Optimising Compilers

Explain what is meant by an effect system for a typed language. Distinguish between immediate effects and possible other effects; also give a typical form of sequent \( \Gamma \vdash e : \langle\text{whatever}\rangle \). [4 marks]

Given the following subset of ML,

\[
e ::= x \mid \lambda x.e \mid e e' \mid \text{let } x = e \text{ in } e' \mid \text{if } e \text{ then } e' \text{ else } e'' \mid \text{ref } e \mid !e \mid e := e',
\]

design an effect system for terms \( e \), for which the (immediate) effects of an expression are any subset of \( \{C,R,W\} \) representing reference creation, dereferencing and assignment to some reference cell. You may assume that the ML-like types \( t \) of the language involve integers, functions and reference types but have no polymorphism. Assume also that assignment returns the value assigned. It suffices to give clauses for \( x, \lambda x.e, e e', \text{if } e \text{ then } e' \text{ else } e'' \) and \( e := e' \). [6 marks]

Explain how your system copes with terms like

\[
\lambda x.\lambda y.\text{if } x \text{ then } y := 1 \text{ else } 0
\]

and

\[
\lambda x.\lambda y.\text{if } x \text{ then } \lambda z.y := z + 1 \text{ else } \lambda z.0.
\]

(If your system cannot handle these cases then instead explain how one might adjust it to do so.) [4 marks]

Explain how the analysis might be used to determine when the optimisation of \( e + e \) to \( \text{let } x = e \text{ in } x + x \) is safe. [3 marks]

Similarly, suggest a criterion on the type or effect of \( f \) in \( \text{let } f = \lambda x.e \text{ in } f(1) + f(2) \) which would enable the two calls to \( f \) to be evaluated concurrently. [3 marks]