Interdefinability of Parallel Operations in PCF [1]

Allen Stoughton

Computer Science and Artificial Intelligence
School of Cognitive and Computing Sciences
University of Sussex
Falmer, Brighton BN1 9QH, England

Abstract. It is shown that the "parallel or" and "parallel conditional" operations are interdefinable elements of the continuous function model of the programming language PCF.

In his seminal paper [3], which the reader is assumed to be familiar with, Plotkin showed that the continuous function model of the programming language PCF contains certain "parallel" elements that are not definable by terms. The most famous of these elements is the "parallel or" operation, $por\colon o \to o \to o$, which is defined by

$$por \perp tt = tt, \qquad por\, tt \perp = tt, \qquad por\, ff\, ff = ff.$$

But two "parallel conditionals" are also of interest, $pif_o\colon o \to o \to o \to o$ (over the booleans) and $pif_\iota\colon o \to \iota \to \iota \to \iota$ (over the natural numbers), which are defined by

$$pif_\kappa \perp x\, x = x, \qquad pif_\kappa\, tt\, x \perp = x, \qquad pif_\kappa\, ff \perp x = x,$$

for $\kappa = o, \iota$. Note that por and pif_o are finite (isolated), whereas pif_ι is infinite.

Plotkin showed that if constants PIf_o and PIf_ι denoting pif_o and pif_ι are added to PCF, then all finite elements of the continuous function model are denotable. Actually, only PIf_o is needed to define $POr = \lambda xy.\, PIf_o\, x\, tt\, y$. Furthermore, given PIf_ι, we can define

$$PIf_o = \lambda xyz.\, Eq\, 1\, (PIf_\iota\, x\, (\supset_\iota y\, 1\, 0)\, (\supset_\iota z\, 1\, 0)),$$

where $Eq\colon \iota \to \iota \to o$ is the easily definable equality test over the natural numbers (strict in both arguments), and we have written 0 and 1 instead of Plotkin's numerals k_0 and k_1.

Abramsky [1] and Curien [2] independently sharpened Plotkin's definability theorem by showing that simply adding a constant denoting por to PCF is enough to make all finite elements denotable. In particular, their theorem shows that pif_o can be defined from por. But this leaves open the question of whether pif_ι can be defined from por. The answer is "yes", as the following proposition shows, with the consequence that an element of the continuous function model is definable in PCF plus a constant denoting por iff it is definable in PCF plus a constant denoting pif_ι.

[1] Appears in *Theoretical Computer Science*, 79:357–358, 1991.

Proposition. The operations por, pif_o and pif_ι are interdefinable elements of the continuous function model of PCF.

Proof. We have already shown that por is definable from pif_o, which in turn is definable from pif_ι. It thus remains to show that pif_ι is definable from por. Suppose that POr is a constant denoting por. Define

$$PIf_\iota = Y_\sigma\, F\, 0,$$

where $\sigma = \iota \to o \to \iota \to \iota \to \iota$ and $F\colon \sigma \to \sigma$ is defined by

$$\begin{aligned} F = \lambda fnxyz.\, \supset_\iota\, &(POr\,(PAnd\,(Eq\,y\,n)\,(Eq\,z\,n))\\ &\qquad\quad (PAnd\,x\,(Eq\,y\,n))\\ &\qquad\quad (PAnd\,(Not\,x)\,(Eq\,z\,n)))\\ &n\\ &(f\,(+1\,n)\,x\,y\,z). \end{aligned}$$

Here, $Not\colon o \to o$ is $\lambda x.\, \supset_o x\, f\!f\ tt$, we have extended POr to three arguments in the obvious way, and $PAnd\colon o \to o \to o$ is the "parallel and" operation, dual to POr:

$$PAnd = \lambda xy.\, Not\,(POr\,(Not\,x)\,(Not\,y)).$$

The reader will have no trouble verifying that PIf_ι does in fact denote pif_ι. $\Box$

The proof of this proposition makes use of ideas from the proofs by Abramsky and Curien of their definability theorem. Combining our proposition with Plotkin's definability theorem, we have an alternative proof of Abramsky and Curien's theorem.

References

[1] S. Abramsky, Unpublished notes, 1984.

[2] P.-L. Curien, *Categorical combinators, sequential algorithms and functional programming,* Research Notes in Theoretical Computer Science (Pitman, London, 1986).

[3] G. Plotkin, LCF considered as a programming language, *Theoretical Computer Science* **5** (1977) 223–255.