DEPENDENCY IN REFINEMENT LOGICS

JONATHAN STERLING

ABSTRACT. We propose a new generalization of LCF-style refinement logics called ***Dependent LCF***, which allows the definition of refinement rules which express a dependency between subgoals, without the use of unification variables.

Over the past four decades, there have been many incarnations of the *LCF* (Logic for Computable Functions) interface, but the one that will concern us here is its extension to *LCF with Validations* as found in Cambridge LCF, which outfits the proving activity with the synthesis of explicit evidence. The Cambridge LCF signature is as follows:

```
signature CAMBRIDGE_LCF =
sig
  type form
  type thm
  type proof = thm list ⇀ thm
  type goal = form list ⊗ form
  type tactic = goal ⇀ goal list ⊗ proof
end
```

The `goal` type represents sequents, where `form` represents logical propositions. The LCF methodology, however, may be used to give a refinement treatment to many kinds of judgments, not just sequents; therefore, it is better to not include this in the core signature at all, and make the type of judgments abstract; for clarity, we will replace the name `thm` with `evidence`.

```
signature LCF =
sig
  type judgment
  type evidence
  type proof = evidence list ⇀ evidence
  type tactic = judgment ⇀ judgment list ⊗ proof
end
```

Then, a tactic is something that refines a judgment to a list of judgments (its subgoals), and synthesizes its evidence (provided the evidence of its subgoals). There are many different tactics which can be implemented generically over this signature; here are a few:

```
signature TACTICALS =
sig
  structure Lcf : LCF
  val ID : Lcf.tactic
  val FAIL : Lcf.tactic
  val THEN : Lcf.tactic ⊗ Lcf.tactic ⇀ Lcf.tactic
  val THENL : Lcf.tactic ⊗ Lcf.tactic list ⇀ Lcf.tactic
end
```

1. Modernized LCF: The Logic of Tactics

In order to study the design space for LCF refiners, we would like to give a judgmental characterization of tactic systems. First, we will recognize that the type of validations (`proof`) in the ML implementation is essentially a HOAS (higher-order abstract syntax) encoding of a hypothetical proof or synthesis. Then, it is clear that the list of subgoals generated by a tactic are a context which explains the variables bound by the hypothetical synthesis.

To make the preceding observations precise, we can recover the character of an LCF refiner in a single form of judgment, $\boxed{J \Vdash \tau \Rightarrow E \dashv \Psi}$, where J is a judgment of the logical theory, τ is a tactic, Ψ is a context of judgments, representing the subgoals generated by the tactic τ, and E is the synthesis of the judgment J, binding variables $|\Psi|$ which represent the syntheses of the subgoals. The meaning of this assertion is that τ is applicable to demonstrating the judgment J, producing synthesis e under the assumptions that the judgments in Ψ can be demonstrated. In addition to the refinement judgment, there is also refinement failure $J \Vdash \tau \Uparrow$, which expresses the inapplicability of τ to J. We require as a matter of policy that the applicability of a tactic be discrete, and in practice, we leave implicit the rules which explain the inapplicability of a tactic.

Remark 1.1. The refinement judgments $J \Vdash \tau \Rightarrow E \dashv \Psi$ and $J \Vdash \tau \Uparrow$ are not higher-order judgments, because the variable J is ranges not over judgments of the refinement theory, but of the object theory.

Let $\mathfrak{J}$ be the open-ended collection of judgments in our logical theory, and let $\mathfrak{R}$ be a collection of rule names. Each rule $R \in \mathfrak{R}$ must be interpretable as a tactic, i.e. the meaning of the assertions $J \Vdash R \Rightarrow E \dashv \Psi$ and $J \Vdash R \Uparrow$ must be explained for $J \in \mathfrak{J}$. We say τ *tactic* in case for all object-judgments $J \in \mathfrak{J}$, the assertion conditions for $J \Vdash \tau \Rightarrow E \dashv \Psi$ and $J \Vdash \tau \Uparrow$ are disjoint, and moreover, if $J \Vdash \tau \Rightarrow E \dashv \Psi$, then $FV(E) \subseteq |\Psi|$.

In practice, we will explain only the assertion conditions for one of $J \Vdash \tau \Rightarrow E \dashv \Psi$ and $J \Vdash \tau \Uparrow$, and implicitly take the other to be its complement.

Numerous general purpose tactics can be defined over a logical theory, including identity, failure, disjunction and sequencing:

$$\frac{}{J \Vdash \mathtt{id} \Rightarrow \alpha \dashv \alpha : J} \qquad \frac{}{J \Vdash \mathtt{fail} \Uparrow}$$

$$\frac{J \Vdash \tau_1 \Rightarrow E \dashv \Psi}{J \Vdash \tau_1 | \tau_2 \Rightarrow E \dashv \Psi} \qquad \frac{J \Vdash \tau_1 \Uparrow \quad J \Vdash \tau_2 \Rightarrow E \dashv \Psi}{J \Vdash \tau_1 | \tau_2 \Rightarrow E \dashv \Psi}$$

$$\frac{J \Vdash \tau_1 \Rightarrow E \dashv \Phi \quad |_{\alpha}\, \Phi(\alpha) \Vdash \tau_2 \Rightarrow F_\alpha \dashv \Psi_\alpha \;\; (\alpha \in |\Phi|)}{J \Vdash \tau_1 ; \tau_2 \Rightarrow [F \,/\, |\Phi|]\, E \dashv \bigoplus_{|\Phi|} \Psi}$$

Remark 1.2. The nominal treatment that we have given here allows for a much more economical presentation of the standard tacticals, which are quite ardruous to define in the HOAS treatment used in ML implementations.

Theorem 1.3. *The above rules all define valid tactics:.*

(1) $\mathtt{id}$ *tactic.*
(2) $\mathtt{fail}$ *tactic.*
(3) *If* τ_1 *tactic and* τ_2 *tactic, then* $\tau_1 | \tau_2$ *tactic.*
(4) *If* τ_1 *tactic and* τ_2 *tactic, then* $\tau_1 ; \tau_2$ *tactic.*

Proof. It suffices to verify that the synthesis E of each tactical is well-scoped in Ψ; in each case, this follows by induction.

(1–2) Immediate.
(3) By induction on derivations of $J \Vdash \tau_1 | \tau_1 \Rightarrow \Psi \dashv E$.
Case $J \Vdash \tau_1 \Rightarrow E \dashv \Psi$: Validity follows from the inductive hypothesis τ_1 *tactic.*
Case $J \Vdash \tau_1 \Uparrow$: Validity follows from the inductive hypothesis τ_2 *tactic.*
(4) We need to show that $FV([F \,/\, |\Phi|]\, E) \subseteq \left|\bigoplus_{|\Phi|} \Psi\right|$. From our inductive hypotheses, it is evident that $FV(E) \subseteq |\Phi|$ and that for each $\alpha \in \Phi$, $FV(F_\alpha) \subseteq \left|\Psi_\alpha\right|$; all of the free variables of the term got by substituting each F_α for $\alpha \in |\Phi|$ clearly reside in one of the fibres of the family of contexts Ψ, and so they must comprise a subset of the union $\bigoplus_{|\Phi|} \Psi$ of Ψ's fibres.

$\square$