PLC: Workout 8 [90 points + 10 extra]

Due date: Monday, May 11th by midnight

About This Homework

This assignment is about internal verification in Agda, including using it for implementing mark-and-sweep garbage collection.

How to Turn In Your Solution

Please submit your solution via ICON. As for workout7, please follow this format for submissions:

- Please just submit a zip file for your whole `workout7` directory. This will make it easier to autograde.
- If you are working with a partner, only one of you should submit the workout, and
- that person should include a partners.txt file that lists both the partners' full names (not HawkIds)

Partners Allowed

You may work alone or with one partner. See previous subsection for how to indicate you worked with a partner.

How To Get Help

You can post questions in the `workouts` section on Piazza.

The course staff will be holding office hours by Zoom, at times to be announced on Piazza. Please check Piazza and the course calendar for these:

`https://calendar.google.com/calendar/embed?src=a5d6qokrert25ce093iksp8np0%40group.calendar.google.com&ctz=America%2FChicago`

1 Reading

Review Chapter 5 of Verified Functional Programming in Agda, available for free (on campus or VPN) here:

`https://dl-acm-org.proxy.lib.uiowa.edu/doi/book/10.1145/2841316`

2 Internal verification of operations on sorted lists [50 points]

There are three holes in `sortedLists.agda`. Please read the comments for `sortedList` and throughout the file for an explanation of how the `sortedList n` type works. It enforces that the elements are in non-decreasing order, so any value of type `sortedList n` is really a sorted list. (The number `n` is a lower bound on the data in the list.) You will prove one theorem about the `min'` operation defined in the file, and then write `insert` and `merge` operations.

3 Garbage collection [50 points]

In `gc.agda` you will find definitions modeling memory as a vector of allocated cells. There are several holes to fill in to define a basic version of a mark-and-sweep garbage collector. Internal verification is used to ensure that pointers cannot go outside the memory. We handle the graph traversal in a way that satisfies Agda's termination checker by having the main function (`markh`) take a vector of unmarked addresses as an input. When we mark an address (as in mark-and-sweep), we remove it from that vector, which causes the length of the vector to decrase (this is done by `doMark`). Then we can recurse with the length of that vector of unmarked addresses decreased if we need to.