Analysis and prediction of bug duplicates in KDE bug tracking system

Gregor Leban
IJS

Agenda

* Why be interested in this problem
* ***What*** is the problem
* ***Why*** it is a problem
* How we can not solve it
* How we tried solving it
* What we will try in the future

Why are we interested in this problem

* ALERT EU project 

* Aim is to improve the overall bug resolution process in Open Source environments
* Goal is to build tools that will:
  * Improve the overview of the project
  * Improve the communication between the developers
  * Suggest people who can solve a bug
  * Detect bug duplicates in BTS

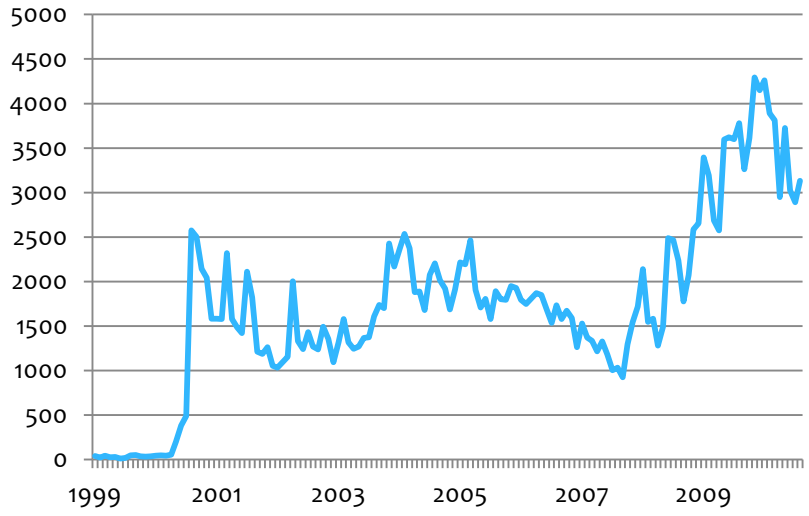
What is the problem

* What are bug duplicates?
* Bug tracking systems are web apps that users of some software can use to report detected issues
  * Bugzilla, Mantis, LaunchPad
  * Demo
* When reporting a bug users don‘t check if the same bug has already been reported
* If they create a bug report that is describing the same problem as some previous bug report they created a *bug duplicate*

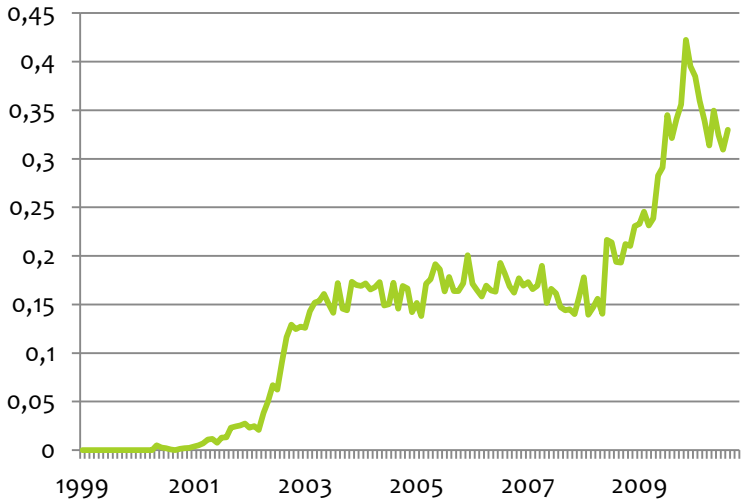
Why are bug duplicates a serious problem

* KDE is a use-case partner on ALERT
  * > 1.800 developers, >6M lines of code
* KDE Bugzilla has **>250.000 bug reports**
* **50.000 bug reports are duplicates**

Number of bug reports per month



Ratio of duplicates vs. all reports



Why are bug duplicates a serious problem (2)

* Similar problem in other large projects:
  * Firefox: **38%** of reports are duplicates
  * Ubuntu: **62%** of reports are duplicates
* Processing bug reports requires a lot of time.
* Bug triagers have to identify if a bug is a duplicate or not
* A few minutes per bug × thousands of bugs = enormous human effort

Importing the data

* To analyse the data we imported it into Contextify
* From each bug report we constructed one document
* Text of the document = bug description + comments
* Documents are stored in the BOW form. We removed the stop words and applied the Porter stemmer
* Along with the BOW we also stored bug‘s meta information (time, who posted it, what product/component it was assigned to, etc.)

Predicting if a bug report is a duplicate

* A temptation: treat the problem as a classification task
* Learning examples are all existing bug reports
* Attributes are the words in the report + meta information
* What is the class?
* 1st option:
  * Binary classification problem: duplicate / non-duplicate
  * Bad because:
    * Each duplicate is a duplicate because of its specific relation to **one** of the reports
    * The bug duplicates don‘t have any inherent property that would separate them from the non-duplicates.
  * If somebody is not convinced: CA: 58%, Prec:31%, Rec: 42%

Predicting if a bug report is a duplicate (2)

* 2nd attempt at classification:
  * Create a class value for each bug report
  * Bad because:
    * Very high number of classes
    * No ability to generalize

If not classify, what can we do?

* Rank
* Given a bug report d_i rank the reports based on likelihood of them being a duplicate of d_i
* One way to compute the ranking is to rank the reports based on their **similarity** with the report d_i
* We don‘t decide if a bug is a duplicate or not, but the user can check the top *n* ranked reports and decide

Computing similarity between reports

* We first weight the terms in the documents using the TF-IDF weighting scheme:

$$TF_{i,j} = f_{i,j} \qquad IDF_i = \log \frac{N}{n_i}$$

$$w_{i,j} = TF_{i,j} \times IDF_i$$

- Similarity between reports d_i and d_j is computed as cosine similarity:

$$\overrightarrow{d_i} = [w_{1,i}, w_{2,i}, \dots, w_{M,i}] \qquad \overrightarrow{d_j} = [w_{1,j}, w_{2,j}, \dots, w_{M,j}]$$

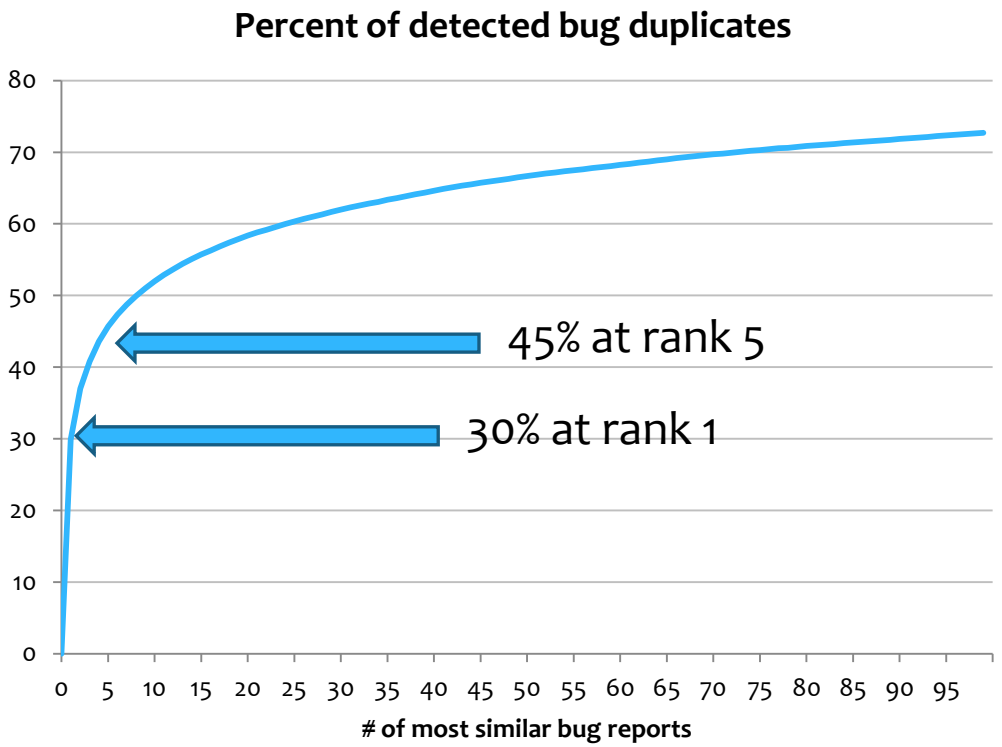
$$sim(\overrightarrow{d_i}, \overrightarrow{d_j}) = \frac{\overrightarrow{d_i} \cdot \overrightarrow{d_j}}{|\overrightarrow{d_i}| \times |\overrightarrow{d_j}|}$$

Are duplicated reports really similar?

* Question: how well can we identify duplicates by ranking the reports by similarity?
* Experiment:
  * For each bug report that has a duplicate we created a ranked list of 100 most similar bug reports
  * If the duplicated reports have similar content we can expect that the duplicate will be ranked high in the list

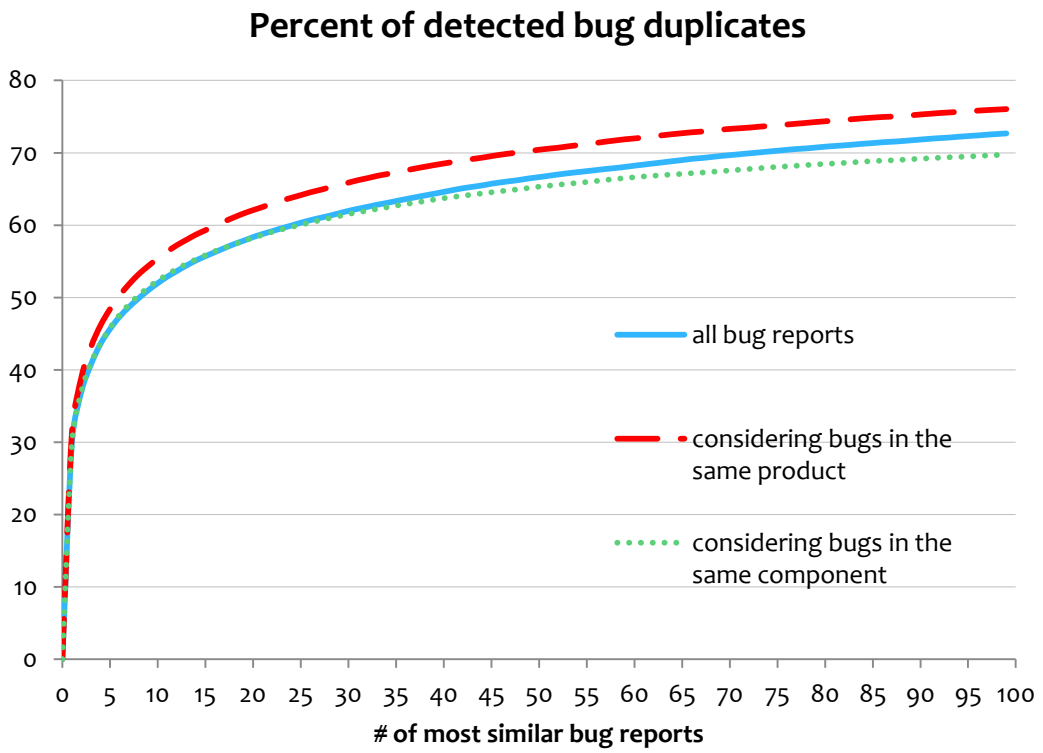
Results (graphical)

* What percent of bug duplicates can we detect by checking *n* highest ranked reports



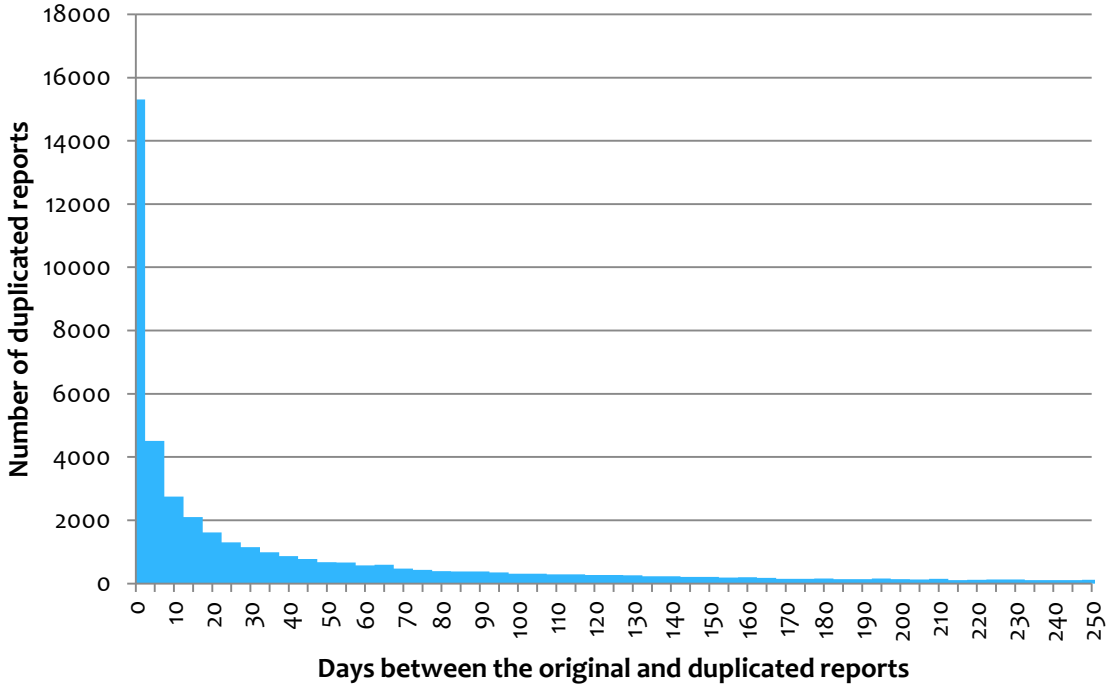
Improving ranking using meta-data

* Bugs are assigned to a product and a component
* What if we considered in ranking only bugs assigned to the same product/component?
* Not so impressive because users can assign the wrong value
* 10% of duplicates assigned the wrong product
* 25% of duplicates assigned the wrong component



Improving ranking using meta-data (2)

* Observation: Original bug reports and the duplicates are often very near in time
* Idea: change the ranking based on the time difference



Future work

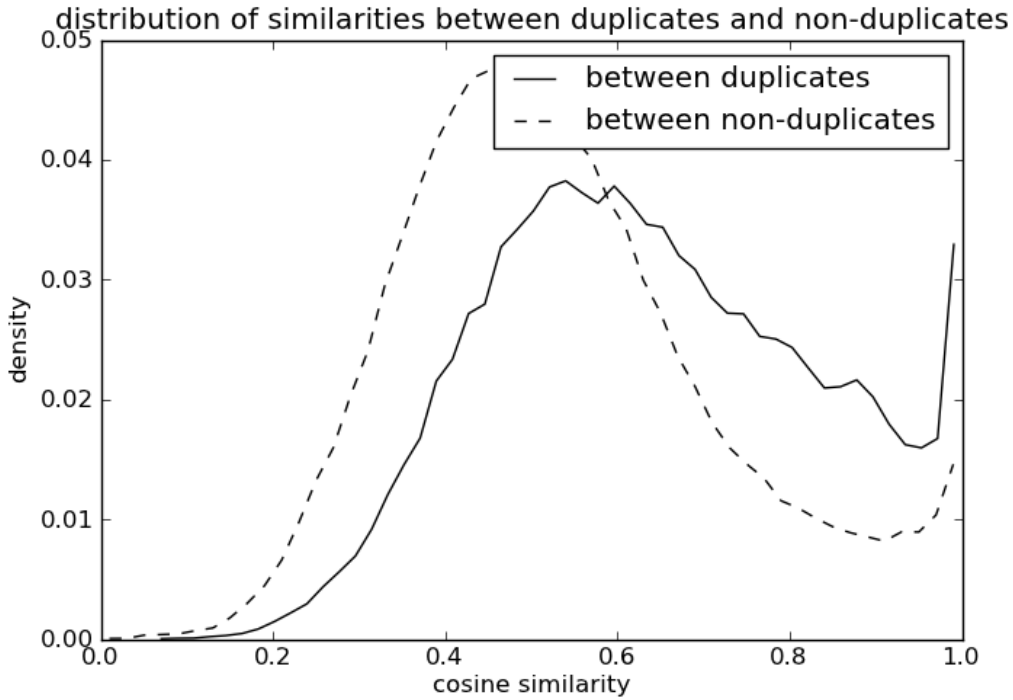
* Using meta information
* Computation of similarities based on concepts not on words
  * Synonym words are represented as a single concept
  * Similarity between reports will be computed based on the contained concepts
* Considering using relational learning

Results

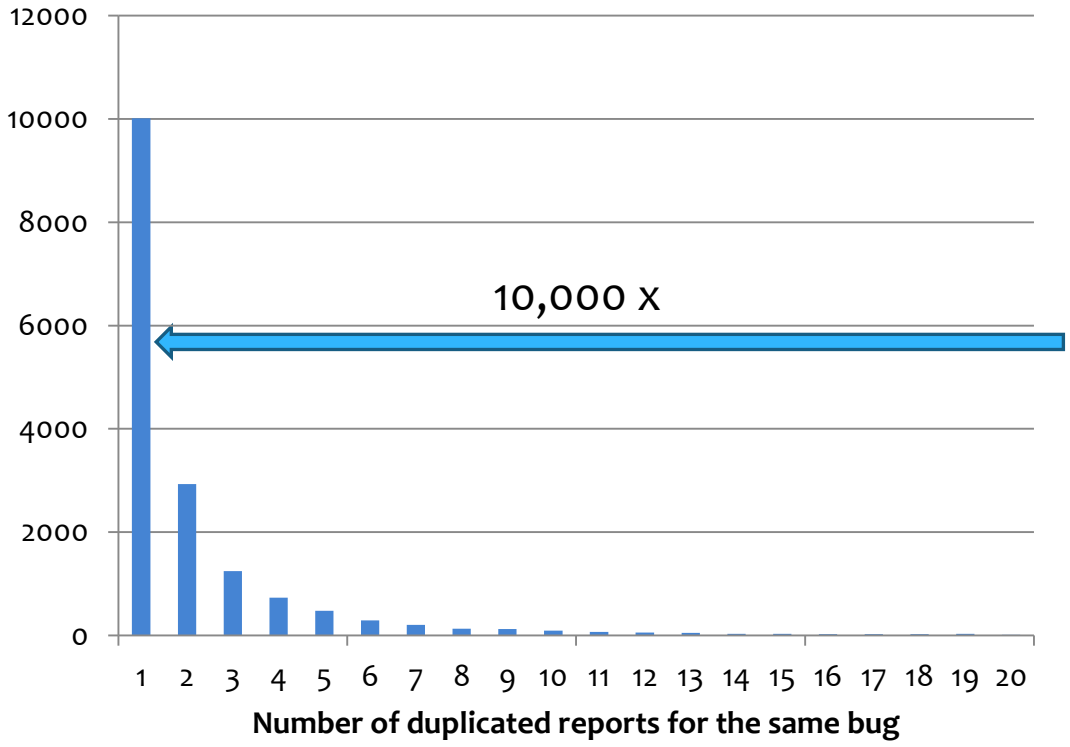
* What evaluation metric to use?
* Precision and Recall are not appropriate for our task
  * There is typically only one correct result (duplicate)
* Our task is similar to the task of question answering
* Typical metric in that domain is mean reciprocal rank (MRR)
* Reciprocal rank is the inverse of the rank of the first correct answer
* $MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$
* **Obtained MRR: 0.374**

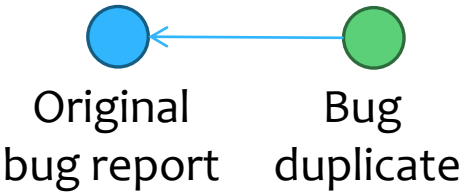
Classifying reports based on their similarities

* Question: How well can we use the similarities to discriminate between duplicates and non-duplicates?
* No treshold that could separate groups
* Possibility of false negatives (duplicates not marked as duplicates)

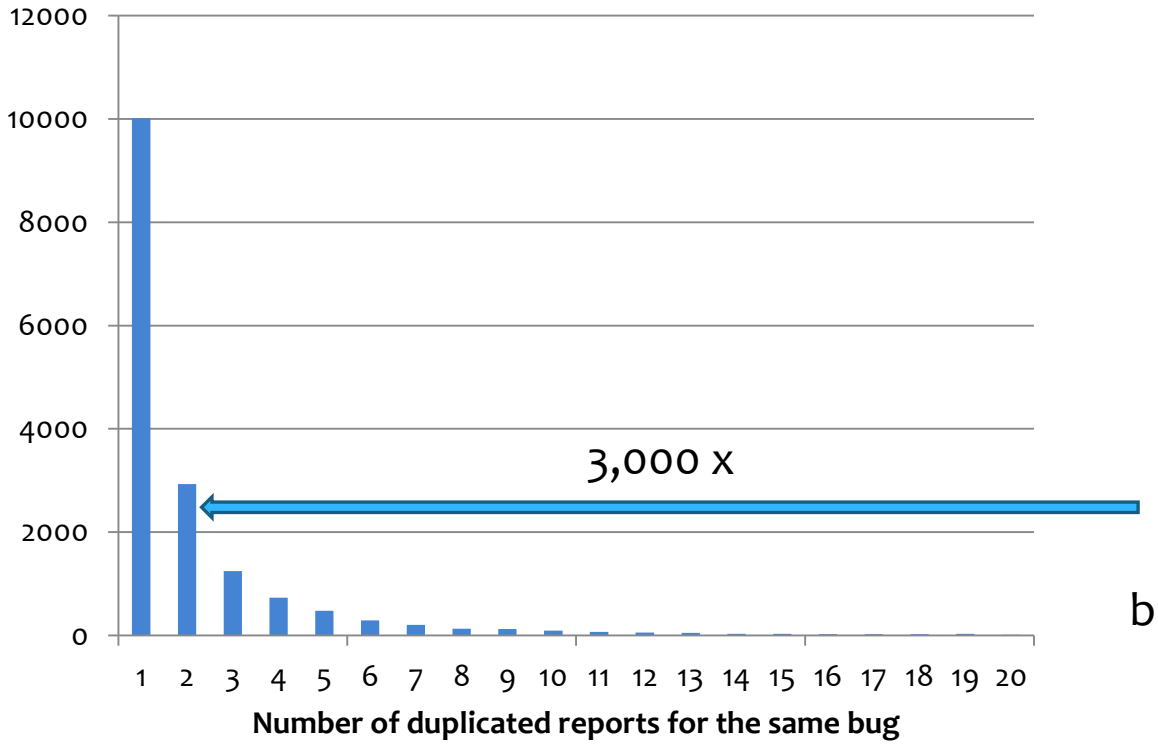


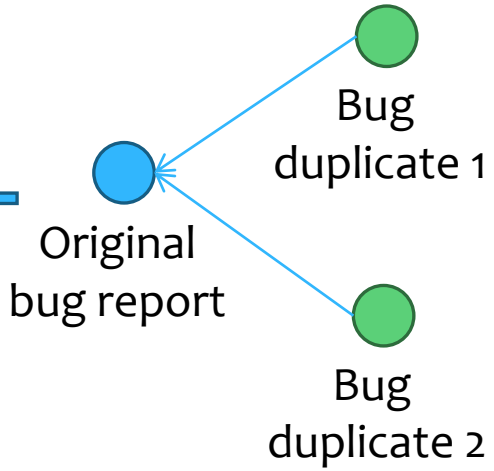
Distribution of the number of duplicated reports



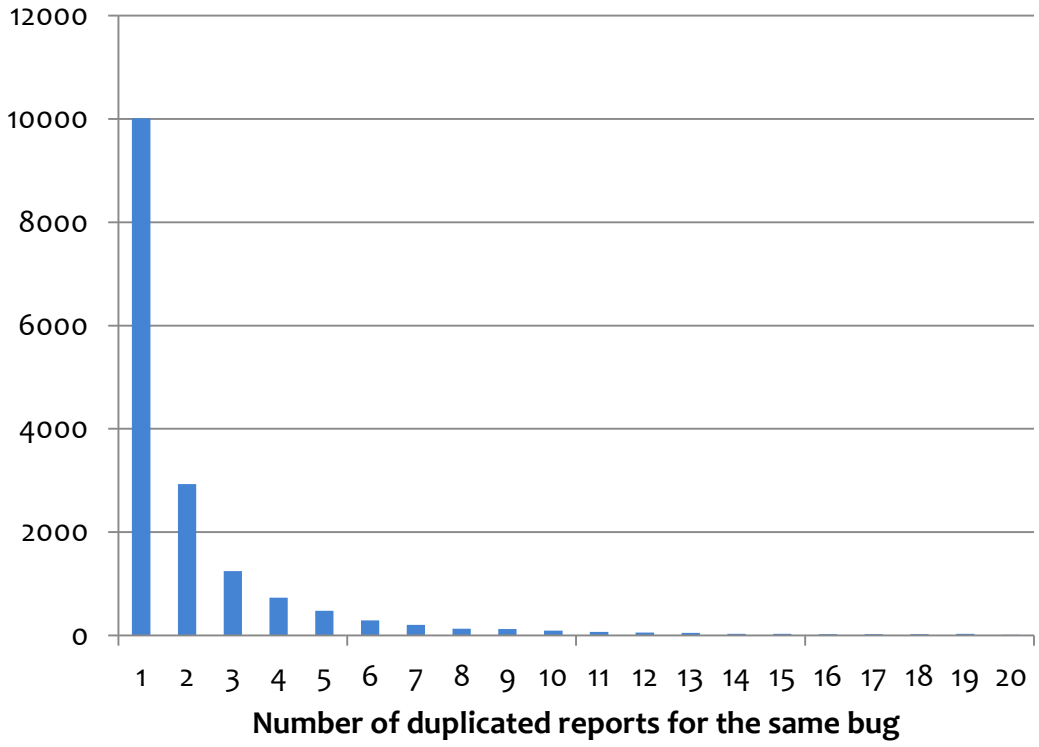


Distribution of the number of duplicated reports





Distribution of the number of duplicated reports



* The distribution has a long tail
* One bug report even has 251 duplicated reports