Readme_Odmark

Version 1 8/22/24

A copy of this file should be included in the github repository with your project. Change the teamname above to your own

1. Team name: Odmark
2. Names of all team members: Grace Odmark
3. Link to github repository: https://github.com/grodmark/fa24-TOC-godmark_project02/tree/main
4. Which project options were attempted: Tracing NTM Behavior
5. Approximately total time spent on project: 4 hours
6. The language you used, and a list of libraries you invoked. Python
    a. csv, argparse, itertools, sys
7. How would a TA run your program (did you provide a script to run a test case?)
    a. python3 traceTM_odmark.py <transitions csv file name> <input string> <optional --max_depth value>
        i. python3 traceTM_odmark.py equal01s.csv 01
        ii. python3 traceTM_odmark.py a_plus.csv aaa
8. A brief description of the key data structures you used, and how the program functioned.
    a. The key data structures I used were lists which were passed to the check_next function each time it ran. Then, I iterated through the list of transitions and compared current state and input with the transition options to determine how to proceed. I also tracked visited states with a visited set, and this helped ensure that the program didn't stall or run at too high of a depth.
9. A discussion as to what test cases you added and why you decided to add them (what did they tell you about the correctness of your code). Where did the data come from? (course website, handcrafted, a data generator, other)
    a. Below are the two test cases I used. These tested both a very simple machine (the a*) and a more complex machine with the equal number of 0s and 1s program. The added complexity came from having to change the tape and tracking previously visited states.
        i. python3 traceTM_odmark.py equal01s.csv 01
        ii. python3 traceTM_odmark.py a_plus.csv aaa
10. An analysis of the results, such as if timings were called for, which plots showed what? What was the approximate complexity of your program?
    a. The complexity is based on how many states there are in the machine and how long the string is. As there become more options of state transitions with the same current state and input, the complexity will increase immensely because of the nature of Depth-First Searches and how much parsing must be done.
11. A description of how you managed the code development and testing.
    a. I managed the code development and testing by first considering how to recurisvely call the check_next function, and when to accept a transition. Something that was important in my development process was knowing how to

modify variables in a way that wouldn't mess things up if multiple transitions were possible on the same input and state. In testing, I observed whether the path taken for the string was something that would really happen, and whether or not a string should actually be tested.

12. Did you do any extra programs, or attempted any extra test cases
    a. No