Mathematical Reasoning with Diagrams

Mateja Jamnik

ISBN: 1-57586-324-3

Contents

Foreword

The advent of the modern computer in the nineteen-fifties immediately suggested a new research challenge: to seek ways of programming these versatile machines which would make them behave as much like intelligent human beings as possible. After fifty years or so, this quest has produced some intriguing results, but until now progress has been disappointingly slow. This book is a welcome and encouraging sign that things may at last be about to change.

To be sure, what might be called the *purely logical* approach has recently produced some noteworthy successes. Consider the following two examples:

On October 10, 1996, a rigorous proof of the Robbins Conjecture was found by William McCune's theorem proving program EQP at the Argonne National Laboratory. This problem had been unsolved since the mid-nineteen-thirties.

On May 11, 1997, the (then) world chess champion Garry Kasparov lost a six game match against the computer program Deep Blue with a score of 2.5 to 3.5: two wins for Deep Blue, one win for Kasparov and three draws.

Both these programs, however, use quite *non-human* methods. Neither of them is at all based on how the mind of the human expert actually works. It is in fact very difficult to find out what the natural intellectual processes of expert humans really are. To program a computer to solve the kind of problems that such experts can solve, the *purely logical* approach has hitherto been found more effective than the *heuristic* approach: to invent systematic algorithms for solving the problems rather than trying to discover, and then to imitate, the relevant human skills. Not that the heuristic approach has been ignored. On the contrary, heuristic problem-solving and the programming of "expert system" have been prominent computational methodologies in Artificial

Intelligence and Operations Research from the beginning. But in mathematical theorem proving, at least, the *purely logical* approach has far outpaced the heuristic approach.

The fact is that the latter has been severely hampered by a shortage of insights into mathematical cognition and ratiocination. Professor Alan Bundy's group at the University of Edinburgh has for some time now been patiently and insightfully seeking to remedy this shortage. Mateja Jamnik developed the ideas described in this book as a member of Bundy's group, and the book beautifully illustrates what the group has been doing.

What is novel about Mateja Jamnik's work is that she has found an explanation of at least part of the mystery of how humans are able to "see" the truth of certain mathematical propositions merely by contemplating appropriate diagrams and constructions. This ability to "see" is one of the really fundamental components of the human mathematical cognitive repertoire. As the late great mathematician G. H. Hardy put it: in the last analysis there is no such thing as "proof" – all a mathematician really does is *observe what is there*. To convince others of what he observes to be the case all he can do is *point* and say: *do you see?* Mateja Jamnik's program Diamond "sees", for example, as we do, that a 5×5 array of dots is also a nest of 5 "ells" containing respectively 1, 3, 5, 7, and 9 dots, and that there is nothing special about this special case. It and we can see that the case of 5 is but one instance of the *general* pattern whereby, *for any* n, an $n \times n$ array of dots is also a nest of n "ells" containing respectively $1, 3, 5, 7, \ldots, 2n-1$ dots. In this way it and we can directly see, as a kind of mathematical sense datum, the truth of the mathematical theorem that the number n^2 is the sum of the first n odd numbers. This act of seeing is analyzed in the program as a set of alternative decompositions of a given square array; we can see it either as a *row of columns*: or as a *column of rows*, or as a *nest of frames*, or as a *nest of ells*, or as an *array of subsquares*, and so on and on. Each of these perceptions is the same as seeing the truth of a corresponding mathematical theorem.

It is as though the program Deep Blue had been given some of the very same abilities to "see" the right chess move as Kasparov – literally to *see what he sees* when he looks at the board. But we don't know what it is that Kasparov sees.

Impressive though some of its achievements have been (such as that of EQP mentioned above), the *purely logical* approach to mathematical theorem proving is limited in scope and scientifically unedifying. The present book is an encouraging demonstration that its scope can be much widened, and its explanatory power expanded, by a fearless and patient exploration of the details of actual mathematical cognition. The logical and the heuristic approaches are beginning to come together in a most fruitful way. Mateja Jamnik is among the pioneers of a fresh new approach to an old problem.

I welcome the elegant and surprising insights of this book as ushering in a new generation of discoveries in the understanding of mathematical reasoning.

J. A. Robinson
Professor Emeritus, Syracuse University

Preface

This book started as my doctoral dissertation (Jamnik 1999) under the supervision of Alan Bundy and Ian Green at the University of Edinburgh, in what was then known as the Department of Artificial Intelligence, and had by the end of my PhD studies become the Division of Informatics.

This book is written for a mathematically minded audience. The intention is to present an exploration into the subset of the world of mathematics which can be solved with the use of pictures. Some familiarity with logic is required to understand the entire contents of the book. However, readers with little or no knowledge of logic should be able to safely omit parts of three particularly technical chapters: most of Chapter 8 and Appendix B, and parts of Chapter 4. Chapter 8 and Appendix B, in particular, are not essential to understand the general line of argument taken in the book. The specialist audience that the book is intended to attract is the automated reasoning community. The general audience that it is intended to attract is a community of scientists in artificial intelligence, computer scientists, mathematicians, philosophers, psychologists, cognitive scientists, teachers of mathematics and anybody interested in mathematical recreations.

I have attempted to make this book self-contained, and have included a comprehensive survey of other related work (see Chapter 2). In order to explain any possibly unfamiliar or unconventional use of terminology, I have included a Glossary of such terms at the end of the book.

Parts of the work described in this book have appeared in press in the past, in particular in Jamnik et al. 1999, Jamnik et al. 1998, Jamnik et al. 1997b and Jamnik et al. 1997a.

Acknowledgments

First and foremost, I would like to thank Alan Bundy for his invaluable advice, guidance and encouragement in my research. He has been an inspiration for me during these years.

I am indebted to Alan Robinson for not only being the first person to encourage me to publish this work as a book, but also for writing the foreword and giving me useful comments on a draft.

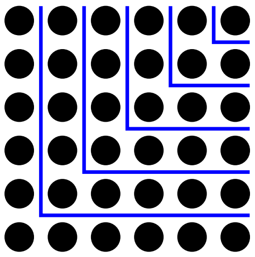
I would also like to thank Ian Green, Predrag Janičić, Nigel Shadbolt, Aaron Sloman and Keith Stenning for many useful discussions and comments on my work. Thanks to Manfred Kerber, Bob Lindsay and Toby Walsh for their comments on a draft of this book.

I especially thank Gavin Bierman for his love, understanding and encouragement throughout this time.

Lastly, to my family: *najlepša hvala moji družini za njihovo stalno ljubezen in podporo.*

This work was supported by a studentship from the Department of Artificial Intelligence at the University of Edinburgh, by a supplementary grant from the Slovenian Scientific Foundation, by a studentship from the British Overseas Research Scheme, and by EPSRC grant GR/M22031.

1 Introduction



$$n^2 = 1 + 3 + 5 + \cdots + (2n - 1)$$

— Nicomachus of Gerasa (*circa* A.D.100)
in Nelsen's *Proofs Without Words*

This book is about mathematical reasoning with diagrams. Human mathematicians often informally use diagrams when proving theorems. Diagrams seem to convey information which is easily understood by humans. For example, it requires only basic secondary school knowledge of mathematics to realize that the diagram above is a proof of a theorem about the *sum of odd natural numbers.* We call such proofs diagrammatic proofs. In this book we present an investigation into formalizing and mechanizing diagrammatic reasoning, and a concrete result of this investigation, a semi–automatic formal proof system, called Diamond (**Dia**gra**m**matic Reas**on**ing and **D**eduction), which facilitates a user to prove theorems of arithmetic using diagrams.

1.1 Motivation

It is an interesting property of diagrams that helps us to "see" and understand so much just by looking at a simple diagram. Given some basic mathematical training and familiarity with spatial manipulations, we not only know what theorem the diagram represents, but we also understand the proof of the theorem represented by the diagram and believe it is correct.

Is it possible to simulate and formalize this sort of diagrammatic reasoning on machines? Or is it a kind of intuitive reasoning particular to humans that mere machines are incapable of? Roger Penrose claims that it is not possible to automate certain diagrammatic proofs.[1] We are taking his position as a challenge and are trying to capture the kind of diagrammatic reasoning that Penrose is talking about so that we will be able to emulate some simple examples of it on a computer. Our primary motivation is not to discover diagrammatic proofs, but to study them in order to understand them better and be able to formalize them.

The importance of diagrams in many domains of reasoning has been extensively discussed by Larkin and Simon (1987), who claim that "a diagram is (sometimes) worth *ten* thousand words". The advantage of a diagram is that it concisely stores information, explicitly represents the relations among the elements of the diagram, and it supports a lot of perceptual inferences that are very easy for humans. Diagrams have been extensively used in the history of mathematics to *aid informal mathematical reasoning.* The use of diagrams in explanations of theorems and proofs of geometry dates back at least to Ancient Greece, and the time of Aristotle and Euclid. Thus it is surprising perhaps that more recently, starting with the invention of formal axiomatic logic in the sense of Frege, Russell and Hilbert, diagrams have been denied a *formal* role in theorem proving. It is generally thought by logicians that diagrams have no accepted syntax nor semantic theory which would make them rigorous enough to be used in formal proofs. Hence, in the past century, only symbolic proofs of some logic have been considered to be *formal*, and proofs that use diagrams have been considered *informal.* Only very recently, in the last two decades, have there been efforts to fill this gap and investigate whether and how diagrams can be used in formal proofs (for instance, see Funt 1980, Sowa 1984, Kaufman 1991, Barker-Plummer and Bailin 1992, Barwise and Etchemendy 1994, Shin 1995, Hammer 1995, Stenning and Oberlander 1995).

Alongside the revival of research on formal aspects of using diagrams, investigations have also been carried out in other directions with different perspectives on the use of diagrams. These can be characterized into two groups of research perspectives, namely computational and cognitive perspectives.

From a computational perspective, Lindsay (1998) devises a compu-

[1]Roger Penrose presented his position in the lecture at the International Centre for Mathematical Sciences in Edinburgh, in celebration of the 50th anniversary of UNESCO on 8 November, 1995. His point of view is elaborated in Penrose 1994a.

tational model of human reasoning with diagrams, and claims that diagrams are sometimes more efficient for solving problems than some logical machinery. Glasgow and Papadias (1992) make a distinction between visual and spatial reasoning: visual reasoning is concerned about *what* a diagram looks like, whereas spatial reasoning deals more with *where* a diagram is located relative to other diagrams. Stenning and Oberlander (1995) introduce computational models for interpreting Euler's circles (Euler 1795). They also carry out a comparative analysis of the expressiveness of diagrammatic and symbolic representations in Stenning and Oberlander 1992. One of the aspects of the computational perspective is also the issue of knowledge representation. A lot of work on various kinds of representations has been carried out by Sloman and Hayes (see Sloman 1971, Hayes 1974, Sloman 1996). Related to this work and to Glasgow's work mentioned above is an unresolved debate on the characterization of diagrammatic (or graphical or visual) and symbolic (or sentential) representations (e.g., see Narayanan 1992, Olivier 1996, Blackwell 1997, Anderson 1997, Anderson et al. 2000, Anderson et al. 2001). Since there is no consensus on the definition of each type of representation, it seems that researchers adopt their own definitions suitable for their work.

From a cognitive perspective, Johnson-Laird (1983), and Hegarty and Just (1993) argue that humans, at least in some cases, use diagrams in their mental models of a situation. Mental imagery has been studied by Pylyshyn (1981), Pinker (1985) and Kosslyn (1993), amongst others. Pylyshyn is particularly critical of mental imagery and questions claims that humans use diagrams in cognition – we may think we do, he says, but there is no conclusive evidence that the brain uses diagrammatic representations; we may even be using symbolic logical representations (Pylyshyn 1973).

Our work contributes to research on formal and computational aspects of the use of diagrams, especially in automated reasoning systems. Automated reasoning systems have their roots back in the fifties when the first programs were written that could automatically prove simple theorems of propositional logic. As a result of growing interest in the research on automated reasoning we have today many sophisticated systems such as the theorem prover of Boyer and Moore (see Boyer and Moore 1990) or Isabelle (see Paulson 1989) in which one can prove complex theorems of mathematics.

However, during all these years, perhaps due to the influence of axiomatic logic, the majority of researchers have concentrated their efforts on improving the exact, rigorous and formal proof searching algorithms for a particular formal system of logic. In their efforts they have neglected the beauty and power of informal, intuitive reasoning of human mathematicians. There are exceptions including work by Gelernter (1963) and Bundy (1983). Bundy argued that in order to progress in computational logic, we need to go further and consider these informal aspects of human reasoning (Bundy 1983).

Our work supports this argument. We investigate informal human reasoning with diagrams and use it as an inspiration for formalizing diagrammatic reasoning so that it can be carried out on machines. We build a meta-theory in which diagrammatic proofs are formal. The issues which are addressed in this process include formality, informality and the rigor of diagrams in proofs. We hope to gain an insight into the understanding of at least a simple subset of diagrammatic proofs.

1.2 Aims

The concise storage of information, the intuitive representation of relations amongst elements of diagrams, and the support of perceptual inferences that humans seem to find easy to understand, are the characteristics of diagrams that we exploit in this book. We make the claim that most diagrams are "intuitive and easy to understand" informally, and support it only by anecdotal evidence from both, our own experience and that of some other people.[2] As mentioned before, there are at least two approaches to investigating diagrammatic reasoning corresponding to the perspectives on the diagrams research. One approach is to clarify and model the processes that are going on in humans when they use diagrams in mathematical reasoning – this can be described as a cognitive approach to investigating diagrams. Another approach is to design and implement a system which uses diagrammatic reasoning – this can be described as a formal and computational approach to investigating diagrams. In this book we take the second approach – our aim is to formalize diagrammatic reasoning and to show that diagrams can be used for proofs in a formal system.

Diagrams are concrete in nature. Unless we use *abstraction devices*[3] to represent the generality of a diagram, the diagram is a particular

[2]An experimental study into quantifying "intuitiveness" of diagrams and their use in mathematical proofs, and examining whether people find them "easier" to understand than symbolic logical proofs would be an interesting cognitive investigation.

[3]Note that in this book the word *abstraction* has two meanings due to a lack of two different appropriate words. First, an abstraction refers to some *abstraction device*, such as ellipsis (ellipsis is a term used for the "…" notation). Second, it refers to the *abstraction mechanism* which constructs a general proof from examples of a proof. The use of both meanings will always be clear from the context. Definitions of some terminology specific to this book, to which the reader is advised to pay special attention, can be found in Glossary on page 185.

instance of the general class to which it belongs – it is a typical representative instance for this classes. Abstraction devices are tools for representing the continuation of some pattern and are often used in objects to represent their generality. Examples of abstraction devices include ellipsis, or the summation of numbers sign $\sum$, or labelling of objects with variables. The use of abstraction devices in diagrams seems to be problematic, because it is difficult to keep track of them while manipulating a diagram. It is not clear if humans manipulate such abstraction devices or they reason with concrete objects and infer the generality in some other way. We aim to capture diagrammatic proofs which do not use abstraction devices on a computer. We use the concreteness property of diagrams and look into how theorems of mathematics can be expressed as diagrams for some concrete values, i.e., ground instantiations of a theorem.

The initial diagrams which represent (part of) a theorem are manipulated using some geometric operations which deconstruct diagrams in different ways, but preserve certain properties. For instance, if a diagram represents a natural number, then the collection of diagrams which is a result of applying some operation to the initial diagram represents the same natural number. This is true, because the operations are defined so that they preserve the natural number that the diagrams represent. The sequence of geometric operations on a diagram represents the *inference steps* of a diagrammatic proof. This is a novel approach to proving arithmetic theorems, which to the best of our knowledge, has not been undertaken before in other research on the automation of diagrammatic reasoning (see the overview of the past research in this field in Chapter 2). Rather than using symbolic formulae of some logic to prove a mathematical theorem, we use manipulations of diagrams. Our intuition is that the fact that the operations are visual seems to make them intuitively easier to understand and use for humans. No specialized knowledge of logic is required, just some familiarity with spatial manipulations. A concrete proof instance is called an *example-proof*, and consists of a sequence of operations applied to the concrete diagram. The set of all available diagrams and operations defines the proof search space.

Since manipulating abstraction devices to infer the generality of a diagram, or a theorem and its proof that the diagram conveys, can be problematic and can lead to ambiguous results, we need to find an alternative mechanism to capture a general proof of a theorem at hand. We do so by extracting a general pattern from several proof instances, and capture it in a recursive program, called a *schematic proof.* This recursive program allows us to construct a general diagrammatic proof for the universally quantified theorem at hand.

Finally, a general schematic proof which is inferred from the instances has to be shown to be correct. It seems that humans sometimes omit this step all together.[4] Human machinery for extracting a general argument is usually convincing enough to reassure them that the general argument is correct, e.g., consider the proof at the beginning of this chapter. In an automated reasoning system, we need to show the correctness of the induced general argument. This confirms that a diagrammatic schematic proof is indeed a correct formal proof of a theorem. We use the *constructive ω-rule*, an existing technique in logic (Sundholm 1983), to justify the step from schematic proofs to theoremhood. Baker et al (1992) investigated this rule in the domain of arithmetic theorems. The constructive ω-rule allows us to capture infinitary concepts in a finite way using the diagrams. In this book we aim to investigate the entire process of constructing examples, constructing a general proof, and showing that the general proof is correct. Together, all three stages constitute our formalization of *diagrammatic proofs.*

Having formalized the use of diagrams in proofs, it is no longer true that diagrammatic proofs can only be informal proofs. It is now interesting to investigate the relation between symbolic and diagrammatic proofs. Usually, theorems are symbolically proved with the use of inference steps which often do not convey an intuitive notion of truthfulness to humans in quite as easy way as diagrams do. The inference steps of a formal symbolic (as opposed to diagrammatic) proof are statements that follow the rules of some logic. The reason we trust that they are correct is that the logic has been previously proved to be sound. Following and applying the rules of such a logic guarantees that there is no mistake in the proof. We hope to have such a guarantee in our proof system, and moreover, to gain an insight into the intuitive understanding, the correctness and such properties of our diagrammatic proof. Ultimately, the entire process of diagrammatically proving theorems will illuminate the issues of formality, rigor, truthfulness and power of diagrammatic proofs, and perhaps more generally, of any sort of proof.

1.3 Some Original Contributions

There are three main contributions made by our work. First, our research introduces a novel approach to automated reasoning about mathematical theorems. There has been no work done on the automation of systems which use diagrams in such a direct way as our system Diamond, and the manipulations of diagrams lead to a correct proof of a theorem. All of the traditional formal rules of some logic which are expressed as

[4]Some anecdotal evidence will be given later in §4.5 and §4.6.

symbolic formulae, are completely replaced by geometric operations on diagrams. Thus, all the inference rules of DIAMOND are diagrammatic.

Second, the work presented in this book shows that diagrams *can* be used for *formal* proofs. Moreover, formal proofs are not just aided by diagrams, but can be constructed using only diagrams and operations on them. Although some people have claimed that diagrams can be given a rigorous function in reasoning and some people have disputed it, we are the first to show *how* it can be done in a particular subfield of mathematics so that formal diagrammatic rather than symbolic logical proofs can be generated. We formalize diagrammatic reasoning in a particular domain of mathematics (see Chapter 3), and implement a reasoning system DIAMOND which is capable of diagrammatically proving a number of theorems (Chapter 9). These proofs are guaranteed to be correct.

Finally, we show how the constructive ω-rule can be used to reason with particular instances of diagrams rather than with abstraction devices in general diagrams. We demonstrate how this technique can be used to capture general diagrammatic proofs (Chapter 4).

These three contributions are embodied in an implementation of a diagrammatic proof system called DIAMOND which automates diagrammatic reasoning and applies it to problem solving in mathematics. DIAMOND is a body of Standard ML code which interactively, via a graphical user interface, allows a user to construct diagrammatic proofs.

The construction of diagrammatic proofs in DIAMOND consists of three steps.

- The user interactively constructs example-proofs by choosing initial diagrams which represent the theorem (Chapter 5), and then applies diagrammatic operations (Chapter 6) to build these example-proofs.
- DIAMOND then automatically constructs a general pattern from these instances of proofs, and captures it in a recursive program, called a schematic proof. (Chapter 7)
- The final step is to check if the schematic proof is correct. DIAMOND automatically verifies a given schematic proof. (Chapter 8)

The main limitation of DIAMOND is that its expressiveness of diagrammatic rules is restricted. There are rules which cannot be expressed as manipulations of diagrams with the current repertoire. Indeed, there are theorems which consist of terms that cannot be expressed as diagrams. To overcome these weaknesses DIAMOND needs to be extended with some additional types of concrete diagrams and operations on them. Finally, DIAMOND is a proof checker, it is not a discoverer. The user of DIAMOND provides most of the intelligence by constructing example-proofs. Therefore, in order to enable Diamond to find proofs for itself, we could extend Diamond to a fully automated theorem prover which *discovers* diagrammatic proofs (Chapter 10) – this remains an interesting direction for future work.

There is a potential for the ideas we present in this book to be used for exploring human intuitive reasoning in a novel way. We think that humans find diagrammatic proofs easier to understand and more compelling than their symbolic logical counterparts. We have only anecdotal evidence to support our belief. However, some comparative psychological validity experimental study could be carried out. We propose that such a study could use Diamond to provide an architecture where the diagrammatic proofs can be constructed and explored in order to gain an insight into the understanding of the proof.

1.4 Layout of the Rest of This Book

Here is the organization and the layout of the rest of this book. It should give the readers an overall picture of the topics discussed in this book, and point them to a specific subject of interest.

In Chapter 2, *The History of Diagrammatic Systems*, we describe several other diagrammatic systems which have been implemented in the past. They all use diagrams for reasoning in some way: to store information, to reject false facts, to infer new facts, etc. We concentrate in more detail on Gelernter's Geometry Machine, Koedinger and Anderson's DC, Barker-Plummer and Bailin's Grover, Barwise and Etchemendy's Hyperproof, Lindsay's Archimedes, Furnas' Bitpict, and Anderson and McCartney's IDR, because these seem to be closest to our work with respect to the use of diagrams for problem solving.

In Chapter 3, *Diagrammatic Theorems and the Problem Domain*, we present some examples of theorems which can be represented and proved in a diagrammatic way. Diagrams are often perceived as an informal rather than formal aid to reasoning, so we discuss their use in proofs, and the general issues about the formal and informal role of diagrams in proofs. We then present some examples of theorems that can be proved diagrammatically by showing the diagrams and the manipulations on them. Based on these examples, a taxonomy of diagrammatic proofs is introduced. Another factor which is considered in our choice of the problem domain is the use of abstraction devices (e.g., ellipsis) in diagrams. Finally, the taxonomy helps us choose the domain of problems that we subsequently concentrate on in this book.

In Chapter 4, *The Constructive ω-rule and Schematic Proofs*, we give a way of capturing diagrammatic proofs without the need to resort to diagrams containing abstraction devices. The mathematical basis for capturing the generality of the proof is in the use of the constructive ω-rule in schematic proofs, which is explained in detail.

In Chapter 5, *Designing a Diagrammatic Reasoning System*, we describe the Diamond system which is an embodiment of the ideas presented in this book. Diamond is a diagrammatic proof checker, which interactively proves theorems of mathematics by applying geometric operations to diagrams. In this chapter some of the design issues for the implementation of this proof system are discussed. These include: the architecture of Diamond, the basic notion of a diagrammatic proof, the construction of example-proofs, the representation of diagrams, and Diamond's graphical interface.

In Chapter 6, *Diagrammatic Operations*, we present the geometric operations, which are available in Diamond. These operations capture the inference steps of a diagrammatic proof. We define them here and give some examples.

In Chapter 7, *The Construction of Schematic Proofs*, the notion of a diagrammatic proof is presented. A diagrammatic proof is captured in a recursive program, referred to as a schematic proof. When a schematic proof is run, it generates a proof of $P(n)$ for each input value of number n. In this chapter we describe how general schematic proofs are automatically constructed from example-proofs, and how they are formalized in Diamond.

In Chapter 8, *The Verification of Schematic Proofs*, we present a method which enables us to prove the correctness of schematic proofs for *particular* theorems. The mechanism for construction of a schematic proof is an inductive inference algorithm. It is a machine's attempt to make an "intelligent" guess of what the general proof is. This "guess" needs to be verified and shown to be correct. In this chapter we define a way of carrying out the verification, in particular, we devise a theory of diagrams where we can check the correctness of a schematic proof.

In Chapter 9, Diamond *in Action*, we present a running example of the construction, formalization and verification of a diagrammatic proof in Diamond. We also comment on the general results in Diamond such as the range and depth of theorems it can interactively prove, and the limitations of Diamond.

In Chapter 10, *Complete Automation*, we propose some possible future directions for the work discussed in this book. In particular, we give an indication of how to make Diamond a completely automated theorem prover capable of discovering diagrammatic proofs. Finally, we make some concluding remarks.

In Appendix A, *More Examples of Diagrammatic Theorems*, we give additional examples of theorems and their diagrammatic proofs, which are analyzed to motivate the taxonomy of diagrammatic theorems used to choose the problem domain discussed in this book.

In Appendix B, *The ω-Rule*, we define and motivate the use of the ω-rule in logic. The problems with its use in automation lead us to the use of its constructive version (explained in Chapter 4).

In Glossary we give some definitions of technical terms used in this book that might prove useful. Notice that in the literature, the terms induction, abstraction and generalization are often used interchangeably for the same concept. We have three different notions for these terms, and hence define them here precisely. We urge the reader to pay particular attention to the use of these terms.