Improving Trace-Based JIT Optimisation using Whole-Program Information

Maarten Vandercammen, Coen De Roover
Software Languages Lab, Vrije Universiteit Brussel, Belgium

Trace-based JIT Compilation

Program
1  function f(a) {
2      var b = a * 0 ? 1 : 2;
3      return a + b;
4  }
5
6  var n = 100;
7  var result = 0;
8  var a = user_input();
9  function loop() {
10      while (n > 1) {
11          result += f(a);
12          n -= 1;
13      }
14  }
15
16  loop();
17  a = 10;
18  loop();

Execution Trace
loop:
  LOAD n
  LOAD CONST 1
  CMP_GREATER
  GUARD_TRUE
  LOAD a
  LOAD_CONST 0
  CMP_GREATER
  GUARD_TRUE
  LOAD a
  LOAD_CONST 1
  ADD_INT
  LOAD result
  ADD_INT
  LOAD n
  DEC
  JUMP loop

AOT Compilation

C++

Imprecise static information

Hybrid:

Refine static information with observed information

Time
Run-time + Compile-time analysis

Scope
Local + whole-program analysis

My Approach

Extended trace optimisations

Refinement

Compile time
Initial Analysis

Run time
1st run-time analysis

2nd run-time analysis

Initial + Run-Time analysis 1

Initial + Run-Time analysis 1 & 2

Scope

No Analysis

Initial Analysis

Analysis

Constant Variables Found

Initial

(a)

RT 1

(a, b)

RT 2

(a, b, c)

Future Work

1) Analysis Launch Point?
2) Extent of Scope?