Turing Machine Simulator

Rachelle Burgos (rachelle.burgos27@myhunter.cuny.edu)

March 15, 2023

1 Introduction

This Turing Machine simulator is a program written in C++ that simulates a one-tape, one-track, two-way infinite tape, deterministic Turing Machine. Given a file containing the Turing Machine description (otherwise known as the transition function), this program will simulate the behavior of the Turing Machine on a given input string consisting of only 0's and 1's. The file description contains five-tuples that describe the Turing Machine in the format `currentState currentTapeSymbol newState newTapeSymbol direction`.

For every new configuration the Machine is on, the program will output an ID, where the format is If the user would like to halt the program and input a new input string for the same Turing Machine description, press `h` to be prompted for the new input.

2 How It Works

The main function reads in the input string, checks if it is valid, and prompts the user to enter a new input string if necessary. It then creates an instance of the `TuringMachine` class and calls the run function to simulate the Turing Machine.

The transition function is created by parsing the input file containing the Turing Machine's description and storing each line in a hash map of five-tuples. The hash map representing the transition function has a key containing the current state and the current symbol. Each key maps to the new state, the new tape symbol, and the direction in which to move the read/write head.

In the given implementation of the Turing Machine, the tape is represented as a vector of characters, which stores the symbols on the tape in order. To mock the two-way infiniteness of the tape, the data structure used to implement the tape is able to dynamically resize if the read/write head moves past either end of the allocated tape, with blank symbols `B` inserted at either end.

The Turing Machine always begins at the initial state `0`. During the simulation, the program looks up the current state and current tape symbol in the hash map to determine the next transition. The user is able to halt the program by pressing `h`, which prompts the user for a new input string for the same machine description.

If there is an entry for the current state and tape symbol in the hash map, the simulator updates the current state, writes the new symbol to the tape, and moves the read/write head in the specified direction. If there is no entry for the current state and tape symbol in the hash map, the simulator halts and reports that the input was not accepted by the Turing Machine. If the current state is `f`, which represents the final state, the simulator halts and reports that the input was accepted by the Turing Machine. If the Turing Machine does not halt, it will loop on forever unless the user presses `h` or quits the program.

3 Compilation Instructions

- To compile the Turing Machine simulator using the provided Makefile, type `make all`, which will produce an executable named `run.me`.
- To run the program, type `./run.me <TM-Description>` in the terminal, where `<TM-Description>` is the file containing the description of the Turing Machine. If no file for the Machine's description is provided, the program will output an error message.

– To clean up the object files and executable, type `make clean`.

– If you make changes to the source code, you can recompile the program by typing `make rebuild`. This will clean up the object files in addition to recompiling the program.

4 References

In order to have the program halt properly when the user presses the `h` key, I used the function _kbhit() by Morgan McGuire. Initially, I tried using C++ standard library functions such as `cin.peek()` and programming techniques such as threading, but neither method worked as I intended. Therefore, after researching different options, I decided to use `_kbhit()` because it allowed me to detect keyboard input without having to wait for the user to press enter. Otherwise, all other included libraries are from the C++ standard library.