A Fix for the Proof of Fixed Point Induction in PFPL

Yicheng Ni Yuting Wang

Shanghai Jiao Tong University

1 Introduction

We are building a Coq library for reasoning about the locally nameless representation based on the locally nameless sets proposed by Andrew Pitts. When we are formalizing in Coq the theory about equational reasoning in Practical Foundations for Programming Languages (PFPL), we find the proof of a theorem in the book problematic.

To be more specific, the proof of reflexivity of logical equivalence for PCF (also known as the fundamental theorem) in Chapter 47 depends on a theorem called Fixed Point Induction (Theorem 47.8 in PFPL). The proof of Fixed Point Induction does not encompass all possible cases, making the proof incomplete.

The rest of this note discusses a solution to this problem. It is organized as follows. We first reiterate related definitions and notations in PFPL in Section 2. We state the problem in more detail in Section 3. We then try a fix suggested by the author of PFPL and illustrate where we got stuck in Section 4. Finally, we propose our solution in Section 5.

2 Definitions

We try to keep the definitions and notations the same as those in PFPL.

Definition 1 (Kleene equality)**.** *Two terms e and e' are kleene equal, written as $e \simeq e'$, iff for every $n \geq 0$, $e \longmapsto_* \overline{n}$ iff $e' \longmapsto_* \overline{n}$. where $\longmapsto_*$ is the transitive closure of the small-step reduction.*

By this definition, two terms are kleene equal iff they converge to the same value or they both diverge.

Definition 2 (Logical equivalence)**.** *A logical equivalence $e \sim_\tau e'$ between closed expressions of type τ is defined by induction on τ as follows:*

$$\begin{array}{lll} e \sim_{nat} e' & \textit{iff} & e \simeq e' \\ e \sim_{\tau_1 \rightharpoonup \tau_2} e' & \textit{iff} & e_1 \sim_{\tau_1} e_1' \textit{ implies } e(e_1) \sim_{\tau_2} e'(e_1'). \end{array}$$

Definition 3 (General recursion)**.** *The construct of general recursion (fixed point) in PCF is written as* $\text{fix } x : \tau \text{ is } e$.

The reduction rule of general recursion is defined as follows:

$$\text{fix } x : \tau \text{ is } e \longmapsto [\text{fix } x : \tau \text{ is } e/x]e.$$

Definition 4 (Bounded recursion)**.** *A bounded recursion* $\text{fix}^m\ x : \tau \text{ is } e$ *is defined by induction on* $m \geq 0$ *as follows:*

$$\begin{aligned} \text{fix}^0\ x : \tau \text{ is } e &\triangleq \text{fix } x : \tau \text{ is } x \\ \text{fix}^{m+1}\ x : \tau \text{ is } e &\triangleq [\text{fix}^m\ x : \tau \text{ is } e/x]e. \end{aligned}$$

PFPL employs the following notations: $f^{(\omega)} \triangleq \text{fix } x : \tau \text{ is } e_x$ and $f^{(m)} \triangleq \text{fix}^m\ x : \tau \text{ is } e_x$ for some given e_x in the context.

3 Problem

The problem lies in the proof of Fixed Point Induction, which is stated as follows:

Theorem 1 (Fixed Point Induction)**.** *Suppose that* $x : \tau \vdash e : \tau$.
If $(\forall m \geq 0)\ \text{fix}^m\ x : \tau \text{ is } e \sim_\tau \text{fix}^m\ x : \tau \text{ is } e'$*, then* $\text{fix } x : \tau \text{ is } e \sim_\tau \text{fix } x : \tau \text{ is } e'$.

By Fixed Point Induction, if for any m the m-bounded recursion of two terms are logically equivalent, then the general recursion of these two terms are logically equivalent. To prove this theorem, PFPL introduces applicative contexts and tries to prove a Generalized Fixed Point Induction theorem. A context is a term with a hole to be filled with another term. The notation $A\{\text{fix } x : \tau \text{ is } e\}$ denotes a term generated by filling the context A with term $\text{fix } x : \tau \text{ is } e$. The definition of applicative contexts can be found in the book which we do not repeat here. The important point about applicative contexts for this note is that $A\{\text{fix } x : \tau \text{ is } e\}$ is equal to $[\text{fix } x : \tau \text{ is } e/y](A\{y\})$, where y is a fresh variable.

The Generalized Fixed Point Induction theorem is stated as follows:

Theorem 2 (Generalized Fixed Point Induction)**.**
If $(\forall m \geq 0)\ A\{\text{fix}^m\ x : \tau \text{ is } e\} \sim_{\tau_0} A'\{\text{fix}^m\ x : \tau \text{ is } e'\}$*, then* $A\{\text{fix } x : \tau \text{ is } e\} \sim_{\tau_0} A'\{\text{fix } x : \tau \text{ is } e'\}$.

If Generalized Fixed Point Induction holds, then we can derive the original Fixed Point Induction theorem by instantiating A and A' with the empty context.

The generalized theorem is proved by induction on the structure of type τ_0. However, we got stuck at the base case where τ_0 is nat. According to the definition of $\sim_{nat}$, we have to show $A\{\text{fix } x : \tau \text{ is } e\} \simeq A'\{\text{fix } x : \tau \text{ is } e'\}$. Here, PFPL introduces a lemma (which is Corollary 47.17 in PFPL):

Lemma 1. *Suppose that* $y : \tau \vdash e : nat$*. There exists* $m \geq 0$ *such that* $[f^{(\omega)}/y]e \simeq [f^{(m)}/y]e$.

Lemma 1 is a corollary of the Compactness theorem (Theorem 47.16 in PFPL). In the original proof, Lemma 1 is applied twice in the proof of Generalized Fixed Point Induction: first instantiating e in Lemma 1 with $A\{y\}$ to get some m_1 such that $A\{\text{fix } x : \tau \text{ is } e\} \simeq A\{\text{fix}^{m_1} x : \tau \text{ is } e\}$ and second instantiating e with $A'\{y\}$ to get some m_2 such that $A'\{\text{fix}^{m_2} x : \tau \text{ is } e'\} \simeq A'\{\text{fix } x : \tau \text{ is } e'\}$. The proof can be concluded by the transitivity of $\simeq$ if $A\{\text{fix}^{m_1} x : \tau \text{ is } e\} \simeq A'\{\text{fix}^{m_2} x : \tau \text{ is } e'\}$ holds. This can be derived from the assumption when $m_1 = m_2$. However, it is not clear what we should do if $m_1 \neq m_2$.

4 An Attempt to Fix this Problem

To fix the above proof, one may consider picking the larger number from m_1 and m_2 and pumping the smaller bounded fixed point in the given proposition to match the larger one.[1] For example, the above proof can be finished if we can prove the following proposition:

Proposition 1. *If* $[f^{(\omega)}/y]e \simeq [f^{(m)}/y]e$*, then* $\forall m' \geq m, [f^{(\omega)}/y]e \simeq [f^{(m')}/y]e$.

We tried to prove this proposition by case analysis on whether $[f^{(\omega)}/y]e$ converges. If $[f^{(\omega)}/y]e$ converges, the proposition can be proved by Lemma 47.15 in PFPL. However, we do not know how to prove the proposition if $[f^{(\omega)}/y]e$ diverges. Suppose $m = 0$ and $e = y$, we need to prove $f^{(\omega)} \simeq f^{(m')}$ for all $m' \geq 0$. We have the assumption that $f^{(\omega)} \simeq f^{(0)}$ and both $f^{(0)}$ and $f^{(\omega)}$ diverge. This assumption does not carry enough information for us to finish the proof. That is, to prove the divergence of general recursion implies the divergence of all bounded recursions.

5 Our Solution

The problem of the proof for the Generalized Fixed Point Induction is that Lemma 1 was applied *twice* which generates incompatible assumptions. We propose a solution that applies Lemma 1 *only once* and employs a converse of the compactness theorem to finish the proof. Below we discuss this solution in details. The problem with the original proof is that the compactness theorem is applied twice which generates incompatible bounding numbers for recursion.

Theorem 3 (Compactness)**.** *Suppose that* $y : \tau \vdash e : \tau$ *nat where* $y \notin f^{(\omega)}$.
If $[f^{(\omega)}/y]e \longmapsto_* \overline{n}$*, then there exists some* m *such that* $[f^{(m)}/y]e \longmapsto_* \overline{n}$.

The Compactness theorem states that convergence of general recursion implies convergence of some bounded recursion. To fix the proof for the Generalized Fixed Point Induction, we propose the Converse of Compactness theorem:

Lemma 2 (Converse of Compactness)**.** *Suppose that* $y : \tau \vdash e : \tau$ *nat where* $y \notin f^{(\omega)}$.
If $[f^{(m)}/y]e \longmapsto_* \overline{n}$ *for some* m*, then* $[f^{(\omega)}/y]e \longmapsto_* \overline{n}$.

[1] Based on a discussion with the author of PFPL through email.

It states that the convergence of bounded recursion implies the convergence of general recursion. This theorem is the contrapositive proposition of the goal we got stuck in Section 4. In our proof for the Generalized Fixed Point Induction, Compactness and the Converse of Compactness theorem are both applied once. Lemma 1, which is applied twice in the original proof of Generalized Fixed Point Induction in PFPL, is unnecessary in our proof since we directly perform case analysis on the divergence of terms. In the rest of this section, we first repair the Generalized Fixed Point Induction in 5.1 and then prove Converse of Compactness theorem in 5.2.

5.1 Repair of the Generalized Fixed Point Induction

Recall that the Generalized Fixed Point Induction is proved by induction on the type τ. In the base case where $\tau = \mathit{nat}$, the goal is to show that $A\{\text{fix } x : \tau \text{ is } e\} \simeq A'\{\text{fix } x : \tau \text{ is } e'\}$.

We analyze whether $A\{\text{fix } x : \tau \text{ is } e\}$ and $A'\{\text{fix } x : \tau \text{ is } e'\}$ diverge. When both $A\{\text{fix } x : \tau \text{ is } e\}$ and $A'\{\text{fix } x : \tau \text{ is } e'\}$ diverge, the proof is trivial. Otherwise, either $A\{\text{fix } x : \tau \text{ is } e\}$ converges or $A'\{\text{fix } x : \tau \text{ is } e'\}$ converges. Suppose $A\{\text{fix } x : \tau \text{ is } e\} \longmapsto_* \overline{n}$. By Compactness Theorem, $A\{\text{fix}^m\ x : \tau \text{ is } e\} \longmapsto_* \overline{n}$ for some m. By the assumption of the Generalized Fixed Point Induction and the definition of $\simeq$, we have $A'\{\text{fix}^m\ x : \tau \text{ is } e'\} \longmapsto_* \overline{n}$. By Converse of Compactness, we can get $A'\{\text{fix } x : \tau \text{ is } e'\} \longmapsto_* \overline{n}$. Since both $A\{\text{fix } x : \tau \text{ is } e\} \longmapsto_* \overline{n}$ and $A'\{\text{fix } x : \tau \text{ is } e'\} \longmapsto_* \overline{n}$ holds, we conclude the definition of $\simeq$. For the remaining case that $A'\{\text{fix } x : \tau \text{ is } e'\}$ converges, the proof is symmetric.

5.2 Proof of Converse of Compactness

Similar to the proof of Compactness Theorem in PFPL, we prove Converse of Compactness by proving its generalized version described by using the stack machine for PCF:

Lemma 3 (Generalized Converse of Compactness)**.**
If $[f^{(m)}/y]k \triangleright [f^{(m)}/y]e \longmapsto_* \epsilon \triangleleft \overline{n}$*, then* $[f^{(\omega)}/y]k \triangleright [f^{(\omega)}/y]e \longmapsto_* \epsilon \triangleleft \overline{n}$.

The definitions of the stack machine are given in Chapter 28 of PFPL. If Generalized Converse of Compactness holds, we can the derive Converse of Compactness by first instantiating k with the empty stack and then applying the soundness and completeness theorems (in Chapter 28 of PFPL) which state that the reduction relation of terms is equivalent to the transition relation in stack machines.

The proof of Generalized Converse of Compactness follows the proof of Lemma 47.15 in PFPL. First by induction on m, and then induction on $\longmapsto_*$.

In the case that $m = 0$, the proof is similar to that of Lemma 47.15. The reflexivity case of $\longmapsto_*$ is trivial ($\triangleright \neq \triangleleft$). In the head case where $[f^{(0)}/y]k \triangleright [f^{(0)}/y]e \longmapsto s'$ and $s' \longmapsto_* \epsilon \triangleleft \overline{n}$ holds, we do case analysis on the transition $[f^{(0)}/y]k \triangleright [f^{(0)}/y]e \longmapsto s'$. The cases can be divided into two categories according to the state of s' (evaluation state, represented as $\triangleright$, or return state, represented as $\triangleleft$). When s' is a evaluation state, the goal is proved by the induction hypothesis for $\longmapsto_*$. When s' is a return state, the induction hypothesis for $\longmapsto_*$ cannot be directly applied since it is only applicable for

$\longmapsto_*$ with evaluation state on the left side. However, the goal can be proved by further induction on the stack k to get $s' \longmapsto s''$. If s'' is still a return state, we apply induction hypothesis for the stack k; if s'' turns out to be a evaluation state, we apply induction hypothesis for $\longmapsto_*$.

In the case where $m = S\ m'$, we have the assumption that $[([f^{(m')}/x]e_x)/y]k \triangleright [([f^{(m')}/x]e_x)/y]e \longmapsto_* \epsilon \triangleleft \overline{n}$. Syntactically, $[([f^{(m')}/x]e_x)/y]k = [f^{(m')}/x][e_x/y]k$ and $[([f^{(m')}/x]e_x)/y]e = [f^{(m')}/x][e_x/y]e$. By induction hypothesis, we can get $[([f^{(\omega)}/x]e_x)/y]k \triangleright [([f^{(\omega)}/x]e_x)/y]e \longmapsto_* \epsilon \triangleleft \overline{n}$, and we need to show $[f^{(\omega)}/y]k \triangleright [f^{(\omega)}/y]e \longmapsto_* \epsilon \triangleleft \overline{n}$. It is proved by applying the following lemma with $m = f^{(\omega)}$, $m' = [f^{(\omega)}/x]e_x$ and $f^{(\omega)} \longmapsto [f^{(\omega)}/x]e_x$.

Lemma 4 (Converse of Compactness for Reduction I)**.** *Suppose that* $m \longmapsto m'$. *If* $[m'/y]k \triangleright [m'/y]e \longmapsto_* \epsilon \triangleleft \overline{n}$, *then* $[m/y]k \triangleright [m/y]e \longmapsto_* \epsilon \triangleleft \overline{n}$.

Lemma 4 cannot be proved alone. Instead, it is proved together with Lemma 5:

Lemma 5 (Converse of Compactness for Reduction II)**.** *Suppose that* $t \longmapsto t'$ *and* $m \longmapsto m'$. *If* $[m'/y]k \triangleright t' \longmapsto_* \epsilon \triangleleft \overline{n}$, *then* $[m/y]k \triangleright t \longmapsto_* \epsilon \triangleleft \overline{n}$.

Lemma 4 and Lemma 5 imply that convergence of reduced terms implies convergence of the original terms. These two lemmas are proved by mutual induction on $\longmapsto_*$. The intuition behind mutual induction is that after finite steps of transition, the configuration in Lemma 4 matches the configuration in Lemma 5 and vice versa.

In both lemmas, the reflexivity case is trivial ($\triangleright \neq \triangleleft$).

For the head case of Lemma 4, we have $[m'/y]k \triangleright [m'/y]e \longmapsto s'$ and $s' \longmapsto_* \epsilon \triangleleft \overline{n}$. Similar to the proof of Lemma 3, we do case analysis on the transition $[m'/y]k \triangleright [m'/y]e \longmapsto s'$. Each case can be divided into two subcases according to whether $e = y$. In the cases that $e = y$, the goal can be proved by applying Lemma 5. Notice that if we give explicit steps for $\longmapsto_*$ and assume $[m'/y]k \triangleright t' \longmapsto_* \epsilon \triangleleft \overline{n}$ needs $i' + 1$ steps, the steps for the instance of Lemma 4 is also $i' + 1$. In the cases that $e \neq y$, the proof is similar to Lemma 3.

For the head case of Lemma 5, we have $[m'/y]k \triangleright t' \longmapsto s'$ and $s' \longmapsto_* \epsilon \triangleleft \overline{n}$. Similar to the proof of Lemma 3, we do case analysis on the transition $[m'/y]k \triangleright t' \longmapsto s'$. In each case, inversion the $t \longmapsto t'$ relation. The subcases can be divided into two categories according to whether the inversion produces sub-relation of $t \longmapsto t'$. For example, if $t = S(e)$ and $t' = S(e')$ then the inversion produces the sub-relation $e \longmapsto e'$; if $t = ifz(z; z; e)$ and $t' = z$ then the inversion does not produce sub-relation. In the subcases where the inversion produces sub-relation, the goal can be proved by the induction hypothesis for $\longmapsto_*$ for the sub-relation $\longmapsto$. In the subcases where the inversion does not produce sub-relation, the goal can be proved by Lemma 4; notice that if we give explicit steps for $\longmapsto_*$ and assume $[m'/y]k \triangleright t' \longmapsto_* \epsilon \triangleleft \overline{n}$ needs $i' + 1$ steps, the steps for the instance of Lemma 4 is i'.

Combining the proof of Lemma 4 and Lemma 5, the $\longmapsto_*$ relation is decreasing. Therefore, the mutual induction is valid.

6 Coq Formalization of Equality for PCF

We have formalized the theorems in Chapter 47 of PFPL for proving that contextual equivalence and logical equivalence imply each other in Coq. The proof scripts are located at `https://zenodo.org/records/13918934`. This formalization makes use of a Coq library we developed for reasoning about the locally nameless representation. The library extends the theory of Locally Nameless Sets introduced by Andrew Pitts to generically and automatically prove locally nameless lemmas. See the upcoming paper "Generic Reasoning of the Locally Nameless Representation" in APLAS 2024 for details.