1. Evaluation

This file simply typesets the benchmark results analyzed through R.

Measurement Setup. We measured how much time was required to construct indexes. For comparison, we also present the time needed for loading the bytecode processed:

Name	Elapsed time
Method name	6.57 ± 0.02
Exception handlers	7.48 ± 0.10
Instruction type	4.67 ± 0.05

Below we measure in all cases the pure query time. We discuss the runtime overhead of the optimizer itself separately below.

Base vs. reference implementation. We set out to compare SQUOPT to FindBugs. Therefore, we measured *startup performance* [Georges et al. 2007], that is the performance of running the queries only once, to minimize the effect of compiler optimizations.

Name	Elapsed time (s)	CPU time (s)
FindBugs	43.00 ± 0.57	90.18 ± 1.90
SQUOPT	22.88 ± 0.23	58.01 ± 1.46

Interpretative overhead and optimization potential. We present the results of our benchmarks in Table 2. We see that, in its current implementation, SQUOPT causes an interpretation overhead between 1x and 218.1x.

References

A. Georges, D. Buytaert, and L. Eeckhout. Statistically rigorous Java performance evaluation. In *Proc. Int'l Conf. Object-Oriented Programming, Systems, Languages and Applications*, OOPSLA '07, pages 57–76, New York, NY, USA, 2007. ACM.

Identifier	Description
PROTECTED_FIELD	Class is final but declares protected field
NO_CLONE	Class implements Cloneable but does not define or use clone method
SUPER_CLONE_MISSING	The clone method does not call super.clone()
NOT_CLONEABLE	Class defines clone() but doesn't implement Cloneable
COVARIANT_COMPARETO	Covariant compareTo() method defined
GC_CALL	Explicit garbage collection; extremely dubious except in benchmarking code
RUN_FINALIZERS_ON_EXIT	Method invokes dangerous method runFinalizersOnExit
COVARIANT_EQUALS	Abstract class defines covariant equals() method
FINALIZER_NOT_PROTECTED	Finalizer should be protected, not public
UNUSED_PRIVATE_FIELD	The value of a private field is not read
DONT_CATCH_IMSE	Dubious catching of IllegalMonitorStateException

Table 1. Implemented Analyses

Name	Base impl. (in ms)	Modular impl	Optimiz. time	IS	OS	OS-Opt	Performance (relati
SUPER_CLONE_MISSING	0.76±0.04	0.00±0.00	19.0±0.40	0.2x	0.1±0x	0±0x	
PROTECTED_FIELD	0.19±0.00	0.58±0.00	2.0±0.04	0.4x	0.8±0x	0.1±0x	
UNUSED_PRIVATE_FIELD	31.66±0.15	32.12±1.08	13.0±0.31	0.3x	0.4±0x	0.3±0x	
NO_CLONE	0.00±0.00	0.00±0.00	10.0±0.07	0.0x	0±0x	0±0x	
COVARIANT_COMPARETO	0.00±0.00	0.00±0.00	41.0±0.15	0.0x	0±0x	0±0x	
NO_SUITABLE_CONSTRUCTOR	0.00±0.00	0.00±0.00	13.0±0.07	0.0x	0±0x	0±0x	
GC_CALL	22.99±0.19	38.68±0.50	26.0±0.49	0.4x	16.9±1.3x	0.8±0x	
RUN_FINALIZERS_ON_EXIT	24.33±0.89	39.79±0.43	9.5±0.15	0.3x	45.5±4.5x	2.4±0.1x	
NOT_CLONEABLE	0.71±0.00	0.00±0.00	9.4±0.13	0.1x	9.7±0.1x	0.1±0x	
COVARIANT_EQUALS	0.71±0.00	0.00±0.00	9.7±0.18	1.0x	10.8±0.2x	0.1±0x	
FINALIZER_NOT_PROTECTED	0.59±0.01	2.09±0.00	5.2±0.16	0.7x	12.6±0.3x	0.1±0x	
DONT_CATCH_IMSE	14.74±0.29	0.00±0.00	7.0±0.05	1.0x	551.5±14.8x	2.1±0x	

Performance given as average±standard deviation in milliseconds; plot whiskers denote standard deviation
IS: interpretation slowdown for SQ (bigger is better) OS: optimization speedup for SQUOPT (bigger is better)
OS-Opt: optimization speedup, considering the optimization time (SQUOPT + Opt) (bigger is better)
The plot shows performance relative to the slowest performance: Base SQ SQUOPT

Table 2. Performance measurements (in ms)