

## Re: Grammatical Recursion

**Source:** <http://sci.tech-archive.net/Archive/sci.lang/2004-12/3569.html>

---

**From:** Tak To (*takto\_at\_alum.mit.edu.-*)

**Date:** 12/20/04

Date: Mon, 20 Dec 2004 09:27:49 -0500

Tak To <takto@alum.mit.edu.-> wrote:

> *One can only talk about the difference between recursion and  
> iteration in the context of specific computation models.*

Greg Lee wrote:

> *Yes, because that's what they are -- computation models.*

That was not what I meant. Sorry, my bad. I should have said "formalism" in instead of "model" to avoid the confusion. Recursion and iteration are different types (or idioms) of algorithms in a formalism. For the term "recursion" to be meaningful, the formalism must have some notion of "function", "invocation", etc; and for "iteration" to be meaningful, the formalism must have "loop", "repeat", etc. The formalism defined by a high level programming language such as "C" would have both; so does Kleene Recursive Functions with its mu operator. However, formalisms such as the Turing Machine, Lambda Calculus, or machine instruction sets without stack operations do not have either (or only one of them); and comparing iteration and recursion in the context of these formalisms would be rather meaningless. In particular, in the Post (after Emil Post) Production System (which forms the basis of mathematical language processing as well as the programming language SNOBOL), there is really no notion of iteration, and only a weak notion of recursion -- in the sense that a grammar can be *\_recursive\_* (meaning that a production of a symbol can reference the symbol itself). Thus, IMHO, comparing iteration and recursion in this context is rather meaningless.

Tak

--

-----+-----  
Tak To

takto@alum.mit.eduxx

-----^^  
[taode takto ~{LU5B~}]

NB: trim the xx to get my real email addr