

Re: Java outperforms C++?

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2005-04/msg01046.html>

- *From:* Jerry Coffin <jcoffin@xxxxxxxxxx>
 - *Date:* Wed, 20 Apr 2005 00:30:18 -0600
-

In article <d43c8a\$ed1\$2\$8302bc10@xxxxxxxxxxxxxxxxxxxx>, \$Scott@xxxxxxxxxxxxxxxxxxxx says...

[...]

- > So you're saying, having to save data to memory because of aliasing doesn't
- > matter, it's entirely free? MOV instructions don't take any time because
- > they are going to an on chip cache so the fact that C++ has to do them but
- > Java doesn't won't affect the speed?
- > I think you should think again,
- > differences in compilers depend on loosing an instruct here or there and
- > this is a definite plus for Java. Even if the instruction is completely
- > overlapped it is stopping some other instruction from overlapping.

Nobody (least of all me) has claimed that it has no effect. The effect is, however, quite small — it started out small, and has been shrinking for years.

- > Which C++ compiler uses GC?

I've used it with VC++, gnu and Borland. Offhand I don't remember for sure whether I've used it with Comeau or not, but I kind of doubt it. By the time I started using Comeau, I'd learned better ways of doing things.

- > How does it handle say locked database resources which would normally be
- > released when a destructor is called? It is hard to see how this could be
- > implemented as a compiler option since it would force programmers to behave
- > as if CG was present (for the benefit of releasing critical resources) even
- > when the option was off. For all practical purposes C++ and GC are
- > incompatible – indeed a C++ which did use it would not be ANSI compliant..

Not so — GC simply deals only with memory. Using it the global delete becomes a NOP, but other parts of dtors continue to operate normally.

OTOH, you do a fine job of pointing out one of the major shortcomings of Java — a language such as Java that assumes GC, is almost invariably clumsy and restricted compared to C++.

Re: Java outperforms C++?

- >> Run-time compilation: here he has three sub-points:
- >> 1) The VM can contain code specifically for the CPU it's running on.
- >>
- >> ...and so can machine code.
- >>
- > You're missing the point here. The burden is not on the programmer to write
- > machine specific code.

Correct -- again, about both VMs and machine code. Using '-G6' or '-G7' is hardly what a sane person could call a burden.

- > The optimizations which can be done will always be
- > better when the compilation is done at runtime.

Theoretically, JIT affords a few optimization opportunities that don't exist otherwise.

Unfortunately, this doesn't mean much. The problem is that optimization is difficult and expensive. With JIT compilation, you only get to amortize that expensive, difficult work across a single execution, so in a typical case spending much time on it is a net loss.

Decoupling the optimization from execution allows the optimization to be amortized across all the executions, justifying the expenditure of MUCH greater effort.

- > Optimizations for specific
- > CPUs could be done as in C++ but that would produce either a
- > separate executable for each processor or a very large executable to select
- > different optimizations for different CPUs in a limited set like say the
- > Intel processor range.

I guess I won't worry too much about those executables that attempt to optimize for both SPARC and Intel, since I haven't had to deal with any that targeted both to start with.

Realistically, when you're doing this you typically want to do something like putting the critical code into DLLs. This critical code that's open to substantial CPU-specific enhancement is rarely large enough to occupy even a whole penny's worth of drive space at current hard drive prices. I've never dealt with a case where it was anywhere close as big as a JVM.

- >> 2) the VM might know more about what calls need to be virtual.
- >>
- >> ...or it might not. At least in my testing, most don't.
- >
- > Which VMs and Java versions have you tested and how did you determine that
- > the VM did not know that some particular call didn't need to be virtual?

Re: Java outperforms C++?

Re: Java outperforms C++?

I've tested JVMs from Sun, Microsoft, HP and IBM, from some of the earliest betas up through one downloaded from Sun within the last month. I did the determination in the only way that's really effective: trace through it's execution with a kernel debugger so I can see exactly what it's doing.

[...]

- > Branch prediction at compile time might allow the compiler to inline some
- > procedure or even 'procedurize' some chunk of code to maximize cache hits.
- > These are all new techniques, but the scope is enormous, just as C compilers
- > eventually got to produce code which runs faster than most assembler
- > programmers (due largely to the complexity of the instruction set, its
- > timing pipeline effects etc) so a similar potential exists for run-time
- > compilation.

As an aside, an "assembler programmer" is a person who writes assemblers, which is distinct from an assembly language programmer, who writes assembly language code (much as a "compiler programmer" is a person who writes compilers, distinct from somebody who writes code that gets compiled).

In the end, we're left with one simple fact: as fascinating as all these discussions about why Java *might* be able to keep up with C++, they're all little more than hand-waving. The facts come out in real tests -- and so far, you've yet to show a single instance in which reasonably written C++ has been slower than Java.

I've said before that I suspect such instances probably do exist, and I certainly believe that they should. The fact remains that the cases we've seen where it was claimed that C++ lost, what was being tested wasn't really C++ at all -- it was just Java written in a strange dialect that happens to be (sort of) acceptable to a C++ compiler.

--

Later,
Jerry.

The universe is a figment of its own imagination.

.

• *References:*

- ◆ *Java outperforms C++?*
 ◇ *From: benben*
- ◆ *Re: Java outperforms C++?*
 ◇ *From: \$Scott*
- ◆ *Re: Java outperforms C++?*
 ◇ *From: Jerry Coffin*

Re: Java outperforms C++?

- ◆ [Re: Java outperforms C++?](#)
 - ◇ From: \$Scott
- ◆ [Re: Java outperforms C++?](#)
 - ◇ From: Bo Persson
- ◆ [Re: Java outperforms C++?](#)
 - ◇ From: \$Scott
- ◆ [Re: Java outperforms C++?](#)
 - ◇ From: Jerry Coffin
- ◆ [Re: Java outperforms C++?](#)
 - ◇ From: \$Scott

- Prev by Date: [Re: Java outperforms C++?](#)
- Next by Date: [Re: variant t constructor bug?](#)
- Previous by thread: [Re: Java outperforms C++?](#)
- Next by thread: [Re: Java outperforms C++?](#)
- Index(es):
 - ◆ [Date](#)
 - ◆ [Thread](#)