Research Notes in Artificial Intelligence

DAVID S TOURETZKY

The Mathematics of Inheritance Systems

Pitman, London

Morgan Kaufmann Publishers, Inc., Los Altos, California

David S Touretzky
Computer Science Department
Carnegie-Mellon University

The Mathematics of Inheritance Systems

Pitman, London

Morgan Kaufmann Publishers, Inc., Los Altos, California

PITMAN PUBLISHING
128 Long Acre, London WC2E 9AN

First published 1986, reprinted 1988

Available in the Western Hemisphere from
MORGAN KAUFMANN PUBLISHERS, INC.,
2929 Campus Drive, San Mateo, California 94403

ISSN 0268-7526

British Library Cataloguing in Publication Data

Touretzky, David S.
The mathematics of inheritance systems.—
(Research notes in artificial intelligence,
ISSN 0268-7526).
1. Artificial intelligence—Mathematics
I. Title II. Series
006.3 Q335

ISBN 0-273-08765-7

Library of Congress Cataloging in Publication Data

Touretzky, David S.
The mathematics of inheritance systems.

(Research notes in artificial intelligence)
Revision of the author's thesis (doctoral)—
Carnegie-Mellon University, 1984.
Bibliography: p.
1. Computer architecture. 2. Artificial intelligence.
3. Logic, Symbolic and mathematical. I. Title.
II. Title: Inheritance systems. III. Series.
QA76.9.A73T67 1986 006.3 86-2954
ISBN 0-934613-06-0

Reproduced and printed by photolithography
in Great Britain by Biddles Ltd, Guildford

The Mathematics of Inheritance Systems

Preface

This book is a revised version of my doctoral dissertation, completed at Carnegie-Mellon University in 1984. Its primary aim is to present a formal mathematical theory of a popular reasoning strategy that to date has been defended mostly by appeals to intuition: multiple inheritance with exceptions to inherited properties. Virtually all knowledge representation schemes and object-oriented programming languages include some sort of inheritance mechanism. Common examples are FRL, KRL, KLONE, NETL, Simula, Smalltalk, Flavors, LOOPS, and Ada. But the lack of a formal theory of inheritance hid some defects in existing inference algorithms. One was the incorrect treatment of networks with multiple consistent theories; another was a tendency to reason incorrectly when true but redundant statements are present. Both these problems can be eliminated once a more rigorous understanding of inheritance has been achieved.

Reasoning with exceptions is complicated because it involves operations outside of classical first order logic. The formalism I have developed to express the nonstandard inference rules that underlie inheritance bears some relation to default and nonmonotonic logics, but it includes an important hierarchical notion these other systems lack. The formalism and the definitions that follow allow us to prove theorems about the consistency, uniqueness, and constructability of inheritance theories, and lead to a formal semantics for inheritance in terms of constructable lattices of predicates.

The second major component of this thesis is the application of the inheritance theory to the analysis of a connectionist computer architecture, parallel marker propagation machines (PMPM's), of which the best-known example is Fahlman's NETL Machine. PMPM's and related connectionist schemes have aroused considerable interest as high speed inference engines for AI. The formal theory serves as a correctness specification for PMPM inheritance algorithms and allows us to show that a PMPM can reason correctly only for certain limited network topologies. However, through a technique known as "conditioning," the topology of a network can be altered to force the PMPM to produce correct results in the more general case.

The final task of this thesis is to demonstrate that the topological, network-oriented approach to reasoning as found in property inheritance systems can be successfully applied to other inference problems. We consider the problem of inheritable relations such as "bigger than" in the sentence "elephants are bigger than bread boxes," and the sorts of inferences we should be able to make from them, *e.g.*, that particular elephants are bigger than particular bread boxes, modulo known exceptions. This type of reasoning can be formalized as an extension to property inheritance, and after this is done we can return to the analysis of PMPM architectures and produce new inference algorithms, correctness specifications, and theoretical results.

D.S.T.

December, 1985

Acknowledgements

The work reported here, although applicable to inheritance systems in general, began as an attempt to answer certain questions raised by Scott Fahlman's parallel knowledge representation system, NETL. I am grateful to Scott for creating such a pleasing and stimulating intellectual puzzle, and for allowing me to happily explore its intricacies as his graduate student. One piece of the puzzle is solved now, but others remain. Scott also aided and abetted most of my other adventures as a graduate student. The combination of freedom and unhesitating support he provided was invaluable.

Jon Doyle, the second member of my committee, taught me to write mathematics. (I, however, take credit for any remaining flaws in the writing.) Over a two year period Jon and I worked together on finding the right intuition for inheritance and rigorously formalizing it. The mathematical analysis I present here would not have been possible without his guidance. Jon read countless versions of the early chapters of the thesis; his high level of enthusiasm was a wonderful antidote for occasional discouragement.

I am grateful to Dana Scott for asking some tough questions about the meaning of inheritance, which helped guide me down the path to a formal analysis. Dana also contributed to the lattice theory part of the thesis.

James Allen, the fourth member of my committee, provided insight and encouragement in several useful discussions and helped publicize the work after the defense.

I thank David Etherington of the University of British Columbia, who was kind enough to corresponded with me periodically on such topics as inheritance and default logic, for the insights he provided and for his careful reading of the final draft of the thesis.

For the revised edition published by Pitman, Sandy Koi turned almost a hundred crudely scrawled diagrams into professional quality illustrations. Readers familiar with the original dissertation will no doubt appreciate her talents as much as I now do.

During five of my six years in graduate school I was supported as a fellow of the Fannie and John Hertz Foundation. It is a pleasure to be able to acknowledge here the Foundation's generosity.

Finally, I thank my family, and my friends: Lars Ericson, Loretta Ferro, Cynthia Lamb, Al Rotella, Andi Swimmer, and Cindy Wood.

This thesis is dedicated to the Allegheny County Airport in West Mifflin, Pennsylvania, where I spent many hours as a pilot and flight instructor — a welcome respite from thinking about inheritance.

Contents

1 Inheritance Hierarchies

"This structure of concepts is formally called a hierarchy and since ancient times has been a basic structure for all Western knowledge."

— Robert M. Pirsig, *Zen and the Art of Motorcycle Maintenance*

"How anybody can get useful work done when restricted to hierarchical inheritance is beyond me; the world just doesn't work hierarchically."

— Daniel L. Weinreb, Symbolics, Inc.

1.1 Introduction

An inheritance system is a representation system founded on the hierarchical structuring of knowledge. Virtually all knowledge representation languages and object-oriented programming languages are organized around such systems. As Weinreb notes, inheritance is often extended to more complex domains than pure hierarchies (which are just tree structures), but even so, the essential idea of a hierarchical ordering of objects remains. Well-known systems with inheritance include FRL (Roberts and Goldstein, 1977), KRL (Bobrow and Winograd, 1977), SRL (Wright and Fox, 1983), KLONE (Brachman and Schmolze, 1985; Brachman, *in press*), NETL (Fahlman, 1979), Simula (Dahl, 1968), Smalltalk (Borning and Ingalls, 1982), Flavors (Weinreb, 1981), LOOPS (Bobrow and Stefik, 1981), and Ada (DoD, 1982). Until recently, despite their widespread use, inheritance systems with exceptions remained unformalized; the lack of a formal theory hid some defects in the behavior of existing systems. This thesis presents a formal mathematical theory of inheritance with exceptions and shows how to correct the flaws in existing inheritance systems. In later chapters, the formal theory of inheritance is applied to the formal analysis of a massively parallel computer architecture known as a parallel marker propagation machine, of which the most well-known example is Fahlman's NETL Machine (Fahlman, 1979). This machine has been proposed as a high speed inference engine for applications in AI.

1.2 Taxonomic hierarchies

In AI, as in other endeavors at organizing knowledge, regularity can be exploited by creating abstractions. For our purposes, an abstraction is a collection of properties shared by the members of a set. For example, if a knowledge base contains many references to gray, long-nosed, four-legged, peanut-eating jungle dwellers, we might be motivated to create an abstraction with these properties, perhaps giving it a name such as "elephant." Abstractions can also share properties, as individuals do. Elephants and sheep have some properties in common: both are warm-blooded and bear live young. One abstraction that includes both elephants and sheep is mammal. When abstractions organized by inclusion relations form a tree, the result is known as a taxonomic hierarchy, or in AI, an inheritance hierarchy. More complex organizations are possible when the tree is replaced by a general directed graph.

1.3 Advantages of hierarchical structuring

The primary advantage of hierarchical structuring is that it is an efficient method of representation. In the case of the gray, long-nosed, four-legged *et ceteras* mentioned above, the naive method of representing them would list the properties of each individual separately, but after creating the elephant abstraction listing the properties the individuals have in common, we can fully describe each one simply by saying that he or she is an elephant. To efficiently represent lions and tigers and bears as well as elephants, we could create a higher class, such as mammal, to describe the properties common to all these animals.

A second advantage of hierarchical structuring, after representational compactness, is that it makes searching more efficient. Just as we can search a binary tree of alphabetized names faster than an unordered list of names, a set of assertions organized hierarchically can be searched faster than an unordered list of assertions. Often we will have more than one retrieval task in mind, with each task requiring a different organization of the hierarchy. This calls for multiple, overlapping, orthogonal groupings of properties, and is known as multiple inheritance. Multiple inheritance provides tremendous representational flexibility, but it also introduces semantic problems that do not arise in tree-structured (simple inheritance) systems. These problems will be discussed later in the chapter.

1.4 The necessity of exceptions

Mandatory inheritance of properties is too inflexible for representing real-world knowledge (Fox, 1979). The real world contains exceptions to almost every generalization. Although most people's ideal elephant is a gray, four-legged, peanut-eating jungle dweller, there are non-gray elephants, three-legged elephants, elephants who don't eat peanuts, and elephants who don't live in jungles. If we required an abstraction to hold true for all members of a class, very few properties could be placed there. Instead, most inheritance systems allow individuals to override the properties of an abstraction that do not apply to them. For example, if we assert that Clyde is an elephant but is not gray, Clyde may inherit four-leggedness, long nosedness, jungle dwelling, and other properties from the elephant abstraction, but he will not inherit grayness.

Classes as well as individuals may have exceptional properties. It is useful to assert that mammals have four legs, but humans are mammals with only two. If we make four-leggedness a property of mammals, then lions and tigers and bears and elephants will become four-legged by inheritance. Humans won't if we state explicitly that they are two-legged. Ahab, a one-legged human, is an exception to an exception.

1.5 Two actual inheritance systems

FRL is a typical frame-based inheritance language (Roberts and Goldstein, 1977). Figure 1.1 shows how elephants, mammals, humans, and their respective numbers of legs are represented in FRL. Abstractions are encoded in FRL as *frames*, and the inclusion relation between them is called an AKO (A Kind Of) link. The properties associated with a frame are called its *slots*. In figure 1.1, each frame has a "number of legs" slot. Mammal has the value "4" in its number of legs slot, while human has the value "2". The other frames have no value in their number of legs slot.

To determine how many legs Clyde has in figure 1.1, FRL first checks the number of legs slot of the Clyde frame. Finding no value there, it proceeds up the AKO hierarchy to search the number of legs slots of higher frames. At the elephant frame the number of legs slot is also empty. At the mammal frame the value "4" is found, so FRL concludes

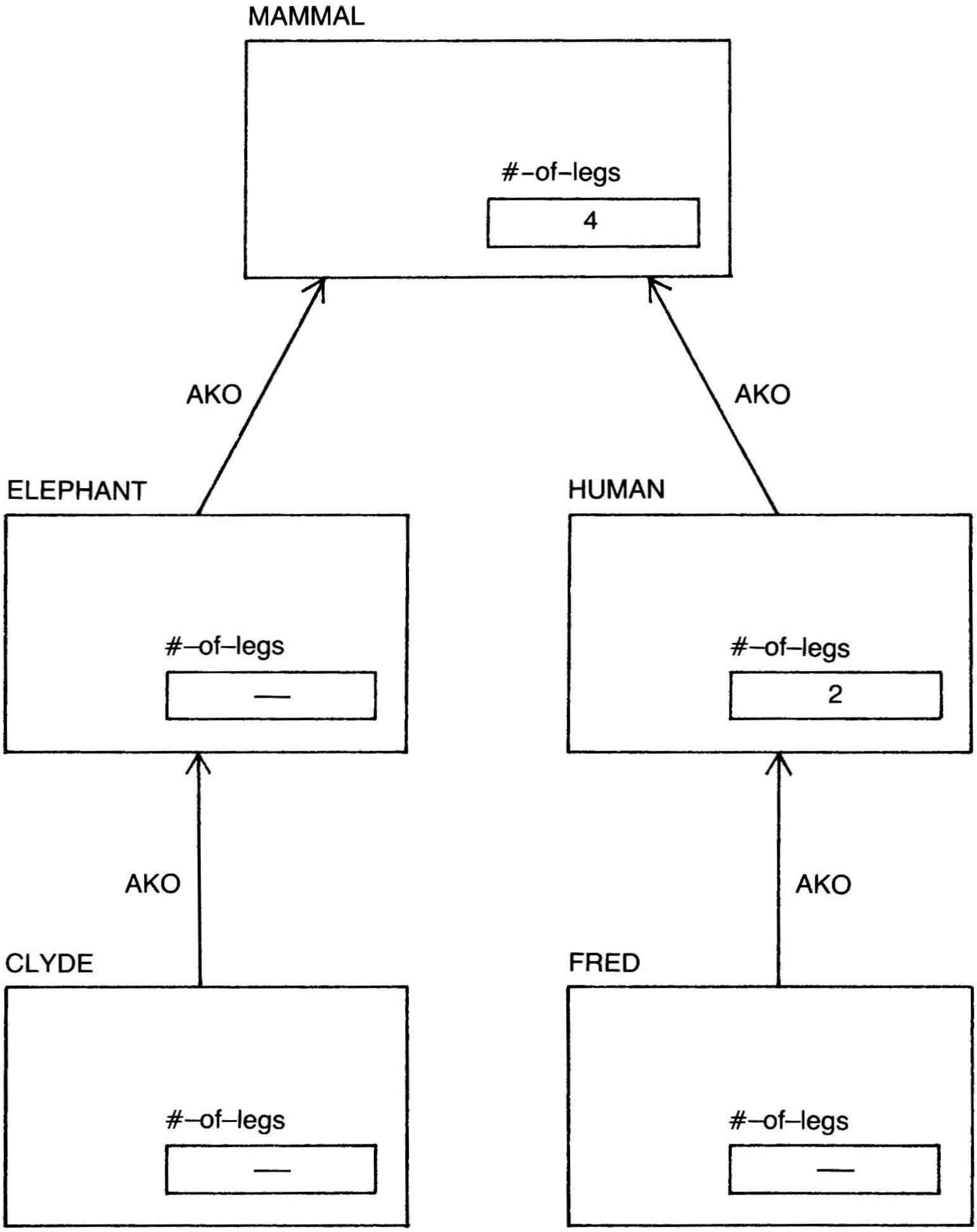

Figure 1.1: A frame-based representation of mammals, elephants, humans, and their respective numbers of legs.