Many software projects incorporate bug repositories so that developers, testers, and users can report bugs that they have encountered, and call for more useful features or make suggestions for revision.

This form of testing is asynchronous and loosely organized, due to a project’s reliance on various developers, testers, and users.

The cost of bug reporters’ searching the repository (to determine if their problem has been reported) can be higher than the cost of creating a new bug report. We suggest automatically retaining a small subset of existing bugs.

To avoid the same bug being addressed by multiple bug fixers, it is necessary for a triager to inspect each submitted bug report to determine whether it is a duplicate. However, such inspect efforts can be very expensive or ineffective.

Browser-Closing Bug:
- Bug-260331: After closing Firefox, the process is still running. Cannot reopen Firefox after that, unless the previous process is killed manually.

Document-Contain-No-Data Bug:

Quantity of existing bug reports:
- The large number of existing bug reports makes it challenging for the triager to examine all existing bug reports to detect duplication.

To address the challenge, the triager can retrieve a small subset of automatically suggested bug reports and compare the new bug report with each retrieved bug report to see whether the new bug report is a duplicate.

Quality of the list of suggested duplicate bug reports:
- Existing approaches adopt info-retrieval techniques to measure the similarity between bug reports using NL info. Although these approaches already provide some practical help triagers, there is still a need to improve these approaches due to their low recalls.

To address the challenge, both NL info and exec info can be mined for improving detection of duplicate bug reports.

We used the bug repositories of two large open source projects: Eclipse and Firefox to evaluate different combinations of parameters:

- Ways of using the NL info from each bug report: sum, sum+des, and 2sum+des.

Heuristics for retrieving potential target reports using three kinds of similarities:
- Bheur: a combined similarity of NL-S and E-S by calculating the arithmetic average of NL-S and E-S.
- CBHeur: a combined similarity of NL-S and E-S based on distinguishing which kind of info source is the dominant factor.
- RCBHur: a variant of CBHeur, where exec-info-dominant bug reports are ranked higher than NL-dominant bug reports.

Three upper sub-figures: if we fix the parameter of how we weight the NL info, CBHeur always outperforms the other two heuristics (BHeur and RCBHur).

Three lower sub-figures: if we fix a specific heuristic in bug-report retrieval, neither way of using the NL info always outperforms the other two heuristics.

Right two figures: compared with the best performance of approaches using only NL info, our calibrated approach (with the CBHeur and using only the summary) leads to an increase of 11-20% and an increase of 18-26% in recall rates on the two experimental bug-report sets, respectively.