

# Java Performance and Replay Compilation

Note Title

10/28/2008

Java app performance affected by:

true  
for  
all  
perf.  
meas.

- ① input
- ② hardware and OS (host platform)
- ③ language compiler, libraries, runtime  
→ includes GC algorithms
- ④ heap size
- ⑤ nondeterminism

all perf.  
meas. →

- ⑥ warmup  
→ timer-based sampling  
derived JIT compilation  
→ thread scheduling  
with more runs, app is  
more optimized.

To manage nondeterminism of  
JIT optimizations,

a current idea is to use  
REPLAY COMPILATION

↳ ① execute app  $g$  times  
collect a profile run

list of optimization  
level for each func.

② restart JVM.

③ repeat step ①  $n$  times.

now you have  $n$  profile runs.

④ From the  $n$  profile runs,

build 1 COMPILATION PLAN,  
which is a fixed list of  
optimization levels for each  
function.

Compilation plan can be the optimal  
profile run or some merged  
version of all  $n$  profile runs.

⑤ run experiments  
by replaying fixed compilation  
plan

# Java Performance Evaluation through Rigorous Replay Compilation

---

Andy Georges et al,  
Ghent Univ, OOPSLA 2008

---

Jikes Research VM

- Supports replay compilation
- compilation plan called  
an "advice file"

Using Jikes RVM, they show

Performance Results for one  
compilation plan may not be representative  
of other compilation plans

using specjvm98 and DaCapo,

They found

- little overlap between

  - 10 compilation plans  
each computed from  
a 10-iteration profile run

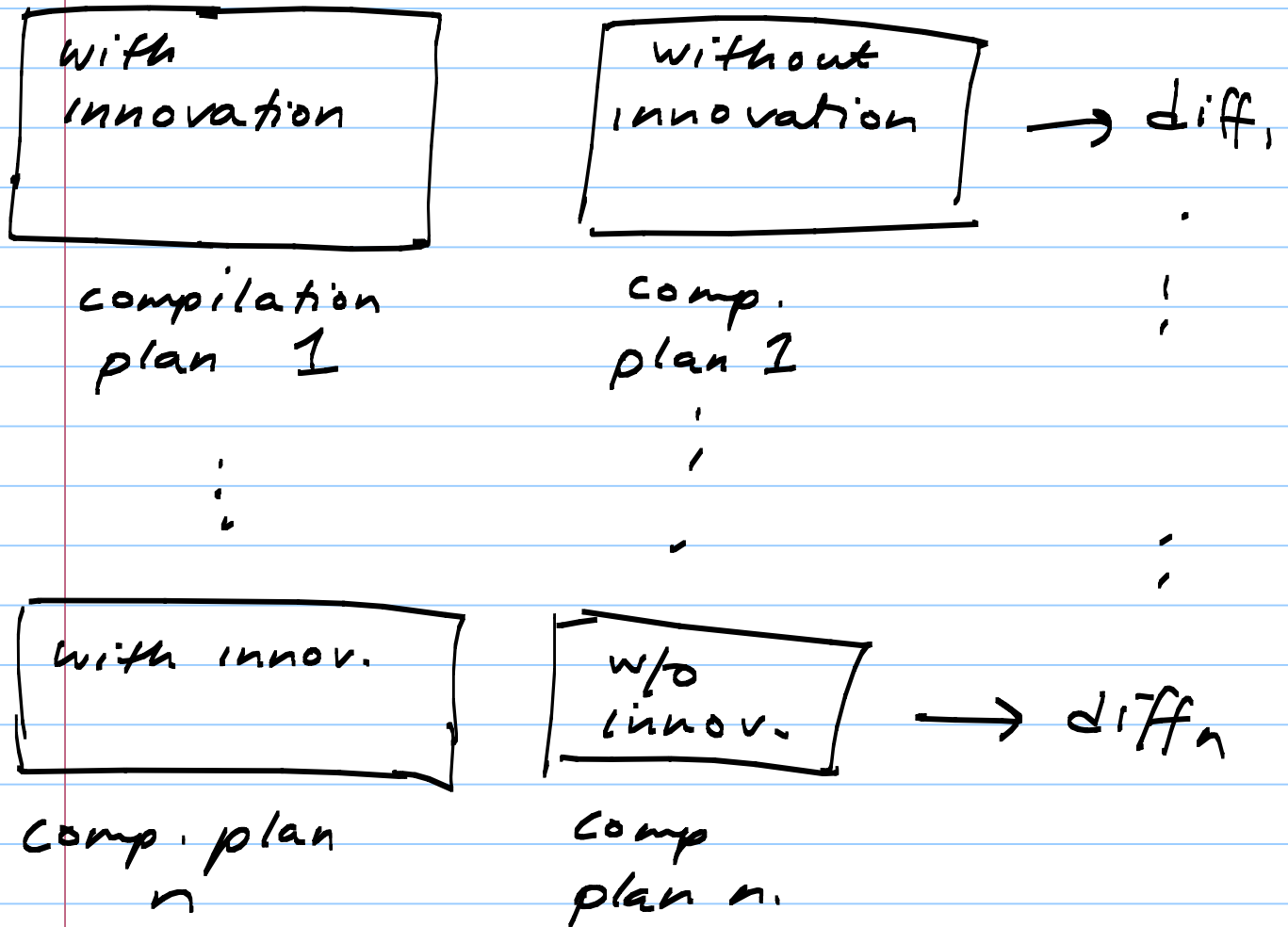
- different compilation  
plans may not agree  
on which GC runs fastest

- and of course, even running  
the same compilation plan  
multiple times will give you  
different execution times

**BOTTOM LINE:**

You need multiple compilation  
plans.

# Statistical Analysis



do a within-subjects t-test

