Relazione lavoro svolto

Pietro Ghiglio

September 15, 2020

Table of Contents

Table of Contents

Scopo

- Propagare a livello di codice sorgente informazioni contenute tramite profiling.
- Associare ad ogni riga di codice il numero di istruzioni eseguite durante la run del programma, associate alla linea di codice.
- Associare ad ogni chiamata a funzione, il numero di istruzioni eseguite a partire dalla chiamata.

Implementazione

- Stack di callsites, push prima di ogni chiamata, pop dopo la chiamata.
- Ad ogni esecuzione di un basic block: stampa id basic block + dump dello stack.

Propagazione a livello di codice sorgente

Dato un output dell'instrumentazione corrispondente all'esecuzione di un basic block: bbId callsite1 callsite2 ... callsiteN

- Per ogni istruzione llvm contenuta nel basic block corrispondente all'id, assegnare il costo dell'esecuzione dell'istruzione llvm alla location di codice sorgente corrispondente.
- Per ogni callsite nel dump dello stack, assegnare il costo dell'istruzione llvm al callsite, a meno di funzioni ricorsive.

Call Graph

- Grafo in cui ogni vertice è una funzione.
- Un edge (v,u) rappresenta il fatto che la funzione v chiama u.
- Label sugli edge con callsite.
- Una call (v,u) è ricorsiva se u == v o se da u è possibile richiamare v. ((v,u) è parte di un ciclo).

Esempio

Table of Contents

Metriche utilizzate

Al momento le metriche utilizzabili sono il numero di istruzioni LLVM o il numero di istruzioni Assembly.
Ad entrambe potrebbe essere associato un costo energetico o diretto (istruzioni LLVM) o dato dalla somma del costo delle istruzioni assembly corrispondenti.
Richiede un energy model della target architecture, con le varie considerazioni sulla fattibilità in base alla complessità dell'architettura.

Table of Contents

Mapping

- Mapping source $\rightarrow$ LLVM direttamente dalle debug information delle API LLVM.
  Alcune istruzioni (es. malloca all'inizio della funzione) non hanno debug info.
- Mapping LLVM $\rightarrow$ Assembly ottenuto tramite un pass che sostituisce le informazioni di debug riguardo alla linea di codice sorgente con un id dell'istruzione stessa.
  Molti metodi delle API LLVM per la modifica delle informazioni di debug sono privati.
  Le informazioni di debug sostituite vengono recuperate effettuando disassembly dell'eseguibile.