Automating inductive proof

► ALAN BUNDY, *Automating inductive proof.*

School of Informatics, University of Edinburgh, 10 Crichton St, Edinburgh, UK.
E-mail: A.Bundy@ed.ac.uk.
URL Address: http://homepages.inf.ed.ac.uk/bundy/.

The automation of inductive proof plays a pivotal role in the formal development of ICT systems: both software and hardware. It is required to reason about all forms of *repetition*, which arises in: recursive and iterative programs; parameterised hardware; traces of program runs; program invariants; etc. Since formal proof is a highly skilled and time-consuming activity, industry requires as much automation as possible to enable formal methods to be used cost effectively.

Unfortunately, inductive reasoning is much harder to automate than, for instance, first-order reasoning. Negative results from mathematical logic underpin these difficulties. These results include incompleteness, the undecidability of termination and the absence of cut elimination. Of these, the absence of cut elimination creates the most practical problems. The proofs of even some very simple and obviously true conjectures require the injection of cut formulae. These formulae typically take the form of intermediate lemmas, generalisations of the conjecture or non-standard induction rules. Cut rule steps are generally assumed to require human intervention with an interactive prover to provide an appropriate cut formula.

We have developed a proof technique called *rippling* [Bundy *et al*, 2005] that guides the manipulation of the induction conclusion until the induction hypothesis can be used in its proof. In fact, rippling can be used in any situation where a given embeds in a goal. It rewrites the goal while preserving and re-grouping the embedding until an instance of the given appears as a sub-expression of the goal.

The main contribution of rippling, however, is not its guidance of the step case, but the way it informs the application of the cut rule. It provides a strong expectation of the direction of the proof, but is not always successful. When it fails, an analysis of the failure suggests an appropriate application of cut: the form of a missing lemma, a generalisation or a non-standard induction rule [Ireland & Bundy, 1996]. This increases the scope of inductive-proof automation, which has economic implications for the use of formal methods in the ICT industry.

[Bundy *et al*, 2005]Bundy, A., Basin, D., Hutter, D. and Ireland, A. (2005). *Rippling: Meta-level Guidance for Mathematical Reasoning*, volume 56 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press.

[Ireland & Bundy, 1996]Ireland, A. and Bundy, A. (1996). Productive use of failure in inductive proof. *Journal of Automated Reasoning*, 16(1–2):79–111.