Retrospective Document
Sprint-5

Work & Test Progress

Integration of the System Components - 90%

K-gram generation code integration with Lucene, tests if correctly generates the grams after integration
Newline-removing filter’s integration to the Lucene analyzer, tests if it correctly removes after integration
Frequency generation code’s integration to the Lucene code, then speed related tests while the number of docs increased.
Calling Lucene indexing mechanism from C++ code and user input tests related to it.
Removing the grams from index, tests if it actually removes them from index by checking its size each time.
Space-removing filter’s integration to the Lucene analyzer, tests if it correctly removes the spaces after integration
Prefix and suffix filter, tests yet to be completed, not 100% done.
Case sensitivity filter integration and its tests.

Team Progress

List the team members along with their contribution percentages.

Fatih Belce – 25%
Mustafa Güven – 25%
Oğuzhan Demir – 25%
Özgür Baskın – 25%

Left-overs (Backlog)

Integration of the System Components:
We had to improve some of the system components and were obliged to deal with some bugs. Plus we needed to implement our own analyzer with additional filters on the way, which rooted us to the place where we were before, the base index implementation period. However; this time was used to enhance our index to an advanced one. This came with a dozens of different tests, at the end all of the mentioned took more time than we thought. Therefore, we could complete 90% of the milestone. We promised to integrate apriori code, k-gram code and frequency finder in a single place. They are there and working as expected, considering the completed tests so far. Though, as a left-over we still need to get the info from physical plan generator and bring it to the index, but it is not 100% completed yet.

Next Sprint

Selectivity Algorithm:
We need a selection mechanism for k-grams to be able to understand which one is useful or not. In order to accelerate the system, non-useful k-grams are eliminated. Hence, we have to develop our selectivity algorithm by observing some theoretical, practical results and making calculations and have to implement it.

Optimization of the system components:
Optimizing takes a lot of effort and tests and because of the data pool we have it will take a lot of our times as well. Indexing and finding frequency of the grams take so much time when we do not invest much attention to the size of the data. We want it to take less time to make more tests; on the other hand, we need it to be as realistic as possible. Those two do not come in one package. Therefore we will try to find a middle ground and share the workload of optimizing different components to different persons, so that we could work in a parallel manner. Those components would include from start to the end of every single piece of code that could take some CPU cycles. Parsing regex, writing to file, indexing, finding frequencies of words, generating trees, calling one piece of code from another one and so on.

Comments

This sprint every member of the group had different tasks to complete from different courses. Midterms hit to us as well. At these kind of time intervals (hws, midterms and other tasks), group members are having less of an attention because of an exhausted state of mind, and they cannot find the time to relax their minds. Therefore a short come of this dense sprint was to work in very small time intervals and all group had hard time completing the tasks assigned to them. In spite of this, we were able to make the most of ourselves and almost completed our big milestone, but leaving some optimization problems. Next sprint will be spent to find a solution to this problem.

Assistant’s Evaluation

Assistant’s (Team Leader’s) comments regarding to this completed sprint.

Supervisors’s Evaluation

Supervisor’s (Team Leader’s) comments regarding to this completed sprint.