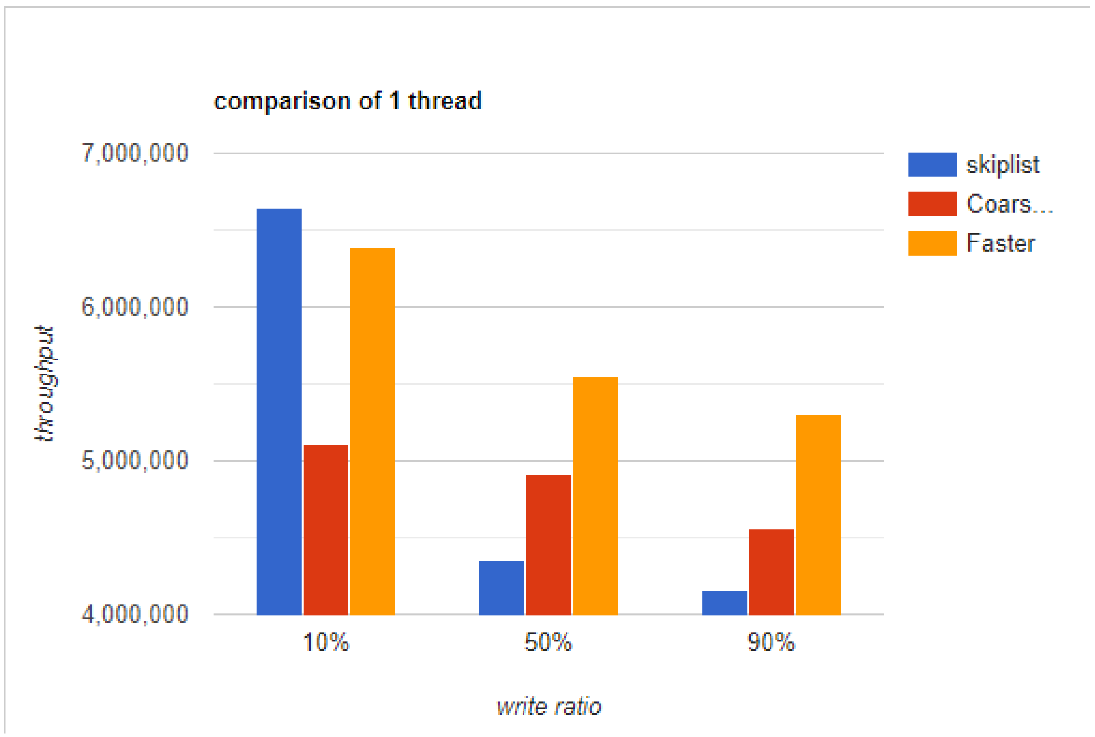
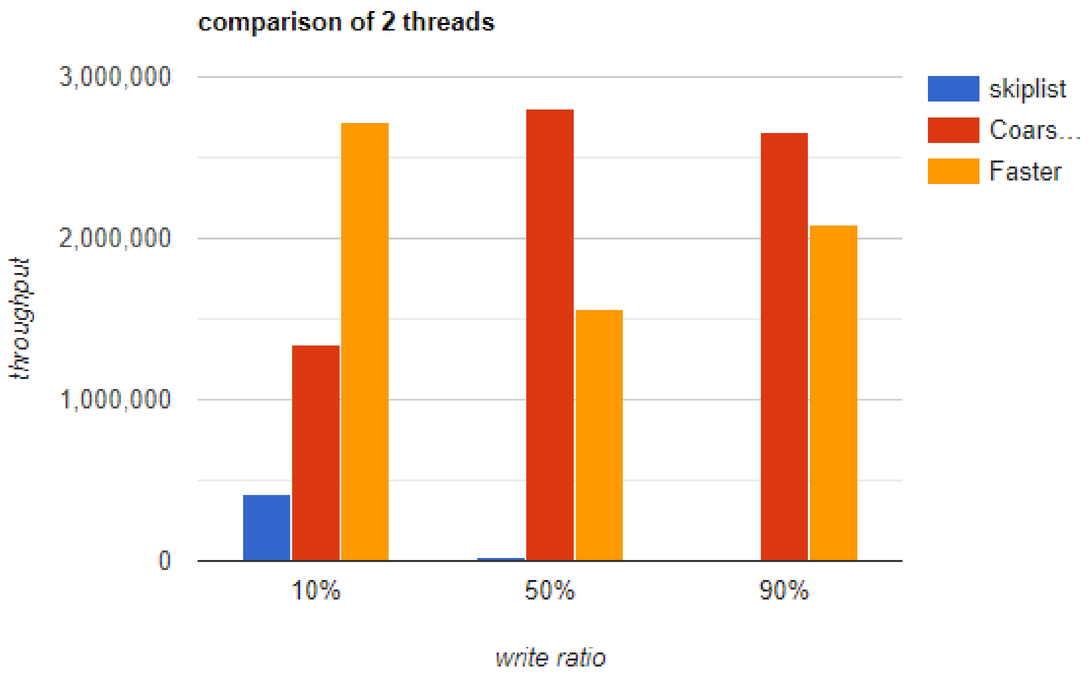
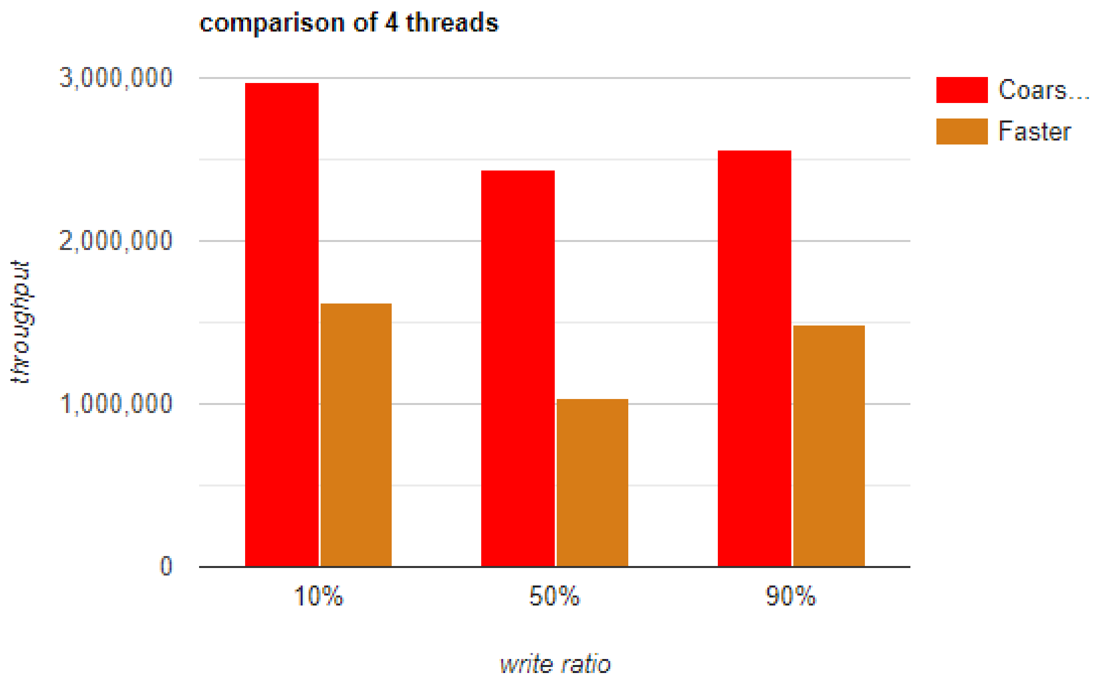
Soft3410 assignment 2 report

Shuwei Zhang

Observation of 3 different data structures:







Note: the thread-unsafe skiplist has only went through tests on 1 and 2 threads.

Note2: some of the benchmarks are done on a different computer which has a better cpu!!! (i7-6700k vs i7-5500u) These benchmarks includes the whole CoarseGrained ones, and the 1t-10%wr, 2t-90%wr benchmark of non-sync skiplist. They are supposed to have more outputs.

Note3: the **faster skiplist** used a read-write lock.

From the observation, we can summarize:

1. As the write ratio increases, the throughout of all 3 data structures tends to decrease.

2. The **non-synchronized skiplist**'s processing efficiency drop drastically as writing ratio and thread number increases, although it can still stay accurate when there's only 1 thread. (ignore the 1t-10wr data)

3. Considering the **Coarse-grained skiplist** was run on a better computer, it should have similar efficiency to the **faster skiplist** when there're multiple thread. However, when

there’s only 1 thread, the faster skiplist run much more faster.

Explanation of observation:

1. Since the writing operation(add and remove) in the skiplist has to perform an additional operation about adding/removing levels, these operations are therefore costing more time to perform than reading operations. Skiplist primarily saves time on reading operations, therefore if there’re too many writing operations, it will cost more time for the data structure to handle than simple data structures like linked list.

2. The non-synchronized skiplist is failing tests on higher threads because it’s not thread safe. Therefore having huge difference on results between reality and expectation. For example, when multiple threads tries to insert 5 and 6 between 3 and 7, a conflict will happen and there will only be 1 node inserted successfully. On higher writing ratios and threads, the data structure have to constantly spend time re-doing operations that should have been done before(using same example, it may insert 6 again after a failed insertion that should be successful if locked properly), therefore decreasing its efficiency drastically.

3. The faster skiplist uses a read-write lock, which allows multiple reading process to happen together but only 1 writing process at one time, therefore slightly increasing the performance, comparing to the coarsegrained lock, which has only 1 process accessing the data structure on both writing and reading operations.