Implementazione di una macchina astratta

Per implementare una macchina astratta M, è necessario avere a disposizione una macchina ospite M_O, oppure realizzare un circuito in grado di eseguire direttamente L_M.

Gli approcci possibili sono:

- interprete puro scritto in L_{M_O}. Poco efficiente, paga a tempo di esecuzione l'overhead di decodifica e analisi sintattica;
- compilatore puro che traduce da L_M a L_{M_O} (M non viene realizzata). Solitamente il codice generato è di grandi dimensioni perché le primitive di M_O non sono pensate per eseguire L_M;
- realizzazione di una macchina intermedia M_I con relativo interprete, e compilazione da L_M a L_{M_I}. Casi estremi: se $M_I = M$ allora abbiamo un interprete puro, se $M_I = M_O$ un compilatore puro.

Macchina intermedia e RTS

Il vantaggio dell'approccio misto è che possiamo definire M_I come un'estensione di M_O, in modo che l'interpretazione di L_{M_I} sia efficiente.

La collezione di strutture dati e sottoprogrammi che vengono aggiunti a M_O prende il nome di *supporto a tempo di esecuzione*: $M_I = M_O$+RTS. Le primitive di M_I che non sono in M_O diventano quindi chiamate a routine inserite nel RTS. Un esempio comune sono le procedure di I/O.

Le macchine intermedie facilitano anche la portabilità su più macchine ospiti, visto che si deve reimplementare solo M_I.