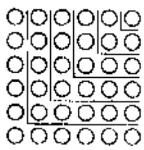
Automation of Diagrammatic Proofs in Mathematics

Mateja Jamnik
Supervisors: Prof. Alan Bundy, Dr. Ian Green
Department of Artificial Intelligence, 80 South Bridge
Edinburgh, EH1 1HN, UK
matejaj@dai.ed.ac.uk

Introduction



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

It requires only basic secondary school knowledge of mathematics to realise that the diagram above is a proof of a theorem about the sum of odd naturals.

It is an interesting property of diagrams that allows us to "see" and understand so much by looking at a simple diagram. Not only do we know what theorem the diagram represents, but we also understand the proof of the theorem and believe it is correct. Is it possible to simulate and formalise this sort of diagrammatic reasoning on machines? Or is it a kind of intuitive reasoning peculiar to humans that mere machines are incapable of? Roger Penrose claims that it is not possible to automate such diagrammatic proofs.[1] We are taking his position as an inspiration and are trying to capture the kind of diagrammatic reasoning that Penrose is talking about in order to be able to simulate it on a computer.

Theorems in automated theorem proving are usually proved by logical formal proofs which often do not convey an intuitive notion of truthfulness to humans. The inference steps are just 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 us that there is no mistake in the proof.

However, there is a subset of problems which humans can prove in a different way by the use of geometric operations on diagrams, so called diagrammatic proofs (such as the one given above). Insight is more clearly perceived in these than in the corresponding algebraic proofs: they capture an intuitive notion of truthfulness that humans find easy to see and understand. We are identifying and automating this diagrammatic reasoning on mathematical theorems. The user gives the system, called DIAMOND (DIAgraMmatic reasONing and Deduction), a theorem and then interactively proves it by the use of geometric manipulations on the diagram. These operations are the "inference steps" of the proof. DIAMOND then automatically derives from these example proofs a generalised proof. The constructive w-rule is used as a mathematical basis to capture the generality of inductive diagrammatic proofs. In this way, we explore the relation between diagrammatic and algebraic proofs. Ultimately, the entire process of diagrammatically proving theorems will illuminate the issues of formality, rigour, truthfulness and power of diagrammatic proofs.

[1] Roger Penrose presented his position in the lecture at International Centre for Mathematical Sciences in Edinburgh on 8 November, 1995.

Proposed work plan

There are several issues that need to be addressed during the course of this research project:

- Investigate and formalise geometric operations and heuristics on diagrams (April 96-July 96).
- Implement basic reasoning part of DIAMOND which will accept example proof steps instructed by the user (August 96-November 96).
- Formalise and implement representations of diagrams (December 96-March 97).
- Code diagrammatic operations (April 97-July 97).
- Formalise and implement generalisation mechanism (constructive w-rule) (August 97-November 97).
- Improve user interface (December 97-January 98).
- Testing (February 98-March 98).
- Write up the thesis (April 98-September 98).

The programming language used for the implementation of DIAMOND is Standard ML of New Jersey, Version 109. The code is available upon request to the author.

Progress to date

So far, several diagrams have been represented and several geometric operations on them are available to the user of DIAMOND. Thus, the interactive construction of proofs and automatic generalisation from example proofs have been implemented. We can prove a few theorems: sum of odd naturals, sum of all naturals, sum of Fibonacci squares. We are working on more examples.