

Concepts of Programming Language Design

Linear Type Systems

Tom Smeding, Gabriele Keller

January 22, 2025

1. **Linear Types** Linear type systems can be used to model stateful computations in a declarative way. An alternative way, implemented in Haskell, is to do so by using a monad.
 - (a) What could a linear type equivalent of the (in this example monomorphic) `IORef` interface below look like?

```
newIORef    :: Int -> IO (IORef Int)
readIORef   :: IORef Int -> IO Int
writeIORef  :: IORef Int -> Int -> IO ()
```
 - (b) In your opinion, which one is more convenient from the user perspective? Why?
 - (c) What is the type of $\lambda(x : \text{Int}) \rightarrow \text{get } x \text{ arr}$ under the environment $\Gamma = \{\text{get} : \text{Int} \rightarrow \text{!Array} \rightarrow \text{!Pair Float !Array}, \text{arr} : \text{!Array}\}$.
2. **Linear Types and Type Constructors** Linear data types can contain linear and non-linear types, but non-linear types can only contain non-linear types.
 - (a) Explain the statement above. Why would it be a problem if a non-linear data type contains items of linear type, and why is not a problem if a linear data type contains non-linear fields?
 - (b) How does that affect functions which accept linear values as arguments? For example, what would be the type of a function which accepts to linear lists of identical length and returns a list of the pair-wise sum (i.e., a linear version of `zipWith (+)`)?