Linear types can change the world!

Philip Wadler
University of Glasgow[*]

Abstract

The linear logic of J.-Y. Girard suggests a new type system for functional languages, one which supports operations that "change the world". Values belonging to a linear type must be used exactly once: like the world, they cannot be duplicated or destroyed. Such values require no reference counting or garbage collection, and safely admit destructive array update. Linear types extend Schmidt's notion of single threading; provide an alternative to Hudak and Bloss' update analysis; and offer a practical complement to Lafont and Holmström's elegant linear languages.

An old canard against functional languages is that they cannot change the world: they do not "naturally" cope with changes of state, such as altering a location in memory, changing a pixel on a display, or sensing when a key is pressed.

As a prototypical example of this, consider the world as an array. An array (of type *Arr*) is a mapping from indices (of type *Ix*) to values (of type *Val*). For instance, the world might be a mapping of variable names to values, or file names to contents. At any time, we can do one of two things to the world: find the value associated with an index, or update an index to be associated with a new value.

Of course it is possible to model this functionally; we just use the two operations

$$\begin{aligned} \mathit{lookup} &: \mathit{Ix} \to \mathit{Arr} \to \mathit{Val}, \\ \mathit{update} &: \mathit{Ix} \to \mathit{Val} \to \mathit{Arr} \to \mathit{Arr}. \end{aligned}$$

A program that interacts with the world might have the form

$$\mathit{main} : \mathit{Args} \to \mathit{Arr} \to \mathit{Arr},$$

where the first parameter is the list of arguments that make up the command line, the second parameter is the old world, and the result is the new world. An example of a program is

$$\mathit{main}\ \mathit{files}\ a = \mathit{update}\ \text{"stdout"}\ (\mathit{concat}\ [\mathit{lookup}\ i\ a \mid i \leftarrow \mathit{files}])\ a.$$

[*] Author's address: Department of Computing Science, University of Glasgow, G12 8QQ, Scotland. Electronic mail: wadler@cs.glasgow.ac.uk.

Presented at *IFIP TC 2 Working Conference on Programming Concepts and Methods*, Sea of Gallilee, Israel, April 1990; published in M. Broy and C. B. Jones, editors, *Programming Concepts and Methods*, North Holland, 1990.