2008 Paper 9 Question 16

Optimising Compilers

Consider the language

$$e ::= x \mid \lambda x.e \mid e_1 e_2 \mid e_1; e_2 \mid \xi?x.e \mid \xi!e_1.e_2 \mid \texttt{if } e_1 \texttt{ then } e_2 \texttt{ else } e_3.$$

in which ξ represents a communication channel (from a fixed set), and the forms $e_1; e_2$, $\xi?x.e$ and $\xi!e_1.e_2$ respectively represent sequencing, reading from a channel (binding x) and writing to a channel.

(a) Construct an *effect system* for the above language where effects, F, are represented as *sets* of actions of the form $\xi?$ or $\xi!$ representing side-effects of input from or output to ξ. Explain the two principal occurrences of effects in the judgement form of your system. [8 marks]

(b) Assess the *safety* of your analysis making clear any respects in which execution behaviour may fail to match your analysis. [2 marks]

(c) Let us say a general program analysis framework is *any-path*-like (as opposed to *all-path*-like) if the analyses of $\texttt{if } e_1 \texttt{ then } e_2 \texttt{ else } e_3$ and $e_1; e_2; e_3$ coincide. Is your effect system any-path-like? Justify your answer. [2 marks]

(d) Augment the above language with constructs

$$e ::= \texttt{letchan } \xi \texttt{ in } e \mid \texttt{parsum}(e_1, e_2)$$

which allow a *local channel* to be created, and also inter-thread communication (e_1 and e_2 are evaluated in parallel and their sum returned when both have completed). Extend your effect system to the augmented language, noting that reads and writes to local channels are *not* to be reflected in the overall effect of a `letchan`. [6 marks]

(e) Suggest an alternative data structure for F that might enable effects of the form "after getting two inputs from channel ξ_1 *or* getting one input from channel ξ_2 then an output is written to channel ζ" to be represented. [A modified effect system is not required.] [2 marks]