Improving Trace-Based JIT Optimisation Using Whole-Program Information
Vandercammen, Maarten; De Roover, Coen

Publication date: 2016

Citation for published version (APA):
Improving Trace-Based JIT Optimisation using Whole-Program Information

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

Program

```python
function f(a) {
    var b = a > 0 ? 1 : 2;
    return a + b;
}
```

Execution Trace

```c
loop:
    LOAD n
    LOAD_CONST 1
    CMP_GREATER
    GUARD_TRUE
    LOAD a
    LOAD_CONST 0
    CMP_GREATER
    GUARD_TRUE
    LOAD loop
    LOAD result
    ADD_INT
    LOAD n
    DEC
    JUMP loop
```

Trace-based JIT Compilation

- Precise: dynamic information
- Run-time analysis
- Local analysis
- Trace optimisations

AOT Compilation

- Imprecise: static information
- Compile-time analysis
- Whole-program optimisation

My Approach

- Hybrid: Refine static information with observed information
- Time: Run-time + Compile-time analysis
- Scope: Local + whole-program analysis
- Extended trace optimisations

Example

Initial Analysis

```c
loop:
    LOAD n
    LOAD_CONST 1
    CMP_GREATER
    GUARD_TRUE
    LOAD a
    LOAD_CONST 0
    CMP_GREATER
    GUARD_TRUE
    LOAD loop
    LOAD result
    ADD_INT
    LOAD n
    DEC
    JUMP loop
```

1st run-time analysis

```c
loop:
    LOAD n
    LOAD_CONST 1
    CMP_GREATER
    GUARD_TRUE
    LOAD a
    LOAD_CONST 0
    CMP_GREATER
    GUARD_TRUE
    LOAD loop
    LOAD result
    ADD_INT
    LOAD n
    DEC
    JUMP loop
```

2nd run-time analysis

```c
loop:
    LOAD n
    LOAD_CONST 1
    CMP_GREATER
    GUARD_TRUE
    LOAD a
    LOAD_CONST 0
    CMP_GREATER
    GUARD_TRUE
    LOAD loop
    LOAD result
    ADD_INT
    LOAD n
    DEC
    JUMP loop
```

Refinement

- Compile time
  - Initial Analysis
  - 1st run-time analysis
  - 2nd run-time analysis

Run time

- Compute heavyweight initial analysis
- Refine initial analysis with run-time information
- Further refine run-time analysis with extra run-time information

Initial Analysis

```c
loop:
    LOAD n
    LOAD_CONST 1
    CMP_GREATER
    GUARD_TRUE
    LOAD a
    LOAD_CONST 0
    CMP_GREATER
    GUARD_TRUE
    LOAD loop
    LOAD result
    ADD_INT
    LOAD n
    DEC
    JUMP loop
```

Initial Analysis

```c
loop:
    LOAD n
    LOAD_CONST 1
    CMP_GREATER
    GUARD_TRUE
    LOAD a
    LOAD_CONST 0
    CMP_GREATER
    GUARD_TRUE
    LOAD loop
    LOAD result
    ADD_INT
    LOAD n
    DEC
    JUMP loop
```

Future Work

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