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 `synthesis`.

```
signature LCF =
sig
  type judgment
  type synthesis
  type proof = synthesis list ⇀ synthesis
  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 syntheses 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, which we will call ***Modernized LCF***. To begin with, note 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 of a judgment. With this insight in hand, we are in a position to unify the list of subgoals and the validation generated by a tactic into a single concept, namely that of a *hypothetical proof* E whose free variables are explained in a context Ψ of subgoals.

To make the preceding observations precise, we can characterize the behavior of a ***Modernized LCF*** refiner judgmentally via two forms of judgment, $\boxed{J \Vdash \tau \Rightarrow E \dashv \Psi}$ and $\boxed{J \Vdash \tau \Uparrow}$, 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 $J \Vdash \tau \Rightarrow E \dashv \Psi$ is that τ is applicable to demonstrating the judgment J, producing synthesis E under the assumptions that the judgments in Ψ can be demonstrated. The divergence judgment $J \Vdash \tau \Uparrow$ expresses the inapplicability of τ to J. 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.

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 ranges not over judgments of the refinement theory, but of the object theory.

In order to properly treat generic judgments whose syntheses bind variables, we will commit up front to using the abstract binding trees logical framework as a universal domain of discourse, with a distinguished sort jdg for object-judgments; and to each judgment J such that $\Omega \rhd \Upsilon \parallel \Gamma \vdash J \in_{\mathsf{wf}} \mathsf{jdg}$, a *valence* $\mathfrak{v}(J)$ shall be associated which describes the binding structure of its synthesis.

Then, we will need to define the wellformedness of subgoal contexts Ψ, simultaneously associating to Ψ a metavariable context Ω which will stand for the collected syntheses of

Ψ:

$$\frac{}{\cdot\ ctx \rightsquigarrow \cdot} \qquad \frac{\Psi\ ctx \rightsquigarrow \Omega \quad \cdot \rhd \cdot \parallel \cdot \vdash J \in_{\mathbf{wf}} \mathsf{jdg}}{\Psi, \alpha : J\ ctx \rightsquigarrow \Omega, \alpha : \mathfrak{v}(J)}$$

Remark 1.2. Note how the context-wellformedness judgment artificially requires that J be wellformed in the empty metavariable context.

Then, the judgments $J \Vdash \tau \Uparrow$ and $J \Vdash \tau \Rightarrow E \dashv \Psi$ shall presuppose $\cdot \rhd \cdot \parallel \cdot \vdash J \in_{\mathbf{wf}} \mathsf{jdg}$, and the latter shall postsuppose $\Psi\ ctx \rightsquigarrow \Omega$ and $\Omega \rhd \cdot \parallel \cdot \vdash E \in_{\mathbf{wf}} \mathfrak{v}(J)$. When E binds no variables or symbols, we shall write E instead of $\{\cdot\}[\cdot].E$.

Let $\mathfrak{J}$ be the set $\{J \mid \cdot \rhd \cdot \parallel \cdot \vdash J \in_{\mathbf{wf}} \mathsf{jdg}\}$, 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 $\boxed{\tau\ tactic}$ 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 well-defined and disjoint.

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

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

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

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

Remark 1.3. 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.

The sequencing tactical $\tau_1; \{\tau_\alpha\}_{\alpha \in I}$ corresponds to `THENL` in LCF; the uniform sequencing tactical `THEN` may be recovered as a definitional extension, with its second argument given as the constant family:

$$\tau_1; \tau_2 \triangleq \tau_1; \{\tau_2\}$$

Theorem 1.4. *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 for any* $\alpha \in I$, τ_α *tactic, then* $\tau_1; \{\tau_\alpha\}_{\alpha \in I}$ *tactic.*

Proof. Disjointness for each tactical is immediately evident; it suffices, then, to verify the postsuppositions $\Psi\ \mathit{ctx} \rightsquigarrow \Omega$ and $\Omega \rhd \cdot \parallel \cdot \vdash E \in_{\mathbf{wf}} \mathfrak{v}(J)$ in each case.

(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) It suffices to verify the cases where $|\Phi| \equiv I$, since otherwise $J \Vdash \tau_1; \{\tau_\alpha\}_{\alpha \in I} \Uparrow$. 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 $\alpha \in \Phi$, $FV(F_\alpha) \subseteq |\Psi_\alpha|$; all of the free metavariables 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.

□

2 ***Modernized LCF*** and the constructible subgoals property

An LCF-style refiner has a property called *constructible subgoals*, which means that the subgoals incurred by a rule or tactic may be constructed independently, using only the statement of the main goal. One unfortunate consequence of this principle is that it is impossible to define a refinement rule which expresses a dependency between subgoals. The canonical example is the introduction rule for dependent pairs in Type Theory:

$$\frac{M \in A \qquad N \in [M \,/\, x]\, B}{\langle M, N \rangle \in (x : A) \times B}\ \mathit{PairIntro}$$

However, consider what a refinement rule for this would look like; we would like to take the goal $(x : A) \times B\ \mathit{true}$ to two subgoals (one for each conjunct); the first subgoal $A\ \mathit{true}$ is clear enough, but it is not possible to even write down the second subgoal until we know the synthesis of the first one.

Because a tactic in ***Modernized LCF*** produces only a *context* of independent subgoals, we cannot give a refinement treatment to this rule, and must instead write a family of refinement rules `PairIntro`$\{w\}$ fibred over witnesses of the left conjunct w:

$$(x : A) \times B\ \mathit{true} \Vdash \texttt{PairIntro}\{w\} \Rightarrow \langle w, \beta \rangle \dashv \begin{cases} \alpha : w \in A \\ \beta : [w \,/\, x]\, B\ \mathit{true} \end{cases}$$

This is clearly unsatisfactory, since it breaks the natural flow of proof development, whereby multiple goals may be refined simultaneously without committing in advance to a particular solution. However, a more palatable rule that allows A *true* to be demonstrated by refinement is simply not expressible in ***Modernized LCF***, since the sense of the second subgoal cannot be expressed except by referring to the synthesis of the first subgoal.

3 *Dependent LCF* and generalized refinement rules

At the crux of our problem is the fact that a tactic produces a context of subgoals without any dependencies; if we were to construe the judgment $J \Vdash \tau \Rightarrow E \dashv \Psi$ as synthesizing a *telescope* Ψ rather than a mere context, a proper refinement rule for *PairIntro* would be within reach. In fact, whilst the ML signature for LCF refiners rules out this interpretation, the notation we have used for ***Modernized LCF*** immediately suggests this generalization. Going forward, we will call the theory ***Dependent LCF*** when we take Ψ to be a telescope rather than a context.

Briefly, we can adjust the postsuppositions of the refinement judgment to account for this generalization, by first characterizing the wellformed telescopes:

$$\frac{}{\cdot\ \mathit{tele} \rightsquigarrow \cdot} \qquad \frac{\Psi\ \mathit{tele} \rightsquigarrow \Omega \quad \Omega \rhd \cdot \parallel \cdot \vdash J \in_{\mathbf{wf}} \mathsf{jdg}}{\Psi, \alpha : J\ \mathit{tele} \rightsquigarrow \Omega, \alpha : \mathfrak{v}(J)}$$

Now, $J \Vdash \tau \Rightarrow E \dashv \Psi$ shall postsuppose $\Psi\ \mathit{tele} \rightsquigarrow \Omega$ and $\Omega \rhd \cdot \parallel \cdot \vdash E \in_{\mathbf{wf}} \mathfrak{v}(J)$.

3.1 Dependent Tacticals

All that remains is to give a new definition of the sequencing tactical $\tau_1; \{\tau_\alpha\}_{\alpha \in I}$ which accounts for this dependency. To this end, we will need to implement an auxiliary judgment $\boxed{E \dashv \Phi \Vdash^\star \{\tau_\alpha\}_{\alpha \in I} \Rightarrow E' \dashv \Phi'}$ that allows us to apply the family of tactics $\{\tau_\alpha\}_{\alpha \in I}$ pointwise to a proof state, simultaneously propagating refinements rightward through metavariable substitution. This judgment, which presupposes $I \equiv |\Phi|$, is defined by recursion on the telescope Φ, viewed as a *cons*-list:

$$\frac{}{E \dashv \cdot \Vdash^\star \{\} \Rightarrow E \dashv \cdot}$$

$$\frac{J \Vdash \tau_\alpha \Rightarrow E_\alpha \dashv \Psi_\alpha \quad [E_\alpha / \alpha]E \dashv [E_\alpha / \alpha]\Phi \Vdash^\star \left\{\tau_\beta\right\}_{\beta \in \Phi} \Rightarrow E' \dashv \Phi'}{E \dashv \alpha : J, \Phi \Vdash^\star \left\{\tau_\beta\right\}_{\beta \in \{\alpha\} \cup |\Phi|} \Rightarrow E' \dashv \Psi_\alpha \oplus \Phi'}$$

Now, the sequencing tactical is readily definable, presupposing $I \equiv |\Phi|$:

$$\frac{J \Vdash \tau_1 \Rightarrow E \dashv \Phi \quad E \dashv \Phi \Vdash^\star \{\tau_\alpha\}_{\alpha \in I} \Rightarrow E' \dashv \Phi'}{J \Vdash \tau_1; \{\tau_\alpha\}_{\alpha \in I} \Rightarrow E' \dashv \Phi'}$$

4 Case Study: Constructive Type Theory

The refinement logic for CTT is a sequent calculus with a single form of judgment, $\boxed{H \gg P}$, which means that the proposition P is true (and functionally so) under the assumptions in the telescope H. The synthesis of this judgment is a hypothetical witness for the truth or P (or *extract term*) with free variables bound in H.

The type theory CTT is then approximated in the refinement logic by adding valid refinement rules for $H \gg P$. For the sake of familiarity, we will re-use Nuprl's rule notation such that

$$\begin{array}{l} H \gg P \left\lfloor \textbf{ext}\ E(\alpha_0, \alpha_1, \ldots, \alpha_n) \right\rfloor \\ \quad \textbf{by}\ \texttt{RuleName} \\ \qquad H_0 \gg Q_0 \left\lfloor \textbf{ext}\ \alpha_0 \right\rfloor \\ \qquad H_1(\alpha_0) \gg Q_1(\alpha_0) \left\lfloor \textbf{ext}\ \alpha_1(\alpha_0) \right\rfloor \\ \qquad \vdots \\ \qquad H_n(\alpha_0, \alpha_1, \ldots, \alpha_{n-1}) \gg Q_n(\alpha_0, \alpha_1, \ldots, \alpha_{n-1}) \left\lfloor \textbf{ext}\ \alpha_n(\alpha_0, \ldots, \alpha_{n-1}) \right\rfloor \end{array}$$

shall mean

$$H \gg P \Vdash \texttt{RuleName} \Rightarrow E(\alpha_0, \alpha_1, \ldots, \alpha_n) \dashv \left\{ \begin{array}{l} \alpha_0 : H_0 \gg Q_0 \\ \alpha_1 : H_1 \gg Q_1 \\ \qquad \vdots \\ \alpha_n : H_n \gg Q_n \end{array} \right.$$

Let us first give some of the refinement rules for the dependent pair type:

$$\begin{array}{l} H \gg (x : A) \times B \left\lfloor \textbf{ext}\ \langle \alpha, \beta \rangle \right\rfloor \\ \quad \textbf{by}\ \texttt{PairIntro} \\ \qquad H \gg A \left\lfloor \textbf{ext}\ \alpha \right\rfloor \\ \qquad H \gg [\alpha \,/\, x]\, B \left\lfloor \textbf{ext}\ \beta \right\rfloor \\ \qquad H, x : A \gg B \in \mathbb{U} \left\lfloor \textbf{ext}\ \gamma(x) \right\rfloor \end{array}$$

$$\begin{array}{l} H, z : (x : A) \times B, J \gg C \left\lfloor \textbf{ext}\ \mathsf{spread}(z; u, v.\alpha(u, v)) \right\rfloor \\ \quad \textbf{by}\ \texttt{PairElim}\{z\} \\ \qquad H, z : (x : A) \times B, u : A, v : [u \,/\, x]\, B, [\langle u, v \rangle \,/\, z]\, J \gg C \left\lfloor \textbf{ext}\ \alpha(u, v) \right\rfloor \end{array}$$

We can also give refinement rules for dependent function introduction and elimination; in the same way ***Modernized LCF*** could not express a refinement rule for pair introduction, a refinement rule for function elimination likewise only becomes attainable via the generalization to ***Dependent LCF***.

$H \gg (x : A) \to B \ \lfloor \textbf{ext}\ \lambda(x.\alpha(x)) \rfloor$

by `FunIntro`

$\quad H, x : A \gg B \ \lfloor \textbf{ext}\ \alpha(x) \rfloor$

$\quad H \gg A \in \mathbb{U} \ \left\lfloor \textbf{ext}\ \beta \right\rfloor$

$H, z : (x : A) \to B, J \gg C \ \left\lfloor \textbf{ext}\ \beta(\mathsf{ap}(z; \alpha), \mathsf{Ax}) \right\rfloor$

by `FunElim`$\{z\}$

$\quad H, z : (x : A) \to B, J \gg A \ \lfloor \textbf{ext}\ \alpha \rfloor$

$\quad H, z : (x : A) \to B, y : [\alpha \,/\, x]\, B, p : y = \mathsf{ap}(z; \alpha) \in [\alpha \,/\, x]\, B, J \gg C \ \left\lfloor \textbf{ext}\ \beta(y, p) \right\rfloor$