Lab 9: Concurrent Double-Linked Lists

Instructor: Mack Joyner

Course wiki: `http://comp322.rice.edu`

Staff Email: comp322-staff@rice.edu

Goals for this lab

- Convert a sequential implementation of double-linked lists into a concurrent one
- Make sure that the concurrent implementation runs faster than a single-threaded sequential one

1 Downloads

As with lab 1, the provided template project is accessible through your private GitHub classroom repo at: `https://classroom.github.com/a/QtwajBr4`

For instructions on checking out this repo through IntelliJ or through the command-line, please see the Lab 1 handout. The below instructions will assume that you have already checked out the lab6 folder, and that you have imported it as a Maven Project if you are using IntelliJ.

2 Double Linked List Implementation

The code you are given has a correct sequential implementation of a double-linked list. You can check that it works by running the Lab9BaselineTest. However, this implementation is not thread-safe! There are no mechanisms to ensure that concurrent accesses to the double-linked list are handled correctly. You can check this by running the Lab9CorrectnessTest. The testListInsertions and/or testListDeletions test will probably fail. If it doesn't (it's possible, on any given run, that none of the concurrency issues get exposed and that all tests succeed) run it again until it does fail. The testListPerformance test will probably succeed, but that's a small comfort when we have a non-working implementation.

2.1 Fixing the Concurrency Issues

Implement the concurrentInsert, concurrentRemove and concurrentContains methods in the DoubleLinkedList class. Right now, they are just calling the non-thread-safe insert, remove and contains methods. Once both the testListInsertions and testListDeletions tests are consistently passing, you can be pretty confident that you have handled the concurrent accesses to the list correctly. However, the testListPerformance test might now be failing.

2.2 Fixing the Performance

If the performance test is failing, then you need to think about how you are handling the concurrent accesses to the list. Are you allowing different tasks to access the list at the same time? Are there any operations that can happen on the list at the same time? If the performance test is failing, that probably means that you fixed the concurrency, but at the cost of limiting the parallelism too much. The performance test fails only if you cannot achieve a speedup of more than 1.3x, which should be possible even if you have an old 2-core laptop.

3 Turning in your lab work

For each lab, you will need to turn in your work before leaving, as follows.

1. Show your work to an instructor or TA to get checked off for this lab during lab or office hours by Monday, Apr 10th at 3pm. They will want to see your passing unit tests on your laptop.
2. Commit and push your code to GitHub.