CMPT 383: Vitamin #10

Anders Miltner
miltner@cs.sfu.ca

Due Nov 23

Introduction

This Vitamin is to help you practice concurrency in Rust. The test suite is provided in `src/lib.rs`. You should fill out the function definitions in `src/functions.rs`.

This submission will be partially autograded. There are some portions of the assignment that are ungraded, and some that will be graded. We provide a (partial) test suite for partial validation. You can run these tests by opening a terminal in the v10 directory, and running `cargo test`.

We have included all relevant imports. If you import additional functions, you may get a zero on the assignment. You will be implementing a variety of forms of distributed incrementers.

1 `shared_state_incr`

The `shared_state_add` function is a function that a number of threads will run. These threads will call into `shared_state_add`. Each `shared_state_add` call should increment the provided integer reference by 1.

2 `distributed_receive_incr`

The `distributed_receive_incr` function is provided a receiver that receives functions that take in i32 inputs, and produce i32 values. It is also provided a mutable integer. The `distributed_receive_incr` function should receive i32 to i32 functions until the channel is closed. It should incrementally apply each of these functions to the variable x. Once the channel is closed, return x.

3 `distributed_send_incr`

The `distributed_send_incr` function will receive a vector of functions that transform i32 values to i32 values. It should transform these values into a vector of JoinHandles, where each JoinHandle corresponds to a thread that sends one of the i32 to i32 functions over a channel. The receiver of that channel should be provided in the second argument. No tests are provided for this function.