J Autom Reasoning (2006) 35: 429–431
DOI 10.1007/s10817-006-9027-0

BOOK REVIEW

“Rippling: Meta-Level Guidance for Mathematical Reasoning,” by Alan Bundy, David Basin, Dieter Hutter, and Andrew Ireland, Cambridge University Press, 2005

Nimish Shah

Published online: 18 August 2006

Introduced by Aubin [1] and then developed by Bundy [3], *rippling* is a term used to explain a heuristic for proving inductive theorems in automated theorem proving. In the proof of the inductive step $P(n) \rightarrow P(n+1)$, the inductive hypothesis $P(n)$ can be ‘visualised’ as a pool of undisturbed water. This inductive hypothesis can be ‘disturbed’ by replacing all occurrences of the subterm n by $n+1$, corresponding to a stone being thrown into the water. As the stone causes a water wave-front to be set up, rippling outwards from the stone, and leaving undisturbed water behind, so (and as is often seen in a backward proof of the inductive step) the subterm $n+1$ and its surrounding terms are rewritten so as to leave the subterm n and the original and undisturbed surrounding terms behind.

As a concrete example, consider the theorem $n \times (m+o) = n \times m + n \times o$, (i.e., $\forall n.\ P(n)$) which can be proved by induction on n using Peano’s axioms. Using a context Γ, three rewrite rules are assumed, whence three wave rules can be further computed that orientate them. Here A, B, and C are meta-variables that range over terms.

Γ			**Context**
$s(A) \times B$	$=$	$A \times B + B$	Rewrite Rule 1
$A + (B + C)$	$=$	$(A + B) + C$	Rewrite Rule 2
$(A + B) + C$	$=$	$(A + C) + B$	Rewrite Rule 3
$\boxed{s(\underline{A})}^{\uparrow} \times B$	$\Rightarrow$	$\boxed{\underline{A \times B} + B}^{\uparrow}$	Wave Rule 1.1
$A + (\boxed{\underline{B} + C}^{\uparrow})$	$\Rightarrow$	$\boxed{(\underline{A + B}) + C}^{\uparrow}$	Wave Rule 2.1
$(\boxed{\underline{A} + B}^{\uparrow}) + C$	$\Rightarrow$	$\boxed{(\underline{A + C}) + B}^{\uparrow}$	Wave Rule 3.1
$\boxed{\underline{B} + A}^{\uparrow} = \boxed{\underline{C} + A}^{\uparrow}$	$\Rightarrow$	If $P(n) \equiv B\phi = C\phi$ then $\top$	Strong Fertilization

N. Shah (✉)
England, UK
e-mail: Nimish_Shah@onetel.com

$\forall \mathbf{n}.\ \mathbf{P}(\mathbf{n})$ (where m, o are arbitrary constants)			**To Be Proved**
$n \times (m+o)$	$=$	$n \times m + n \times o$	
$\boldsymbol{\Gamma}, \mathbf{P}(\mathbf{n}) \vdash \mathbf{P}(\mathbf{n}+\mathbf{1})$			**Induction Step**
$\boxed{s(\underline{n})}^{\uparrow} \times (m+o)$	$=$	$\boxed{s(\underline{n})}^{\uparrow} \times m + \boxed{s(\underline{n})}^{\uparrow} \times o$	P(n+1)
$\boxed{\underline{n \times (m+o)} + (m+o)}^{\uparrow}$	$=$	$\boxed{s(\underline{n})}^{\uparrow} \times m + \boxed{s(\underline{n})}^{\uparrow} \times o$	LHS 1.1
$\boxed{\underline{n \times (m+o)} + (m+o)}^{\uparrow}$	$=$	$\boxed{s(\underline{n})}^{\uparrow} \times m + (\boxed{\underline{n \times o} + o}^{\uparrow})$	RHS 1.1
$\boxed{\underline{n \times (m+o)} + (m+o)}^{\uparrow}$	$=$	$\boxed{(\underline{\boxed{s(\underline{n})}^{\uparrow} \times m + n \times o}) + o}^{\uparrow}$	RHS 2.1
$\boxed{\underline{n \times (m+o)} + (m+o)}^{\uparrow}$	$=$	$\boxed{(\underline{(\boxed{\underline{n \times m} + m}^{\uparrow}) + n \times o}) + o}^{\uparrow}$	RHS 1.1
$\boxed{\underline{n \times (m+o)} + (m+o)}^{\uparrow}$	$=$	$\boxed{(\underline{\boxed{(\underline{n \times m + n \times o}) + m}^{\uparrow}}) + o}^{\uparrow}$	RHS 3.1
$\boxed{\underline{n \times (m+o)} + (m+o)}^{\uparrow}$	$=$	$\boxed{(\underline{n \times m + n \times o}) + (m+o)}^{\uparrow}$	Unblocking
	$\top$		Strong Fertilization

The book has six chapters (excluding the introduction), which can be conveniently divided into the following: (i) a detailed description of rippling (Chapters 1 and 2), (ii) cases when rippling fails (Chapters 3 and 5), and (iii) the mathematical details of rippling/annotations (Chapters 4 and 6 and Appendix 1). The notation used is simple. A wave-front is shaded (and its direction of movement indicated by an arrow), a skeleton is not shaded, while a wave-hole is a skeleton inside a wave-front. In the example above, each wave-front expands (one rewrite on the left-hand side (LHS), several on the right-hand side (RHS)) until a skeleton of the induction hypothesis is left in the wave-hole. Strong fertilization (i.e., the removal of the induction hypothesis) then takes place alongside the trivial syntactic matching of the remaining subterms.

Chapter 2 describes the other type of rippling – called *sideways rippling* – where free variables act as sinks that are capable of absorbing subterms that arise during rippling. Both Chapters 1 and 2 are self-explanatory, although the reader is advised to work through the examples presented (especially in parse-tree form) in order to fully appreciate how rippling guides the proof.

In contrast, Chapters 3 and 5 deal with rippling failure. Chapter 5 is the analysis of when rippling is a 'hit' or 'miss' in a wide variety of examples; while Chapter 3 concentrates on the research carried out by Bundy/Ireland at Edinburgh/Heriot-Watt universities. Proof planning is the heuristic (of which rippling is one type covering inductive proofs) that most proofs (within a particular family) follow a similar plan. By analyzing when, where, and how an automatic proof fails, useful information can be obtained. In the case of rippling, a proof may fail because of a missing lemma (say, a rewrite rule) or because a more generalized proof should have been proved instead. Both of these patches are examples of a missing cut formula (A) when the cut theorem is used backward: that is, to prove $\Gamma \vdash \forall n.\ P(n)$ means to prove $\Gamma,\ A \vdash \forall n.\ P(n)$ and $\Gamma \vdash A$. Since rippling carries such a strong expectation of how the proof should proceed, the structure of A can be guessed, while higher-order unification can be used to fill in the structure. This area is the subject matter of the latter part of Chapter 3 (which also briefly discusses unblocking as used in the example above).

The remaining two chapters (excluding the conclusion) deal with the mathematical side of rippling. Chapter 6 advocates the thesis that annotations (used for annotating wave-rules, wave-fronts, wave-holes, etc.) can be used to develop a more powerful proof methodology that abstracts away unwanted details. The chapter also discusses how different methods of annotations (used in the different implementations of rippling) are used to convey semantic information (say, in distinguishing multiple occurrences of a particular variable). The notation used in Chapters 6 and 4 comes from the basic terminology of term rewriting systems [2]. Chapter 4 demonstrates formal properties of rippling, for example, how various measures can be used for proving termination via well-founded induction. The chapter also covers how terms are automatically annotated.

The book is certainly recommended to researchers interested in implementing rippling in their systems.[1] While not a textbook, the book does represent a solid and thorough review of the research that has been accumulated to date by two research groups (one at Edinburgh and the other at Saarbrücken and led by Hutter) that have implemented rippling in their systems (Oyster/Clam and INKA, respectively). Perhaps, the only negative point to mention is the lack of resources available on the http://www.rippling.org website. It would be a very simple task to make many of the original research papers (several of which have been published in this journal) available; and it to be hoped that this shortcoming will soon be remedied.

Contents: 1. *An Introduction to Rippling*, 2. *Varieties of Rippling*, 3. *Productive Use of Failure*, 4. *A Formal Account of Rippling*, 5. *The Scope and Limitations of Rippling*, 6. *From Rippling to a General Methodology*, 7. *Conclusions, Appendix* 1: *An Annotated Calculus and a Unification Algorithm, Appendix* 2: *Definition of Functions Used in This Book*.

References

1. Aubin, R.: Mechanizing Structural Induction. Ph.D. thesis, Edinburgh University (1976)
2. Baader, F., Nipkow, T.: Term Rewriting and All That. Cambridge University Press (1998)
3. Bundy, A.: The use of explicit plans to guide inductive proofs. In: Lusk, E., Overbeek, R. (eds.) 9th International Conference on Automated Deduction (CADE 9), pp. 111–120. Springer, Berlin Heidelberg New York (1988)

[1] As Bundy points out, empirical evidence shows that 50% of the time spent in verification (in an industrial context) is spent on repairing proofs after changes to specifications.