COMPUTER SCIENCE TRIPOS Part II – 2025 – Paper 9

9 Optimising Compilers (tmj32)

Consider the following abstract syntax for a language $\mathcal{L}$ whose types are integers and functions:

$$e ::= x \mid \lambda x.e \mid e_1 e_2 \mid \mathcal{G}(x) \mid \mathcal{G}(x) := e \mid \texttt{if } e_1 \texttt{ then } e_2 \texttt{ else } e_3 \mid \texttt{let } x = e_1 \texttt{ in } e_2$$

where x ranges over variable names, $\mathcal{G}(x)$ reads from global variable x and $\mathcal{G}(x) := e$ evaluates e, writes its result to global variable x and itself evaluates to the value of e.

(a) Provide inference rules for a type-and-effect system for $\mathcal{L}$, where effects are a subset of $\{R_x, W_x \mid x \text{ is a global variable}\}$. [7 marks]

(b) Show how the rules from part (a) assign a type and effect(s) to the following expressions:

(i) $\mathcal{G}(y) := \mathcal{G}(x)$ [1 mark]

(ii) $\texttt{let } f = \lambda x.\mathcal{G}(y) := x \texttt{ in } f\ \mathcal{G}(x)$ [3 marks]

(iii) $\texttt{if } \mathcal{G}(x) \texttt{ then } \lambda x.\mathcal{G}(y) := x \texttt{ else } \lambda x.x$ [3 marks]

(c) Each global variable has its own lock that needs to be taken before reading or writing to it, which is achieved in $\mathcal{L}$ with a new construct:

$$e ::= \texttt{synchronised } e$$

that provides mutual exclusion by taking the locks required for the evaluation of e before e is executed and unlocking them afterwards. The type-and-effect system can be used to help identify which locks should be taken at each `synchronised` expression. Extend your type-and-effect system with an inference rule for this new construct that can help with this analysis and explain this rule. [*Note:* you do not need to provide any inference rules for the locking and unlocking operations themselves.] [4 marks]

(d) Discuss the relative merits of using effect sets compared to effect sequences when generating code to take and release locks for the construct in part (c). [2 marks]