Lab 6: Isolated Statements, Atomic Variables

Instructor: Mack Joyner

Course Wiki: http://comp322.rice.edu

Staff Email: comp322-staff@mailman.rice.edu

Goals for this lab

- Gain experience with using the isolated construct
- Gain experience with using atomic variables

1 Spanning Tree Construction

The idea of a spanning tree was introduced/reviewed in Lecture 20. Given a connected and undirected graph `G` consisting of:

1. A set of nodes `N`
2. A set of undirected edges `E` between pairs of nodes in `N`
3. An arbitrarily defined root node `R` in `N` for graph `G`

A spanning tree is a set of edges `S` in `E` that touch all nodes in `N` and contains no cycles (i.e. is a tree). To have no cycles, we require that `|S|` be exactly `|N| - 1`.

One way to define a spanning tree over `G` is by defining a "parent" node for each node in `N`, with each node having at most one parent and the root node `R` having no parents.

Constructing a spanning tree is generally formulated as a graph traversal algorithm, in which we start at the root node `R` and through a depth-first traversal set the parent of each node if it has not already been set. If it has already been set, we know we already explored that node.

Lecture 20 discussed how this algorithm can be parallelized.

2 Parallelization using Isolated Construct

The Maven project for this lab is located in the following GitHub classroom repository:

https://classroom.github.com/a/0eomIsZd

For instructions on checking out this repository 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.

First, we will parallelize spanning tree construction and use isolated to protect against concurrent accesses.

Recall the following constraints on isolated statements — an isolated statement may not contain any HJ statement that can perform a blocking operation e.g., finish, future get(), and phaser next/wait.

Your first task is to implement parallel spanning tree construction by adding async, finish, and isolated constructs as described in Lecture 20 to the template in `SpanningTreeIsolated.java`. This will consist of:

1. Using an isolated statement (or some variant) to protect against concurrent updates to a node's parent.
2. Changing the calls to `compute()` to be performed in parallel.

You can verify the correctness of your isolated version using the test `Lab6CorrectnessTest.testIsolated`. Note that the test only confirms a functioning algorithm, so a TA will also need to verify your parallel implementation in order to check you off.

3 Parallelization using Atomic Variables

Atomic Variables were also introduced in Lecture 20.

Your task in this section is to modify the `SpanningTreeAtomic.java` program to run spanning tree construction in parallel and use atomic variables to protect against concurrent accesses, similar to how you added isolated constructs to `SpanningTreeIsolated.java`. This will consist of:

1. Using atomic variables to protect against concurrent updates to a node's parent. Note that this may require changes to provided data structures.
2. Changing the calls to `compute()` to be performed in parallel.

Like the previous section, you can verify the correctness of your atomic version using the test `Lab6CorrectnessTest.testAtomic`. Note that the test only confirms a functioning algorithm, so a TA will also need to verify your parallel implementation in order to check you off.

You may find the AtomicReference class useful for this part of the lab.

4 Turning in your lab work

For lab6, you will need to turn in your work before Tuesday, April 6, 2021 at 12 pm (noon), as follows.

1. Show your work to an instructor or TA to get credit for this lab. The TAs will be interested in seeing your changes for the isolated and atomic versions.
2. Commit and push your work to your `lab6` GitHub folder. The only changes that must be committed are your modifications to `SpanningTreeIsolated.java` and `SpanningTreeAtomic.java`. Check that all the work for today's lab is in your `lab6` directory by opening `https://classroom.github.com/a/OeomIsZd` in your web browser and checking that your changes have appeared.