On the Energy Cost of Static Analysis Precision

An Empirical Study of SpotBugs Effort Levels

Sophie van der Linden
Delft University of Technology

Xutong Liu
University of Twente

Andy Zaidman
Delft University of Technology

Carolin Brandt
Delft University of Technology

FSE Companion '26 · DevOpsSustain Workshop · July 2026, Montréal

Motivation

CI pipelines are everywhere, and they consume real energy

- **CI runs continuously.** Every commit triggers build, test, and analysis. Large teams run hundreds of pipeline runs per day.
- **Cumulative cost is substantial.** Prior work shows CI and automated testing consume considerable energy (Zaidman AST'24, Liu et al. Greenvolve'26).
- **But where does the cost come from?** Most studies treat the pipeline as a whole. The contribution of individual phases, like static analysis, is unknown.
- **Static analysis is configurable.** Unlike test execution, analysis depth is not correctness-constrained. Teams can deliberately trade precision for efficiency.

Research gap: the energy cost of configurable static analysis in CI remains largely unmeasured.

SpotBugs Effort Levels

Why SpotBugs?

Most frequently used SAT in our literature survey (2020–2025). Open-source Java tool, integrates with Maven & Gradle, offers a clean effort parameter with four discrete levels.

SpotBugs Effort Levels

Why SpotBugs?

Most frequently used SAT in our literature survey (2020–2025). Open-source Java tool, integrates with Maven & Gradle, offers a clean effort parameter with four discrete levels.

Effort Level	Additional Features Enabled
Min (1)	Conserves space. Minimal analysis depth.
Less (2)	+ Accurate exception flow analysis + Model instanceof in type analysis
More (3)	+ Track guaranteed null dereferences + Track recoverable null values | + Interprocedural analysis (application classes)
Max (4)	+ Interprocedural analysis across referenced classes (most computationally intensive)

Note: once a configuration is adopted, teams rarely change it (Beller et al. 2016), making these choices long-term.

Research Questions

RQ1 How does energy consumption vary across different effort levels of SpotBugs?

Establishes whether configuration depth has a measurable, statistically significant effect on energy use.

RQ2 How does the change in energy consumption relate to the number of reported warnings?

Evaluates the energy-efficiency trade-off: are additional findings from higher effort levels proportional to their energy cost?

Methodology

10 open-source Java projects × 4 effort levels × 30 runs each

Measurement setup

Tool	EnergiBridge (CPU energy via RAPL registers)
Hardware	Dell OptiPlex 7060, Intel i5 8th Gen, Windows 10
Protocol	30 runs · 5-min warm-up · 1-min cool-down
Outliers	Z-score filtering (|z| > 3 removed)
Isolation	Gradle: --rerun-tasks | Maven: spotbugs:spotbugs
Stats	Friedman test + Wilcoxon signed-rank + Holm correction

Subject projects

Gradle

Cruise-control, Elasticsearch, JUnit Framework

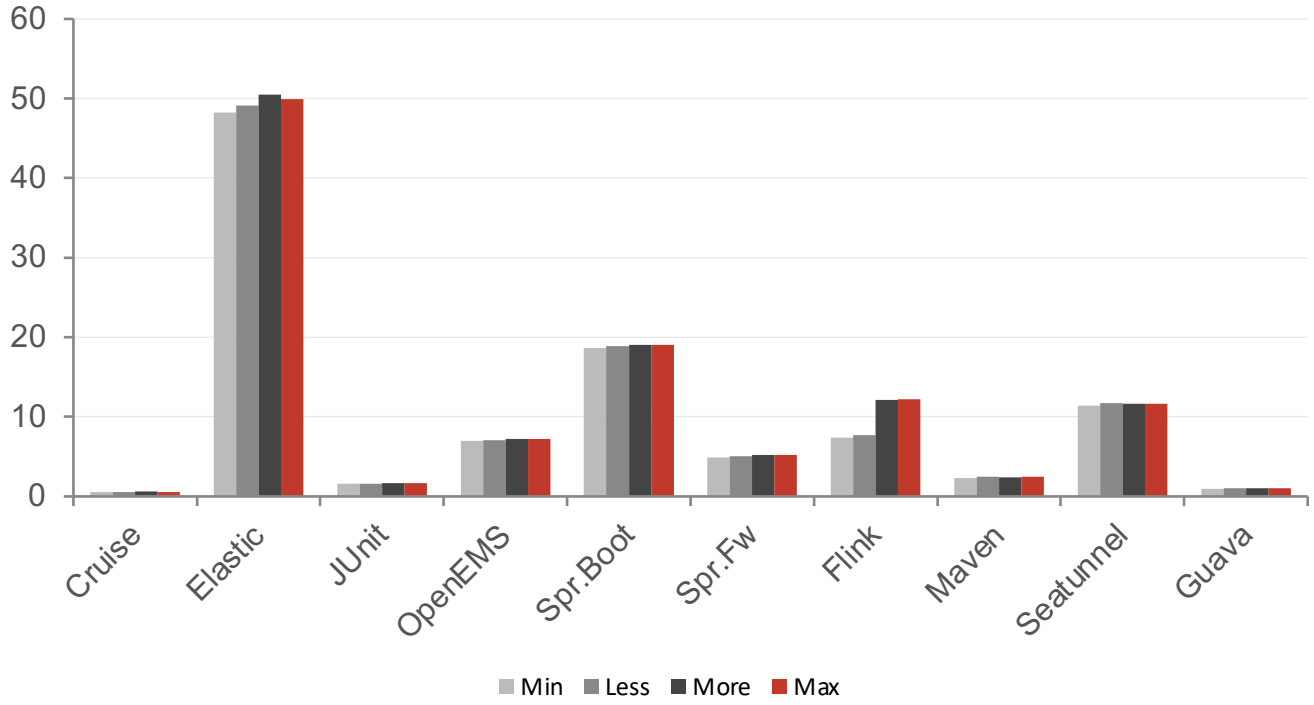
OpenEMS, Spring Boot, Spring Framework

Maven

Apache Flink, Apache Maven

Apache Seatunnel, Google Guava

RQ1: Energy Consumption per Effort Level



- ✓ All 10 projects: significant ($p < 0.01$)
- ✓ Largest ΔE: Min→Less, Less→More
- ✓ More→Max: often not significant
- ! Exception: Flink +65% at Less→More

RQ1: Where Are the Significant Differences?

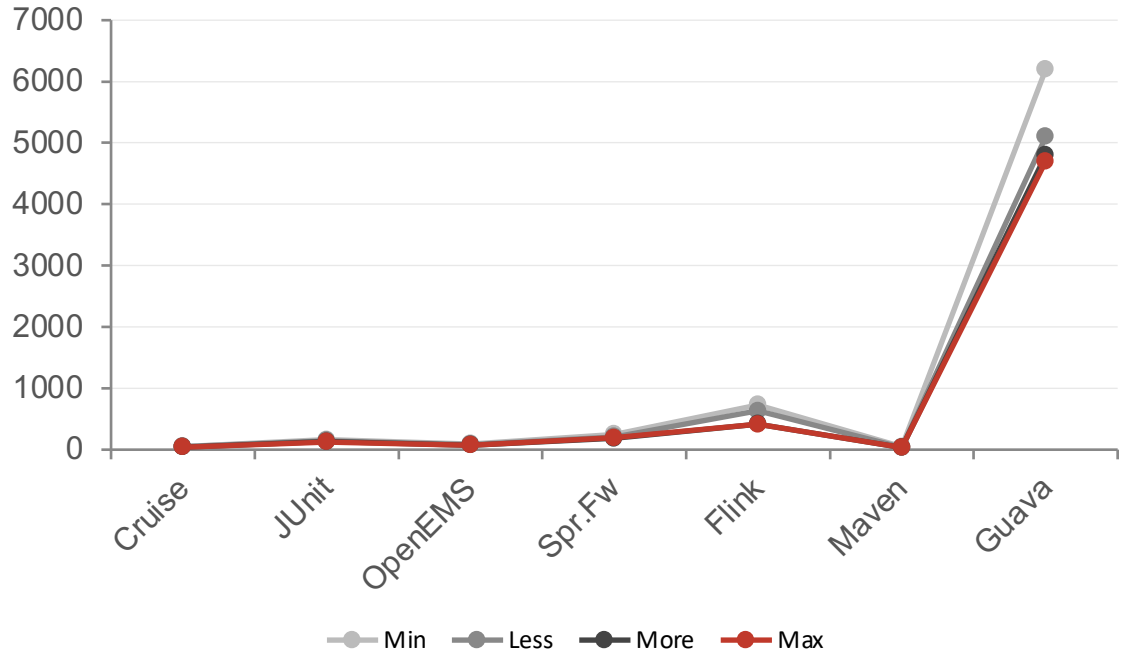
Holm-corrected Wilcoxon signed-rank p-values for adjacent effort comparisons

Project	Min → Less	Less → More	More → Max
Cruise-control	0.358	**0.036 ***	**0.036 ***
Elasticsearch	0.236	**0.009 ****	0.192
JUnit Framework	1.000	0.857	1.000
OpenEMS	**0.002 ****	**2.3e-8 *****	0.187
Spring Boot	**0.014 ***	0.234	0.851
Spring Framework	**0.013 ***	**7.7e-7 *****	0.413
Apache Flink	**6.4e-4 *****	**2.0e-33 *****	0.060
Apache Maven	**1.7e-11 *****	**3.4e-3 ****	0.311
Apache Seatunnel	**0.047 ***	1.000	1.000
Google Guava	**2.1e-7 *****	**2.0e-5 *****	**2.8e-6 *****

Bold = significant ($p < 0.05$). Red = highly significant ($p < 0.001$). Note: More→Max is rarely significant.

RQ2: Warnings per Unit of Energy

Higher effort = more warnings, but are they proportional to the extra energy cost?



Summary

— 9 of 10 projects: “Min” yields most warnings per Wh

— Higher levels produce more warnings in absolute terms

— …but the energy cost increases disproportionately

— Exception: Spring Framework: “Less” is most efficient

— Warning count does not distinguish true vs. false positives

Takeaways

1. Configuration depth has a real, measurable energy cost.

2. The largest energy increases occur at Min→Less and Less→More.

3. Min is the most energy-efficient configuration.

4. Static analysis tools should expose energy trade-offs.

Threats to Validity

Internal validity

– Measurement noise

– Mitigated by 30 repetitions, 5-min warm-up, 1-min cool-down, z-score filtering

– EnergiBridge measures CPU energy only

External validity

– Single Windows desktop machine, not typical for CI (usually Linux/containerized)

– Relative trends across effort levels expected to generalize; absolute values may not

Construct validity

– CPU energy ≠ full-system footprint (no memory, storage, network, cooling contribution)

– Results reflect energy vs. warning volume trade-off, not overall analysis quality

These are exploratory findings — not definitive configuration recommendations.

Threats to Validity

Internal validity

– Measurement noise

– Mitigated by 30 repetitions, 5-min warm-up, 1-min cool-down, z-score filtering

– EnergiBridge measures CPU energy only

External validity

– Single Windows desktop machine, not typical for CI (usually Linux/containerized)

– Relative trends across effort levels expected to generalize; absolute values may not

Construct validity

– CPU energy ≠ full-system footprint (no memory, storage, cooling contribution)

– Results reflect energy vs. warning volume trade-off, not overall analysis quality

Future work

- Fine-grained feature attribution
- True positive rate per Wh (labelled datasets)
- Other SATs (PMD, Checkstyle, SonarQube)
- Linux CI server environments

These are exploratory findings — not definitive configuration recommendations.

The Team

Sophie van der Linden

BSc Student · TU Delft

Andy Zaidman

Full Professor · TU Delft

Xutong Liu

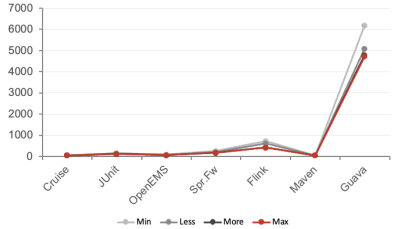
Postdoc · University of Twente

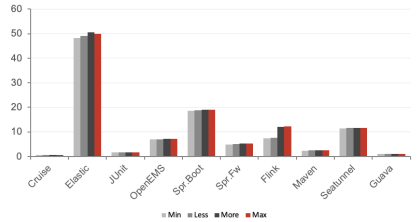
Carolin Brandt

Assistant Professor · TU Delft

Thank you.

Questions & Discussion





On the Energy Cost of Static Analysis Precision: An Empirical Study of SpotBugs Effort Levels

van der Linden · Liu · Zaidman · Brandt · FSE Companion '26

Replication kit: doi:10.5281/zenodo.18516019