**CS/SE 6301 Software Analysis and Comprehension**

**Spring 2016**

**Assignment #2 –February 23, 2016**

**Team 2**

**Ronaldo Goncalves Junior**

**Sungsoo Ahn**

**Eclipse JDT Core**

**Apache Lucene**

1. **Precision and recall @{5,10,20}, and effectiveness for each query of each system**
   1. Apache Lucene 5.4

Precision and recall within top 5, 10, and 20 results as well as the effectiveness are shown in Table 1.

**Table 1: Apache Lucene results.**

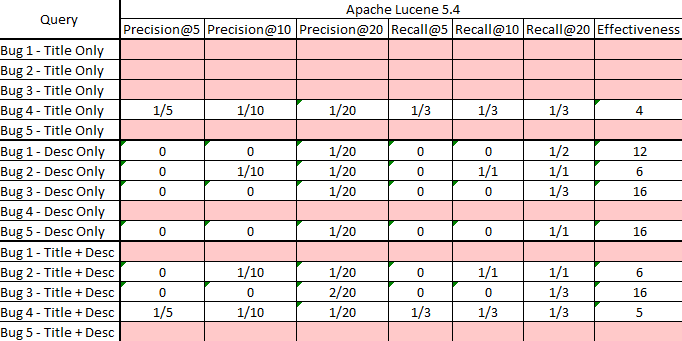
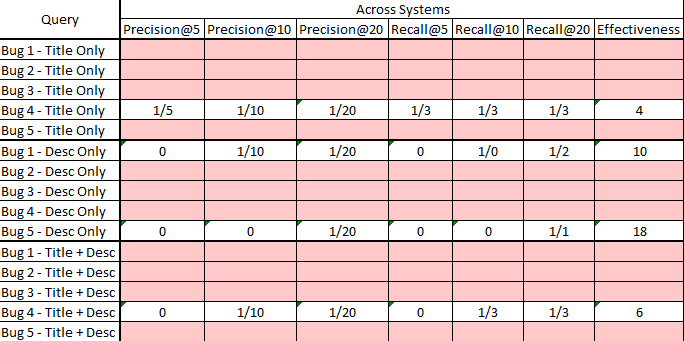


Table 2 shows precision and recall within top 5, 10, and 20 results as well as the effectiveness for Lucene bugs for all systems (across systems).

**Table 2: Lucene bugs across systems results.**



* 1. Eclipse JDT/Core

Precision and recall within top 5, 10, and 20 results as well as the effectiveness are shown in Table 3.

**Table 3: Eclipse JDT Core results.**

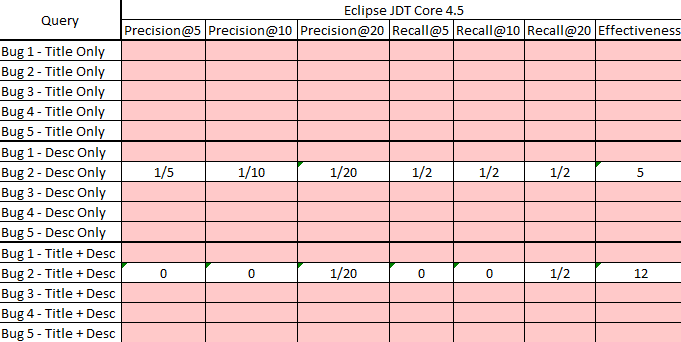
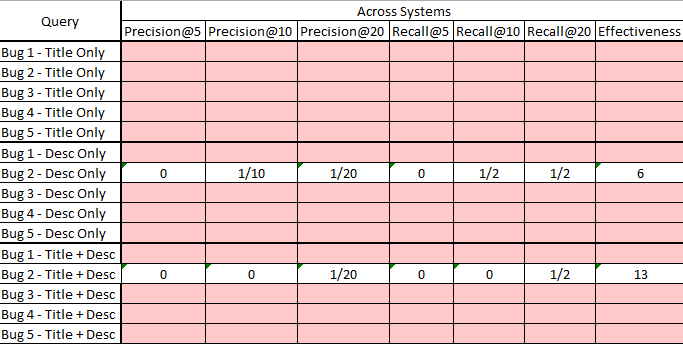


Table 4 shows precision and recall within top 5, 10, and 20 results as well as the effectiveness for Eclipse bugs for all systems (across systems).

**Table 4: Eclipse bugs across systems results.**



1. **Average and median precision and recall @{5,10,20}, and effectiveness for each system**

It is possible to notice in the above tables that due to the amount of unsuccessful query results, both the average and the median for precision and recall for both systems tend to zero. However, that does not mean the analyzer (or the input) is incorrect. The results shows that similar methods are retrieved (e.g., from the same class of the defective method, or the same folder/package. Please see Assignment2Team2.xlsx for more information). This indicates that the search is in the right direction, which could potentially retrieve positive results for 30 or 40 items instead of 5, 10, and 20, for instance.

Although the analyzer (and other source code files) is the same for both systems, we can see that for Lucene there are slightly better results. The major difference lies on the input (bug title and description), since both project are of similar size. This indicates that the input has a strong influence in the results. For the present work, we avoided all bugs that contained source code reference and other implementation details. Therefore, it is possible to state that the results might change drastically when other gold sets are used (“easier to spot” bugs).

1. **Average and median precision and recall @{5,10,20}, and effectiveness for all the systems (i.e., across systems)**

The same information described in the previous section is valid for this section. Nonetheless, it is still possible to notice an impact into the results when the queries are executed across systems. If there is some level of difficulty to retrieve information, the addition of a new system to the corpus might impact positively or negatively. We see that the results are slightly worst (please see section 1 tables). That is, for this case study, we can conclude that across systems evaluation is hindering more than helping.

1. **Comparison of precision, recall, and effectiveness between the three types of queries, defined in step 3.a**

Using Title Only queries have proven to retrieve results with less efficiency, i.e., most fixed bugs queries did not return results at all within top 20. Compared to Title Only, Desc Only (description only) queries generated significantly better results for both systems. Not only the resulting items themselves, but also other elements shows that the analyzer “is going into the right direction”, such as finding methods within the class that contains the defective method, the same package, and so on.

Finally, for the last query type, the “Title + Desc”, it is possible to notice that the results are better than the Tile Only and worst than the Desc Only. This leads to the statement that using the title as a query element (for this case study) does not help the analyzer to track and find the defective method, and rather makes it more difficult to do so.

According to the mentioned above, although the results might be unsatisfactory to a developer trying to locate a defective method, it is important to take into account that the gold set was generated using bugs that might not be descriptive by themselves. For example, the title for the first fixed bug for Eclipse system is: “Support Unicode 7.0 in ECJ”. Also, instead of avoiding all bugs that contain source code within its title and description, if the gold set is comprised of such bugs, the results are expected to be significantly better, which can be validated in future works.