'Here is an accomplishment, my lord the king, which will improve both the wisdom and the memory of the Egyptians'. Ammon replies that, on the contrary, writing is an inferior substitute for memory and understanding. 'Those who acquire it', he says, 'will cease to exercise their memory and become forgetful; they will rely on writing to bring things to their remembrance by external signs instead of on their own internal resources.'

This argument has re-appeared in many contexts, most recently in

connection with today's hand-held calculators. But Ammon's next point is more subtle, and goes to the heart of our topic, namely the nature of knowledge when predicated of an interacting pair of systems when one of them is a person and the other is an inanimate device, in this case a book. 'As for wisdom', says Ammon about book-users, 'your pupils will have the reputation for it without the reality: they will receive a quantity of information without proper instruction, and in consequence be thought very knowledgeable when they are for the most part quite ignorant'. Ammon's crushing finale is: 'And because they are filled with the conceit of wisdom instead of real wisdom they will be a burden to society'.

The objection here is concerned with the simulation of knowledge by the possessor of a rapid-access source. Precisely because the distinction can be blurred between possessing a sufficiently fast knowledge-source and actually possessing the knowledge, we see Ammon's point as not only subtle but also topical. To Ammon the distinction is critical. Herbert Simon's contrasting view, hatched in 1955, is expressed in a passage published in 1971:

> The change in information processing techniques demands a fundamental change in the meaning attached to the familiar verb 'to know'. In the common culture, 'to know' meant to have stored in one's memory in a way that facilitates recall when appropriate. By metaphoric extension, 'knowing' might include having access to a file or book containing information, with the skill necessary for using it. In the scientific culture the whole emphasis in knowing shifts from the storage or actual physical possession of information to the process of using or having access to it.

Note that Simon's position paves the way for a philosophy of knowledge in which both members of a man–machine partnership can be allowed to 'know' things, jointly with, and in some cases independently of, each other, or even — as explained in previous chapters — to create and codify new knowledge for joint use. It is interesting that Socrates goes on to scourge the passive technology of books for failure to be precisely the kinds of products which today's AI scientists are striving for, notably knowledge-bearing devices able to explain their own contents. Socrates is particularly, and rightly, scornful of books in this regard: '... if you ask them what they mean by anything they simply return the same answer over and over again.'

I want now to move a few years on from Socrates and to look over the shoulder of Aristotle as he struggles to pin down a different but related ambiguity in the meaning of the word to know, concerned not so much with whether information is retrieved from an internal or an external store but rather with whether retrieval is effectively instantaneous.

Aristotle's study of knowing and inferring, the *Posterior Analytics*, starts briskly and provocatively: 'All instruction given or received by way of argument proceeds from pre-existent knowledge'. If some of us at the start of the 1960's had marked, learned and inwardly digested this passage, that decade's unfulfilled quest for knowledge-free mechanisms as the key to machine intelligence might have been shortened. Not that we do not need

these mechanisms of general-purpose search and general-purpose deductive inference. We do need them, and derivatives of what was invented then can be discerned in the workings of Prolog and other computer languages of the 1980's. What we do *not* need is the Platonic fancy that in these pure forms lies all that is required for a machine to receive instruction by way of argument. Aristotle was right. There is an additional requirement, namely for the cognitive agent to be first loaded with relevant knowledge.

Obvious now, this insight was far from obvious in 1970, when bold spirits at MIT announced the knowledge approach and began to push it down the reluctant throats of the rest of us. Now that the dust has settled, one of the most elegant worked examples still remains: David Waltz' program for interpreting pictures such as that of Fig. 16.2, i.e. shadowed scenes formed by varied and partially occluding polyhedra resting on a plane landscape. The problem posed by the line drawing in that Figure is: how can a machine identify the solid bodies represented by the draughtsman, and thus in a sense understand the picture? Clearly it must somehow associate every face, edge and vertex with its proper body, while avoiding hypothesized associations which contradict the facts of three-space and the conventions of projection onto two-space. The Aristotelian answer must be in terms of 'pre-existent knowledge', whatever that phrase might mean in terms of interpreting such drawings. We shall approach the meaning conferred by Waltz' program by first digressing back to Aristotle and the notion of pre-existent knowledge.

Aristotle categorizes the notion under two headings, namely facts about individual objects (qualified knowledge) and facts about classes (unqualified). An example he gives of the first is that 'this figure inscribed in the semicircle' is a triangle. His example of the second is that the angles of every triangle are equal to two right angles. He now poses the problem: can a student who knows the second fact, about triangles in general, be said already to know its truth in relation to every particular triangle, including one suddenly shown him which he has never seen before?

Aristotle realises that before the knowledge becomes complete, some computation must be done. First the student has to run a mental recognition routine (to use today's language) so as to know that this object *is* a triangle. Then he must apply his rule about triangles so as to infer that this one too has angles equal to two right angles. Aristotle holds back from Herbert Simon's usage, which credits the student with already having knowledge of the properties of this particular triangle provided that when asked about it he can mobilise the answer with sufficient immediacy. Instead Aristotle sits with one foot on either side of the fence and says of the student: 'Before he was led on to recognition or before he actually drew a conclusion, we should perhaps say that in a manner he knew, in a manner not'. If Aristotle had taken just one more step he might have formed the suspicion that the chains of mental calculation demanded by some tests of student knowledge could be infeasible for the examinee to complete, even in a lifetime, such as for example the primeness or otherwise of some 40-digit integer suddenly shown him. If in such a case Aristotle were to discard the equivocation 'in a sense he knew', he would step over the threshold which Simon crossed in

1955 and of which much of the AI world has still to become aware. This threshold separates the world of unbounded rationality shared by philosophers, mathematicians and physical scientists, and the world of bounded rationality meaning the term very strictly to mean brain-bounded rationality. We must come to terms with the laws of brain-bounded rationality if we are to understand anything at all about knowledge, whether realised in people or in machines.

With this in mind let the reader now turn back to Chapter 16 for the meaning given to the word by Waltz' program. Knowledge here takes the form of a catalogue of classes of vertex, in Table 16.1 of that Chapter, labelled accordng to a line-labelling scheme embodying certain elementary facts of optics and geometry: unqualified knowledge, in Aristotle's terminology. The immediately following Table shows the respective sizes of the combinatorial spaces which must be searched under two alternative assumptions.

Alternative 1 says that the cognitive agent (let us suppose some visually naive student) has no knowledge of the above-mentioned elementary facts, but only of a rule for deciding whether a given allocation of labels to local features is globally legal. If asked to say, by analogy with Aristotle's triangle test, whether a given drawing does or does not depict a physically possible arrangement of polyhedra, the only strategy open to him is that of 'generate and test'. During the student's execution of this laborious phase, Aristotle has to say that in a manner he knows, in a manner not. Unfortunately from the standpoint of this method the computational complexity of even simple line drawings is intractably large. For the drawing shown in Chapter 16 the student could not possibly complete a generate-and-test strategy within a life-time. So Aristotle would presumably not in this case say 'in a manner he knew'. Like the rest of us he might rather conclude that the student did not know at all, and never would know unless he were to abort the attempt to answer the question and take more knowledge on board.

Is promptness in supplying good answers, then, the only credential required to support a claim to knowledge, even when the claim is advanced on behalf of a machine? Not so. Certain structural criteria must additionally be met if the machine's answers are to accredit a claim that it 'knows'. Suppose that one were to run on a supercomputer the naive generate-and-test algorithm for Waltz' pictures, regularly obtaining correct answers in less than a second. Many a bystander, including even from the ranks of AI, might be tempted to say of each answer: 'In a manner the machine knew ...' — so long as no-one asked it: 'But *how* do you know?'. An answer such as 'I generate and test, but I do it very fast' would, one supposes, fail to satisfy most examiners.

A brain or a computer program gives solutions to problems in a given domain on the basis of some stored complex of relevant procedures, methods and facts. For a person knowledgeable about the domain to credit another agent with *knowledge*, as opposed to mere problem-solving ability, the stored operational descriptions employed by the two parties must satisfy

some minimal matching relation. More particularly, description A and description B must show a level of match not only in what they DENOTE (they must give the same answers to the same questions) but also in what they CONNOTE (the two descriptive complexes must be essentially similar).

A more detailed basis can be found in Chapter 18 for analysing what is and what is not knowledge in a machine. Here we simply point out that customary usage of the term 'knowledge-based' includes the requirement for such a system that it be able to explain the process by which a conclusion was reached. Moreover this process must be such as to support the generation of explanatory comments which match the user's way of thinking about the problem. Plainly Alternative 1 cannot be made the basis of a knowledge-based system at all, but only of a super-clever black box.

Alternative 2 says that the cognitive agent has in memory the facts concerning labelling constraints and physical possibility and uses them to confine the amount of generate-and-test to what can be quickly accomplished by even so sluggish a calculating engine as the brain. If the agent, whether machine or student, then answers promptly and correctly then it seems reasonable to say 'In a manner he knew', even though Student (2) strictly speaking has no more initial *information* about the answer than Student (1): both have all that is needed (in principle) to answer the question. What has changed in the transition from 1 to 2 is not the information-content concerning the answer but the amount of information which the brain can mobilise about its value at short notice. In a neo-Aristotelian system of definitions we might call this latter quantity the knowledge-content of the stored materials. Notice that when operating under Alternative 2 the supercomputer's reply (this time, one supposes in a fraction of a nanosecond) *can* reasonably be credited to knowledge. The machine this time observes a regime of brain-bounded rationality in the selection of facts and representations held in memory. Hence its explanatory trace is intelligible to any equally informed user imprisoned in the same bounds.

The phrase 'bounded rationality' is from Herbert Simon's introduction to the first section of his book 'Models of Thought' in which his papers published before the year 1979 are collected. By way of a definition suited to my theme we can say that AI is

THE COMPUTATIONAL MODELLING OF
BRAIN-BOUNDED RATIONALITY

MODELLING BRAIN-BOUNDED RATIONALITY

The elaboration of Simon's 'bounded' may seem needless, since Simon plainly refers to the bounds imposed by various information-processing limitations of the brain, in particular the size of short-term memory and the speed of sequential operations. But it is necessary to distinguish between Simon's boundedness and that of computational physicists and computer people susceptible to their way of thinking, who prefer to model a rationality

bounded only by hardware technology and the constraints of solid-state physics. I made earlier mention of circumstances which have retarded the development of an effective science of AI. I mentioned resistance from other disciplines. A greater retardant, though, has been neglect even by some AI practitioners of Simon's qualification concerning the brain's boundedness, and the inescapable consequences of this for the forms which the modellers must follow. Yet the Nobel Prize awarded to Simon in 1979 was for his use in the mid-1950s of precisely this qualification to demolish the models of von Neumann and Morgenstern, which were based on the unbounded rationality of an imaginary being, Economic Man. In the key paper of 1956, Rational Choice and the Structure of the Environment, we find:

> A comparative examination of the models of adaptive behaviour employed in psychology (e.g. learning theories) and of the models of rational behaviour employed in economics shows that in almost all respects the latter postulate a much greater complexity in the choice mechanisms, and a much larger capacity in the organism for obtaining information and performing computations than do the former. Moreover, in the limited range of situations where the predictions of the two theories have been compared ..., the learning theories appear to account for the observed behaviour rather better than do the theories of rational behaviour.

Yet realization has come only slowly that the potent historical example set by the physical and mathematical sciences is the worst possible example for AI. Artificial intelligence must build its models around the very same peculiar shapes and constraints of brain-bounded cognition which the physical scientists seek so rigorously to exclude. To make the exhortation more specific, I have set out in Table 22.1 some of these peculiar constraints.

Table 22.1 — Some information-processing parameters of the human brain. Estimation and other errors can be taken to be around 30 per cent

1.	Rate of information transmission along any input or output channel	30 bits per second
2.	Maximum amount of information explicitly storable by the age of 50	10^{10} bits
3.	Number of mental discriminations per second during intellectual work	18
4.	Number of addreses which can be held in short-term memory	7

Between the time of Socrates and that of Aristotle we find the first machine builders. As already related in Chapter 16, the mathematicians Eudoxus and Archytas enjoy a double priority, in the first place for devising

mechanical aids to reasoning and in the second place for provoking the first of the succession of eminent counter-blasts which have punctuated AI history. These two employed their art, in Plutarch's words, 'as a means of sustaining experimentally, to the satisfaction of the senses, conclusions too intricate for proof by words and diagrams', much to the annoyance of Plato at their meddlesome encroachment on the preserve of pure thought.

I shall not follow the further progress of special-purpose computing engines, which continued long into the Roman era and included such elaborate wonders as complete clockwork simulators of the solar system. Rather, in keeping with my chosen Aristotelian emphasis, only that strand will be traced which bears on the inductive side, on the synthesis of generalities from particulars, discovery of new concepts, and machine learning.

How does one get knowledge from a knowledgeable human source, i.e. an expert, into a computer? Two ways are open:

(1) let the expert tell his rules of thought and rules of thumb to a programmer who then codes up what he has been told;

(2) let the expert teach the machine by showing, that is by feeding it preclassified examples of expert decisions.

In the case of (2) the machine must have the power of inducing rules from examples; it must be able to generalize. Table 22.2 shows the results of an

Table 22.2 — Expert system generated exclusively from examples compared with hand-crafted variants (from Chilausky, Jacobsen, and Michalski 1976, *Proc. VI Internat. Symp. on Multi-Variable Logic*, Utah). The AQ11 induction procedure was coded in the PL1 programming language

AQ11 in PL1	120K bytes of program space
SOY-BEAN DATA:	19 diseases 35 descriptors (domain sizes 2–7) 307 cases (descriptor sets with confirmed diagnosis)
Test set:	376 new cases
machine runs using rules of different origins	>99% accurate diagnosis with *machine rules* 83% accuracy with *Jacobsen's rules* 93% accuracy with interactively *improved rule*

early experiment conducted by Ryzsard Michalski and colleagues at the University of Illinois on how to build a machine diagnostician of soy-bean diseases.

By this and by much careful follow-up work Michalski decisively established that method (1) is not as effective as might be hoped, and is also rather costly, while (2) is not only cheap but also effective by the highest standards. A feature of special interest in view of the brain-bounded nature of human rationality is that Michalski's machine-synthesised rules are intelligible to the soy-bean experts and also mentally executable by them. In spite of their largely synthetic origin, these products could reasonably be called concept-expressions. This demonstration must rank as a highly significant milestone in the practical modelling of brain-bounded rationality.

Let us go back in time to the first milestone along the trail of machine concept formation. We find it in 13th Century Spain. Here was made what Martin Gardner has described in his book *Logic machines and diagrams* as

> ... the earliest attempt in the history of formal logic to employ geometrical diagrams for the purpose of discovering non-mathematical truths, and the first attempt to use a mechanical device — a kind of primitive logic machine — to facilitate the operation of a logic system.

The tale of Ramon Lull's long, tempestuous, and almost unbelievable career, and of his cognitive contrivances, is enjoyably told in Gardner's book. Lull's fundamental idea was the generation of new and complex concepts by mechanical assortment and recombination of pre-existing simpler ones. This he accomplished by elaborate systems of concentric spinning disks. Around the edges of the disks could be inscribed the names of component concepts from which more complex multiple conjunctions were generated. For the less intellectual audiences, perhaps for site visits to his theological laboratory (one of whose products was a set of 100 sample sermons generated by his spinning disks), he prepared some simplified popular versions. But in Martin Gardner's words 'the method reaches its climax in a varicolored metal device called the *figura universalis* which has no less than fourteen concentric circles! The mind reels', Gardner concludes, 'at the number and complexity of topics that can be explored by this fantastic instrument.'

Lull's influence in exciting the minds of his own and succeeding generations was immense, both *pro* and *contra*. Four hundred years later, as a lampoon on Lull and his method Jonathan Swift describes in *Gulliver's Travels* a generate-and-test treatise-writing machine. The professor in charge reminded Gulliver that

> everyone knew how laborious the usual method is of attaining to arts and sciences; whereas by his contrivance the most ignorant person at a reasonable charge, and with a little bodily labour, may write books in philosophy, poetry, politicks, law, mathematicks and theology, without the least assistance from genius or study.

On the other side of the Lullian controversy, Leibnitz was much taken by Lull's combinatorial machines, and employed similar methods exhaustively to crank out formulas of propositional logic which were then checked for validity, on the generate-and-test principle. His own faith in what might be accomplished through machine-style symbol processing was astonishing:

> If one could find characteristics or signs for expressing all our thoughts as clearly and exactly as arithmetic expresses lines, we could in all subjects, in as far as they are amenable to reasoning, accomplish what is done in arithmetic and geometry.

Commenting on this passage, the British logician John Shepherdson writes: 'His full programme would have embraced not only pure and applied mathematics but also grammar, law, politics, physiology, theology, the art of discovery etc. He was rather optimistic about the time it would take:

> I think that a few selected men could finish the matter in five years. It would take them only two however to work out by an infallible calculus the doctrines most useful for life, those of morality and metaphysics.'

Three hundred years before recognition of the notion of the intrinsic complexity of computations, whether mental or otherwise, Leibnitz exhibits in its most florid form the nonsense into which the neglect of such bounds can still lead the AI enthusiast. The kind of neglect assailed in 1955 by Simon had even earlier been satirised by Claude Shannon in the context of the theory of games applied to chess. Shannon's point was not that grandmaster chess, or even move-perfect chess, was unimplementable, for that would be over-dogmatic. His point was that this goal, as with Leibnitz' more grandiose enterprise, needs more than a sound mechanisation of in-principle solutions. It needs what today we call the 'knowledge approach', for which both Shannon and his great contemporary Alan Turing made specific, prescient, and largely neglected proposals.

Turing created, in the intellectual sense, the concept of the universal machine. In design and implementation, however, the chapter was opened a century earlier, by Charles Babbage. He in turn was undoubtedly inspired by the 'universal machine' idea implicit in Leibnitz' great knowledge project. It is noteworthy that Babbage, as passionately devoted to the aim of machine intelligence as Leibnitz, was also Leibnitz' great admirer and champion. Babbage was largely instrumental in gaining acceptance of Leibnitz' 'd's to replace Newton's 'dots' in the teaching and practice of the calculus in England. He also showed a campaigning zeal to compel acceptance of the feasibility of machine intelligence by effecting some dramatic demonstration. As with Shannon and Turing in a later age, his mind was drawn to the game of chess, for which he advanced proposals for formalizing principles of play.

Certain writings of Turing, in particular the unpublished Lecture of 1947 to the London Mathematical Society ('The Automatic Computing Engine', typescript in King's College Library, Cambridge), have not received atten-

tion – understandably since they have yet to acquire relevance in relation to the more immediate and obvious developments of his main work. The multi-billion-dollar industry set to dominate the century's closing years traces directly from the Universal Turing Machine (UTM) formalism of his 1937 paper. Andrew Hodge's biography provides a good route-map of developments already become so dazzling as to eclipse from view the more speculative themes. It was these themes, however, which Turing perceived as the wave of the ultimate future. In the UTM the rest of the world saw a formal model for a revolutionary concept: programmability. To its author this was the obvious part. What Turing saw as the revolutionary part was the concept of *self-programmability*. Since not everyone outside the ranks of AI is aware of this, I close with a pertinent passage from the 1947 Lecture:

> Let us suppose that we have set up a machine with certain initial instruction tables, so constructed that these tables might on occasion, if good reason arose, modify those tables. One can imagine that after the machine had been in operation for some time, the instructions would have been altered out of recognition, but nevertheless still be such that one would have to admit that the machine was still doing very worthwhile calculations. Possibly it might still be getting results of the type desired when the machine was first set up, but in a much more efficient manner. In such a case one would have to admit that the progress of the machine had not been foreseen when its original instructions were put in. It would be like a pupil who had learnt much from his master, but had added much more by his own work. When this happens I feel that one is obliged to regard the machine as showing intelligence.

The passage ends on a note of expectation, robbed of fulfilment by his early death:

> As soon as one can provide a reasonably large memory capacity it should be possible to begin to experiment on these lines.

Turing died in 1954. A. L. Samuel was then just beginning to take over from Christopher Strachey the programming groundwork for experimenting with the game of checkers, and to embark on some compelling illustrations of the idea of table-modifying tables. A year or two later, as described in Chapter 1, my own impatience to experiment had me immersed in glass beads and matchboxes. Learning of this, Strachey visited me. Might I not, he enquired gently, advance with greater speed by the use of more modern equipment? Strachey and I were soon to collaborate in a Government initiative led by Lord Halsbury to implement certain thoughts at that time new to Whitehall, namely that

— computation is a subject of scientific study, on a par with physics, chemistry, biology etc;
— scientific study requires enablement;
— enablement has to include the provision of appropriate equipment.

Extraordinary as it may seem today, none of these propositions was seen as in the least self-evident. Each had to be buttressed in detail and argued through. In 1964 on commission from the nascent Science Research Council I conducted a poll in British Universities of some hundred under-40 computer scientists. I asked respondents to rank different subdivisions of computing according to estimated importance and personal interest. It is interesting to recall that even at that early stage the two topics which dominated the replies were the man-machine interface and machine intelligence.

In these and other ways it came about that Britain's part in the first 2400 years was re-animated. A selective sample of the activity which ensued has contributed some of the subject matter of this book.

Index