Lecture Notes in Artificial Intelligence 1630

Subseries of Lecture Notes in Computer Science
Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science
Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Singapore
Tokyo

Matthew M. Huntbach Graem A. Ringwood

Agent-Oriented Programming

From Prolog to Guarded Definite Clauses

Springer

Series Editors
Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Authors

Matthew M. Huntbach
Graem A. Ringwood
Department of Computer Science, Queen Mary and Westfield College
Mile End Road, London E1 4NS, UK
E-mail: {mmh/gar}@dcs.qmw.ac.uk

Cataloging-in-Publication data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Huntbach, Matthew M.:
Agent-oriented programming : from prolog to guarded definite clauses / Matthew M. Huntbach ; Graem A. Ringwood. - Berlin ; Heidelberg ; New York ; Barcelona ; Hong Kong ; London ; Milan ; Paris ; Singapore ; Tokyo : Springer, 1999
(Lecture notes in computer science ; Vol. 1630 : Lecture notes in artificial intelligence)
ISBN 3-540-66683-4

CR Subject Classification (1998): I.2, D.1.6, D.1.3, D.3, C.2.4

ISBN 3-540-66683-4 Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera-ready by author
SPIN 10703359   06/3142 – 5 4 3 2 1 0   Printed on acid-free paper

Preface

A book that furnishes no quotations is, me judice, no book – it is a plaything.

TL Peacock: Crochet Castle

The paradigm presented in this book is proposed as an agent programming language. The book charts the evolution of the language from Prolog to intelligent agents. To a large extent, intelligent agents rose to prominence in the mid-1990s because of the World Wide Web and an ill-structured network of multimedia information. Agent-oriented programming was a natural progression from object-oriented programming which C++ and more recently Java popularized. Another strand of influence came from a revival of interest in robotics [Brooks, 1991a; 1991b].

The quintessence of an agent is an intelligent, willing slave. Speculation in the area of artificial slaves is far more ancient than twentieth century science fiction. One documented example is found in Aristotle's *Politics* written in the fourth century BC. Aristotle classifies the slave as "an animate article of property". He suggests that slaves or subordinates might not be necessary if "each instrument could do its own work at command or by anticipation like the statues of Daedalus and the tripods of Hephaestus". Reference to the legendary robots devised by these mythological technocrats, the former an artificer who made wings for Icarus and the latter a blacksmith god, testify that the concept of robot, if not the name, was ancient even in Aristotle's time. Aristotle concluded that even if such machines existed, human slaves would still be necessary to render the little personal services without which life would be intolerable.

The name *robot* comes from the Czech words for serf and forced labor. Its usage originates from Karel Capek's 1920s play *Rossum's Universal Robots* in which Rossum, an Englishman, mass-produced automata. The play was based on a short story by Capek's brother. The robots in the play were not mechanical but grown chemically. Capek dismissed "metal contraptions replacing human beings" as "a grave offence against life". One of the earliest film robots was the replica Maria in Fritz Lang's 1927 classic *Metropolis*. The academic turned science fiction writer Isaac Asimov (1920–1992) introduced the term *robotics* when he needed a word to describe the study of robots in *Runaround* [1942]. Asimov was one of the first authors to depart from the Frankenstein plot of mad scientist creating a monster and to consider the social implications of robots.

An example of an automaton from the dark ages is a vending machine for holy water proposed by Hero of Alexandria around 11 AD. A modern reincarnation is Hoare's choc machine [Hoare, 1985] developed to motivate the computational model CSP (Communicating Sequential Processes). The word *automaton*, often used to describe computers or other complex machines, comes from the same Greek root as automobile meaning self-mover. Modern science owes much to the Greek tradition. Analysis of the forms of argument began with Empedocles and the importance of observation stems from Hippocrates. The missing ingredients of Greek science compared with the science of today were supplied by the Age of Reason. These were the need for deliberately contrived observation - experiments; the need for inductive argument to supplement deduction; and the use of mathematics to model observed phenomena. The most important legacy of seventeenth century science is technology, the application of science. Technology has expanded human capability, improved control over the material world, and reduced the need for human labor. Willing slaves are, perhaps, the ultimate goal of technology.

Industrial robots appeared in the late 1950s when two Americans, Devol and Engelberger, formed the company Unimation. Take-up was slow and Unimation did not make a profit for the first fourteen years. The situation changed in the mid-1980s when the automobile industry, dissatisfied with trade union disruption of production, turned to robot assembly. However, the industrial robot industry overextended as governments curtailed trade union power and the market saturated. Many firms, including Unimation, collapsed or were bought out by end product manufacturers. Today, the big producer is Japan with 400 000 installed robots compared to the US with over 70 000 and the UK with less than 10 000.

With pre-Copernican mentality, people will only freely admit that humans possess intelligence. (This, possibly, should be qualified to mean most humans on most occasions.) Humans can see, hear, talk, learn, make decisions, and solve problems. It seems reasonable that anyone attempting to reproduce a similar artificial capability would first attempt emulating the human brain. The idea that Artificial Intelligence (AI) should try to emulate the human nervous system (brain cells are nerve cells) was almost taken for granted by the twentieth century pioneers of AI. Up until the late 1960s talk of *electronic brains* was common place.

From Rossum's Universal Robots in Carel Kapek's vision to HAL in the film *2001*, intelligent machines provide some of the most potent images of the late twentieth century. The 1980s were, indeed, a good time for AI research. In the 1970s AI had become something of a backwater in governmental funding, but all that changed dramatically because of the Japanese Fifth Generation Initiative. At the beginning of the 1980s, MITI, the Japanese equivalent of the Department for Trade and Industry, announced that Japan was to concentrate on knowledge based systems as the cutting edge of industrial development. This sent tremors of commercial fear through the corridors of power of every country that had a computing industry. These governments had seen national industries such as shipbuilding, automobile manufacturing, and consumer electronics crumble under intensive Japanese competition. In what retrospectively seems to be a halfhearted attempt to target research funds to industrially relevant information technology, a few national and multinational research programs were initiated. A major beneficiary of this funding was AI. On short timescales, commercial products were supposed to spring forth fully armed from basic research.

Great advances in computer hardware were made in this decade with computing power increasing a thousandfold. A computer defeated the world backgammon champion and a computer came in joint first in an international chess tournament, beating a grandmaster along the way. This, however, did not augur the age of the intelligent machine. Genuine progress in AI has been painfully slow and industrial take-up has been mainly limited to a few well-publicized expert systems.

In the mid-1980s, it was envisaged that expert systems that contain thousands of rules would be widely available by the end of the decade. This has not happened; industrial expert systems are relatively small and narrowly focused on specific domains of knowledge, such as medical diagnosis. As researchers tried to build more extensive expert systems major problems were encountered.

There are two reasons why game playing is the only area in which AI has, as yet, achieved its goal. Though complex, chess is a highly regular, codifiable problem compared with, say, diagnosis. Further, the algorithms used by chess playing programs are not usually based on expert systems. Rather than soliciting knowledge from chess experts, successful game playing programs rely mainly on guided brute force search of all possible moves using highly powerful conventional multiprocessor machines. In reality, AI has made as much progress as other branches of software engineering. To a large extent, its dramatic changes of fortune, boom and bust, are due to fanatical proponents who promise too much. The timescale predictions of the Japanese now look very fanciful indeed. AI has been oversold more than once.

A common reaction to the early efforts in AI was that successful replication of human skills would diminish human bearers of such skills. A significant outcome of AI research is how difficult the simplest skills we take for granted are to imitate. AI is a long-term problem, a marathon, and not a sprint competition with the Japanese. Expert systems are only an early staging post on the way to developing intelligent machines.

AI pioneered many ideas that have made their way back into mainstream computer science. These include timesharing, interactive interpreters, the linked list data type, automatic storage management, some concepts of object-oriented programming, integrated program development environments, and graphical user interfaces. Whatever else it achieved, the Japanese Initiative provoked a chain of increased governmental funding for Information Technology reaction around the world from which many, including the authors, benefited.

According to Jennings et al. [1998], the fashion for agents "did not emerge from a vacuum" (who would have imagined it would?) Computer scientists of different specializations artificial intelligence, concurrent object-oriented programming languages, distributed systems, and human-computer interaction converged on similar concepts of agent. Jennings et al. [1998] state, "Object-oriented programmers fail to see anything novel or new in the idea of agents," yet they find significant differences between agents and objects. This is because their comparison only considers (essentially) sequential object-oriented programming languages such as Java. Had they considered concurrent object-oriented programming languages they would have found fewer differences.

Three languages have been promoted for agent development: Java, Telescript, and Agent-TCL. None of these are *concurrent* object-oriented languages. Java, from SUN Microsystems, is advocated for agent development because it is platform independent and integrates well with the World Wide Web. Java does, however, follow the tradition of interpreted, AI languages but is it not sympathetic to symbolic programming. Telescript, from General Magic, was the first commercial platform designed for the development of mobile agents. The emphasis is on mobility rather than AI applications. Agent-TCL [Gray et al., 1996] is an extension of TCL (Tool Command Language) which allows mobile code. While string based, TCL does not have a tradition of AI applications. Programs are not inductively defined, as is the case with Lisp or Prolog.

This monograph describes a concurrent, object-oriented, agent programming language that is derived from the AI tradition. A working knowledge of Prolog is necessary to fully appreciate the arguments. The monograph is divided into two parts. The first part, Chaps. 1–5, describes the evolution of the paradigm of Guarded Definite Clauses (GDC). If the paradigm is serious, and more than a fashion, then it is necessary to to describe its applications. This is done in the second part of the monograph, Chaps. 6–10. To set the paradigm in context, Chap. 1 provides an irreverent survey of the issues of AI. Chap. 2 completes the background to the paradigm with a retrospective rationale for the Japanese Fifth Generation Initiative. Chap. 3 describes how the paradigm evolved from Prolog with the environment change of multiprocessor machines. Included in this chapter is a chronology of the significant developments of GDC. Chap. 4 explores the manifestations of the vital ingredient of the paradigm - event driven synchronization. Chap. 5 compares and contrasts the language evolved with actor languages. The main difference is that GDC is an actor language with the addition of inductively defined messages.

The second part of the book begins with Chap. 6, which illustrates the advantages of GDC in parallel and distributed search. Chap. 7 describes the specialization to distributed constraint solving. Chap. 8 generalizes the chapters on search to meta-interpretation. An affinity for meta-interpretation has long been a distinguishing feature of AI languages. Chap. 9 describes how the overhead of meta-interpretation can be assuaged with partial evaluation. Chap. 10 concludes with the application of GDC to robotics and multi-agent systems.

While GDC as such is not implemented, it differs only marginally from KL1C, a language developed by the Japanese Fifth Generation Computer Systems Initiative. The Institute for New Generation Computer Technology (ICOT) promoted the Fifth Generation Computer Systems project under the commitment of the Japanese Ministry of International Trade and Industry (MITI). Since April 1993, ICOT has been promoting the follow-on project, ICOT Free Software (IFS), to disseminate the research:

> *According to the aims of the Project, ICOT has made this software, the copyright of which does not belong to the government but to ICOT itself, available to the public in order to contribute to the world, and, moreover, has removed all restrictions on its usage that may have impeded further research and development in order that large numbers of researchers can use it freely to begin a new era of computer science.*

AITEC, the Japanese Research Institute for Advanced Information Technology, took over the duties of ICOT in 1995. The sources of KL1 and a number of applications can be obtained via the AITEC home page: http://www.icot.or.jp/AITEC. KL1C runs under Linux and all the GDC programs in this monograph will run with little or no modification.

Despite their best efforts, the reader will find that the authors' cynicism shows through since they, like Bernard Shaw, believe that all progress in scientific endeavor depends on unreasonable behavior. In Shaw's view the common perception of science as a rational activity, in which one confronts evidence of fact with an open mind, is a post-rationalization. Facts assume significance only within a pre-existing intellectual structure that may be based as much on intuition and prejudice as on reason. Humility and reticence are seldom much in evidence and the scientific heroes often turn out to be intellectual bullies with egos like carbuncles.

The authors are very grateful to Jean Marie Willers and Peter Landin for the onerous task of proof reading earlier drafts of this monograph. Thanks are also due to our editors at Springer-Verlag, Ingrid Beyer, Alfred Hofmann, and Andrew Ross. Each author would like to say that any serious omissions or misconceptions that remain are entirely the fault of the other author.

January 1999

Matthew M Huntbach
Graem A Ringwood

Contents