MACHINE INTELLIGENCE 12

Towards an automated logic of human thought

edited by

J. E. HAYES

Research Associate, Turing Institute

D. MICHIE

Chief Scientist, Turing Institute

and

Ė. TYUGU

Head of Software,
Institute of Cybernetics,
Estonian Academy of Sciences

CLARENDON PRESS · OXFORD
1991

Oxford University Press, Walton Street, Oxford OX2 6DP
Oxford New York Toronto
Delhi Bombay Calcutta Madras Karachi
Petaling Jaya Singapore Hong Kong Tokyo
Nairobi Dar es Salaam Cape Town
Melbourne Auckland
and associated companies in
Berlin Ibadan

Oxford is a trade mark of Oxford University Press

Published in the United States
by Oxford University Press, New York

British Library Cataloguing in Publication Data
Machine intelligence.
12, Towards an automated logic of human thought
1. Artificial intelligence
I. Hayes, J. E. (Jean Elizabeth) 1928– II. Michie, Donald 1923–
III. Tyugu, Ė. (Ėnn) 1935–
006.3

ISBN 0-19-853823-5

Library of Congress Cataloging in Publication Data
Machine intelligence 12: towards an automated logic of human thought /edited by J. E. Hayes, D. Michie, and Ė. Tyugu.
p. cm.
Includes index.
1. Artificial intelligence—Data processing. 2. Machine learning.
I. Hayes, Jean E. II. Michie, Donald. III. Tyugu, Ė. Kh. (Ėnn Kharal'dovich), 1935– . IV. Title: Machine intelligence twelve.
Q336.M33 1991 006.3—dc20 90-7320

ISBN 0-19-853823-5

Typeset by Cotswold Typesetting Ltd, Cheltenham
Printed in Great Britain by St. Edmundsbury Press, Bury St. Edmunds

MACHINE INTELLIGENCE 12

MACHINE INTELLIGENCE

Machine Intelligence 1 (1967) (eds N. Collins and D. Michie) Oliver & Boyd, Edinburgh

Machine Intelligence 2 (1968) (eds E. Dale and D. Michie) Oliver & Boyd, Edinburgh

(1 *and* 2 *published as one volume in* 1971 *by Edinburgh University Press*) (eds N. Collins, E. Dale, and D. Michie)

Machine Intelligence 3 (1968) (ed. D. Michie) Edinburgh University Press, Edinburgh

Machine Intelligence 4 (1969) (eds B. Meltzer and D. Michie) Edinburgh University Press, Edinburgh

Machine Intelligence 5 (1970) (eds B. Meltzer and D. Michie) Edinburgh University Press, Edinburgh

Machine Intelligence 6 (1971) (eds B. Meltzer and D. Michie) Edinburgh University Press, Edinburgh

Machine Intelligence 7 (1972) (eds B. Meltzer and D. Michie) Edinburgh University Press, Edinburgh

Machine Intelligence 8 (1977) (eds E. W. Elcock and D. Michie) Ellis Horwood, Chichester/Halsted, New York

Machine Intelligence 9 (1979) (eds J. E. Hayes, D. Michie, and L. Mikulich) Ellis Horwood, Chichester/Halsted, New York

Machine Intelligence 10 (1982) (eds J. E. Hayes, D. Michie, and Y.-H. Pao) Ellis Horwood, Chichester/Halsted, New York

Machine Intelligence 11 (1988) (eds J. E. Hayes, D. Michie, and J. Richards) Oxford University Press, Oxford

Machine Intelligence 12 (1991) (eds J. E. Hayes, D. Michie, and Ė. Tyugu) Oxford University Press, Oxford

FOREWORD

Evgeni Velikhov
Vice President of the Academy of Sciences of the USSR

It is a pleasure to contribute an introduction to this twelfth volume of the international *Machine Intelligence* series. My own work has, at times, cast me in the scientific roles of experimenter, instrumentation designer, and administrator. In all these roles I have seen the growing pervasiveness of the new tools of information technology. As a visitor to the 1987 meeting in Milan of the International Joint Conference on Artificial Intelligence I received a vivid impression of the role that machine intelligence in particular seems destined to play in this, the final decade of this century. Economic growth is increasingly dependent on new technologies in which the intelligence of machines plays a leading role. The concept of machine intelligence itself acquires a new semantic content. This is demonstrated by the evolution of new disciplines such as mechatronics as well as by the increasing importance of intelligent tools in manufacturing. It seems extremely important for the future of the human species that the mind that machines develop grows faster than the muscles, that is the energy parameters.

Looking at the eleven previous impressive volumes of *Machine Intelligence* one observes that the MI conferences have covered a significant part of the world including the Soviet Union where, before MI-12, the ninth MI conference was also held in 1977. I believe that the style and the content of the *Machine Intelligence* series will continue to reflect the much needed dialogue between various societies.

PREFACE

For almost twenty-five years I have, as editor of these volumes, presided over the inquisitiveness of the newly arrived. The young delight to get their noses into everything. But unbounded promise must sooner or later confront the emergence of what in ecology and in entrepreneurial commerce are known as 'niches'. Possibly, machine intelligence has, during all this time, been sleepwalking towards its own true niche. In any case, the series must now select one. Along what line do we see the future commitment of the *Machine Intelligence* series?

In 1986 a Steering Committee was formed to set a direction and to initiate the formation of an international editorial board with an executive editor and two associate editors to support the work and to organize the workshops themselves. These will resume their initial annual tempo. As editor-in-chief I am privileged to welcome our future executive editor Dr Stephen Muggleton. With regard to directions, the central theme will be the design of automated support for intellectual discovery and its application. Sophistication of computing aids is a conspicuous feature of today's scientific scene. From the astrophysicist's supercomputer to the field worker's pocket machine, the race has been to automate every function but one. That one is scientific reasoning itself, whilst AI has been the laggard.

More than a quarter of a century ago, the Nobel Prize-winning chemical microbiologist Joshua Lederberg had a vision of intelligent machines as partners in the scientific quest. In Stanford's DENDRAL project he initiated the first inroad into organized empirical enquiry. The tools of that time were too weak to accomplish more than the planting of a series of signposts, some of which appear in earlier MI volumes. Among these the MetaDENDRAL module set a crucial pointer to the need, reflected in this volume, to mechanize the inductive as well as the deductive component of the cycle of scientific inference.

A modern scientist can fairly be described as an inductive agent loaded to breaking point by complexity. Reporting from a sector where the strain is especially severe, Ross King describes in this volume an application of computer induction to the prediction of protein folding. Elsewhere he has written that 'it was once possible to discover the meaning of new data by carefully examining it by eye'. That time is, of course, long past. Today, decision supports from statistical data analysis are pressed into service. But now even these impressive constructions are proving inadequate to such complex requirements as those of biotechnology for empirical theories of structure–activity relationships, and the requirement for better models of our planet as a basis of rational plans for the next century.

At some stage in the mechanized analysis of any sufficiently complex problem, further progress (as indicated for example in the chapter by Mozetič, Bratko, and Urbančič) has to await intelligible mechanization of the underlying relations of cause and effect. The wheel here comes full circle. John McCarthy's paper of just 30 years ago, 'Programs with common sense', placed at the core of AI's coming tasks the need for a machine-oriented logic capable of expressing causality in everyday life. Progress has subsequently been made, but in its unrestricted form McCarthy's plan remains ambitious. By restricting the aim of mechanizing causal reasoning to defined domains of scientific study we may find both a measure of tractability and also uncommon rewards.

Not the least reward must surely be the sense of mutual usefulness among disciplines, which forms the living cement of our invisible college.

June 1990

Donald Michie
Editor-in-Chief

ACKNOWLEDGEMENTS

Our grateful thanks are due to the Estonian Academy of Sciences, in particular the Institute of Cybernetics who hosted the meeting which provided the basis for this volume in their beautiful city of Tallinn. Thanks are also due to Knowledgelink (Intelligent Terminals Limited), to the Turing Institute for help with travel and resources for the work of editing the papers printed here, and the many who helped with the conference organization and the editorial process. In particular we would like to thank Irene Brebner, Pearl Guthrie, Liina Kesküla, and Catherine McCrae.

CONTENTS

MECHANICS OF KNOWLEDGE PROCESSING

INDUCTIVE FORMATION OF PROGRAMS AND DESCRIPTIONS

OPTIMALITY AND ERROR IN LEARNING SYSTEMS

QUALITATIVE REPRESENTATIONS OF KNOWLEDGE

MECHANICS OF KNOWLEDGE PROCESSING

1

Modularity of Knowledge

Ė. Tyugu
Institute of Cybernetics,
Estonian Academy of Sciences, USSR

When you lose a game of chess to a computer then don't pretend that you didn't think at the game.

S. Maslov

Abstract

Merging different kinds of knowledge in problem-solving is discussed. Several formal calculi are considered as knowledge representation means, and uniting calculi in the NUT programming system is described.

This paper has been inspired by the last book of Sergei Maslov [1] where he described a tower of deductive systems as a representation of scientific knowledge about the world. He illustrated the usage of formal calculi by numerous examples from biology, economics, and technology.

1. INTRODUCTION

We shall discuss here modularity of knowledge in the large. This is not breaking the whole of available knowledge into uniformly represented parts. We are interested in merging various kinds of knowledge and using them all together for achieving some hard goal. The question is: 'How to combine different knowledge representations and handling techniques in problem-solving systems?'

Experience shows that no universally efficient knowledge representation and handling technique exists. On the contrary—a number of very different methods have been developed for solving practically interesting problems in various domains. When considering human intelligence one can also distinguish basically different knowledge-handling mechanisms that are associated with the left and right parts of the brain, that is with logical and intuitive ways of thinking. We can hope that using different knowledge-handling methods in combination will help us to improve the intellectual capabilities of Artificial Intelligence (AI) systems designed for practical applications.

At first glance, blackboard systems seem to be a good example of modularity of knowledge in the large. This is true only when we are considering aspects of implementation. Blackboard systems provide a framework for implementing modularity of knowledge, but they do not help us in finding suitable forms of knowledge representation.

This paper is based on an assumption that any *knowledge system* (KS) which is knowledge representation plus inference engine can be reduced to a formal calculus that adequately represents knowledge processing in this KS. This statement become trivial as soon as we loosen the requirement of adequacy: on a sufficiently low level we can use Turing machines or Post's systems for representing information processing in computers.

Secondly, this paper elaborates on an observation that any successful AI system contains more than one KS, that is it is based on several calculi combined with each other in non-trivial ways. The latter means that there is no obvious natural way to build a single calculus preserving the requirement of adequacy. Putting together calculi of various knowledge systems mechanically would give us a tower of Babel of formal languages—a calculus that is incomprehensible as well as inefficient.

Nevertheless, all purely procedural forms of knowledge can be represented by a single calculus. Any representation of computable functions together with application rules for functions can be used for this purpose (Markov's normal algorithms, recursive functions, etc.). Let us call the calculus chosen the calculus of computable functions (CCF). It seems that CCF is present in any sufficiently general knowledge-based system, because procedural knowledge is a convenient means for providing extensibility to a knowledge-based system.

As soon as we intend to apply procedural knowledge automatically, another calculus is needed for invoking programs. We have good examples of systems where two calculi are used, one for procedural knowledge and another for control of computations. PROLOG combines Horn clause logic with CCF, structural synthesis of programs uses intuitionistic propositional calculus (IPC) for control and CCF for the procedural part.

We have developed programs that contain more than two KS's. The system PRIZ [2] and MicroPRIZ [3], besides the CCF and IPC, also use a rewriting system as a user-friendly front end. It transforms specifications written in a high-level specification language into a set of specific axioms of a formal calculus.

One more calculus is added to those mentioned above in the systems ExpertPRIZ and NUT [4]. ExpertPRIZ is an extention of MicroPRIZ that combines inductively built knowledge bases supporting simple decision-tree logic with the three calculi of PRIZ. The NUT system combines first order calculus of productions with PRIZ calculi.

2. FORMAL CALCULI AS KNOWLEDGE REPRESENTATION MEANS

There are many papers on using logic for knowledge representation [5].

Our thesis is that in all cases when we use knowledge, making inferences step by step, we can build a calculus that represents this knowledge and the inference engine. It is obvious that this thesis cannot be proved formally. However, looking at numerous examples we can find good evidence in favour of this thesis. First of all, making inferences means using knowledge in a deductive way, and in his book [1], Maslov has described a number of calculi, called also deductive systems, which are formalizations of knowledge in various domains. He has defined the concept of calculus in a very general way that suits us well:

> There are a certain number of initial objects and a certain number of rules for generating new objects from the initial objects and from those already constructed. To put it another way: There are an initial position (state) and 'rules of the game' (rules for transition from one state into another). A system of this kind is called a *deductive system*, or a *calculus*.

Let us consider semantic networks as an example of knowledge systems and let us try to build a calculus for them. There are various kinds of semantic networks and different inference mechanisms for working on these networks.

Bearing in mind that any semantic network is a marked graph, we can represent it as a collection of arcs. For instance, Figure 1 shows a representation of explicit and implicit time-relations in the following text:

> John must pick up his report in the morning and have a meeting after lunch. After the meeting he will give the report to me.

The arcs of the network will be objects of the calculus we are building. In this example we have the following objects:

before (lunch, morning)
after (morning, lunch)
after (lunch, have a meeting)
after (have a meeting, give)
at-the-time (morning, pick up)

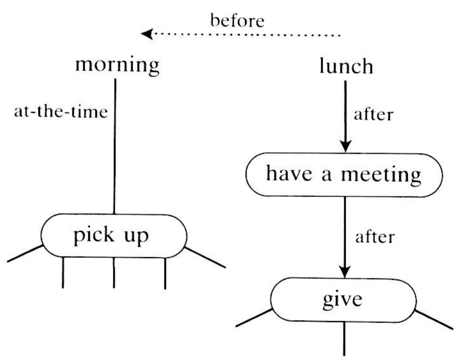

Figure 1. Time-relations.

Sometimes it is more convenient to use another representation of the network, looking at nodes as relations that bind all their neighbouring nodes. In this case the objects will be nodes, used as relations between the other nodes (their neighbours).

Inference on a semantic network is done by propagating facts (or, more generally, 'pieces of knowledge') along the arcs of the network. As a result, the network itself or the marking of its nodes is changed. Inference rules, as usual, are schemes of the following form:

$$S_1, \ldots, S_k \vdash S$$

where objects $S_1, \ldots, S_k$ are premises and the object S is a conclusion. In our case there are rules for transitivity of some time-relations, for instance:

before (x,y), before $(y,z) \vdash$ before (x,z).
after $(x,y) \vdash$ before (y,x).
at-the-time (x,z), before $(y,z) \vdash$ before (y,x).

Applying these rules we can make inferences like:

after (lunch, have a meeting) $\vdash$ before (have a meeting, lunch) at-the-time (pick up, morning), before (lunch, morning) $\vdash$ before (lunch, pick up)

and add new arcs to the graph.

3. HOW TO COUPLE CALCULI?

First of all, let us consider briefly the implementational aspects of coupling different calculi. There are well-known ways of implementing a system which consists of several interacting experts:

(1) building a network of communicating actors (processes, experts) [6];
(2) using blackboard architecture [7, 8, 9];
(3) using broadcasting as a means of communication between the experts.

All these ways can be used for writing knowledge systems, each of which will be then represented as an 'expert' with its own knowledge representation forms and inference engine.

A network of experts can be efficient for loosely coupling several knowledge systems. Object oriented programming is suitable for this purpose, because message passing can be directly used for communication between the KS's. Actually, any tools for programming communicating sequential processes can be used.

Another way to achieve the same goal is to use 'broadcasting'. In this way we can organize knowledge systems to show collective behaviour that mimics the behaviour of a group of co-operating human experts.

The closest co-operation between the knowledge systems can be provided in a blackboard system. In this case, a considerable amount of knowledge (the blackboard) is visible for all knowledge systems. The question remains, how does each KS understand the knowledge on the blackboard? But this is one of the principal questions that need to be considered when writing the KS.

In order to choose one or other of the architectures we must consider the principles of writing a KS. Some useful hints can be obtained from pure logic.

In proof theory we can find examples of successful decomposition of theories. Roughly speaking, sometimes a theory can be split into several parts, so that different inference methods can be applied and efficiency of search can be significantly improved. The following two techniques are worthy of mention:

(1) constructing a set of admissible inference rules;

(2) using a metatheory.

Both these techniques have analogies in knowledge-based systems.

Yet another useful way of combining calculi comes from logic. Let us take a constructive non-categoric theory (that is, a theory that has more than one model). Models of constructive theories can again be considered as calculi. So we have a non-trivial relation of interpretation ('to be a model of') between the calculi. Probably the relation of interpretation is the most widely used relation between the calculi in knowledge-based systems.

In papers on algebraic data types which are represented as heterogeneous algebras, we can find a number of relations between algebras [10]: abstraction, concretization, extension, restriction, enrichment, etc. To an extent, these relations are also meaningful for calculi of knowledge systems.

4. UNITING LOGIC WITH PROCEDURAL KNOWLEDGE

A good example of uniting logic and procedural knowledge is PROLOG. It combines Horn clause logic (HCL) with a procedural knowledge system (CCF). Connection between the HCL and the CCF in PROLOG is established through the realization of functional constants and some predicates as pre-programmed procedures.

A text in PROLOG, that is, the logical part of a PROLOG 'program' consists of clauses

$$A \,\&\, \ldots \,\&\, B \rightarrow C$$