

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 its limitations (e.g. structural changes, formatting changes, etc.)

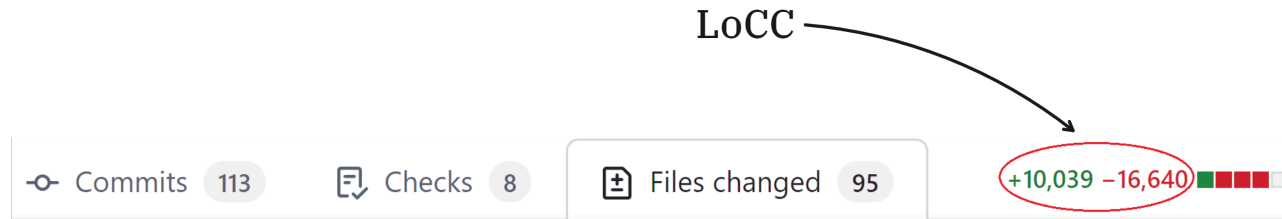


Figure 1: LoCC in a GitHub Pull Request

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

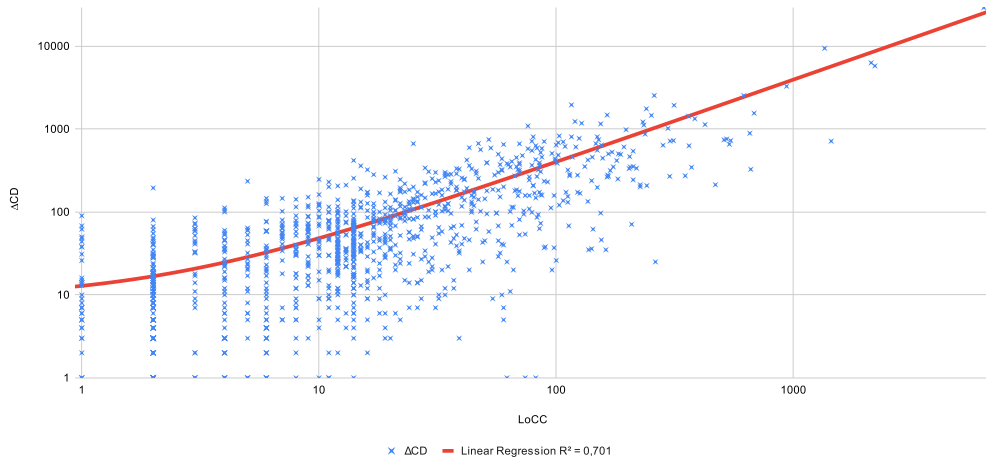
- Partial linear correlation, $R^2 = \{0.8, 0.7\}$
- For Commitizen¹ repo, **features** and **bug fixes** stand apart
- Robust to structural changes, survivorship bias / 250× slower than LoCC, scaling challenges

¹<https://github.com/commitizen-tools/commitizen/>

RQ1: Δ CD correlation with LoCC

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

LoCC vs Δ CD for commitizen-tools/commitizen (github)



✓ Δ CD and LoCC **partially correlate** → Δ CD captures more than raw line changes

RQ2: Commit Type Discrimination

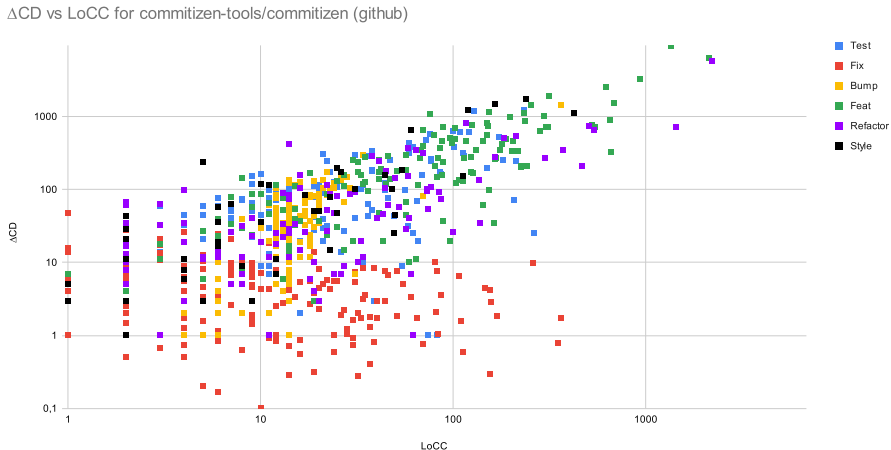
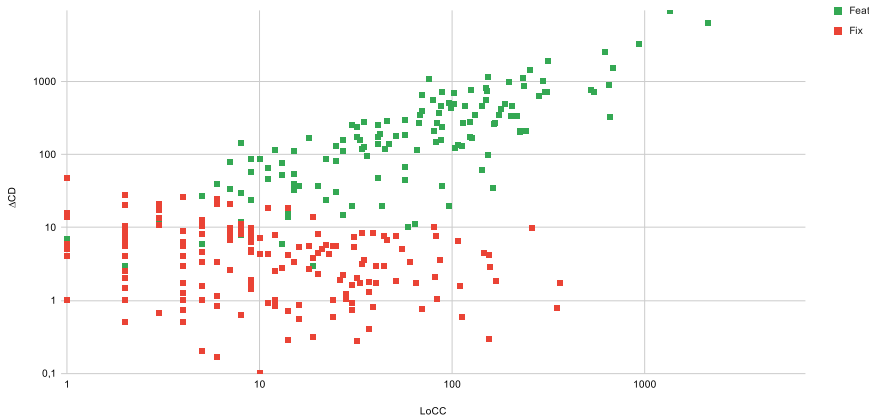


Figure 3: Commits in the Commitizen repository categorized using conventional keywords²

²<https://www.conventionalcommits.org/en/v1.0.0/>

RQ2: Commit Type Discrimination

Δ CD vs LoCC for commitizen-tools/commitizen (github)



Bug Fixes: lower Δ CD, changes to existing code

Features: higher Δ CD, typically novel code

✓ Δ CD can partly **discriminate** between some **commit types**, at least for this project

RQ3: Robustness to structural changes

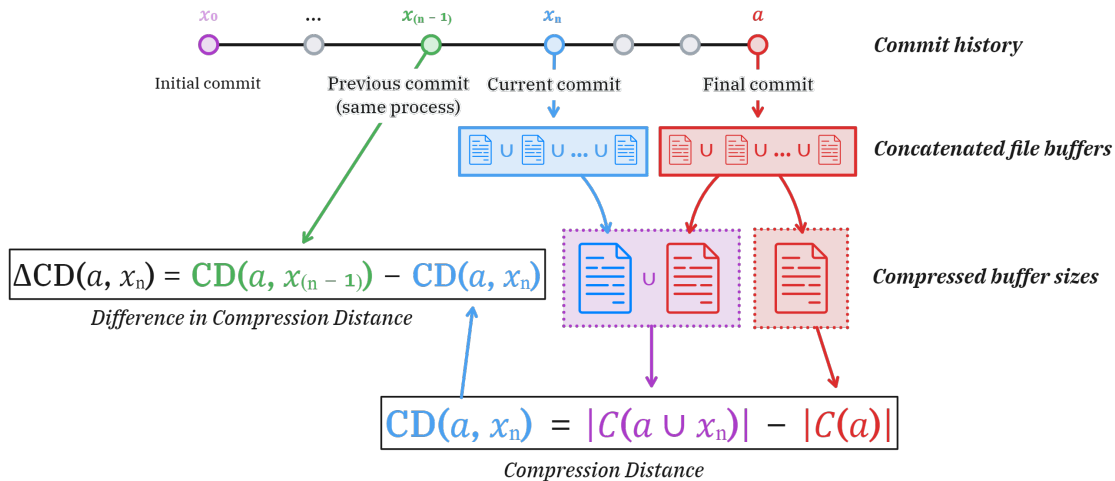
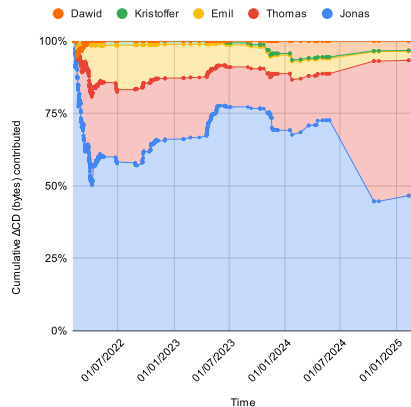
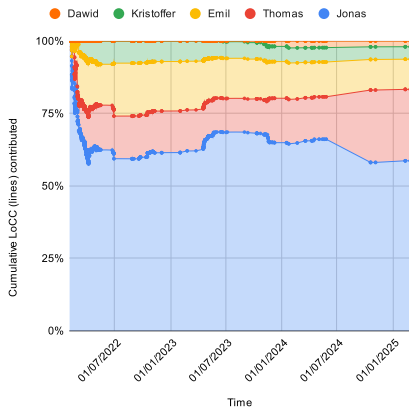


Figure 5: ΔCD (Difference in Compression Distance) Explained

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

RQ3: Survivorship Bias

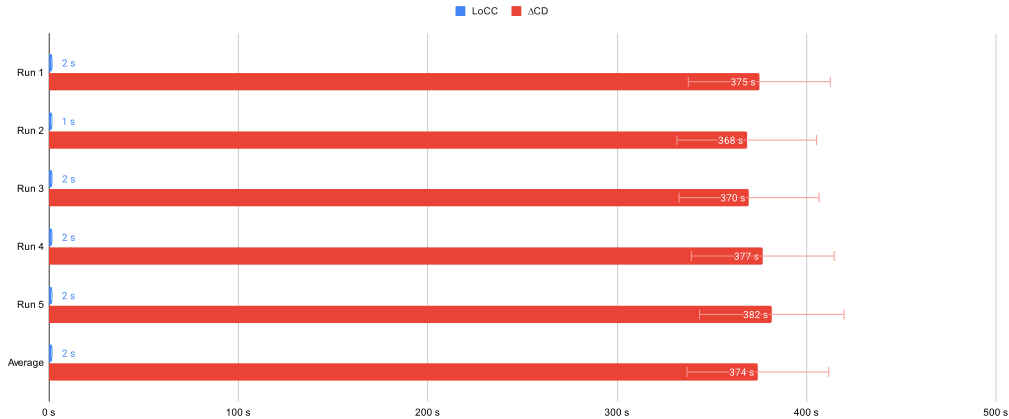
- Example: **Thomas' thesis work in Git Truck**
- According to LoCC (left), Thomas is responsible for 25% of the contributions project
- According to Δ CD (right), Thomas is responsible for 46% of the final revision



✓ Δ CD reflects **lasting impact** on the codebase using survivorship bias

RQ3: Performance and Scalability

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



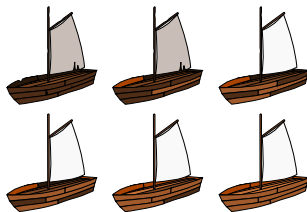
Future work

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