Virtual Machine Optimization

Asked 16 years, 4 months ago Modified 12 years, 5 months ago

Viewed 4k times

12

I am messing around with a toy interpreter in Java and I was considering trying to write a simple compiler that can generate bytecode for the Java Virtual Machine. Which got me thinking, how much optimization needs to be done by compilers that target virtual machines such as JVM and CLI?

Do Just In Time (JIT) compilers do constant folding, peephole optimizations etc?

java jvm jit cil

Share

Improve this question

Follow

edited Aug 21, 2008 at 14:42

Greg Hurlman

17.8k ● 6 ● 55 ● 87

asked Aug 21, 2008 at 14:41

grom

16.1k ● 19 ● 65 ● 67

7 Answers

Sorted by: Highest score (default)

6

I'm just gonna add two links which explain Java's bytecode pretty well and some of the various optimization of the JVM during runtime.

Share Improve this answer

Follow

edited Dec 17, 2009 at 18:50

Dinah

54k • 30 • 134 • 149

answered Aug 21, 2008 at 19:22

dlinsin

19.6k • 13 • 43 • 53

5

Optimisation is what makes JVMs viable as environments for long running applications, you can bet that SUN, IBM and friends are doing their best to ensure they can optimise your bytecode and JIT-compiled code in an efficient a manner as possible.

With that being said, if you think you can pre-optimise your bytecode then it probably won't do much harm.

It is worth being aware, however, that JVMs can tend towards performing better (and not crashing) when presented with just the sort of bytecode the Java compiler tends to construct. It is not unknown for optimisations to be missed or even for the JVM to crash when permutations of bytecode occur that are correct but unlike what would be produced by javac. Hopefully that sort of thing is more in the past now, but may be something to be aware of.

Share Improve this answer

Follow

answered Sep 17, 2008 at 0:36

Mike Tunnicliffe

10.8k ● 3 ● 33 ● 46

4

> Optimising bytecode is probably an oxymoron in most cases

I don't think that's true. Optimizations like hoisting loop invariants and propagating constants can never hurt, even if the JVM is smart enough to do them on its own, by simple virtue of making the code do less work.

Share Improve this answer

Follow

answered Aug 21, 2008 at 14:52

DrPizza

18.3k ● 7 ● 42 ● 53

2

Obfuscators such as ProGuard will perform many static optimisations on your bytecode for you.

Share Improve this answer

Follow

edited Jul 3, 2012 at 13:22

answered Dec 17, 2008 at 11:49

Damon

The HotSpot compiler will optimize your code at runtime better than is possible at compile-time - it has more

1

information to work with, after all. The only time you should be optimizing the bytecode instead of just your algorithm is when you are targeting mobile devices, such as the Blackberry, where the JVM for that platform is not powerful enough to optimize code at runtime and just executes the bytecode.

Share Improve this answer

Follow

answered Dec 18, 2008 at 12:24

Nathaniel Flath

16k ● 19 ● 73 ● 94

0

Optimising bytecode is probably an oxymoron in most cases. Unless you control the VM, you have no idea what it does to speed up code execution, if anything. The compiler would need to know the details of the VM in order to generate optimised code.

Share Improve this answer

Follow

edited Jul 3, 2012 at 13:22

user142162

answered Aug 21, 2008 at 14:47

Skizz

71k ● 10 ● 74 ● 109

0

Note to Aseraphim:

It can also be useful to optimise bytecode for non-embedded applications in some limited cases:

1. When delivering code over the wire, eg for WebStart apps, to minimise deliverable/cache size and because you don't necessarily know the capability/speed of the client.
2. For code that you know is performance critical and used at start-up before (say) HotSpot has had time to gather any stats.

Again, the transformations that a good optimiser/obfuscator performs can be very helpful.

Share Improve this answer Follow

edited Jul 3, 2012 at 13:22

answered Dec 31, 2008 at 8:56

Damon

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.