Planning and Patching Proofs

Alan Bundy & Lucas Dixon
School of informatics
University of Edinburgh

Why Inductive Reasoning is Necessary

Proof by mathematical induction required for reasoning about repetition, e.g. in:

- recursive datatypes: numbers, lists, sets, trees, etc;
- iterative or recursive computer programs;
- electronic circuits with loops or parameterisation.

So needed for proof obligations in formal methods.

Proof Plans: What Are They?

- Attempt to capture common structure of family of proofs.
- Used to guide search for new proofs from same family.
- Three parts: tactic, method and critics.

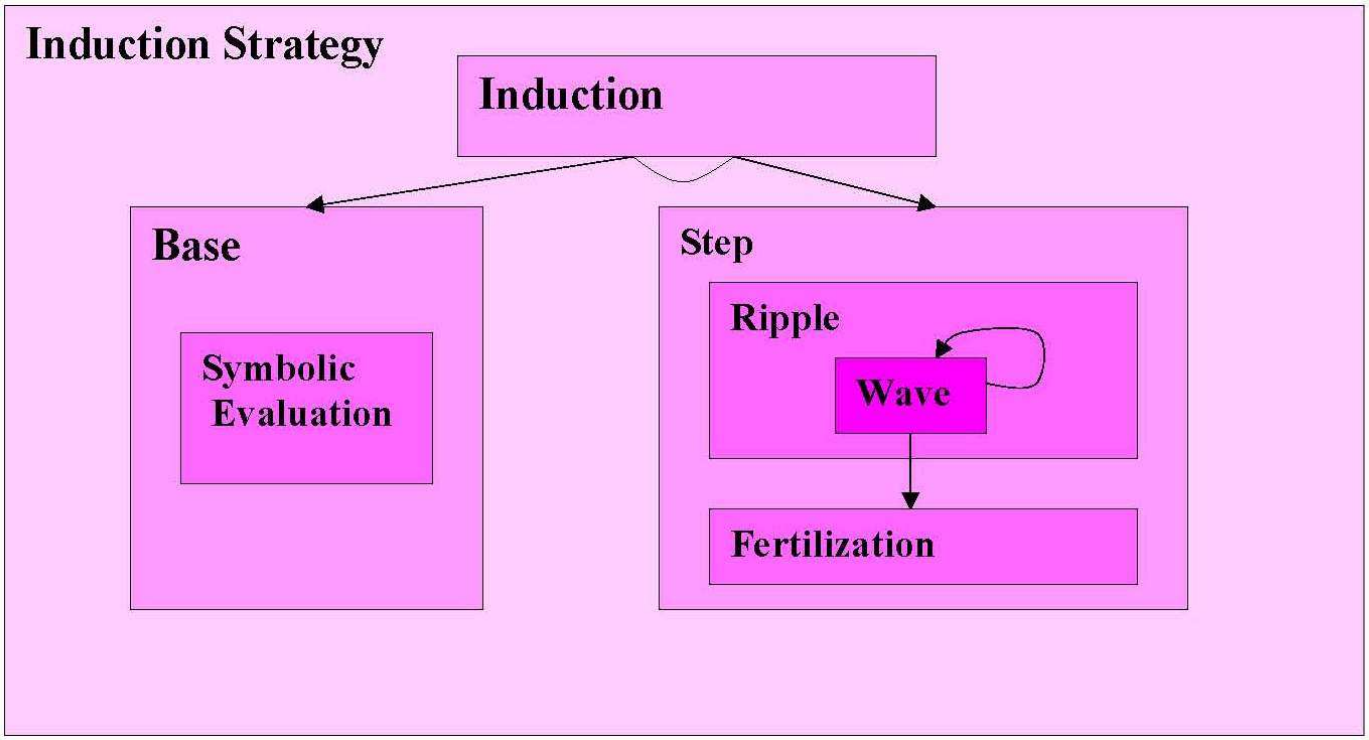
  Tactic is computer program for applying rules of inference.

  Method is meta-logical specification of tactic.

  Critic analyses failure and suggests patch.
- Use AI plan formation to construct special-purpose proof plan for conjecture using general-purpose sub-proof plans.
- Allows flexible application of heuristics.
- Understanding gained suggests extensions of heuristics.

General-Purpose Proof Plans

A Strategy for Inductive Proof: ind_strat



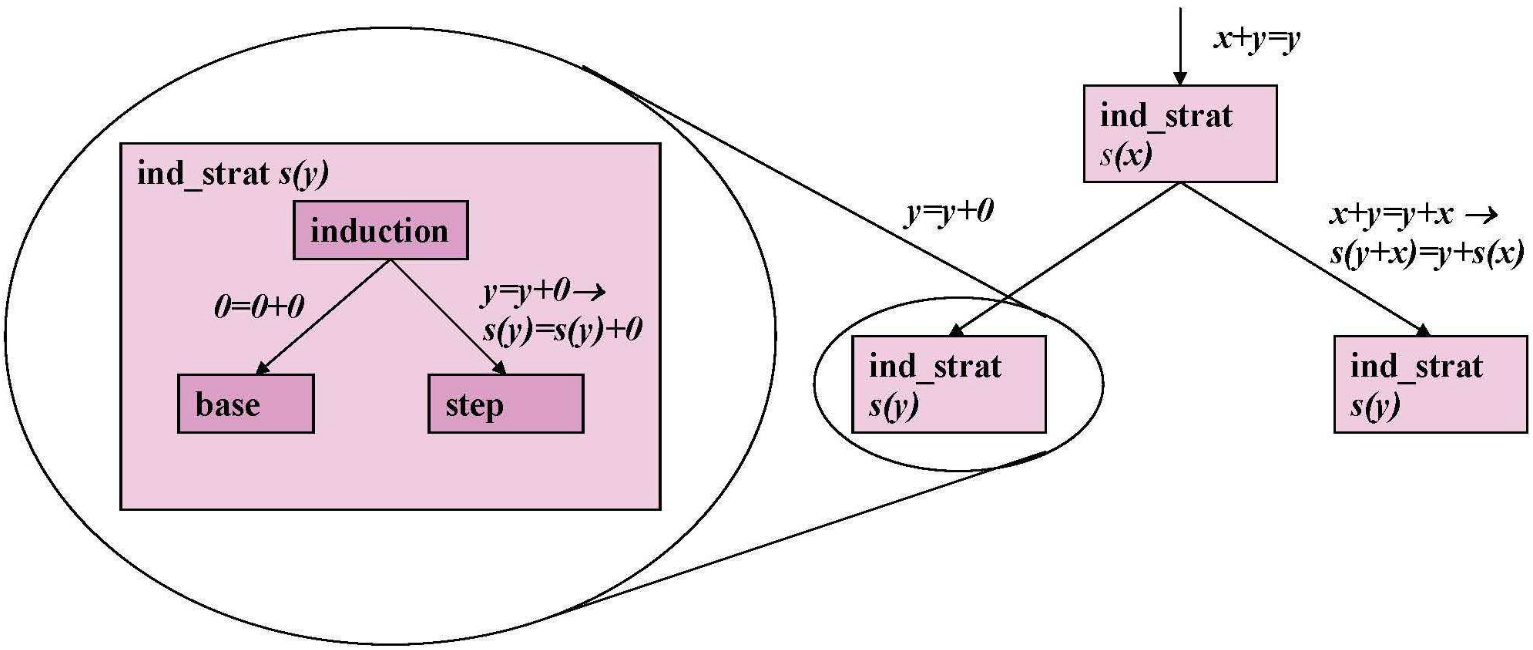
Preconditions:

Declarative: Rippling must be possible in step cases.

Procedural: Look-ahead to choose induction rule that will permit rippling.

Special-Purpose Proof Plan

Commutativity of $+$:



Empirical Success of Proof Planning

- Implemented in *Clam*/λ*Clam*, Ωmega and IsaPlanner proof planners.
- Successfully tested on a wide range domains:
  induction, analysis, residue classes, diagonalisation, summing series, equational reasoning, process algebras, transfinite ordinals, completeness proofs, ...
- Applied outwith mathematics: computer configuration and bridge.
- Solution of eureka problems using critics,
  e.g. lemma discovery and generalisation.
- Applications to software/hardware verification/synthesis/transformation.
  e.g. verification of Gordon computer,
  construction of loop invariants via critics.
- Ωmega linked to 3^{rd} party provers, CAS, constraint solvers, *etc.*
- IsaPlanner part of Isabelle tactic-based theorem prover.

Need for Intermediate Lemmas

Conjecture:

$$\forall l{:}list(\tau).\ rev(rev(l)) = l$$

Rewrite Rules:

$$rev(nil) \Rightarrow nil$$

$$rev(H :: T) \Rightarrow rev(T) @ (H :: nil)$$

Step Case:

$$rev(rev(t)) = t \vdash rev(\boldsymbol{rev(h :: t)}) = h :: t$$

$$\vdash \underbrace{\boldsymbol{rev(rev(t)@(h :: nil))}}_{\textbf{blocked}} = h :: t$$

Introducing an Intermediate Lemma

Lemma Required:

$$rev(X@Y) \Rightarrow rev(Y)@rev(X)$$

Cut Rule: introduces this:

Original: $\Gamma \vdash rev(rev(l)) = l$

New:

$\Gamma, \boldsymbol{rev(X@Y) \Rightarrow rev(Y)@rev(X)} \vdash rev(rev(l)) = l$

Justification: $\Gamma \vdash \boldsymbol{rev(X@Y) \Rightarrow rev(Y)@rev(X)}$

Heuristics needed: to speculate lemma.

Step Case Unblocked

$$
\begin{aligned}
rev(rev(t)) = t \vdash\ & rev(\boldsymbol{rev(h :: t)}) = h :: t \\
\vdash\ & \boldsymbol{rev(rev(t)@(h :: nil))} = h :: t \\
\vdash\ & \boldsymbol{rev(h :: nil)}@rev(rev(t)) = h :: t \qquad \textit{lemma applied} \\
\vdash\ & (\boldsymbol{rev(nil)}@(h :: nil))@rev(rev(t)) = h :: t \\
\vdash\ & (\boldsymbol{nil@(h :: nil)})@rev(rev(t)) = h :: t \\
\vdash\ & \boldsymbol{(h :: nil)@rev(rev(t))} = h :: t \\
\vdash\ & h :: (\boldsymbol{nil@rev(rev(t))}) = h :: t \\
\vdash\ & \boldsymbol{h :: rev(rev(t)) = h :: t} \\
\vdash\ & h = h \wedge \boldsymbol{rev(rev(t)) = t}
\end{aligned}
$$

fertilization now possible.

Rippling in the Step Case

$$\boldsymbol{t@(Y@Z) = (t@Y)@Z}$$

$$\vdash \boxed{h :: \underline{t}}^{\uparrow} @(y@z) = (\boxed{h :: \underline{t}}^{\uparrow} @y)@z$$

$$\vdash \boxed{h :: \underline{t@(y@z)}}^{\uparrow} = \boxed{h :: \underline{t@y}}^{\uparrow} @z$$

$$\vdash \boxed{h :: \underline{t@(y@z)}}^{\uparrow} = \boxed{h :: \underline{(t@y)@z}}^{\uparrow}$$

$$\vdash \boxed{h = h \wedge \underline{t@(y@z) = (t@y)@z}}^{\uparrow}$$

- changing bits in $\boxed{\textit{orange boxes}}^{\uparrow}$ (wave-fronts).
- unchanging bits in ***red*** (skeleton).

Wave-Rules

$$\boxed{H :: \underline{T}}^{\uparrow} @ L \Rightarrow \boxed{H :: \underline{T @ L}}^{\uparrow}$$

$$rev(\boxed{H :: \underline{T}}^{\uparrow}) \Rightarrow \boxed{\underline{rev(T)} @ (H :: nil)}^{\uparrow}$$

$$\boxed{X_1 :: \underline{X_2}}^{\uparrow} = \boxed{Y_1 :: \underline{Y_2}}^{\uparrow} \Rightarrow \boxed{X_1 = Y_1 \wedge \underline{X_2 = Y_2}}^{\uparrow}$$

$$X @ \boxed{(\underline{Y} @ Z)}^{\uparrow} \Rightarrow \boxed{(\underline{X @ Y}) @ Z}^{\uparrow}$$

- note preservation of skeleton and
- outward movement of wave-fronts.
- Can be annotated statically or dynamically.

Rippling Sideways and In

$$rev(t)@L = qrev(t, L)$$

$$\vdash rev(\boxed{h :: \underline{t}}^{\uparrow})@\lfloor l \rfloor = qrev(\boxed{h :: \underline{t}}^{\uparrow}, \lfloor l \rfloor)$$

$$\vdash (\boxed{\underline{rev(t)}@(h :: nil)}^{\uparrow})@\lfloor l \rfloor = qrev(t, \boxed{h :: \underline{\lfloor l \rfloor}}^{\downarrow})$$

$$\vdash rev(t)@(\boxed{(h :: nil)@\underline{\lfloor l \rfloor}}^{\downarrow}) = qrev(t, \lfloor h :: l \rfloor)$$

$$\vdash rev(t)@(\lfloor (h :: nil)@l \rfloor) = qrev(t, \lfloor h :: l \rfloor)$$

$$\vdash rev(t)@(\lfloor h :: l \rfloor) = qrev(t, \lfloor h :: l \rfloor)$$

- Fertilization unifies L with $\lfloor h :: l \rfloor$.
- $\lfloor \textbf{\textit{Sinks}} \rfloor$ provide alternative wave-front destination, available when free variables are in hypothesis.
- Wave-fronts have directions: $\boxed{out}^{\uparrow}$ / $\boxed{in}^{\downarrow}$.
- Note that sinks and wave-fronts may need to be simplified, but this is skeleton preserving.

Sideways and Inwards Wave-rules

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

$$\boxed{H :: \boxed{T@L}}^{\downarrow} \Rightarrow \boxed{H :: \boxed{T}}^{\downarrow} @L$$

$$(\boxed{\boxed{X}@Y}^{\uparrow})@Z \Rightarrow X@(\boxed{Y@\boxed{Z}}^{\downarrow})$$

- Note that some equations can be annotated in both directions.

$$\boxed{H :: \boxed{T}}^{\uparrow} @L \Rightarrow \boxed{H :: \boxed{T@L}}^{\uparrow}$$

$$H :: \boxed{\boxed{T}@L}^{\uparrow} \Rightarrow \boxed{\boxed{H :: T}@L}^{\uparrow}$$

$$X@(\boxed{Y@\boxed{Z}}^{\uparrow}) \Rightarrow (\boxed{\boxed{X}@Y}^{\downarrow})@Z$$

Preconditions of the Wave Method

1. The induction conclusion contains a wave-front,

   $e.g.\ \ldots = qrev(\boxed{h :: \underline{t}}^{\uparrow}, \lfloor l \rfloor).$

2. A wave-rule applies to this wave-front.

   $e.g.\ qrev(\boxed{H :: \underline{T}}^{\uparrow}, L) \Rightarrow qrev(T, \boxed{H :: \underline{L}}^{\downarrow}).$

3. Any condition is provable.

   $e.g.\ X \neq H \rightarrow X \in \boxed{H :: \underline{T}}^{\uparrow} \Rightarrow X \in T.$

4. Inserted inwards wave-fronts contain a sink or an outwards wave-front.

   $e.g.\ \ldots = qrev(t, \boxed{h :: \underline{\lfloor l \rfloor}}^{\downarrow}).$

Advantages of Rippling

Selective: not exhaustive rewriting.

skeleton preserving and measure decreasing.

Bi-directional: rewriting.

different annotations in each direction.

Termination: of any set of wave-rules,

despite bi-directionality.

Heuristic basis: for choosing lemmas, generalisations, case splits and inductions.

(Lack of) Cut Elimination

Gentzen's Cut Rule:

$$\frac{\boldsymbol{A}, \Gamma \vdash \Delta, \quad \Gamma \vdash \boldsymbol{A}}{\Gamma \vdash \Delta}$$

lacks subformula property.

Cut Elimination Theorem:

Gentzen showed Cut Rule redundant in FOL.

Kreisel showed necessary in inductive theories.

Practical Consequences:

Need to generalise conjectures.

Need to introduce lemmas.

Ripple-Based Heuristics

Induction Rules: choose induction which best supports rippling.

Lemmas: design wave-rule to unblock ripple.

Generalisation: generalise goal to allow wave-rule to apply.

Lemma Calculation 1

- **Conjecture:**

$$\forall k, l{:}list(\tau).\ \mathbf{rev(k@l)} = \mathbf{rev(l)@rev(k)}$$

- **Wave-Rules:**

$$\boxed{H :: \underline{\mathbf{T}}}^{\uparrow} \mathbf{@L} \Rightarrow \boxed{H :: \underline{\mathbf{T@L}}}^{\uparrow}$$

$$\mathbf{rev}(\boxed{H :: \underline{\mathbf{T}}}^{\uparrow}) \Rightarrow \boxed{\underline{\mathbf{rev(T)}}@(H :: nil)}^{\uparrow}$$

- **Rippling in Step Case:**

$$\mathbf{rev}(\boxed{h :: \underline{\mathbf{t}}}^{\uparrow} \mathbf{@l}) = \mathbf{rev(l)@rev}(\boxed{h :: \underline{\mathbf{t}}}^{\uparrow})$$

$$\mathbf{rev}(\boxed{h :: \underline{\mathbf{t@l}}}^{\uparrow}) = \underbrace{\mathbf{rev(l)@}(\boxed{\underline{\mathbf{rev(t)}}@(h :: nil)}^{\uparrow})}_{\textbf{blocked}}$$

$$\boxed{\underline{\mathbf{rev(t@l)}}@(h :: nil)}^{\uparrow} = \underbrace{\mathbf{rev(l)@}(\boxed{\underline{\mathbf{rev(t)}}@(h :: nil)}^{\uparrow})}_{\textbf{blocked}}$$

Lemma Calculation 2

- **Induction Hypothesis:**

$$\boldsymbol{rev(t@l)} = rev(l)@rev(t)$$

- **Blocked Step Case:**

$$\boldsymbol{rev(t@l)}@(h :: nil) = rev(l)@(rev(t)@(h :: nil))$$

- **After Weak Fertilization:**

$$(rev(l)@rev(t))@(h :: nil) = rev(l)@(rev(t)@(h :: nil))$$

- **Generalise Common Subterm:**

$$(u@v)@w = u@(v@w)$$

i.e., we have calculated that we need associative of @ as a lemma.

Critic: Lemma Speculation

- **Conjecture:**

$$\forall l{:}list(\tau).\ rev(rev(l)) = l$$

- **Wave-Rule:**

$$rev(\boxed{H :: \underline{T}}^{\uparrow}) \Rightarrow \boxed{\underline{rev(T)} @ H :: nil}^{\uparrow}$$

- **Induction Conclusion:**

$$rev(rev(\boxed{h :: \underline{t}}^{\uparrow})) = \boxed{h :: \underline{t}}^{\uparrow}$$

$$\underbrace{rev(\boxed{\underline{rev(t)} @ (h :: nil)}^{\uparrow})}_{\textbf{blocked}} = \boxed{h :: \underline{t}}^{\uparrow}$$

- **Pattern Sought:**

$$rev(\boxed{\underline{X} @ Y}^{\uparrow}) \Rightarrow \boxed{F(\underline{rev(X)}, X, Y)}^{\uparrow}$$

- **Lemma Discovered:**

$$rev(\boxed{\underline{X} @ Y}^{\uparrow}) \Rightarrow \boxed{rev(Y) @ \underline{rev(X)}}^{\uparrow}$$

Failure of Ripple Preconditions

- Precondition 1 is true:

1. The induction conclusion contains a wave-front.

$$\boldsymbol{rev}(\boxed{\boldsymbol{rev}(\boldsymbol{t})@(h :: nil)}^{\uparrow}) = \boxed{h :: \boldsymbol{t}}^{\uparrow}$$

(in fact, two)

- Precondition 2 is false:

2. A wave-rule applies to this wave-front.

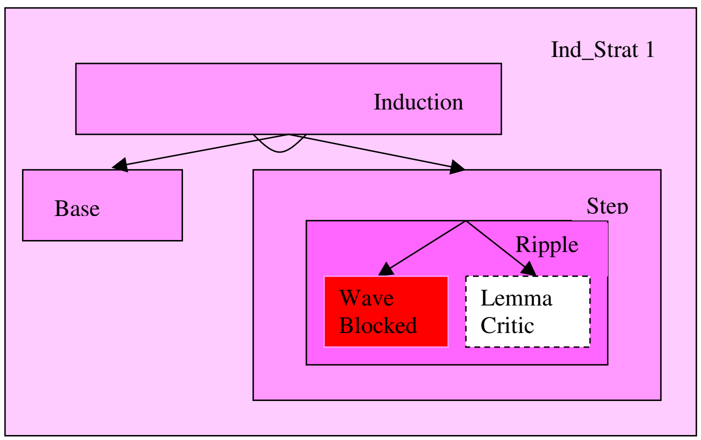
(to neither of them)

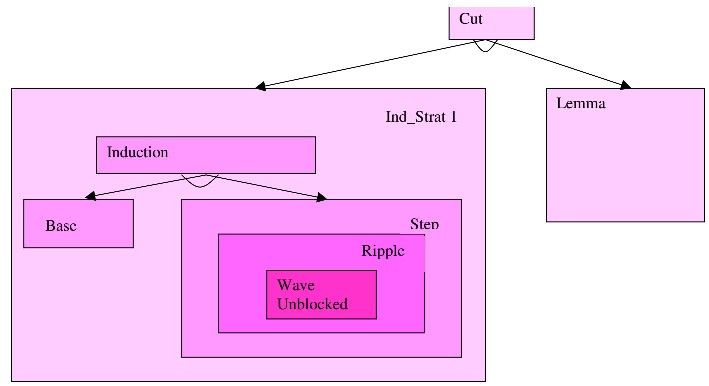
- Preconditions 3 and 4 are inapplicable.

3. Any condition is provable.
4. Inserted inwards wave-fronts contain a sink or an outwards wave-front.

Overview of Lemma Speculation Critic

Critic Invocation: $\Rightarrow$ Critic Applied:





Rippling Failure: Missing Sink

Conjecture:

$$\forall l : list(\tau).\ rev(l) = qrev(l, nil)$$

Wave-Rules:

$$rev(\boxed{H :: \underline{T}}^{\uparrow}) \Rightarrow \boxed{\underline{rev(T)} @ H :: nil}^{\uparrow}$$

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

Induction Conclusion:

$$rev(\boxed{h :: \underline{t}}^{\uparrow}) = qrev(\boxed{h :: \underline{t}}^{\uparrow}, nil)$$

$$\boxed{\underline{rev(t)} @ h :: nil}^{\uparrow} = \underbrace{qrev(\boxed{h :: \underline{t}}^{\uparrow}, nil)}_{\text{missing sink}}$$

Failure of Ripple Preconditions

- Preconditions 1, 2 and 3 are true:

1. The induction conclusion contains a wave-front.

$$\ldots = qrev(\boxed{h :: \underline{t}}^{\uparrow}, nil)$$

2. A wave-rule applies to this wave-front.

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

3. Any condition is provable — trivially, no condition.

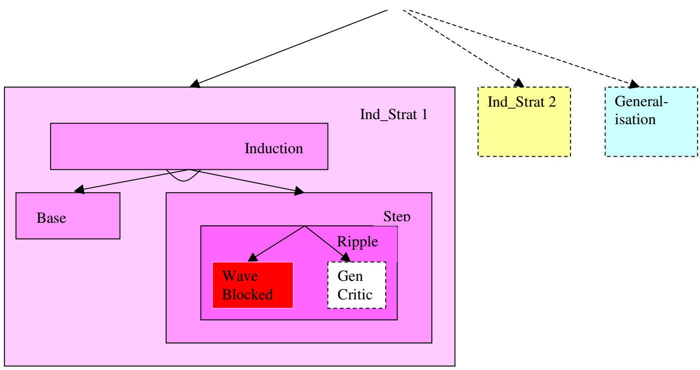
- Precondition 4 is false:

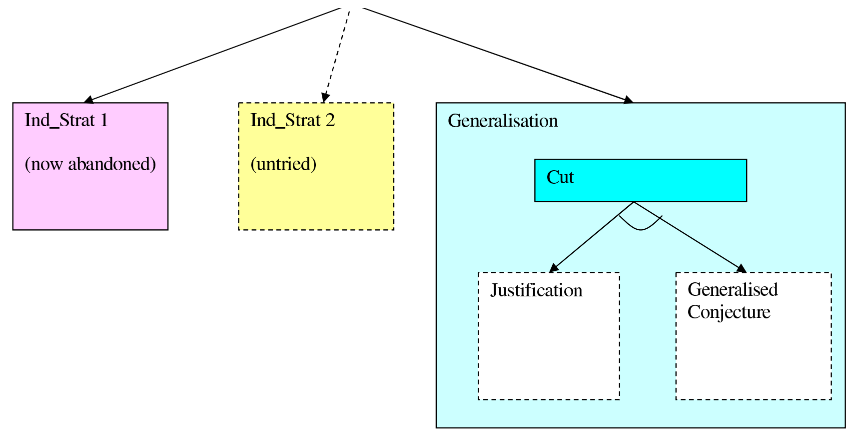
4. Inserted inwards wave-fronts contain a sink or an outwards wave-front.

$$\ldots = qrev(t, \boxed{h :: \underline{nil}}^{\downarrow})$$

Overview of Generalisation Critic

Critic Invocation: $\Rightarrow$ Critic Applied:





Patch: Sink Speculation

Original Conjecture:

$$\forall t{:}list(\tau).\ rev(t) = qrev(t, nil)$$

Disallowed Ripple:

$$\ldots = \boldsymbol{qrev(t,}\ \boxed{h :: \underline{\boldsymbol{nil}}}^{\downarrow}\ \boldsymbol{)}$$

Schematic Conjecture:

$$\forall t{:}list(\tau).\forall l{:}list(\tau).\ F(rev(t), l) = qrev(t, G(l))$$

Induction Hypothesis:

$$\boldsymbol{F(rev(t), L) = qrev(t, G(L))}$$

where F, G and L are meta-variables.

Patch: Instantiating the Meta-Variables

New Step Case:

$$
\begin{aligned}
F(rev(\boxed{h :: \underline{t}}^{\uparrow}), \lfloor l \rfloor) &= qrev(\boxed{h :: \underline{t}}^{\uparrow}, G(\lfloor l \rfloor)) \\
F(\boxed{\underline{rev(t)} @ h :: nil}^{\uparrow}, \lfloor l \rfloor) &= qrev(t, \boxed{h :: \underline{G(\lfloor l \rfloor)}}^{\downarrow}) \\
rev(t)@(\boxed{h :: nil @ \underline{F'(\boxed{\underline{rev(t)} @ (h :: nil)}^{\uparrow}, \lfloor l \rfloor)}}^{\downarrow}) & \\
&= qrev(t, \boxed{h :: \underline{G(\lfloor l \rfloor)}}^{\downarrow}) \\
rev(t)@(\boxed{h :: \underline{F'(\boxed{\underline{rev(t)} @ (h :: nil)}^{\uparrow}, \lfloor l \rfloor)}}^{\downarrow}) & \\
&= qrev(t, \boxed{h :: \underline{G(\lfloor l \rfloor)}}^{\downarrow}) \\
rev(t)@(\lfloor h :: l \rfloor) &= qrev(t, \lfloor h :: l \rfloor)
\end{aligned}
$$

where $F = @$, $F' = \lambda X.\lambda Y.\ Y$ and $G = \lambda X.\ X$.

Key Wave-Rule: $(\boxed{\underline{X} @ Y}^{\uparrow})@Z \Rightarrow X@(\boxed{Y @ \underline{Z}}^{\downarrow})$

Generalised Conjecture: $\forall t{:}list(\tau).\forall l{:}list(\tau).\ rev(t)@l = qrev(t, l)$

Pattern of Failure Suggests Patch

	PC 1	PC 2	PC 3	PC 4
Generalization	$\surd$	$\surd$	$\surd$	$\times$
Case Split	$\surd$	$\surd$	$\times$	
Induction Revision	$\surd$	?		
Lemma Discovery	$\surd$	$\times$		

$\surd$ = success   ? = partial success $\times$ = failure

Advantages of Proof Planning

- Reduction in search: larger steps, fewer options.
- Multi-level proof explanation: supports interaction.
- Non-standard proof exploration,

  least-commitment devices: meta-variables and constraints.

  agent-based proof planning.
- Framework for inter-operating reasoners.

Disadvantages of Proof Planning

- Loss of completeness: limited to methods and critics.
- Lack of serendipity: only anticipated patterns.
- Hard work to discover new proof plans,
  need to invent new concepts, *e.g.* wave-fronts,
  which are then a barrier to human understanding.
- Danger of over-tuning:
  a proof plan for every proof.

Summary

- Negative theoretical results create special search problems.
- These problems common in practice:
  induction rule choice, lemmas & generalisations.
- Proof plan for induction based on rippling.
- Rippling: selective; bidirectional; terminating and
  offers heuristic solution to special problems.
- Ripple breakdowns suggest: induction revision; lemma speculation or generalisation.
  Different patterns of proof breakdown suggest different patches.
- Implemented via proof planning with critics.