

1 CPS review

Starting with the homework from last time, it is important to remember that continuations never get duplicated in same branch of a conditional. Continuations are not instantiated until the top of a computation has been reached. Instead, if we are trying to apply a certain continuation, say k , we would only instantiate k when we had exhausted all computation and are left only with the last computation in a chain of execution.

For example, say we had the following expression: $e_1e_2e_3$. Which e_i will be evaluated first? e_3 will be evaluated first, followed by e_2 and finally e_1 . When rewritten into continuation passing style, the evaluation order becomes more evident:

e_3 (fun a \rightarrow e_2 (fun b \rightarrow e_1 (fun c \rightarrow c b a k))).

2 Project notes

All projects should be written (programmed) in a functional manner: Imperative style programming should be avoided. Side effects should be avoided as well. This means references and mutable values cannot be used. For algorithms that require the use of looping constructs, recursion should be used instead. Fixed size or mutable arrays should not be used either, instead lists can be used or arrays can be modeled through the use of functions. For instance, a function can be declared where the input is an index and the output is the value of an "array" at that index. For additional examples, see how environments were encoded in the paper *A Gentle Introduction to Multistage Programming*.

3 Extending languages to do MSP

How would we go about extending other languages like C or Java with Multi-stage Programming? Our first option could be to explore the connection between macros and MSP. This topic will be covered later on in the course, so we will keep on looking for viable alternatives. Templates in C++ are an attractive alternative. Yet upon inspection we find that templates in C++ only afford us the use of staging, but not true multi-stage programming. Using templates makes it difficult to verify both the generated code and the generator. Finally, it is possible to get ill-typed code using C++ templates. Once this is the case, it is very difficult to debug the source of such problems. As an aside, the use of Makefiles make the analysis and verification of C code very difficult as well. It is important to remember that staging is the preemption of certain inputs and declaring stages is dependent upon the arrival of such inputs.

There have been certain attempts to add multi-stage programming constructs to various languages. For instance, 'C, Tempo, and Popcorn are multi-stage extensions of C, MetaAspectJ is a language that uses aspects in Java, and Metaphor is a multi-stage extension to C#. It is important to remember that when extending any language, the type system must be changed in order to support staging.