REDELMEIER84
Redelmeier, D. Hugh;
Toward Practical Functional Programming;
Technical Report CSRG-158 (May 1984).
This paper is an attempt at illustrating functional programming flavour combined with an introduction to the K language (designed by Redelmeier). It represents current trends in Algol-like programming language design, and discussion of Dijkstra's mini-language (K's ancestor). Hoare's CSP, and the VAL language is included.

The emphasis is placed on dealing with programming without variables, as iteration is replaced by recursion, just as in LISP. The language has been beefed up with a data abstraction mechanism that uses bindings in a Cartesian approach, exception values with their own operators, and "interactive" functional i/o operators.

A major portion of the woek deals with the formal semantics of the language the simplicity of which allows the author to claim that his language is both simple and practical. A good deal of discussion, infact, most of the report is dedicated to examining the alternatives to the approaches or features incorporated in the K language.

Sugestions are made, relating to how functional programming languages can be made more efficient, through compiler improvements which avoid generating code that wastes CPU time by unnecessarily copying values. A collection of remedies called "binding col-allocations" are discussed.

Unfortunately the only reason given for functional programming is the avoidance of "data-flow anomalier" associated with conventional languages and the inability of compilers to statically detect such errors. Extension of this dogma, seems to have influenced the desire of simplicity in the verifiability or correctness proofs of programs in this language. This has "straight jacketted" an otherwise interesting and thought provoking language.