DRLVM Jitrino

DRLVM Jitrino

http://harmony.apache.org/subcomponents/drlvm/JIT.html

1.

1.1.

1.2.

1.3.

1.4.

2.

2.1.

2.2.

3. Jitrino.OPT

3.1.

3.1.1.

3.1.2.

3.1.3.

3.2.

3.2.1.

3.2.2.

3.2.3.

3.2.4.

4. Jitrino.JET

4.1.

4.1.1.

4.2.

4.2.1.

5.

5.1.

5.2.

5.3.

5.3.1.

5.4.

5.4.1.

5.4.2.

5.4.3.

5.4.4.

6.

6.1. JIT_VM

6.1. JIT_EM

7.

	Intel, Nadya Morozova: .	200665

JitrinoDynamic Runtime Layer, DRLDRLVMJITVMDRLVM

DRLVMDRLJIT

DRLVM

- JIT
- Jitrino.OPT
  - Jitrino.OPT
  -
- Jitrino.JET:
  - Jitrino.JET
  -
  - JIT
  - JIT

DRL

JitrinoDRLVMJIT[2]JItrinoJIT

- Jitrino.JETJava
- Jitrino.OPT(IR)(HIR)(LIR)Jitrino.OPTJITHIRLIRJitrinoHIR

Jitrino(JETOPT)

JIT

-
-
-

Jitrino.OPT

-
-
- JitrinoDRLIR (dump)
-

JitrinoJava*1

Jitrino.JETJitrino.OPT

JitrinoJitrino.JETJitrino.OPTJitrino(EM)

Jitrino.JETJava*

Jitrino.OPT— JitrinoJItrino.OPT1

- Java*(HIR)HIRJitrino.OPT
  :

Jitrino(CLI)

- HIR(LIR)(LIR)

Jitrino.JETJitrino.OPT

{{ http://harmony.apache.org/subcomponents/drlvm/images/compilation_process.gif }}
1. Jitrino

Jitrino.OPT

Jitrino.OPT

pipeline management frameworkPMFJitrino.OPTPMFpipelineActions(pass)

Java*JITJava*

Jitrino.OPTVMPMFOPT“”hard-codedEMVMJitrinoJitrinoPMFPMF

---

high-level intermediate representationHIRIR

3.2.1

Java*Java*CLI3.2.2

HLOcode selectorHIRLIRcode generatorCGHIRLIR3.2.3

CGCGHIR3.2.4.

-
- GC
- VM
-

Jitrino.JETJitrino.OPTIRhigh-level intermediate representation HIRlow-level intermediate representation LIR)Jitrino.OPTJava*

:

Java*Java*Jitrino.OPT

Java*:

```
    public static int max(int x, int y) {
        if (x > y) {
            return x;
        }
        return y;
    }
```

Java*

```
    public static int max(int, int);
     Code:
      0:   iload_0
      1:   iload_1
      2:   if_icmple       7
      5:   iload_0
      6:   ireturn
      7:   iload_1
      8:   ireturn
```

Jitrino:

{{

http://harmony.apache.org/subcomponents/drlvm/images/HIR.png

}}

Jitrino:

{{

http://harmony.apache.org/subcomponents/drlvm/images/LIR.png

}}

HIRLIR5.4.Jitrino
IR

JitrinoHIRHIRHIRsingle static assignmentSSA)SSAuse-def(-)HIR

HIR
IR

JitrinoLIR{{Jitrino IA-32/Intel? 64 CG LIR}}HIRSSAIA-32Intel? 64

Jitrino

HIR

- JVM[1]Java*[1]
- HIR[3]

HIRJitrino.OPT

-
-
- HIR

Jitrinoprofile

- —Java*
- Profile—JITprofileJitrino.JET/profileJitrino.OPTprofileprofileprofileEM

Jitrino

- /profile
- profileprofile
  profileJIT

```
-Xem:server
```

Jitrino.OPT

Scope Enhancement Passes

- guarded devirtualization(devirt)
- Inlining
-
  - inliner
    - IR
    -
    -
  -
  -
  - HIR
  -
  -
  IR

```
Inline(HIR_of_compiled_method) {
    current_bytecode_size = HIR_of_compiled_method.get_method().bytecode_size()
    find_inline_candidates(HIR_of_compiled_method)
    while (true) {
        callee = NULL
        while (!inline_candidates.empty()) {
            callee = inline_candidates.pop()
            callee_bytecode_size = callee.bytecode_size()
            if ((current_bytecode_size + callee_bytecode_size) < SIZE_THRESHOLD) {
                current_bytecode_size += callee_bytecode_size
                break;
            }
        }
        if (callee = NULL) {
            break;
        }
        HIR_of_callee = Translator.translate(callee)
        Optimizer.optimize(HIR_of_callee, inliner_pipeline)
        find_inline_candidates(HIR_of_callee)
        HIR_of_compiled_method.integrate(HIR_of_callee)
    }
}

find_inline_candidates(method_HIR) {
    foreach direct_call in method_HIR {
        inline_benefit = compute_inline_benefit(direct_call)
        if (inline_benefit > BENEFIT_THRESHOLD) {
            inline_candidates.push(direct_call)
       }
    }
}
```

- *Lowering*(lower)lowerhelperHIRhelperhelper

Redundancy Elimination Passes

JIT

- Memory optimization(memopt)
  memoptSSASSAmemoptuse-defmemoptscoped
- Lazy exceptions optimization(lazyexc)
-
  - Loop peeling
  - Load invariant hoisting
  - Loop unrolling
- Array-bounds check elimination(abcd)abcdArrayIndexOutOfBoundsExceptionJIT[5]
- Global code motion(gcm)gcmprofile
- *Lowering*(lower)lowerHIRhelperhelperhelper

HIRHIR Simplification Passes

HIRJitrinoHIR

- Unreachable code elimination(uce)uce
- Dead code eliminationSSAuse-def[3]
- Simplification)
  -
  -
  -
  -
  - [3]
- Hash value numbering(hvn)hvn[3]chkzero(), chknull(), and chkcast() HIR

profileStatic profile estimatorstatprof)

---

profileprofilingprofileJITprofileJitrino.OPTstatprofHIRprofileprofilingCFGprofilestatprofprofile

HIRLIR(CS)HIR2.

{{ http://harmony.apache.org/subcomponents/drlvm/images/code_selector.gif }}
2.

:

-
- IR
- IR

CFG

- HIRHIR
- HIRLIRLIR

CSIRCG

3.

{{ http://harmony.apache.org/subcomponents/drlvm/images/code_selection_seq.gif }}
3.

3

- HIR
- CGLIR
- IR
  - CSHIRCSLIR
  - CFG CSHIRCFG CSLIR
  - CFG CSCSCSHIRLIRCSInstruction Selector

{{Jitrino IA-32/Intel? 64}}

LIR

HIRLIRLIR

1. 23
2.
3. 64
4.
5.

LIR

LIR

1.
2. GCGC
3.
4. 64
5. LIR32
6.
7. XMM
8.
9. profileCF G
10.
11.

EM

Code and Data Emission

LIR

1. LIRVM
   1. CG
   2. LIRVM
   3. LIR
   4.
   5. VMtry
   6. VM
   2. CALL 3. GCGC
   : GC 4. GC
      - GC
      - GC
      - Java*

VMJitrinoJitrino.JETJitrino.OPTlock_method Actionunlock_method Action

Lock_methodJITlock_methodCOMPILATION_FINISHEDunlock_methodlock_method

- ActionCOMPILATION_FINISHEDCOMPILATION_FAILED
-
- JavaVMresolve_static_method

Jitrino.JET

Jitrino.JETJitrinoJava*

GCGC

GCGC

- GC()
-

GCGCGCVM helperGCGCGC

Jitrino.JET

Jitrino.JET

Jitrino.JET4

{{ http://harmony.apache.org/subcomponents/drlvm/images/bytecode_to_native.gif }}
4.

:

__(BB)

Jitrino.JET

- athrowreturngotoswitchesretjsr
- __

:

5(BB2)ref_count=1(BB1)2BB1BB2

{{ http://harmony.apache.org/subcomponents/drlvm/images/reference_count.gif }}
5.

Jitrino.JET

Jitrino.JET

1
2Java*
3
4CALLJMP

{{Doxygen}}

JIT

-
- (STL)
-
-
- IR
- profiler(dump)
  :

JITVMJIT

Jitrino.OPT{{jitrino/src/shared/directory}}{{.h}}{{.cpp}}

JIT

{{arenas}}{{MemoryManager}}{{arena}}4096JIT{{arena}}{{arena}}{{arena}}

{{MemoryManager}}

```
    void optABC() {
        //the temporary memory manager used for optABC optimization data
        MemoryManager tmpMM(10000, "mm::optABC");

        StlVector<int> myData1(tmpMM, 1000);
        int* myData2 = new (tmpMM) int[1000];
        //JIT compiler code follows
    }
```

JIT

1__{{MemoryManager}}{{MemoryManager}}
2
3{{MemoryManager}}
4{{MemoryManager}}{{MemoryManager}}

Jitrino.OPT

- JitrinoJIT{{jitrino/src/main/Jitrino.cpp}}{{global_mm}}
- __

{{MemoryManager}}

```
    MemoryManager tmpMM(10000, "myMM");
       int* myData2 = new (tmpMM) int[10];
       myData[10] = 1;
```

{{MemoryManager.cpp}}{{JIT_MEM_CHECK}}{{arena}}{{arena}}

Jitrino__(counters)JitrinoVM

JitrinoJitrino__timers

```
    -XDjit.<JIT>.arg.time=on
```

:

off

VM

:

VM

```
-XcleanupOnExit
```

Jitrino

- Jitrino
-
-

Jitrino PMF

-
-

__Jitrino

`info`	JIT
`rt`	
`ct`	
`irdump`	Jitrino
`dotdump`	.dotJitrino
`dbg`	

PMF

```
-XDjit.<JIT>.<pipeline>.arg.<path>.log=<list of stream names>
```

<path>Jitrino

:

JIT:

```
-XDjit.arg.log=info
```

IR :

```
-XDjit.arg.log=ct,irdump
```

JITJitrinoIRdot.dotCFG3.1.3.dotGraphviz[6]

.dot

```
-XDjit.arg.log=dotdump
```

JitrinoPMF

__(CFG)Jitrino

CFG{{jitrino/src/shared/ControlFlowGraph .h}}{{.cpp}}CFG

-
-
-

- __
- __

IR--

CFG{{Node}}{{Edge}}{{CFGInst}}

CFG{{CFGInst}}{{Node}}{{Edge}}

CFG

- *Block* IR
- *Dispatch*
- *Exit*
  Exit,Exit

CFGEntryReturnUnwindReturnUnwind,Exit

,

- Dispatch
- Unconditionswitch(fall-through)ReturnUnwindExit
- TrueFalse,TrueFalseCFG
- CatchIR,

CFGCFGInst

CFGCFGDispatch

.h.hHIR

CFG

-
-
-
-
-
-

{{Doxygen}}{{ControlFlowGraph.h}}

()DominatorNode,

{{jitrino/src/shared/DominatorTree.h}}{{.cpp}}

:

CFG

`LoopTree LoopNode LoopNode LoopNode`

{{jitrino/src/shared/LoopTree.h}} {{.cpp}}

:

CFGCFG

JITJitrinoJitrinoJIT

JIT_VM

JITJIT_VM

JITVM{{Doxygen}}{{vmcore /include /jit_export.h}}

JITJITGC

:

JITGC

GC {{Doxygen}}{{vmcore/include/jit_export_rt.h}}

JIT

- JITJIT
- VM
- JIT
  3.4

:

JVMTI

JVMTIJITJVMTI

- JITJVMTIJVMTIJITJIT
-
-
  JITJVMTI

VMJIT {{Doxygen}}{{vmcore/include/jit_export_jpda.h and vmcore/include/jit_export.h}}

JIT_EM

JIT

- EM
- EMJITprofileprofile

JIT

JITEM{{Doxygen}}{{include/open/ee_em_intf.h}}

[1] Java* Virtual Machine Specification, http://java.sun.com/docs/books/vmspec/2nd-edition/html/VMSpecTOC.doc.html <ac:structured-macro ac:name="anchor" ac:schema-version="1" ac:macro-id="f2db3153-dbdd-48b6-80a6-8d0f42f94ea1"><ac:parameter ac:name="">ref1</ac:parameter></ac:structured-macro>

[2] JIT Compiler Interface Specification, Sun Microsystems, http://java.sun.com/docs/jit_interface.html <ac:structured-macro ac:name="anchor" ac:schema-version="1" ac:macro-id="32881ac4-43cb-4dd5-9f2f-23c5638382ba"><ac:parameter ac:name="">ref2</ac:parameter></ac:structured-macro>

[3] S. Muchnick, Advanced Compiler Design and Implementation, Morgan Kaufmann, San Francisco, CA, 1997. <ac:structured-macro ac:name="anchor" ac:schema-version="1" ac:macro-id="4b03ae7e-500a-4297-b6f1-7ef6124bfc5d"><ac:parameter ac:name="">ref3</ac:parameter></ac:structured-macro>

[4] P. Briggs, K.D., Cooper and L.T. Simpson, Value Numbering. Software-Practice and Experience, vol. 27(6), June 1997, http://www.informatik.uni-trier.de/~ley/db/journals/spe/spe27.html <ac:structured-macro ac:name="anchor" ac:schema-version="1" ac:macro-id="e9300212-689d-4fd1-8432-188a2ed646bd"><ac:parameter ac:name="">ref4</ac:parameter></ac:structured-macro>

[5] R. Bodik, R. Gupta, and V. Sarkar, ABCD: Eliminating Array-Bounds Checks on Demand, in proceedings of the SIGPLAN '00 Conference on Program Language Design and Implementation, Vancouver, Canada, June 2000, http://portal.acm.org/citation.cfm?id=349342&dl=acm&coll=&CFID=15151515&CFTOKEN=6184618 <ac:structured-macro ac:name="anchor" ac:schema-version="1" ac:macro-id="e333b044-0902-423c-8fc7-ca637715c364"><ac:parameter ac:name="">ref5</ac:parameter></ac:structured-macro>

[6] Graphviz, Graph Visualization Software, http://www.graphviz.org/ <ac:structured-macro ac:name="anchor" ac:schema-version="1" ac:macro-id="48311d8e-ee49-49bc-bb4f-5558ff1a6c85"><ac:parameter ac:name="">ref6</ac:parameter></ac:structured-macro>

- Other brands and names are the property of their respective owners.

Contributors:
(yuzhang@ustc.edu.cn)

(yln@mail.ustc.edu.cn)

(haoyy@mail.ustc.edu.cn))