Termination Orderings for Rippling

David A. Basin
Toby Walsh

MPI–I–94–209 February 1994

Authors' Addresses

David Basin Max-Planck-Institut für Informatik Im Satdwald, D-66123 Saarbrücken, Germany
basin@mpi-sb.mpg.de
Toby Walsh INRIA-Lorraine, 615, rue du Jardin Botanique, 54602 Villers-les-Nancy, France
walsh@loria.fr

Publication Notes

A version of this paper will appear at CADE-12 in Nancy France, June 1994.

Acknowledgements

Many of the ideas described here stem from conversations with members of the Edinburgh MRG group, in particular with Alan Bundy, Ian Green, and Andrew Ireland. We also wish to thank Sean Matthews, Michael Rusinowitch, and Andrew Stevens for comments on earlier drafts. The first author was funded by the German Ministry for Research and Technology (BMFT) under grant ITS 9102. The second author was supported by a SERC and a HCM Postdoctoral Fellowship.

Abstract

Rippling is a special type of rewriting developed for inductive theorem proving. Bundy *et. al.* have shown that rippling terminates by providing a well-founded order for the annotated rewrite rules used by rippling. Here, we simplify and generalize this order, thereby enlarging the class of rewrite rules that can be used. In addition, we extend the power of rippling by proposing new domain dependent orders. These extensions elegantly combine rippling with more conventional term rewriting. Such combinations offer the flexibility and uniformity of conventional rewriting with the highly goal directed nature of rippling. Finally, we show how our orders simplify implementation of provers based on rippling.

1 Introduction

Rippling is a form of goal directed rewriting developed at Edinburgh [5, 3] and in parallel in Karlsruhe [11, 12] for inductive theorem proving. In inductive proof, the induction conclusion typically differs from the induction hypothesis by the addition of some constructors or destructors. Rippling uses special annotations, called *wavefronts*, to mark these differences. They are then removed by annotated rewrite rules, called *wave-rules*. Rippling has several attractive properties. First, it is highly goal directed, attempting to remove just the differences between the conclusion and hypothesis, leaving the common structure preserved. And second, it terminates yet allows rules like associativity to be used both ways.

The contributions of this paper are to simplify, improve, and generalize the specification of wave-rules and their associated termination orderings. Wave-rules have previously been presented via complex schematic definitions that intertwine the properties of structure preservation and the reduction of a well-founded measure (see [3] and §7). As these properties may be established independently, our definition of wave-rules separates these two concerns. Our main focus is on new measures. We present a family of measures that, despite their simplicity, admit strictly more wave-rules than the considerably more complex specification given in [3].

This work has several practical applications. By allowing rippling to be combined with new termination orderings, the power of rippling can be greatly extended. Although rippling has been designed primarily to prove inductive theorems it has recently been applied to other problem domains. We show that in rippling, as in conventional rewriting, the ordering used should be domain dependent. We provide several new orderings for applying rippling to new domains within induction (e.g. domains involving mutually recursive functions) and outside of induction (e.g. equational problem solving). In doing so, we show for the first time how rippling can be combined with conventional rewriting.

Another practical contribution is that our work greatly simplifies the implementation of systems based on rippling. Systems like Clam [4] require a procedure, called a *wave-rule parser*, to annotate rewrite rules. Clam's parser is based upon the complex definition of wave-rules in [3] and as a result is itself extremely complex and faulty. We show how, given a simple modular order, we can build simple modular wave-rule parsers. We have implemented such parsers and they have pleasant properties that current implementations lack (e.g. notions of correctness and completeness); our work hence leads to a simpler and more flexible mechanization of rippling.

The paper is organized as follows. In §2 we give a brief overview of rippling. In §3 we define an order on a simple kind of annotated term and use this in §4 to build orderings on general annotated terms. Based on this we show in §5 how rewrite rules may be automatically annotated. In §6 we describe how new orders increase the power and applicability of rippling. In §7 we compare this work to previous work in this area and discuss some practical experience. Finally we draw conclusions.

2 Background

We provide a brief overview of rippling. For a complete account please see [3].

Rippling arose out of an analysis of inductive proofs. For example, if we wish to prove $P(x)$ for all natural numbers, we assume $P(n)$ and attempt to show $P(s(n))$. The hypothesis and the conclusion are identical except for the successor function $s(.)$ applied to the induction variable n. Rippling marks this difference by the annotation, $P(\boxed{s(\underline{n})})$. Deleting everything in the box that is not underlined gives the skeleton, which is preserved during rewriting. The boxed but not underlined term parts are wavefronts, which are removed by rippling.

Formally, a *wavefront* is a term with at least one proper subterm deleted. We represent this by marking a term with *annotation* where wavefronts are enclosed in boxes and the deleted subterms, called *waveholes*, are underlined. Schematically, a wavefront looks like $\boxed{\xi(\underline{\mu_1}, ..., \underline{\mu_n})}$, where $n > 0$ and μ_i may be similarly annotated. The part of the term not in the wavefront is called the *skeleton*. Formally, the skeleton is a non-empty set of terms defined as follows.

Definition 1 (Skeleton)

1. $skel(t) = \{t\}$ *for* t *a constant or variable*
2. $skel(f(t_1, ..., t_n)) = \{f(s_1, ..., s_n) | \forall i.\, s_i \in skel(t_i)\}$
3. $skel(\boxed{f(\underline{t_1}, ..., \underline{t_n})}) = skel(t_1) \cup ... \cup skel(t_n)$ *for the* t_i *in waveholes.*

We call a term *simply annotated* when all its wavefronts contain only a single wavehole and *generally annotated* otherwise. In the simply annotated case, the skeleton function returns a singleton set whose member we call the skeleton. E.g. the skeleton of $f(\boxed{s(\underline{a})}, \boxed{s(\underline{b})})$ is $f(a, b)$.

We define *wave-rules* to be rewrite rules between annotated terms that meet two requirements: they are skeleton preserving and measure decreasing. This is a simpler and more general approach to defining wave-rules than that given in [3] where these requirements were intertwined into the syntactic specification of a wave-rule.[1] *Skeleton preservation* in the simply-annotated case means that both the LHS (left-hand side) and RHS (right-hand side) of the wave-rule have an identical skeleton. In the multi-hole case we demand that *some* of the skeletons on the LHS are preserved on the RHS and no new skeletons are introduced, i.e. $skel(LHS) \supseteq skel(RHS)$.

Wavefronts in wave-rules are also *oriented*. This is achieved by marking the wavefront with an arrow indicating if the wavefront should move up through the skeleton term tree or down towards the leaves. Oriented wavefronts dictate a measure on terms that rippling decreases. The focus of this paper is on these measures.

Below are some examples of wave-rules (s is successor and $<>$ is infix append).

$$\boxed{s(\underline{U})}^{\uparrow} \times V \Rightarrow \boxed{(\underline{U \times V}) + V}^{\uparrow} \tag{1}$$

$$\boxed{s(\underline{U})}^{\uparrow} \geq \boxed{s(\underline{V})}^{\uparrow} \Rightarrow U \geq V \tag{2}$$

$$\boxed{\underline{U} + V}^{\uparrow} \times W \Rightarrow \boxed{\underline{U \times W} + V \times W}^{\uparrow} \tag{3}$$

$$(\boxed{\underline{U} <> V}^{\uparrow}) <> W \Rightarrow U <> (\boxed{V <> \underline{W}}^{\downarrow}) \tag{4}$$

$$U <> (\boxed{\underline{V} <> W}^{\uparrow}) \Rightarrow \boxed{(\underline{U <> V}) <> W}^{\uparrow} \tag{5}$$

$$\boxed{\underline{U} + \underline{V}}^{\uparrow} = \boxed{\underline{W} + \underline{Z}}^{\uparrow} \Rightarrow \boxed{\underline{U = W} \wedge \underline{V = Z}}^{\uparrow} \tag{6}$$

(1) and (2) are typical of wave-rules based on a recursive definitions. The remainder come from lemmas. Methods for turning definitions and lemmas into wave-rules is the subject of §5. Note that annotation in the wave-rules must match annotation in the term being rewritten. This allows use of rules like associativity of append, (4) and (5), in both directions; these would loop in conventional rewriting. Note also that in (6) the skeletons of the RHS are a strict subset of those of the LHS.

[1]This generalization is, however, briefly discussed in their further work section.

As a simple example of rippling, consider proving the associativity of multiplication using structural induction. In the step-case, the induction hypothesis is

$$(x \times y) \times z = x \times (y \times z)$$

and the induction conclusion is

$$(\boxed{s(\underline{x})}^{\uparrow} \times y) \times z = \boxed{s(\underline{x})}^{\uparrow} \times (y \times z).$$

The wavefronts in the induction conclusion mark the differences with the induction hypothesis. Rippling on both sides of the induction conclusion using (1) yields (7) and then with (3) on the LHS gives (8).

$$(\boxed{\underline{x \times y} + y}^{\uparrow}) \times z \quad = \quad \boxed{\underline{(x \times (y \times z))} + y \times z}^{\uparrow} \tag{7}$$

$$\boxed{\underline{((x \times y) \times z)} + y \times z}^{\uparrow} \quad = \quad \boxed{\underline{(x \times (y \times z))} + y \times z}^{\uparrow} \tag{8}$$

As the wavefronts are now at the top of each term, we have successfully rippled-out both sides of the equality. We can complete the proof by simplifying with the induction hypothesis.

The example illustrates how rippling preserves skeletons during rewriting. Provided rippling does not get *blocked* (no wave-rule applies yet we are not completely rippled-out), we are guaranteed to be able to simplify with the induction hypothesis (called *fertilization* in [2]). This explains the highly goal directed nature of rippling.

We can also ripple wavefronts towards the position of universally quantified variables in the induction hypothesis. Such positions are called *sinks* because wavefronts can be absorbed there; when we appeal to the induction hypothesis, universally quantified variables will be matched with the content of the sinks. Rippling towards sinks at the leaves of terms is called *rippling-in*. Wavefronts are oriented with arrows pointing out (upwards) or in (downwards) indicating if they are moving towards the root or leaves. *Transverse* wave-rules like (4) are used to turn outward directed wavefronts inwards.

3 Ordering Simple Wave-Rules

In this section we consider only simply annotated terms (whose wavefronts have a single wavehole). In the next section we generalize to orders for generally annotated terms with multiple waveholes. We begin with motivation, explaining generally the kinds of orders we wish to define. Afterwards, we propose several concrete measures that are similar, though simpler, to those given by Bundy *et. al.* in [3]. They are able to order all the wave-rules given in [3] and in addition allow rule orientations not possible using the measure given there (see §7).

We consider annotated terms as decorated trees where the tree is the skeleton and the wavefronts are boxes decorating the nodes. See, for example, the first tree in Fig. 1 which represents the term $\boxed{s(\underline{U})}^{\uparrow} \geq \boxed{s(\underline{V})}^{\uparrow}$. Our orders are based on assigning measures to annotation in these trees. We can define progressively simpler orders by simplifying these annotated trees to capture the notion of progress during rippling that we wish to measure.

To begin with, since rippling is skeleton preserving, we needn't account for the contents of the skeleton in our orderings. That is, we can abstract away function symbols in the skeleton, for example, mapping each function to a variadic function constant "*". This gives, for example, the second tree in Fig. 1. In §6.2, we return

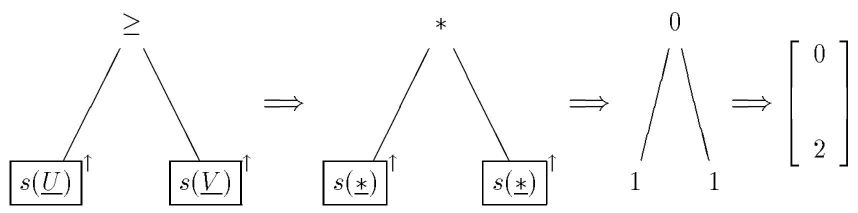

Figure 1: Defining a measure on annotated terms.

to this abstraction and examine termination orderings that do allow the skeleton to be changed during rewriting.

A further abstraction is to ignore the names of function symbols within wave-fronts and assign some kind of numeric weight to wave-fronts. For example, we may tally up the values associated with each function symbol as in a Knuth-Bendix ordering. The simplest kinds of weights that we may assign to wave-fronts measure their *width* and their *size*. Width is the number of nested function symbols between the root of the wavefront and the wavehole. Size is the number of function symbols and constants in a wavefront. For simplicity, we will consider just the width unless otherwise stated. This gives, for example, the third tree in Fig. 1. Of course, there are problem domains where we want our measure to reflect more of the structure of wave-fronts. §6.1 contains an example of this showing how the actual contents may be compared using a conventional term ordering.

Finally, a very simple notion of progress during rippling is simply that wave-fronts move up or down through the skeleton tree. Under this view, the tree structure may be ignored: it is not important which branch a wave-front is on, only its height in the skeleton tree. Under this notion of progress, we can apply an abstraction that maps the tree onto a list, level by level. For instance, we can use the sum of the weights at a given depth. Applying this abstraction gives the final tree in Fig. 1. Again, note that depths are *relative* to the skeleton and not depth in the erased term tree.

To make this more formal and concrete, we introduce some definitions. A *position* is simply a path address (written "Dewey decimal style") in the term tree of the skeleton and the subterm of t at position p is denoted by t/p. If s is a subterm of t at position p, its *depth* is the length of p. The *height* of t, written $|t|$, is the maximal depth of any subterm in t. For an annotated term t, the *out-weight of a position* p is the sum of the weights of the (possibly nested) outwards oriented wavefronts at p. The *in-weight* is defined identically except for inward directed wavefronts. We now define a measure on terms corresponding to the final tree in Fig. 1 based on weights of annotation relative to their depths.

Definition 2 (Out/In Measure) *The* out-measure, *$MO(t)$ (*in-measure, *$MI(t)$) of an annotated term t is a list whose i-th element is the sum of out-weights (in-weights) for all term positions in t at depth i.*

For example, in the following palindrome function over lists ("::" is infix cons)

$$palin(\boxed{H :: \underline{T}}^{\uparrow}, Acc) \Rightarrow \boxed{H :: \underline{palin(T, \boxed{H :: \underline{Acc}}^{\downarrow})}}^{\uparrow} \tag{9}$$

and the skeleton of both sides is $palin(T, Acc)$ and the out-measure of the LHS is [0,1] and the RHS is [1,0]. The in-measures are [0,0] and [0,1] respectively.

We now define a well-founded ordering on these measures which reflects the progress that we want rippling to make during rewriting. Consider, a simple wave-rule like (1),

$$\boxed{s(\underline{U})}^{\uparrow} \times V \Rightarrow \boxed{(\underline{U \times V}) + V}^{\uparrow}.$$

The LHS out-measure is $[0,1]$, and the RHS is $[1,0]$. Rippling has progressed here as the one out-oriented wavefront has moved up the term. In general, rippling progresses if one out-oriented wavefront moves up or disappears, while nothing deeper moves downwards. If the out-measure on a term before rippling is $[l_1, ..., l_k]$ and after $[r_1, ..., r_k]$ then there must be some depth j where $l_j > r_j$ and for all $i > j$ we have $l_i = r_i$. This is simply the lexicographic order on the reverse of the two lists (compared with $>$ on the natural numbers).[2] Progress for in-oriented wavefronts is similar and reflects that these wavefronts should move towards leaves; that is, we use the lexicographic order on the in-measures. Of course, both outward and inward oriented wavefronts may occur in the same rule. For example, consider (9). As in [3], we define a composite ordering on the out and in measures. We order the out measure before the in measure since this enables us to ripple wavefronts out and either to reach the top of the term, or at some point to turn the wavefront down and to ripple it in towards the leaves.

Definition 3 (Composite Ordering) *$t \succ s$ iff $\langle MO(t), MI(t)\rangle >_o \langle MO(s), MI(s)\rangle$ where $>_o$ is the lexicographic order on pairs whose first components are compared with $>_{revlex}$ and the second with $>_{lex}$, the reversed and unreversed lexicographic order on lists of equal length.*

Given the well-foundedness of $>$ on the natural numbers and that lexicographic combinations of well-founded orders are well-founded we can conclude the following.

Lemma 1 *The composite ordering is well-founded.*

We lack space here to discuss implementations of rippling. Two different implementations are considered in [3] and [12]. For both calculi, $\succ$ (and $\succ^*$ of the next section) is monotonic and stable over the substitutions produced during rippling. It follows from standard techniques that if all wave-rules are oriented so that $l \succ r$ then rippling terminates [8].

4 Ordering Multi-Wave-Rules

We now generalize our order for simply annotated terms to those with generalized annotation, that is, multiple waveholes in a single wavefront. Wave-rules involving such terms are called *multi-wave-rules* in [3] and we have already seen an example of this in (6). The binomial equation is another example.

$$binom(\boxed{s(\underline{X})}^{\uparrow}, \boxed{s(\underline{Y})}^{\uparrow}) = \boxed{\underline{binom(X, \boxed{s(\underline{Y})}^{\uparrow})} + \underline{binom(X, Y)}}^{\uparrow} \qquad (10)$$

We define orders for generally annotated terms in a uniform way from the previous ordering by reducing generally annotated terms to sets of simply annotated terms and extending $\succ$ to such sets. This reduction is accomplished by considering ways that general annotation can be *weakened* to simple annotation by "absorbing" waveholes. Weakening a multi-wave term like (10) erases some of the waveholes (underlining) though always leaving at least one wavehole. A wavefront is *maximally weak* when it has exactly one wavehole. A term is *maximally weak* when all

[2]Note that these lists are the same length as the skeletons of both sides are identical; however, when we generalize the measure to multi-holed waves, the skeletons may have different depths and we pad with trailing zeros where necessary.

its wavefronts are maximally weak. Maximally weak terms are simply annotated and this allows us to use the previously defined measure $\succ$ on these terms.

Returning to the binomial example, (10) has only the following two weakenings.

$$binom(\boxed{s(\underline{X})}^{\uparrow}, \boxed{s(\underline{Y})}^{\uparrow}) \quad = \quad \boxed{\underline{binom(X, \boxed{s(\underline{Y})}^{\uparrow}}) + binom(X,Y)}^{\uparrow} \tag{11}$$

$$binom(\boxed{s(\underline{X})}^{\uparrow}, \boxed{s(\underline{Y})}^{\uparrow}) \quad = \quad \boxed{binom(X, s(Y)) + \underline{binom(X,Y)}}^{\uparrow} \tag{12}$$

Both of these are maximally weak as each wavefront has a single hole.

Let *weakenings*(s) be the set of maximal weakenings of a term s. We now define an ordering on generally annotated terms l and r.

Definition 4 (General ordering) $l \succ^* r$ *iff* weakenings(s) $\succ\!\succ$ weakenings(t) *where* $\succ\!\succ$ *is the multiset ordering over the order* $\succ$ *on simply annotated terms.*

This order is sensible as all the elements of the weakening sets are simply annotated and can be compared with $\succ$. Also observe that if l and r are simply annotated then their weakenings are $\{l\}$ and $\{r\}$ and $l \succ^* r$ agrees with $l \succ r$. In general, we will drop the superscript on $\succ^*$ and use context (e.g., at least one argument has multiple holes) to disambiguate.

As the multi-set extension of a well-founded ordering is well-founded [10] we immediately have the following lemma.

Lemma 2 $\succ^*$ *is well-founded.*

As an example, consider (10). The LHS weakenings are

$$\{binom(\boxed{s(\underline{X})}^{\uparrow}, \boxed{s(\underline{Y})}^{\uparrow})\}\,.$$

The RHS weakenings are

$$\{\boxed{\underline{binom(X, \boxed{s(\underline{Y})}^{\uparrow}}) + binom(X,Y)}^{\uparrow}, \boxed{binom(X + s(Y)) + \underline{binom(X,Y)}}^{\uparrow}\}\,.$$

The sole member of the first set is $\succ$-greater than both members of the second set. This equation is measure decreasing and hence a wave-rule when used left to right.

5 Parsing

These orders are simple and admit simple mechanization. We begin with simply annotated terms and then sketch the generalization to multi-waves. We have implemented the routines we describe and in §7 we report on practical experience.

A wave-rule $l \rightarrow r$ must satisfy two properties: the preservation of the skeleton, and a reduction of the measure. We achieve these separately. An *annotation phase* first annotates l and r with unoriented wavefronts so their skeletons are identical; this guarantees that rippling is skeleton preserving. An *orientation phase* then orients the wavefronts so that $l \succ r$. We sum this up by the slogan

$$\mathit{WAVE\text{-}RULE} \quad = \quad \mathit{ANNOTATION} + \mathit{ORIENTATION}\,. \tag{13}$$

5.1 Annotation

To annotate terms we can use the *ground difference unification* algorithm given in [1]. Since parsing is an off-line computation (performed once before theorem proving), it is also reasonable to find skeleton preserving annotation via generate-and-test: generate candidate annotations and test if the resulting terms have the same skeleton. Consider, for example, annotating the recursive definition of the palindrome function. There are four possible skeletons: $palin(T, Acc)$, T, Acc, and H. The first of these corresponds to the annotation

$$palin(\boxed{H :: \underline{T}}, Acc) \Rightarrow \boxed{H :: \underline{palin(T, \boxed{H :: \underline{Acc}})}}. \qquad (14)$$

The remaining annotations are *trivial* in that both sides are completely within wavefronts except for some subterm at the leaves. For example,

$$\boxed{palin(H :: T, \underline{Acc})} \Rightarrow \boxed{H :: palin(T, H :: \underline{Acc})}.$$

Such trivial wave-rules can usually be ignored as they they make no progress moving wavefronts (although they can be used for wavefront normalization, see §6.1).

5.2 Orientation

Given annotated, but unoriented rules, we must now orient them by placing arrows on the wavefronts. We do this by picking an orientation for wavefronts on the LHS of the wave-rule and finding an orientation on the RHS such that $l \succ r$. In Clam the wave-rules used are oriented with wavefronts on the LHS exclusively out or in. Other combinations are, of course, possible. In general the number of wavefronts, n in the LHS is very small, typically one or two in [3]; hence, it is not much extra effort to consider all 2^n orientations and for each of these generate an orientation for the RHS.[3] In practice this is manageable; see §7.

For each orientation of l we must orient r. If l contains at least one outward oriented wavefront there will always be a measure decreasing orientation of r, namely with all wavefronts oriented in. However, orienting wavefronts inwards prohibits later rippling out whilst orienting outwards does not. If rippling-out blocks, we can always redirect wave-rules inwards with the rewrite rule. $\boxed{F(\underline{X})}^{\uparrow} \Rightarrow \boxed{F(\underline{X})}^{\downarrow}$. This rule is structure preserving and measure decreasing. Hence, we orient r's annotation so that it is measure decreasing and $\succ$*-maximal*; that is, for all orientations r_o, if $l \succ r_o$ then $r \succeq r_o$ ($\succeq$ is the union of the identity relation with $\succ$).

One can find a maximal orientation using generate and test, but it is possible to do much better. Below we sketch an algorithm, linear in $|r|$. Its input is two annotated terms l and r where l is oriented and r unoriented. The output is r oriented and $\succ$-maximal. In what follows, suppose $|l|$ (and hence $|r|$) equals k. Let $t_i^{\uparrow}$ be the sum of out-weights at depth i, $t_i^{\downarrow}$ be the sum of in-weights at depth i, and $flip(t, d, n)$ be the operation that non-deterministically flips down n arrows in t at depth d (there may be multiple choices corresponding to different branches or multiple wavefronts at the same position). We assume below that l has at least one wavefront oriented up. If this is not the case then all of r's wavefronts must be oriented down and this is a maximal orientation iff $l \succ r$. Otherwise orientation proceeds as follows. We first orient all the wavefronts in r upwards and then execute the first of the following statements that succeeds.

1. choose the maximum i such that $l_i^{\uparrow} > r_i^{\uparrow}$ and $\forall j \in \{i+1..k\}.flip(r, j, r_j^{\uparrow} - l_j^{\uparrow})$

[3]This requires of course an implementation that efficiently indexes wave-rules so that extra wave-rules do not degrade the performance of rippling.

2. $\forall i \in \{0..k\}.flip(r, i, r_i^\uparrow - l_i^\uparrow)$ and succeed if $\mathrm{MI}(l) >_{lex} \mathrm{MI}(r)$

3. choose the minimum i such that $l_i^\uparrow \neq 0$, $flip(r, i, r_i^\uparrow - l_i^\uparrow - 1)$ and $\forall j \in \{i + 1..k\}.flip(r, j, r_j^\uparrow - l_j^\uparrow)$

Each of the three statements can be executed in linear time. Note that the first two may fail (there does not exist a maximum i in the first case, or in the second the test $\mathrm{MI}(l) >_{lex} \mathrm{MI}(r)$ fails) but the third case will always succeed.

Lemma 3 *The orientation algorithm computes all $\succ$-maximal r where $l \succ r$.*

Proof (sketch): If the first statement succeeds then $\forall j \in \{i + 1..k\}.l_j^\uparrow = r_j^\uparrow$ and $l_i^\uparrow > r_i^\uparrow$ so $\mathrm{MO}(l) >_{revlex} \mathrm{MO}(r)$ and r is maximal. Otherwise, $\forall i.l_i^\uparrow \leq r_i^\uparrow$ so we flip arrows down to equate out-orders and test $\mathrm{MI}(l) >_{lex} \mathrm{MI}(r)$. If this succeeds, we have a maximal r. Otherwise we still have $\forall i.l_i^\uparrow \leq r_i^\uparrow$ but flipping arrows in r to equate out-orders is insufficient as r then has a larger in-order. However, by assumption, l has at least one outward wavefront with a least depth i, so we can flip enough arrows at this depth so $r_i = l_i - 1$. Thus $l \succ r$ and r is maximal. $\Box$

This parser for simply annotated terms is correct (it only returns wave-rules) and complete (it returns all maximal wave-rules under the orderings we define). As an example, consider (9) with the LHS oriented all out. We begin by orienting both wavefronts in the RHS out. The two sides thus have the measures $\langle[0, 1], [0, 0]\rangle$ and $\langle[1, 1], [0, 0]\rangle$ respectively. Hence step 1 fails. Moreover, if we equate the out-measures by turning down the annotation at depth 0, this gives the RHS a measure of $\langle[0, 1], [1, 0]\rangle$ so step 2 fails. Finally we succeed in step 3 by turning down the arrow at depth 1 giving the RHS a measure of $\langle[1, 0], [0, 1]\rangle$. The resulting oriented annotation is given in (9).

5.3 Multi-waves and sinks

The above ideas generalize easily to multi-wave-rules. For reasons of space we only sketch this. We generate skeleton preserving annotations analogous to the single-hole case but allow multi-holed wavefronts. Usually both sides are simply annotated and we may use the above orientation algorithm. Alternatively, after fixing an orientation for the LHS of the wave-rule we may orient the RHS by cycling through possible orientations. For each orientation we compare the weakenings of the two sides under the multi-set ordering over our measure and we pick the RHS orientation with the greatest measure. There are various ways the efficiency of this can be enhanced. E.g. we need only compute weakenings of each side once; with "orientation variables" we may propagate the different orientations we select for the RHS to orientations on the weakening set before comparison under the multi-set measure.

One kind of annotation we haven't yet discussed in our measures is *sinks* (see §2). This is deliberate as we can safely ignore sinks in both the measure and the parser. Sinks only serve to decrease the applicability of wave-rules by creating additional preconditions; that is, we only ripple inwards if there is a sink underneath the wavefront. But if rippling terminates without such a precondition, it terminates with it as well. Sinks (and also recent additions to rippling such as colours [15]) can be seen as not effecting the termination of rippling but rather the *utility* of rippling. That is, they increase the chance that we will be able to fertilize with the hypothesis successfully.

6 Extensions to Rippling

By introducing new termination orders for rippling, we can combine rippling with conventional term rewriting. Such extensions greatly extend the power and applicability of rippling both within and outwith induction. In addition, by design, our orderings are not dependent upon rippling preserving skeletons. This allows us to use rippling in new domains involving, for example, mutual recursion or definition unfolding where the skeleton needs to be modified; such applications were previously outside the scope of rippling. We feel that these extensions offer the promise of the "best of both worlds": that is, the highly goal directed nature of rippling combined with the flexibility and uniformity of conventional rewriting. To test these ideas, we have implemented an *Annotated Rewrite System*, a simple PROLOG program which manipulates annotated terms, and in which we can mix conventional term rewriting and rippling. All the examples below have been proven by this system.

6.1 Unblocking

Rippling can sometimes become blocked. Usually the blockage occurs due to the lack of a wave-rule to move the differences out of the way; in such a situation the wave-rule may be speculated automatically using techniques presented in [13]. However, sometimes the proof becomes blocked because a wavefront needs to be rewritten so that it matches either a wave-rule (to allow further rippling) or a sink (to allow fertilization). This is best illustrated by an example.

Consider the following theorem, where rev is naive reverse, $qrev$ is tail-recursive reverse using an accumulator, $<>$ is infix append, and $::$ infix cons.

$$\forall L, M.\ qrev(L, M) = rev(L) <> M \tag{15}$$

To prove this theorem, we perform an induction on L. The induction hypothesis is

$$qrev(l, M) = rev(l) <> M$$

and the induction conclusion is

$$qrev(\boxed{h :: \underline{l}}^{\uparrow}, \lfloor m \rfloor) \quad = \quad rev(\boxed{h :: \underline{l}}^{\uparrow}) <> \lfloor m \rfloor\ . \tag{16}$$

where m is a skolem constant which sits in a sink, annotated with "$\lfloor\ \rfloor$".

We will use wave-rules taken from the recursive definition of $qrev$, and rev.

$$rev(\boxed{H :: \underline{T}}^{\uparrow}) \quad \Rightarrow \quad \boxed{\underline{rev(T)} <> (H :: nil)}^{\uparrow} \tag{17}$$

$$qrev(\boxed{H :: \underline{T}}^{\uparrow}, L) \quad \Rightarrow \quad qrev(T, \boxed{H :: \underline{L}}^{\downarrow}) \tag{18}$$

On the LHS, we ripple with (18) to give

$$qrev(l, \left\lfloor \boxed{h :: \underline{m}}^{\downarrow} \right\rfloor) \quad = \quad rev(\boxed{h :: \underline{l}}^{\uparrow}) <> \lfloor m \rfloor\ .$$

On the RHS, we ripple with (17) and then (4), the associativity of $<>$ to get

$$qrev(l, \left\lfloor \boxed{h :: \underline{m}}^{\downarrow} \right\rfloor) \quad = \quad rev(l) <> (\left\lfloor \boxed{(h :: nil) <> \underline{m}}^{\downarrow} \right\rfloor). \tag{19}$$

Unfortunately, the proof is now blocked. We can neither further ripple nor fertilize with the induction hypothesis. The problem is that we need to simplify the wavefront on the righthand side. Clam currently uses an ad-hoc method to try to perform wavefront simplification when rippling becomes blocked. In this case (19) is rewritten to

$$qrev(l, \left\lfloor \boxed{h :: \underline{m}}^{\downarrow} \right\rfloor) = rev(l) <> (\left\lfloor \boxed{h :: \underline{m}}^{\downarrow} \right\rfloor) .$$

Fertilization with the induction hypothesis can now occur.

In general, unblocking steps are not sanctioned under the measure proposed earlier, or that given in [3]; their uncontrolled application during rippling can lead to non-termination. But we can easily create new orders where unblocking steps are measure decreasing. These new orders allows us to combine rippling with conventional rewriting of wavefronts in an elegant and powerful way. Namely, unblocking rules will be measure decreasing wave-rules accepted by the parser and applied like other wave-rules.

We define an unblocking ordering by giving (as before) an ordering on simply annotated terms, which can then be lifted to an order on multi-wave terms. To order simply annotated terms, we take the lexicographic order of the simple wave-rule measure proposed above (using size of the wavefront as the notion of weight) paired with $>_{wf}$, an order on the *contents* of wavefronts. As a simply annotated term may still contain multiple wavefronts, this second order is lifted to a measure on sets of wavefronts by taking its multi-set extension. The first part of the lexicographic ordering will ensure that anything which is normally measure decreasing remains measure decreasing and the second part will allow us to orient rules that only manipulate wavefronts. This combination provides a termination ordering that allows us to use rippling to move wavefronts about the skeleton and conventional rewriting to manipulate the contents of these wavefronts.

For the reverse example, the normalization ordering is very simple; we use the following wave-rules.

$$\boxed{nil <> \underline{L}}^{\downarrow} \;\Rightarrow\; L \tag{20}$$

$$\boxed{(H :: T) <> \underline{L}}^{\downarrow} \;\Rightarrow\; \boxed{H :: (T <> \underline{L})}^{\downarrow} \tag{21}$$

The first is already parsed as a wave-rule using our standard measures, but we need to add the second. This rule doesn't change the size of the wavefront or its position but only its form. Hence we want this to be decreasing under some normalization ordering. There are many such orderings; here we take $>_{wf}$ to be the recursive path ordering [7] on the terms in the wavefront where $<>$ has a higher precedence than $::$ and all other function symbols have an equivalent but lower priority. The measure of the LHS of (21) is now greater than that of the RHS as its wavefront is $(H :: T) <> *$ which is greater than $H :: (T <> *)$ in the recursive path ordering (to convert wavefronts into well formed terms, waveholes are marked with the new symbol *).

Unblocking steps which simplify wavefronts are useful in many proofs enabling both immediate fertilization (as in this example) and continued rippling. Wavefronts can even be unblocked using a different set of rules to that used for rippling.

6.2 Mutual Recursion and Skeleton Simplification

Rippling can also become blocked because the skeleton (and not a wavefront) needs to be rewritten. This happens in proofs involving mutually recursive functions, definition unfolding, and other kinds of rewriting of the skeleton. Consider

$$\forall x.\, even(s(s(0)) \times x)$$

where *even* has the following wave-rules.

$$even(\boxed{s(\underline{U})}^{\uparrow}) \Rightarrow odd(U) \tag{22}$$

$$odd(\boxed{s(\underline{U})}^{\uparrow}) \Rightarrow even(U) \tag{23}$$

Note that (22) and (23) are not wave-rules in the conventional sense since they are not skeleton preserving. However, they do decrease the annotation measure. Rules (22) and (23) can be viewed as a more general type of wave-rule of the form $LHS \Rightarrow RHS$ which satisfy the constraint $skeleton(LHS) \equiv skeleton(RHS)$ where $\equiv$ is some equivalence relation. In this case, the equivalence relation includes the granularity relation in which $even(x)$ and $odd(x)$ are in the same equivalence class. Rippling with this more general class of wave-rules still gives us a guarantee of termination. However weakening the structure preservation requirement can reduce the utility of rippling since now we are only guaranteed to rewrite the conclusion into a member of the equivalence class of the hypothesis.

To prove the theorem, we will also need the following wave-rules.

$$\boxed{s(\underline{U})}^{\uparrow} + V \Rightarrow \boxed{s(\underline{U+V})}^{\uparrow} \tag{24}$$

$$U + \boxed{s(\underline{V})}^{\uparrow} \Rightarrow \boxed{s(\underline{U+V})}^{\uparrow} \tag{25}$$

The theorem can be proved without (25) but this requires a nested induction and generalization, complications which need not concern us here.

The proof begins with induction on x. The induction hypothesis is

$$even(s(s(0)) \times n)$$

and the induction conclusion is

$$even(s(s(0)) \times \boxed{s(\underline{n})}^{\uparrow}). \tag{26}$$

Unfortunately rippling is immediately blocked. To continue the proof, we simplify the skeleton of the induction conclusion by exhaustively rewriting (26) using the unannotated version of (1) and the following rules.

$$0 \times V \Rightarrow 0 \tag{27}$$

$$0 + V \Rightarrow V \tag{28}$$

This gives

$$even(\boxed{s(\underline{n})}^{\uparrow} + \boxed{s(\underline{n})}^{\uparrow}). \tag{29}$$

Note that the skeleton was changed by this rewriting. The induction hypothesis can, however, be rewritten using the same rules so that it matches the skeleton of (29). Of course, arbitrary rewriting of the skeleton may not preserve the termination of rippling. To justify these unblocking steps we therefore introduce a new termination order which combines lexicographically a measure on the skeleton with the measure on annotations.[4] We then admit rewrite rules provided their application decreases this combined measure. This new order allows us to combine rippling with conventional rewriting of the skeleton in an elegant and powerful way. In this case, the

[4] With more complex theorems, the height of the skeleton may increase; the addition of the height of the skeleton to the order ensures termination in such situations.

recursive path order on skeletons (with precedence $\times > + > s > 0$) is again adequate. Note that though termination is guaranteed, again skeleton preservation has been weakened. Since the skeleton can be changed during rippling, we are no longer able to guarantee that we can fertilize at the end of rippling. However, provided the skeleton is rewritten identically in both the hypotheses and the conclusion, we will still be able to fertilize.

To return to the proof, rippling (29) with (24) gives

$$even(\boxed{s(\underline{n + \boxed{s(\underline{n})}^{\uparrow}})}^{\uparrow}).$$

Then with (25) gives

$$even(\boxed{s(s(\underline{n+n}))}^{\uparrow}).$$

We now ripple with the mutually recursive definition of even, (22),

$$odd(\boxed{s(\underline{n+n})}^{\uparrow}).$$

Note that this step also changes the skeleton. However, as the measure decreases and as the skeleton stays in the same equivalence class, such rewriting is permitted. Finally rippling with (23) gives

$$even(n+n).$$

This matches the (rewritten) induction hypothesis and so completes the proof.

The power of rippling is greatly enhanced by its combination with traditional rewriting. For example, proofs involving mutually recursive functions, or other kinds of skeleton simplification (e.g., definition unfolding) were not previously possible with rippling. The use of conventional term rewriting to simplify the skeleton is a natural dual to the use of conventional rewriting to simplify wavefronts; indeed they are orthogonal and can be combined to allow even more sophisticated rewriting.

6.3 Other Applications

Rippling has found several novel uses of outside of induction. For example, it has been used to sum series [14], to prove limit theorems [15], and guide equational reasoning [11]. However, new domains, especially non-inductive ones, require new orderings to guide proof. For example, consider the PRESS system [6].[5] To solve algebraic equations, PRESS uses a set of methods which apply rewrite rules. The three main methods are: *isolation*, *collection*, and *attraction*. Below are examples of rewrite rules used by each of these methods.

$$\begin{array}{lrcl}
ATTRACTION: & \boxed{\log(\underline{U}) + \log(\underline{V})}^{\uparrow} & \Rightarrow & \boxed{\log(\underline{U} \times \underline{V})}^{\uparrow} \\
COLLECTION: & \boxed{\underline{U} \times \underline{U}}^{\uparrow} & \Rightarrow & \boxed{\underline{U}^2}^{\uparrow} \\
ISOLATION: & \boxed{\underline{U}^2}^{\uparrow} = V & \Rightarrow & U = \boxed{\pm\sqrt{\underline{V}}}^{\downarrow}
\end{array}$$

PRESS uses preconditions and not annotation to determine rewrite rule applicability. Attraction must bring occurrences of unknowns closer together. Collection must reduce the number of occurrences of unknowns. Finally, isolation must make progress towards isolating unknowns on the LHS of the equation. These requirements can easily be captured by annotation and PRESS can thus be implemented

[5]Due to space constraints, we only sketch this application. The idea of reconstructing PRESS with rippling was first suggested by Nick Free and Alan Bundy.

by rippling. The above wave-rules suggest how this would work. PRESS wave-rules are structure preserving, where the preserved structure is the unknowns. The ordering defined on these rules reflects the well-founded progress achieved by the PRESS methods. Namely, we lexicographically combine orderings on the number of waveholes for collection, their distance (shortest path between waveholes in term tree) for attraction, and our width measure on annotation weight for isolation.

7 Related Work and Experience

The measures and orders we give are considerably simpler than those in [3]. There, the properties of structure preservation and the reduction of a measure are intertwined. Bundy *et al.* describe wave-rules schematically and show that any instance of these schemata is skeleton preserving and measure decreasing under an appropriately defined measure. Mixing these two properties makes the definition of wave-rules very complex. For example, the simplest kind of wave-rule proposed are so-called *longitudinal* wave-rules (which ripple-out) defined as rules of the form,

$$\eta(\boxed{\xi_1(\underline{\mu_1^1},..,\underline{\mu_1^{p_1}})}^{\uparrow},..,\boxed{\xi_n(\underline{\mu_n^1},..,\underline{\mu_n^{p_n}})}^{\uparrow}) \Rightarrow \boxed{\zeta(\underline{\eta(\varpi_1^1,..,\varpi_n^1)},..,\underline{\eta(\varpi_1^k,..,\varpi_n^k)})}^{\uparrow}$$

that satisfy a number of side conditions. These include: each ϖ_i^j is either an unrippled wavefront, $\boxed{\xi_i(\underline{\mu_i^1},\ldots,\underline{\mu_i^{p_i}})}$, or is one of the waveholes, μ_i^l; for each j, at least one ϖ_i^j must be a wavehole. η, the ξ_is, and ζ are terms with distinguished arguments; ζ may be empty, but the ξ_is and η must not be. There are other schemata for *traverse* wave-rules and *creational* wave-rules[6]. These schemata are combined in a general format, so complex that in [3] it takes four lines to print. It is notationally involved although not conceptually difficult to demonstrate that any instance of these schemata is a wave-rule under our size and width measures.

Consider the longitudinal schema given above. It is clear that evey skeleton on the RHS is a skeleton of the LHS because of the constraint on the ϖ_j^i. What is trickier to see is that it is measure decreasing. Under our order this is the case if LHS $\succ^*$ RHS. We can show something stronger, namely, for every $r \in$ *weakenings*(RHS). $\exists l \in$ *weakenings*(LHS). $l \succ r$. To see this observe that any such r must be a maximal weakening of

$$r' = \boxed{\zeta(\eta(\varpi_1^1,\ldots,\varpi_n^1),\ldots,\underline{\eta(\varpi_1^j,\ldots,\varpi_n^j)},\ldots\eta(\varpi_1^k,\ldots,\varpi_n^k))}^{\uparrow}$$

for some $j \in \{1..k\}$. Corresponding to r' is an l' which is a weakening of the LHS where $l' = \eta(t_1,...,t_n)$ and the t_i correspond to the ith subterm of $\eta(\varpi_1^j,\ldots,\varpi_n^j)$ in r': if ϖ_i^j is an unrippled wavefront then $t_i = \varpi_i^j = \boxed{\xi_i(\underline{\mu_i^1},\ldots,\underline{\mu_i^{p_i}})}$, and alternatively if ϖ_i^j a wavehole μ_i^l then $t_i = \boxed{\xi_i(\mu_i^1,\ldots,\underline{\mu_i^l},\ldots,\mu_i^{p_i})}$. Now r is a maximal weakening of r' so there is a series of weakening steps from r to r'. Each of these weakenings occurs in a ϖ_i^j and we can perform the identical weakening steps in the corresponding t_i in l' leading to a maximal weakening l. As l and r are maximally weak they may be compared under $\succ$. Their only differences are that r has an additional wavefront at its root and is missing a wavefront at each ϖ_i^j corresponding to a wavehole. The depth of ϖ_i^j is greater than the root and at this depth the

[6]Creational wave-rules are used to move wavefronts between terms during induction proofs by destructor induction. They complicate rippling in a rather specialized and uninteresting way. Our measures could be easily generalized to include such creational rules.

out-measure of l is greater than r (under any of the weights defined in §3) and at all greater depths they are identical. Hence $l \succ r$.

Similar arguments hold for the other schemata given in [3] and from this we can conclude that wave-rules acceptable under their definition are acceptable under ours. Moreover it is easy to construct simple examples that are wave-rules under our formalism but not theirs; for example, the following two rules are measure decreasing but are not instances of their schema.

$$rot(\boxed{s(\underline{X})}^{\uparrow}, \boxed{H :: \underline{T}}^{\uparrow}, Acc) \Rightarrow rot(X, T, \boxed{H :: \underline{Acc}}^{\uparrow})$$

$$\boxed{0 + \underline{X}}^{\uparrow} \Rightarrow X$$

Aside from being more powerful, there are additional advantages to the approach taken here. Our notion of wave-rules and measures are significantly simpler and therefore easier to understand. As a result, they are easier to implement. The definition of wave-rules given in [3] is not what is recognized by the Clam wave-rule parser as it returns invalid wave-rules under either our definition or that of [3] and misses many valid ones. For example, Clam's current parser fails to find even wave-rules as simple as the following.

$$divides(\boxed{\underline{X} + Y}^{\uparrow}, Y) \quad \Rightarrow \quad \boxed{s(\underline{divides(X, Y)})}^{\uparrow}$$

We have therefore implemented the parser described in §5. The parser is simple, just a couple of pages of Prolog, yet allows new orderings based on different annotation measures to be easily incorporated. Although parsing is in the worst case exponential in the size of the rewrite rule, the parser typically takes under 5 seconds to return a complete set of maximal wave-rules (which seems reasonable for an off-line procedure). The parser is part of our annotated rewrite system and will be shortly integrated into the Clam theorem prover.

8 Conclusions

An ordering for proving the termination of rippling along with a schematic description of wave-rules was first given in [3]. We have simplified, generalized and improved both this termination ordering, and the description of wave-rules. In addition, we have shown that different termination orderings for rippling can be profitably used within and outwith induction. Such new orderings can combine the highly goal directed features of rippling with the flexibility and uniformity of more conventional term rewriting. We have, for example, given two new orderings which allow unblocking, definition unfolding, and mutual recursion to be added to rippling in a principled (and terminating) fashion; such extensions greatly extend the power of the rippling heuristic. To support these extensions, we have implemented a simple *Annotated Rewrite System* which annotates and orients rewrite rules, and with which we can rewrite annotated terms. We have used this system to perform experiments combining rippling and conventional term rewriting. We confidently expect that this combination of rippling and term rewriting has an important rôle to play in many areas of theorem proving and automated reasoning.

References

[1] D. Basin and T. Walsh. Difference unification. In *Proceedings of the 13th IJCAI*. International Joint Conference on Artificial Intelligence, 1993.

[2] R.S. Boyer and J S. Moore. *A Computational Logic*. Academic Press, 1979. ACM monograph series.

[3] A. Bundy, A. Stevens, F. van Harmelen, A. Ireland, and A. Smaill. Rippling: A heuristic for guiding inductive proofs. *Artificial Intelligence*, 62:185–253, 1993.

[4] A. Bundy, F. van Harmelen, C. Horn, and A. Smaill. The Oyster-Clam system. In M.E. Stickel, editor, *10th International Conference on Automated Deduction*. 1990.

[5] A. Bundy, F. van Harmelen, A. Smaill, and A. Ireland. Extensions to the rippling-out tactic for guiding inductive proofs. In M.E. Stickel, editor, *10th International Conference on Automated Deduction*, pages 132–146. Springer-Verlag, 1990.

[6] A. Bundy and B. Welham. Using meta-level inference for selective application of multiple rewrite rules in algebraic manipulation. *Artificial Intelligence*, 16(2):189–212, 1981.

[7] N. Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, March 1982.

[8] N. Dershowitz. Termination of Rewriting. In J.-P. Jouannaud, editor, *Rewriting Techniques and Applications*. Academic Press, 1987.

[9] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B. North-Holland, 1990.

[10] N. Dershowitz and Z. Manna. Proving termination with multiset orderings. *Comms. ACM*, 22(8):465–476, 1979.

[11] D. Hutter. Guiding inductive proofs. In M.E. Stickel, editor, *10th International Conference on Automated Deduction*. 1990.

[12] D. Hutter. Colouring terms to control equational reasoning. An Expanded Version of PhD Thesis: Mustergesteuerte Strategien für Beweisen von Gleichheiten (Universität Karlsruhe, 1991), in preparation.

[13] A. Ireland and A. Bundy. Using failure to guide inductive proof. Technical report, Dept. of Artificial Intelligence, University of Edinburgh, 1992.

[14] T. Walsh, A. Nunes, and A. Bundy. The use of proof plans to sum series. In D. Kapur, editor, *11th Conference on Automated Deduction*. 1992.

[15] T. Yoshida, A. Bundy, I. Green, T. Walsh, and D. Basin. Coloured rippling: the extension of a theorem proving heuristic. Technical Report, Dept. of Artificial Intelligence, University of Edinburgh, 1993. Under review for ECAI-94.