SmartCommit: A Graph-based Interactive Assistant for Activity-Oriented Commits

Supplementary Material

ANONYMOUS AUTHOR(S)

1 FOR SECTION 2

Questions asked in the motivating user study:

- How do you perceive the often stated best practice of submitting task-oriented commits?
- Do you think it is necessary or beneficial to only submit task-oriented commits?
- What clues or information are you looking for when decomposing larger changes?
- For the examples from the observation phase, why did you choose to commit a single large composite change back then?
- What features would an ideal tool have that helps with submitting cohesive commits?

Key insights in the motivating user study:

Table 1: The collected insights from 6 experienced software developers

Topic	Collected insights	Short Summary	Selected Supporting Quotes
Benefits	Easy to understand Easy to describe Safe to revert Easy to integrate	All participants agreed that activity -oriented commits are necessary for software development, maintenance and collaboration.	*"I think small commits are easier to understand, both for me and other colleges."* (P1) *"When using git-bisect to locate bugs, it will be good if the buggy commit is small to safely revert."* (P3)
Causes	Interleaved activities No regulations or guidelines Lack of tool support Time pressure	Only 2 out of 6 participants knew this best practice previously, but they rarely followed it in their daily work.	*"I often find and fix a bug by the way when doing other things."* (P1) *"Our team have regulations about code style but nothing about commit style."* (P2)
Clues	Newly added *def* and *use* Systematic edits Framework structure Optimization/improvement	The dependency between definitions and references was the most frequently used clue, participants also mentioned similar and refactoring changes.	*"New methods or classes often indicate new feature or refactoring."* (P4) *"To fix a buggy condition in if/for/while, I am used to also check other similar ones."* (P6)
Expectations	Informative GUI Intuitive operation Easy to check and adjust Quick in response	All participants expected an intuitive tool for the activity-oriented commits, but anticipated it is hard and unsafe to cover all situations and fully automate the process.	*"I expect the tool to save my time spent on typing Git commands repeatedly."* (P6) *"A few errors are tolerable as long as I can correct them quickly."* (P5)

2 FOR SECTION 4.2.1

Detailed workflow of SmartCommit as Intellij IDEA plugin:[1]

With SmartCommit-plugin installed in Intellij IDEA, it can be invoked on the menu bar or the right-context menu on the project folder. The plugin invokes SmartCommit-core jar by passing the absolute path of the current repository as the argument:

(1) Beginning to analyze the workspace in the background, while the plugin shows a progress bar for the user.
(2) When the analysis is done, it saves some intermediate data in a hidden folder under the user's home directory.
(3) The plugin automatically opens a web GUI in Chrome browser, which is similar with the GUI of the public version.
(4) The user makes adjustments on the suggested groups until deciding to submit or quit.

[1] While the core is open-source, the engineering clients (like the GUI and plugin) are not public due to commercial restriction.

(5) The user can choose part or all of the groups, complete the description and click *Commit* button to submit the selected groups as commits, in the order from left to right.
(6) The successfully submitted groups will disappear from the GUI. The user can continue submitting or click Exit button to close the web page and the plugin.

3 FOR SECTION 3.4.1

Definition and measurement of supported links (Table 1):

Table 1. Link Detection and Weight Calculation Rules

Category	Type	Link Detection and Weight Calculation
Hard Links	Direct dependency	$\exists(h_i, h_j, 1) \mid h_i.file\_type == h_j.file\_type == Java \land \exists(v_i, v_j) \in DependenceGraph, wt = 1.0$
	Indirect dependency	$\exists(h_i, h_j, w) \mid h_i.file\_type == h_j.file\_type == Java \land \nexists(v_i, v_j) \land \exists isReachable(v_i, v_j), wt = 1/min\_hop$
Soft Links	Similarity	$\exists(h_i, h_j, w) \mid h_i.file\_type == h_j.file\_type, wt = (text\_sim(h_i, h_j) + tree\_sim(h_i, h_j)) * 0.5$
	Proximity	$\exists(h_i, h_j, w) \mid h_i.file\_type == h_j.file\_type == Java, wt = 1/max(distance(v_i, LCA, distance(v_j, LCA)$
Refactoring Links	Refactoring	$\exists(h_i, h_j, 1) \mid h_i.file\_type == h_j.file\_type == Java \land h_i, h_j \subseteq same\_refactoring, wt = 1.0$
Cosmetic Links	Reformatting	$\exists(h_i, h_j, w) \mid removeWhiteChars(h_i.base) == removeWhiteChars(h_i.current) \land removeWhiteChars(h_j.base)$ $== \mathrm{removeWhiteChars}(\mathrm{h}_j.current), wt = (text\_sim(h_i.base, h_i.current) + text\_sim(h_j.base, h_j.current)) * 0.5$
	Textual moving	$\exists(h_i, h_j, w) \mid h_i.file\_index == h_j.file\_index \land (h_i.base == h_j.current \lor h_i.current == h_j.base),$ $\mathrm{wt}{=}\lvert \mathrm{h}_i.current.startline - h_j.current.startline \rvert / h_j.current.file.linenumber$
	Cleaning up	$\exists(h_i, h_j, 1) \mid h_i.file\_type == h_j.file\_type == Java \land h_i.current == h_j.current == \emptyset \land indegreeOf(v_i)$ $== \mathrm{indegreeOf}(\mathrm{v}_j) == 0, wt = 1.0$

DependenceGraph: a graph that combines the AST, data dependency, and call graph from the base and current versions of all changed files respectively.

4 FOR SECTION 4.3.3

Questions asked in the focus group interview:

- How do you think the initial decomposition suggested by *SmartCommit*?
- When do you think *SmartCommit* is useful and necessary and when it is replaceable?
- How do you usually adjust the decomposition with the operations of *SmartCommit*?
- Why do you often/seldom use *SmartCommit*, and what improvements do you think would make you use it more?
- Have you ever terminated *SmartCommit* or abandoned it for unbearable waiting time?

5 FOR SECTION 4.3.3

Run time distribution with the number of diff hunks and changed lines (Figure 1):

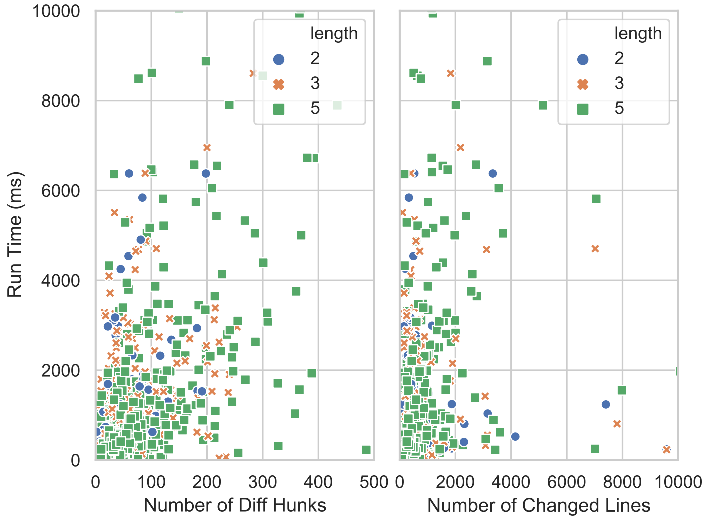

Fig. 1. Run time distribution with the number of diff hunks and changed lines.