Compression of Programs and the Similarity Distance

KIREPRO1PE Research Project, MSc. Computer Science, ITU - 10th of June 2025

Jonas Nim Røssum <jglr@itu.dk>

Background

- *Lines of Code Changed* (LoCC)
  - De facto standard for measuring code changes
  - Has it’s limitations (e.g. structural changes, formatting changes, etc.)

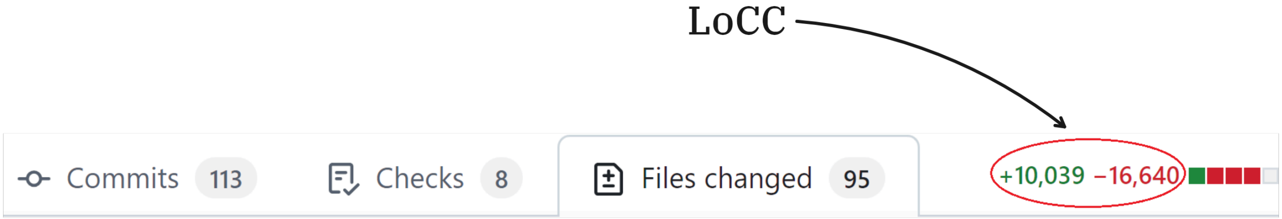


Figure 1: LoCC in a GitHub Pull Request

Project goal and findings

- Find a new metric to address limitations of *Lines of Code Changed* (LoCC)
- *Difference in Compression Distance* (ΔCD)

Research questions

? Is ΔCD correlated with LoCC?

? Can ΔCD discriminate between commit types?

? What are the advantages / limitations of ΔCD?

Findings

→ Partial linear correlation, $R^2 = \{0.8, 0.7\}$

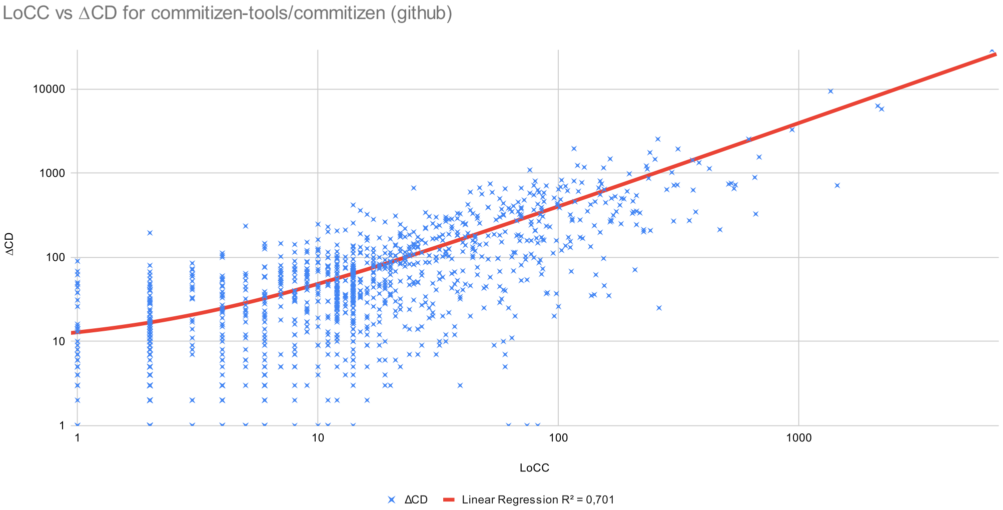
→ For Commitizen[1] repo, features and bug fixes stand apart

→ Robust to structural changes, survivorship bias / 250× slower than LoCC, scaling challenges

[1] https://github.com/commitizen-tools/commitizen/

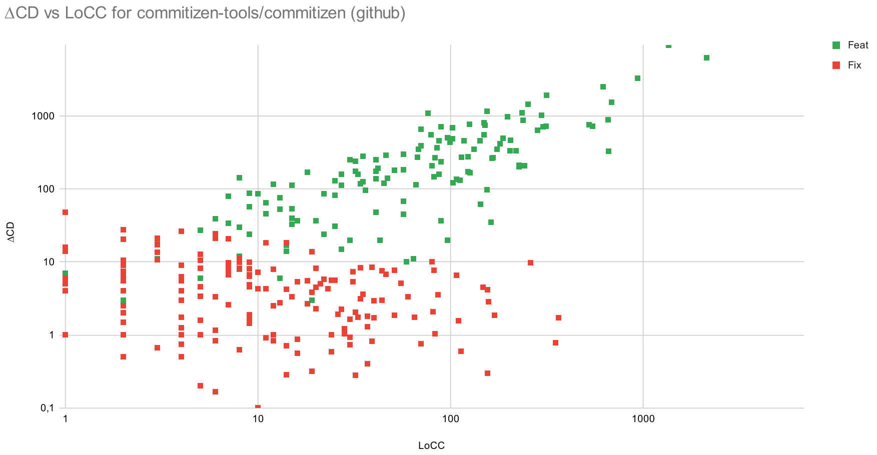
RQ1: ΔCD correlation with LoCC

Linear regression R^2 for **commitizen**: 0.7



✅ ΔCD and LoCC **partially correlate** → ΔCD captures more than raw line changes

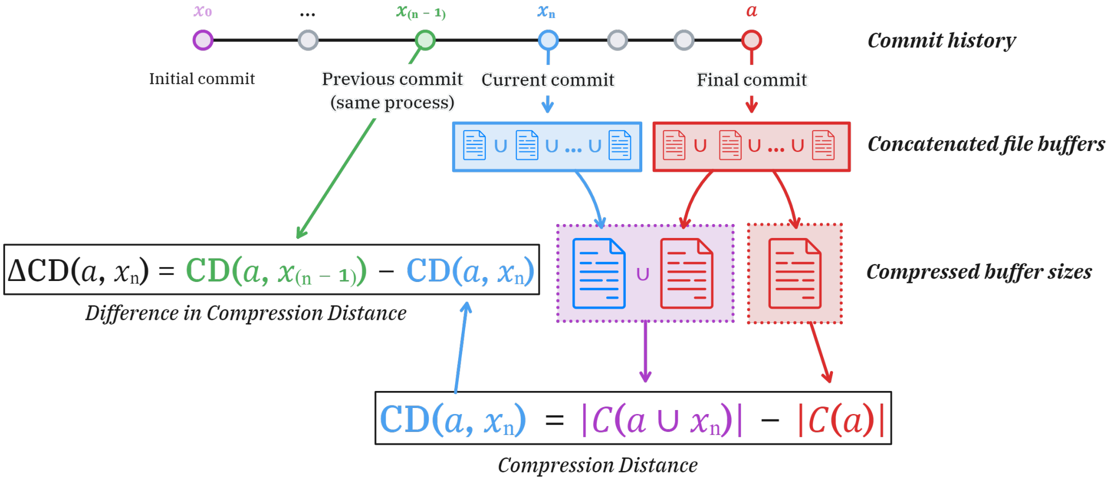
RQ2: Commit Type Discrimination



Bug Fixes: lower ΔCD, changes to existing code **Features**: higher ΔCD, typically novel code

✅ ΔCD can partly **discriminate** between some **commit types**, at least for this project

RQ3: Robustness to structural changes



x_0 … $x_{(n-1)}$ x_n a

Commit history

Initial commit

Previous commit (same process)

Current commit

Final commit

Concatenated file buffers

Compressed buffer sizes

$$\Delta\mathrm{CD}(a, x_n) = \mathrm{CD}(a, x_{(n-1)}) - \mathrm{CD}(a, x_n)$$

Difference in Compression Distance

$$\mathrm{CD}(a, x_n) = |C(a \cup x_n)| - |C(a)|$$

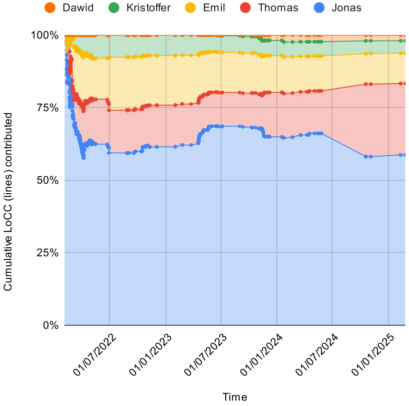
Compression Distance

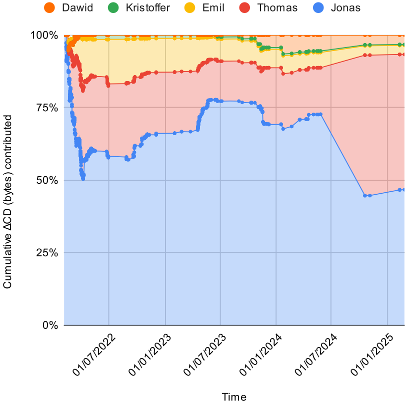
Figure 4: ΔCD (Difference in Compression Distance) Explained

✅ ΔCD is insensitive to **project structure** at commit granularity

RQ3: Survivorship Bias

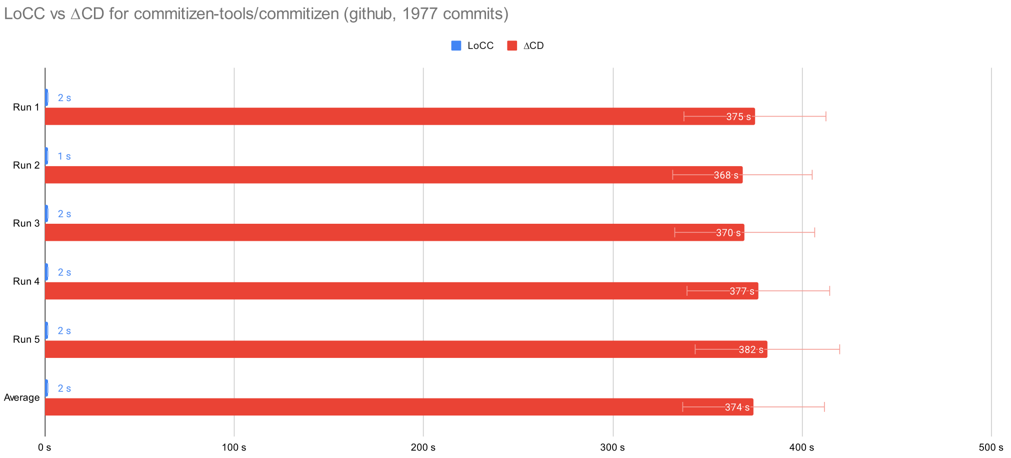
- Example: **Thomas’ thesis work in Git Truck**
- According to LoCC (left), Thomas is responsible for 25% of the contributions project
- According to ΔCD (right), Thomas is responsible for 46% of the final revision





✅ ΔCD reflects **lasting impact** on the codebase using survivorship bias

RQ3: Performance and Scalability



LoCC vs ΔCD for commitizen-tools/commitizen (github, 1977 commits)

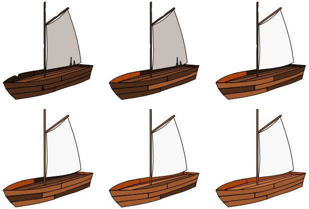
Future work

Performance and scalability

Generalize findings

Robustness to formatting changes etc.

Thank You - Questions?



Project work: Jonas Nim Røssum <jglr@itu.dk>

Original idea: Christian Gram Kalhauge <chrg@dtu.dk>

Source code: github.com/git-truck/git-truck