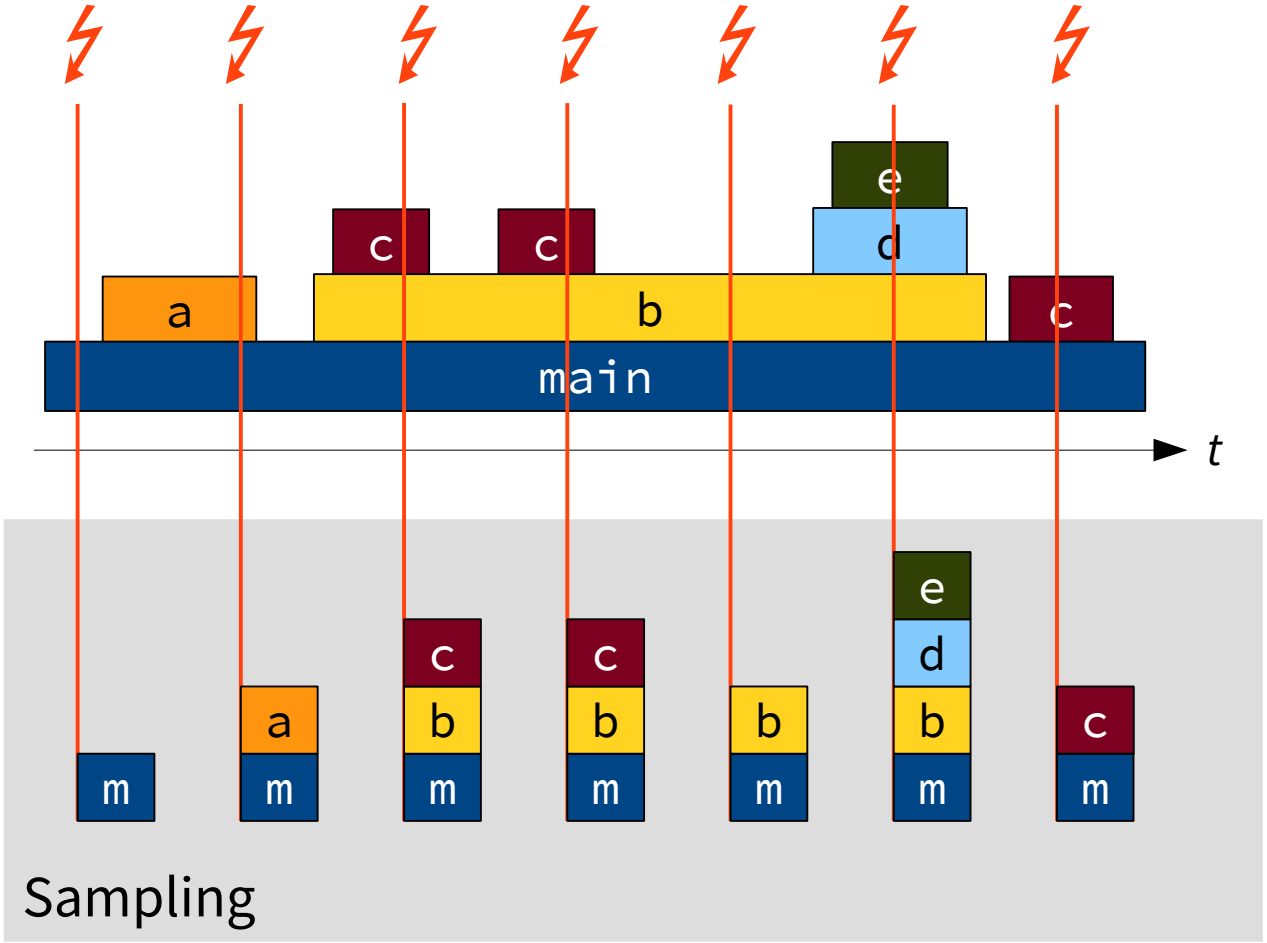
Lightweight Java Profiling with Partial Safepoints and Incremental Stack Tracing

> Peter Hofer
David Gnedt
Hanspeter Mössenböck

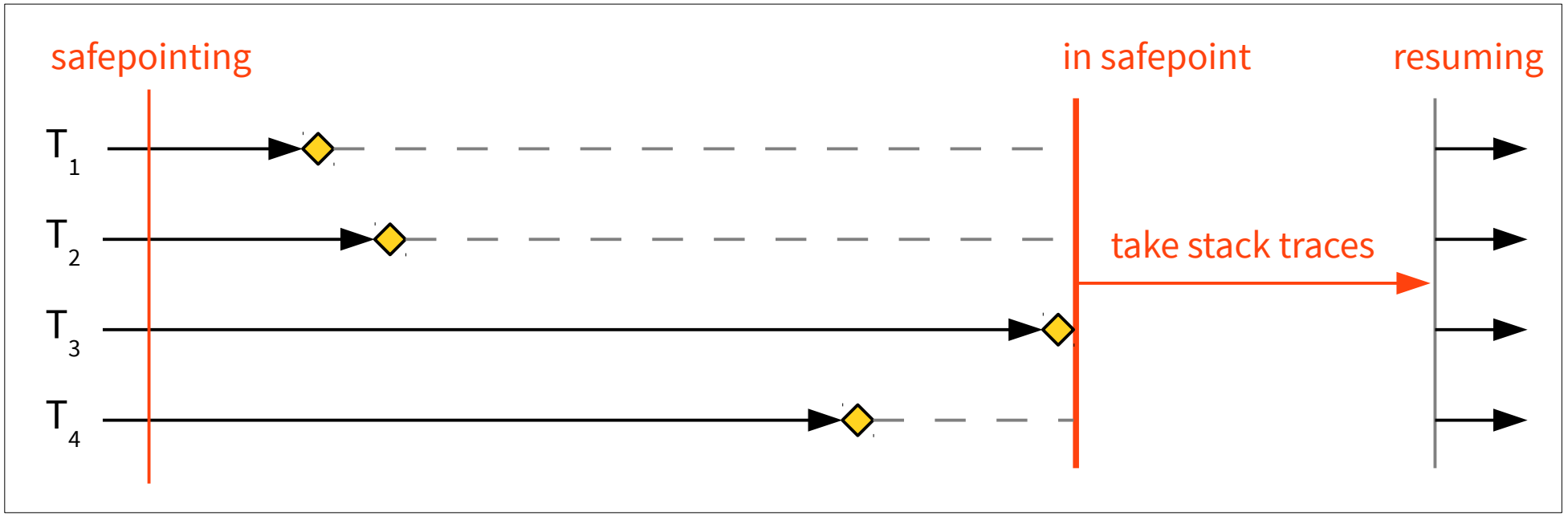
2 February 2015

Profiling

Where does my code spend its time?



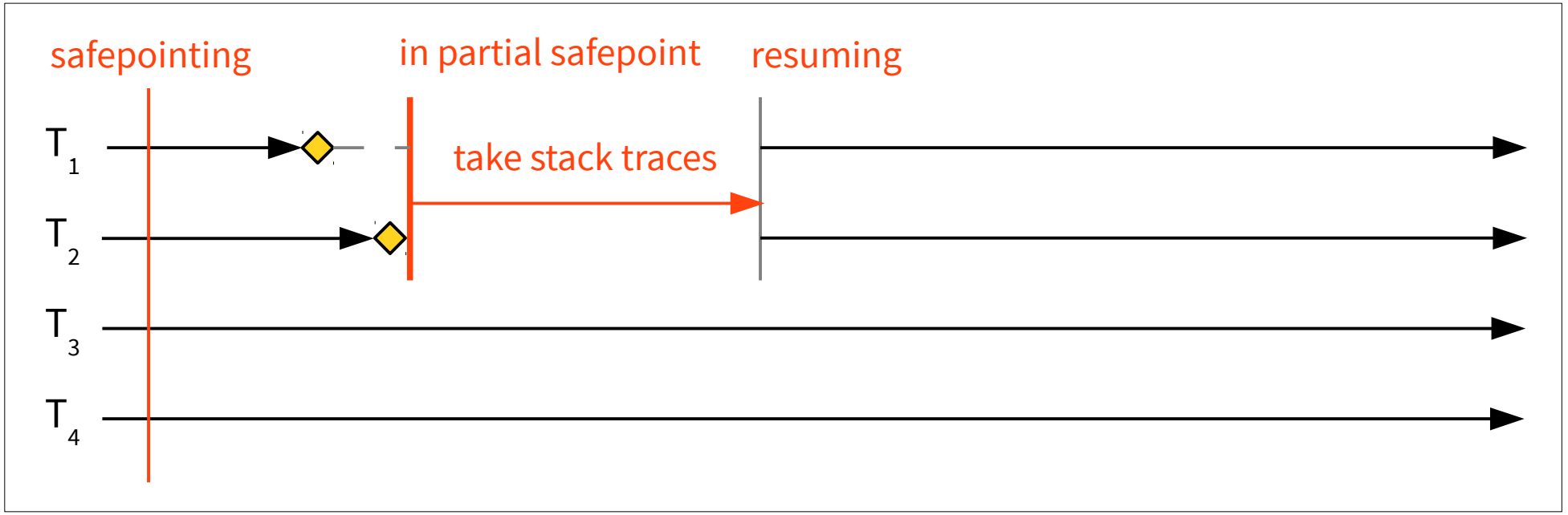
Sampling with Safepoints



Partial Safepoints

Sample first *k* threads that enter

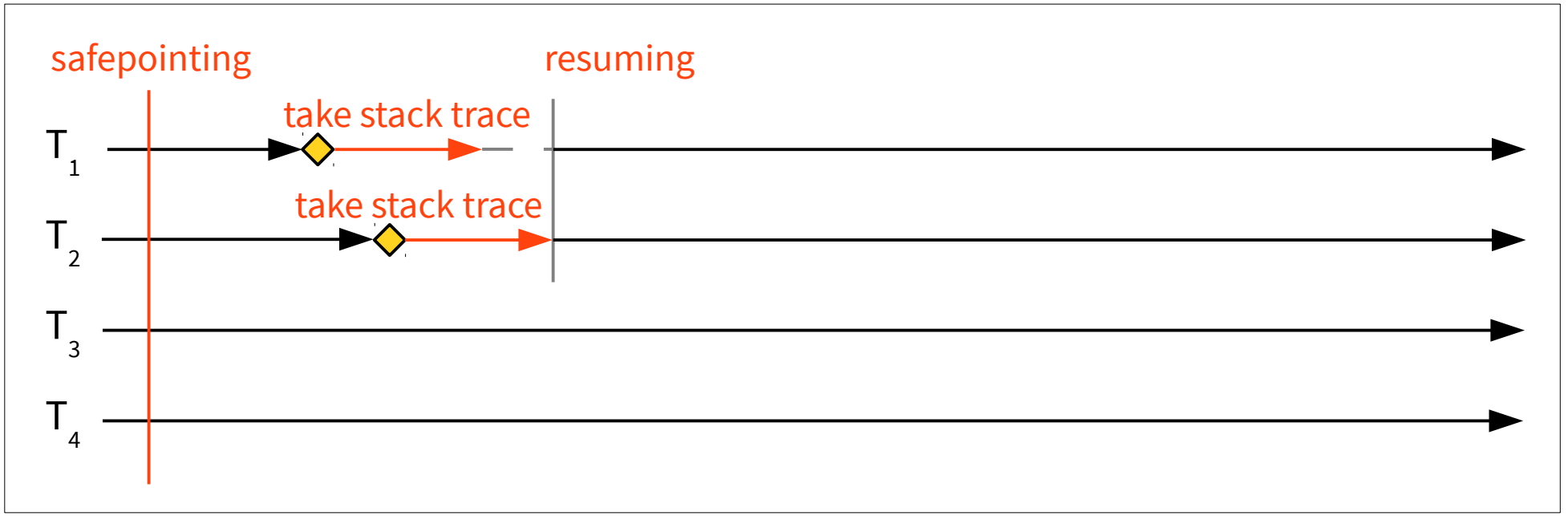
… out of a set of *n* threads of interest



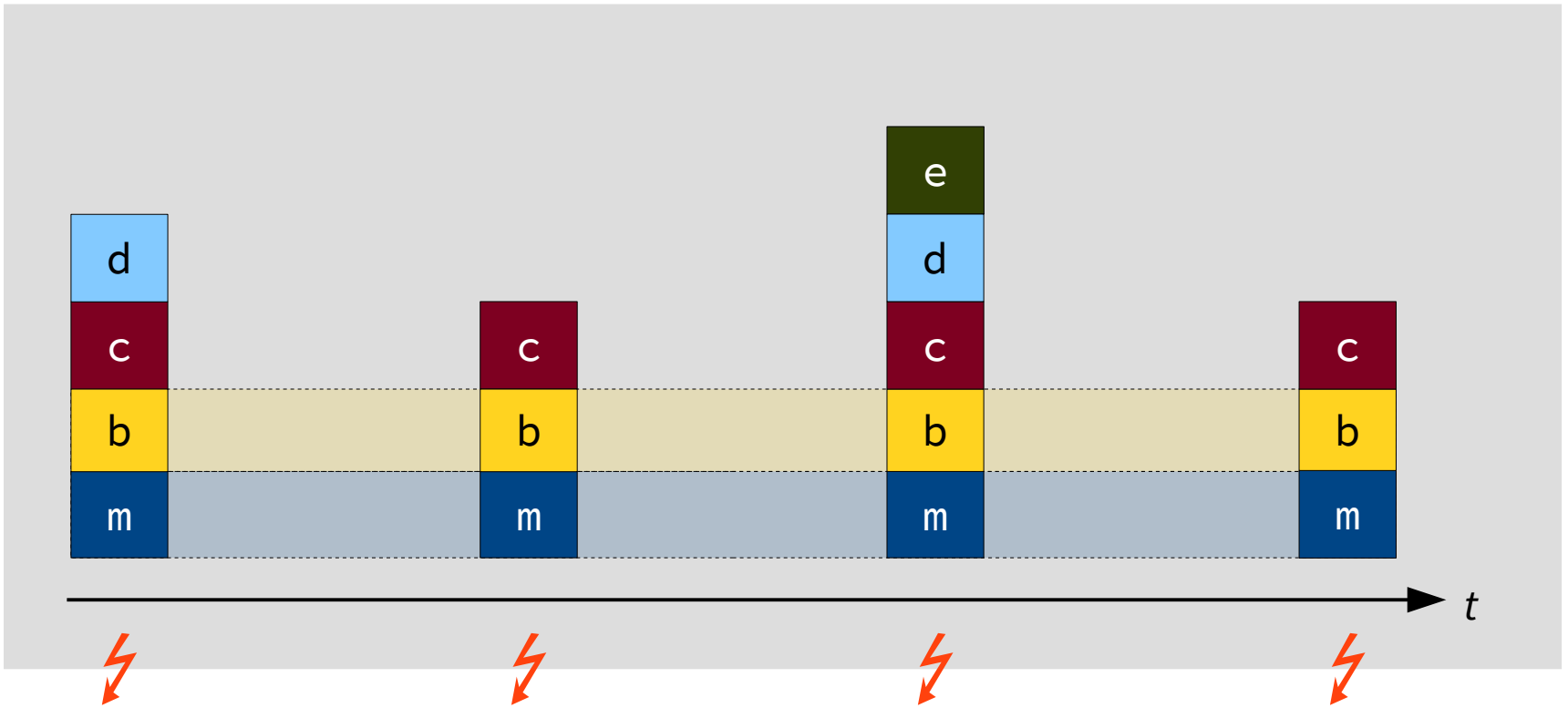
optional: include waiting threads

Partial Safepoints and Self-Sampling

Each thread walks its own stack.

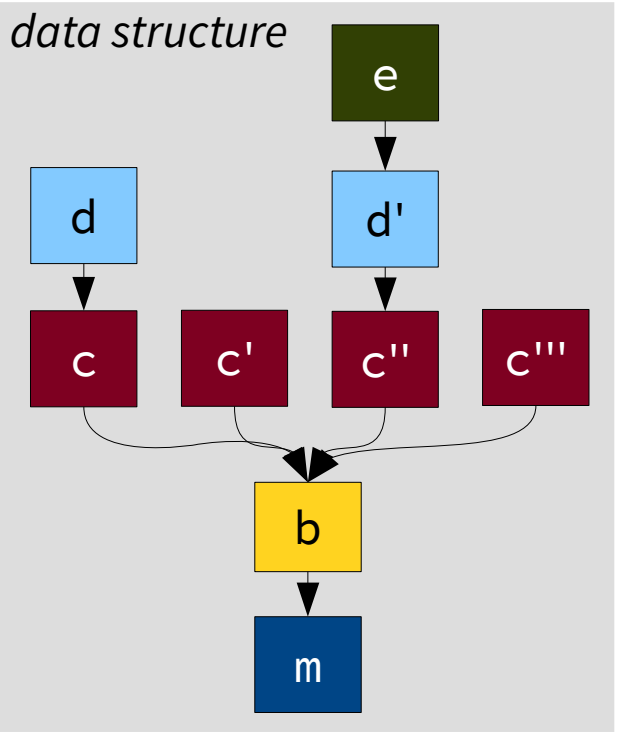


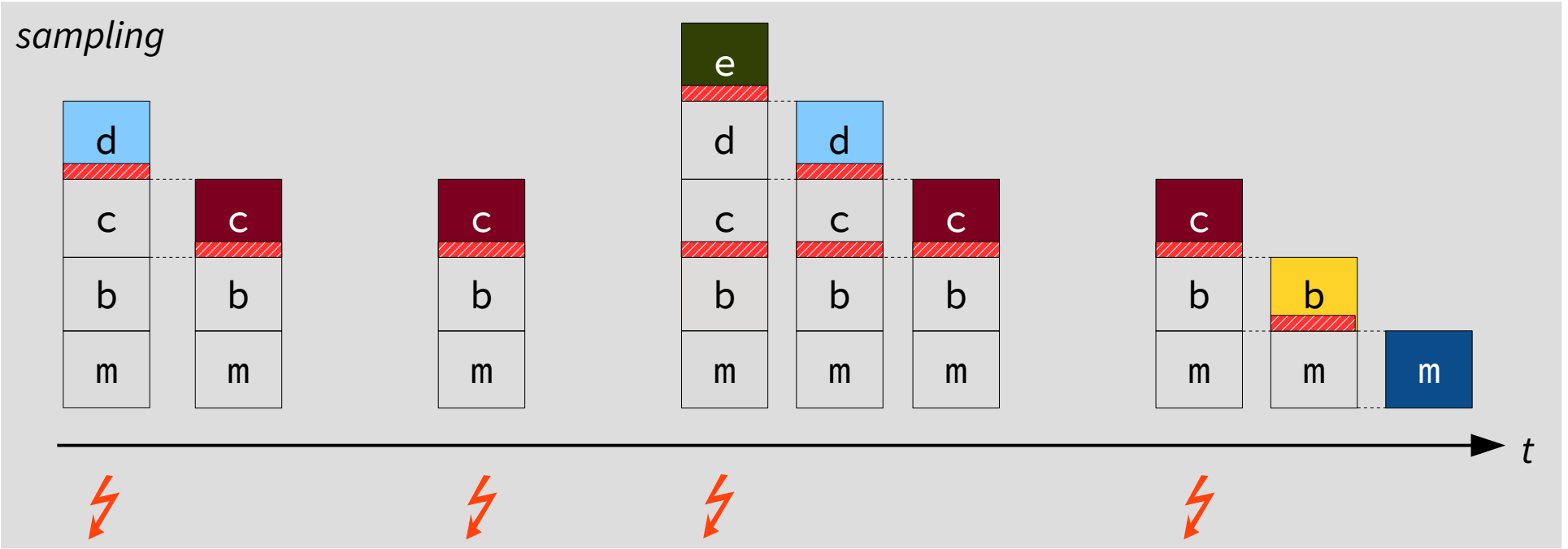
Redundant Stack Tracing Effort



Incremental Stack Tracing

Solution: decode only changed frames.





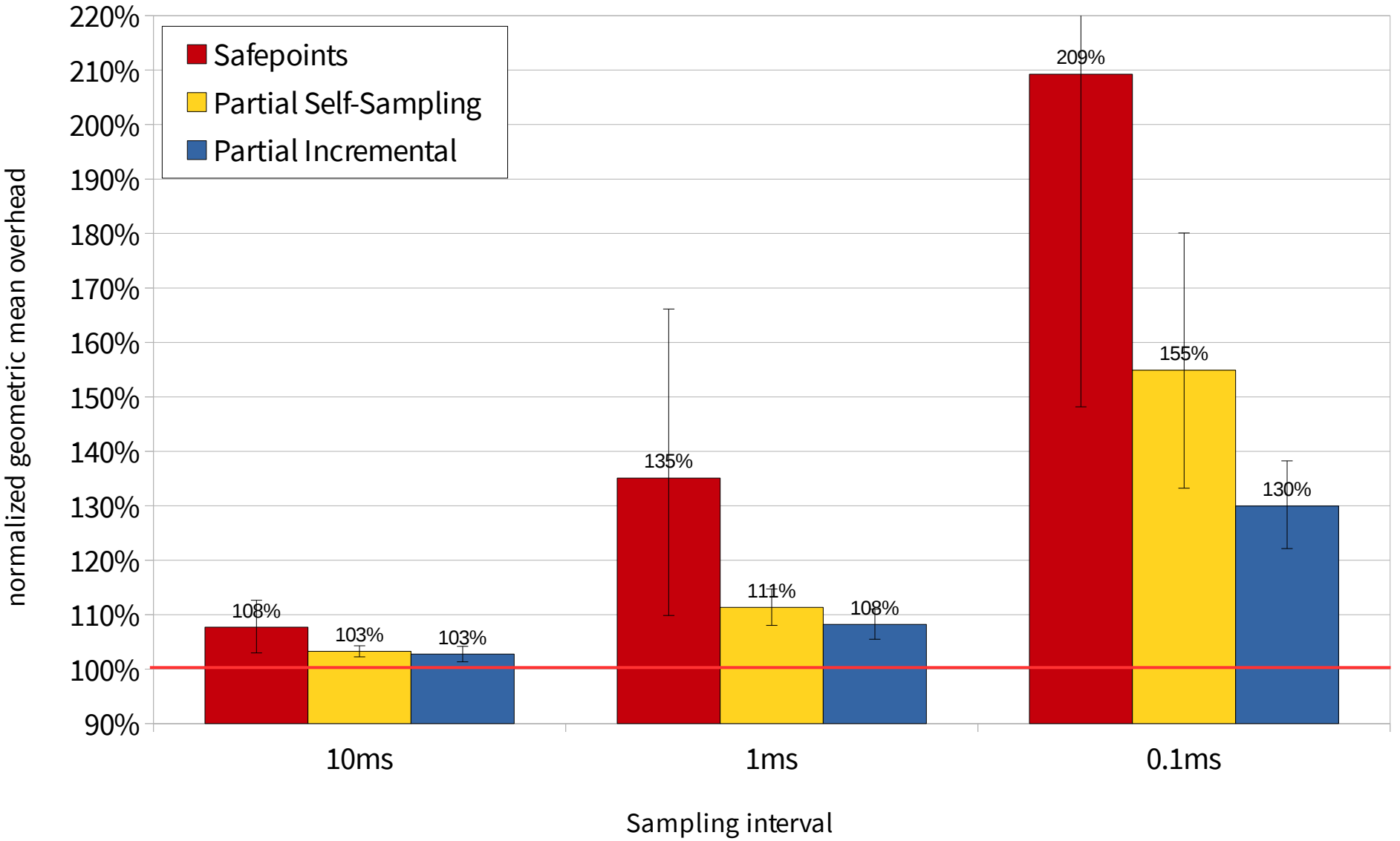
Implementation in HotSpot JVM (OpenJDK)

Challenges:

- Frame layouts
  - interpreter frames, compiled frames
- Inlining
  - multiple methods in one stack frame
- Exceptions
  - trace while unwinding the stack
- Deoptimization and on-stack replacement
  - frame is transformed, patching is lost, ...

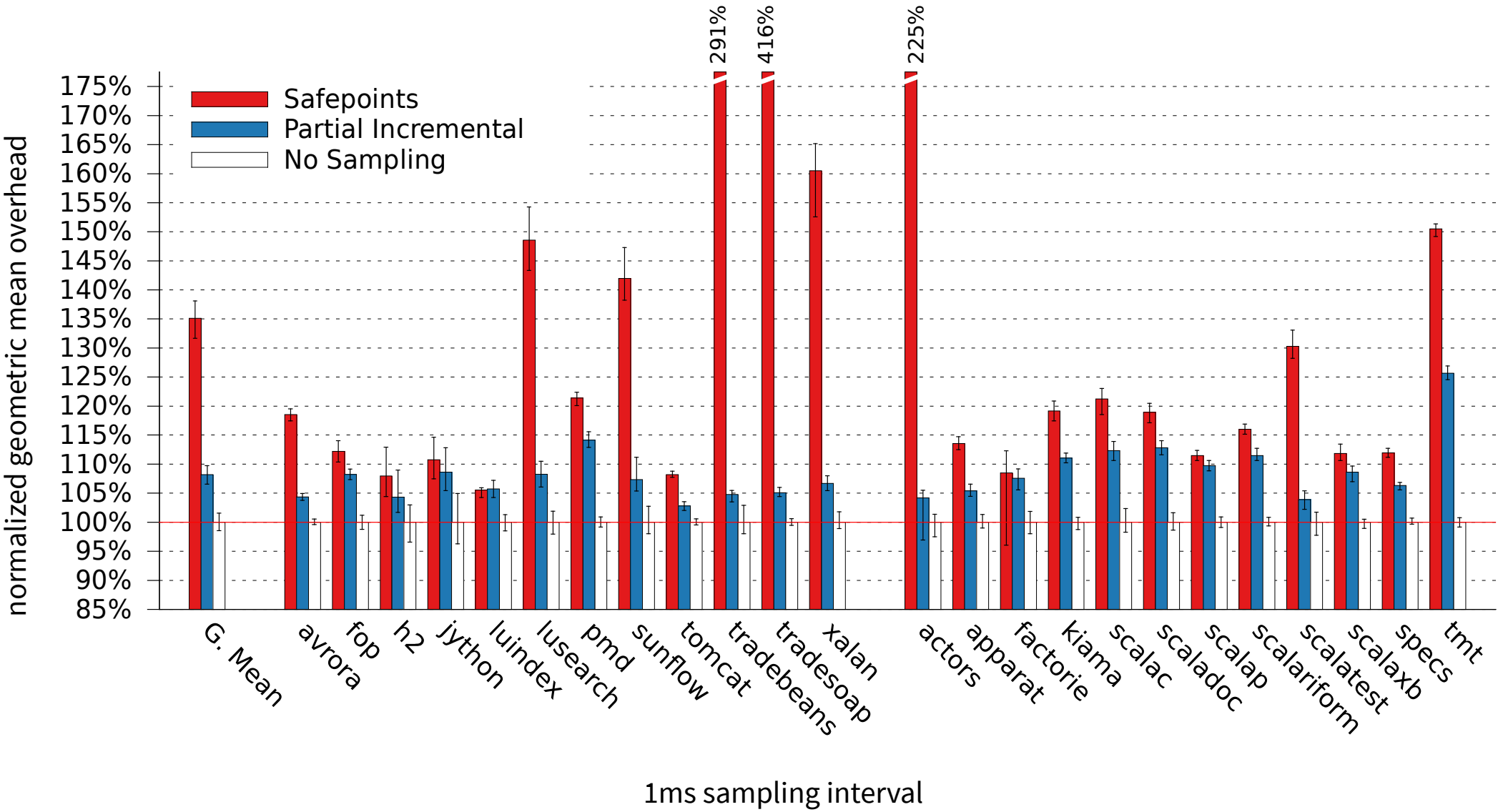
Overhead Comparison

DaCapo and scalabench, k = 4 threads on quad-core CPU

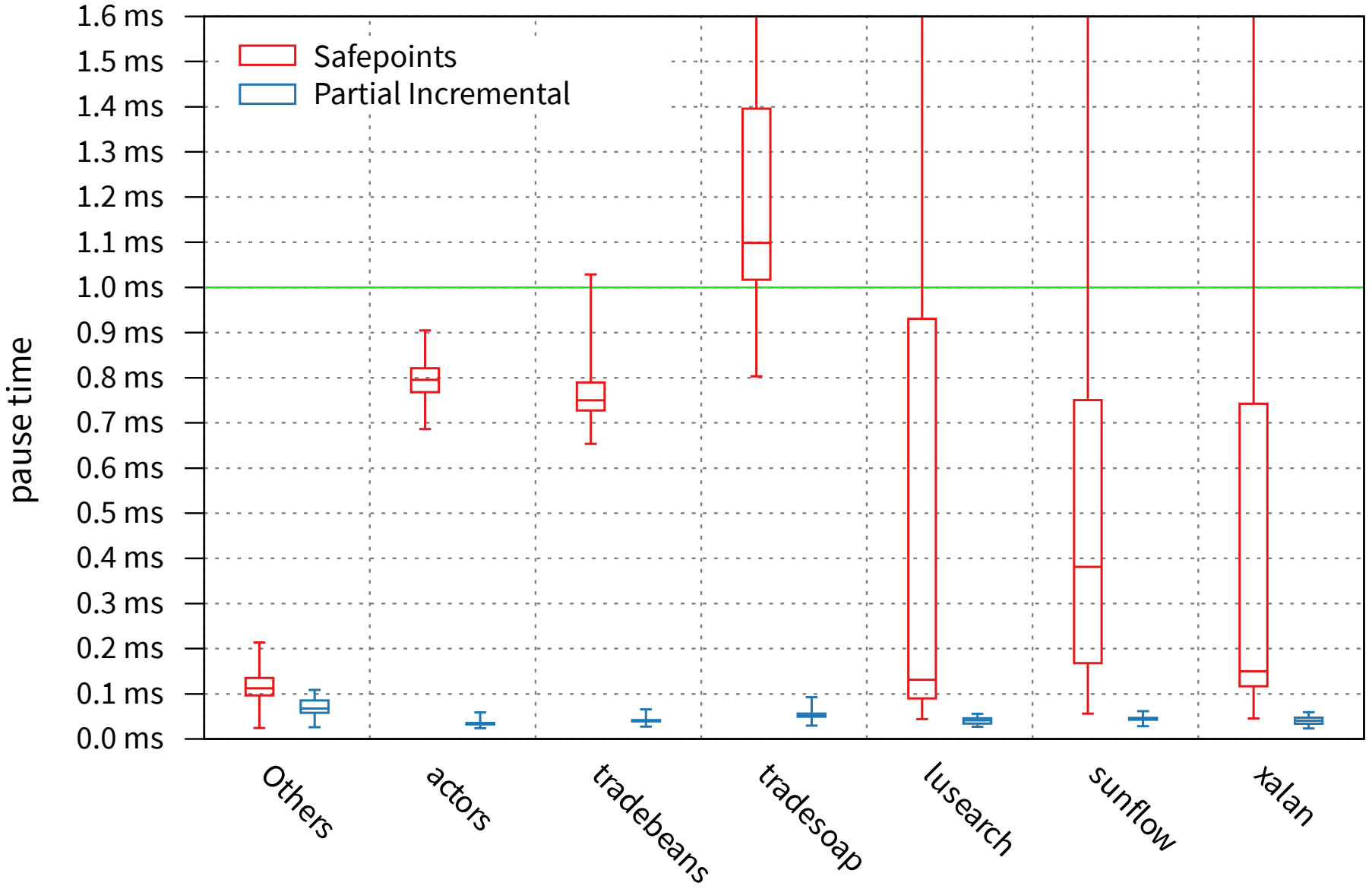


Overhead Comparison

DaCapo and scalabench benchmarks



Pause Times


1.6 ms
1.5 ms
1.4 ms
1.3 ms
1.2 ms
1.1 ms
1.0 ms
0.9 ms
0.8 ms
0.7 ms
0.6 ms
0.5 ms
0.4 ms
0.3 ms
0.2 ms
0.1 ms
0.0 ms
pause time
Safepoints
Partial Incremental
Others
actors
tradebeans
tradesoap
lusearch
sunflow
xalan

Accuracy

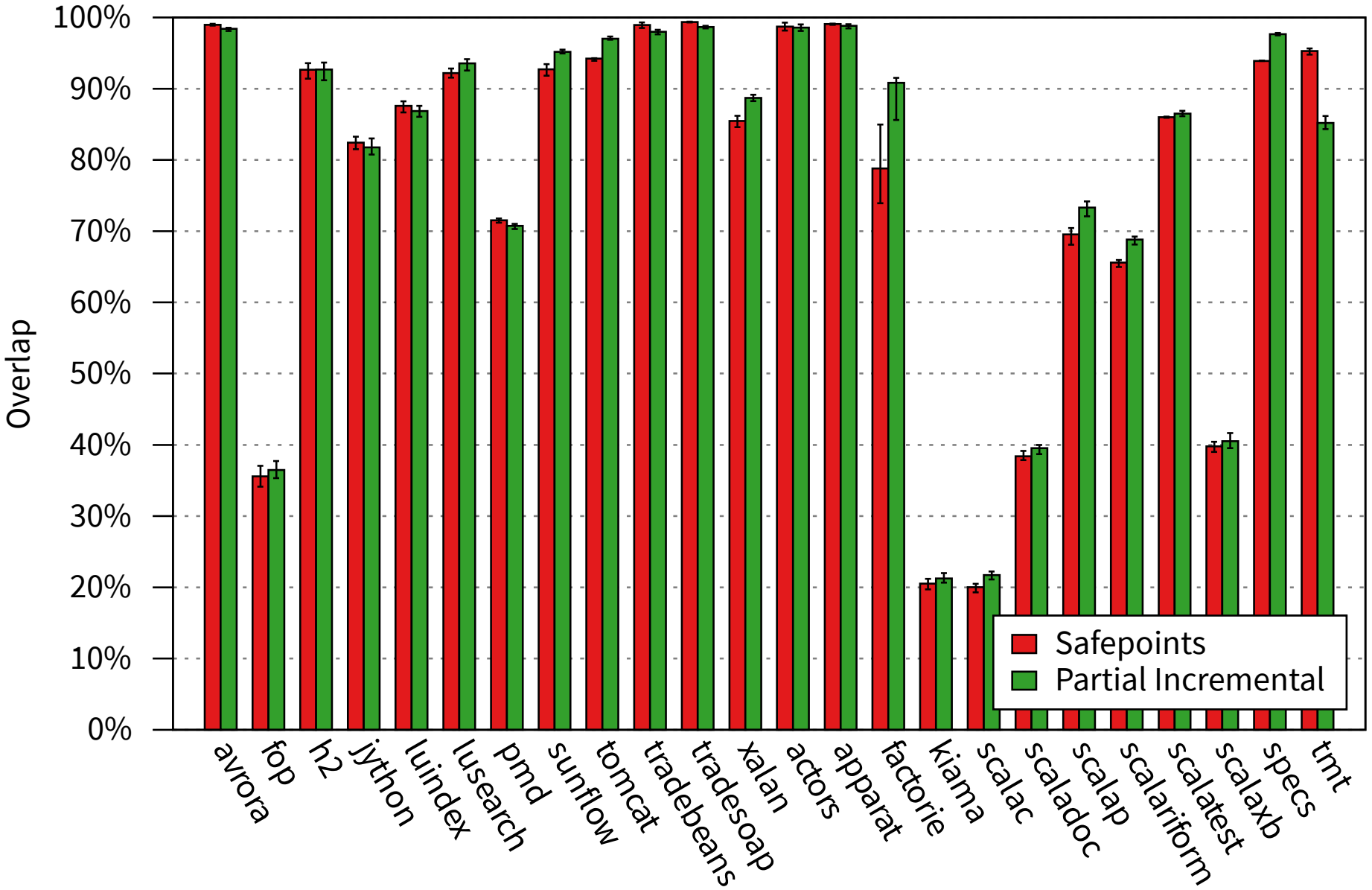
~~Compare to profile from instrumentation?~~

→ Stability and comparison to profile with safepoints

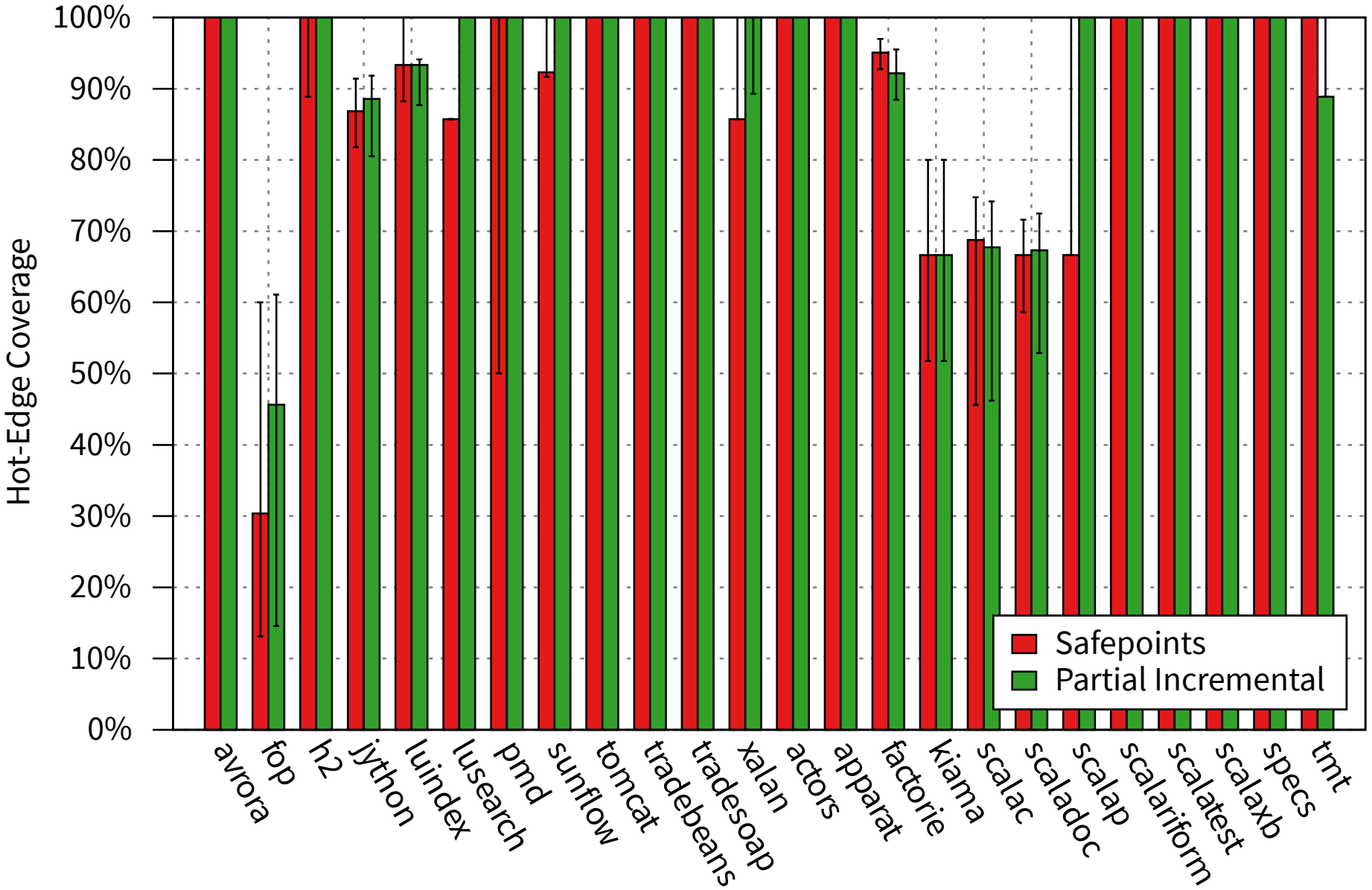
Method

- Collect profiles of multiple executions of a workload
- Merge into a single “average profile”
- Analyze:
  - compare individual profiles to avg profile
  - compare avg. profile to avg. profile with safepoints

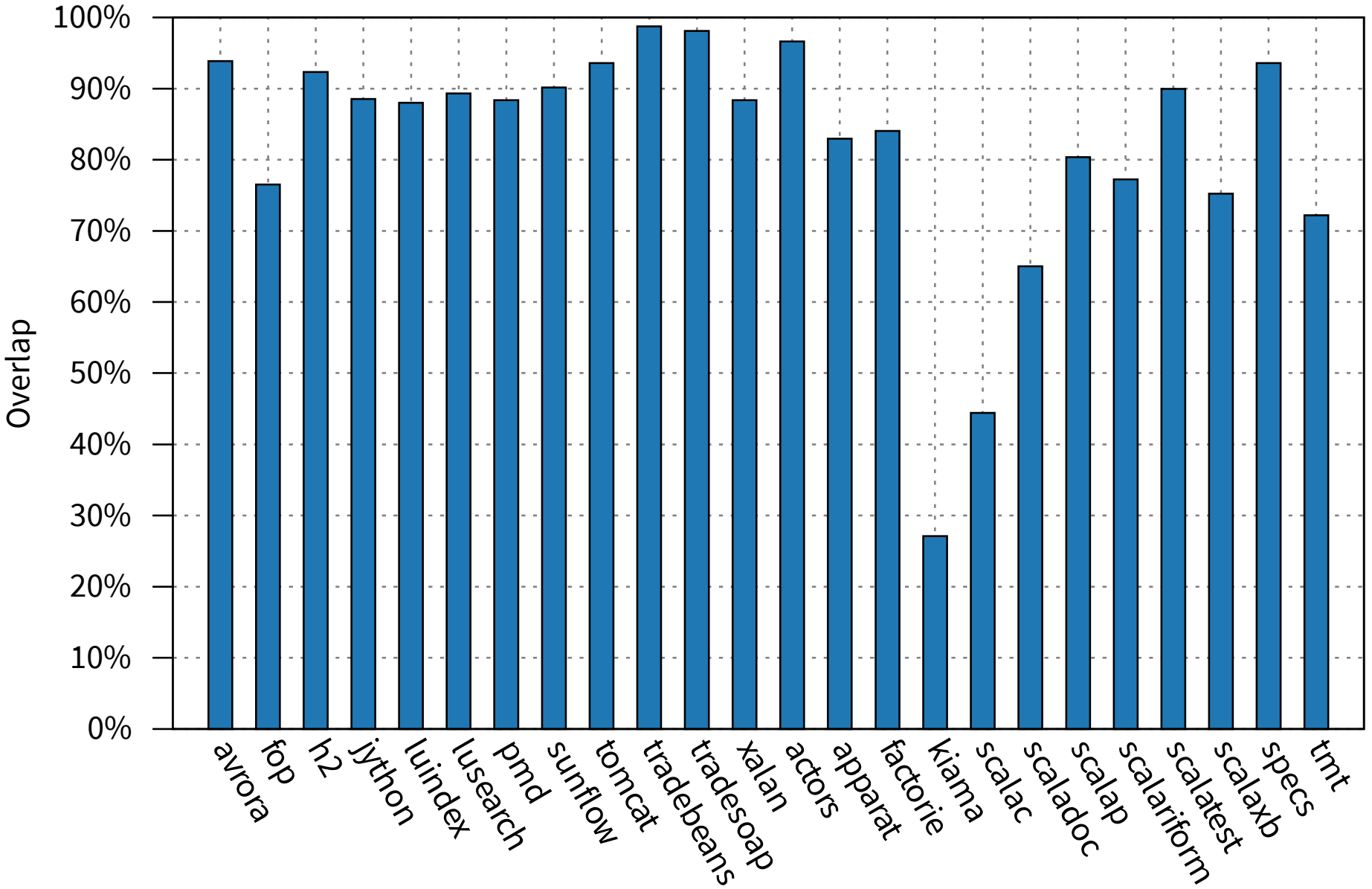
Stability


Overlap
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Safepoints
Partial Incremental
avrora
fop
h2
jython
luindex
lusearch
pmd
sunflow
tomcat
tradebeans
tradesoap
xalan
actors
apparat
factorie
kiama
scalac
scaladoc
scalap
scalariform
scalatest
scalaxb
specs
tmt

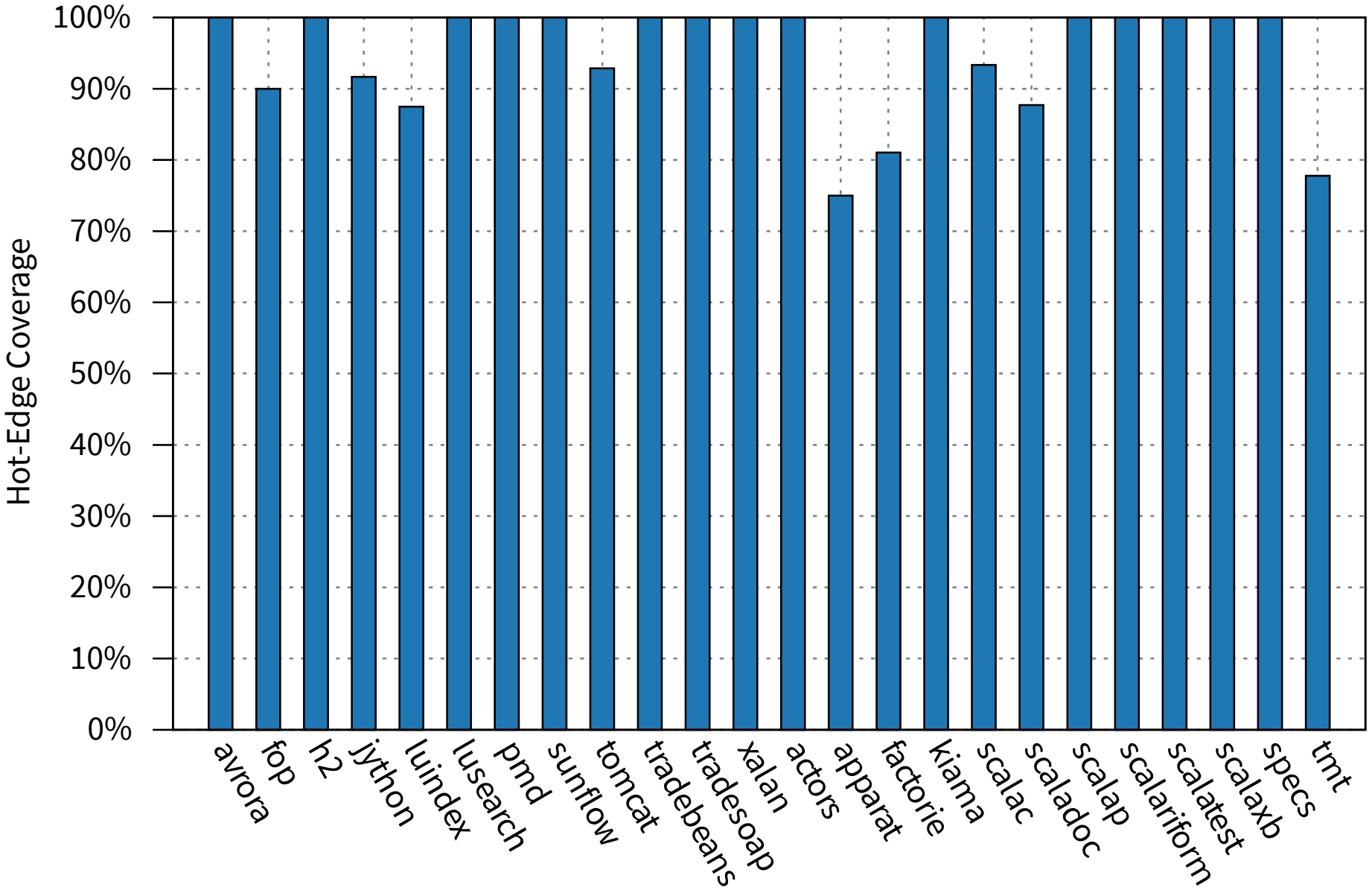
Stability: hot methods



Comparison


Overlap
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
avrora
fop
h2
jython
luindex
lusearch
pmd
sunflow
tomcat
tradebeans
tradesoap
xalan
actors
apparat
factorie
kiama
scalac
scaladoc
scalap
scalariform
scalatest
scalaxb
specs
tmt

Comparison: hot methods



Conclusion

- Techniques
  - Partial Safepoints
  - Self-Sampling
  - Incremental Stack Tracing
- Low overhead
  - without hardware or operating system support
- Short and predictable pause times
- Accuracy unaffected

Questions