Analysis of Collatz Stopping Time Number Sequences Using Multi-Threaded Programming Approach

Joshua Lemon and Levi Shaffer

COP 4634 Systems and Networks I

University of West Florida, Fall 2020

*Abstract*—TODO: Fill in the Abstract

*Index Terms*—Collatz, Computer Science, Multi-Threading, Number Sequences, Race Conditions, Thread Performance

# Introduction

As software has become more and more complex, one of the major challenges facing software and hardware engineers is optimizing software performance for many hardware platforms. One of the ways to increase performance other than increasing the cycle speed of the CPU clocks is by creating multicore CPUs that are capable of processing multiple threads in parallel to take a complex computational problem and breaking it down into multiple smaller problems to solve that then get joined together at process completion.

For the Collatz project, the goal is to utilize a multi-threaded implementation to find the stopping time of the Collatz number sequences up to a passed argument, N. The Collatz conjecture states that for a number N, if the number is even, divide by two. If the number is odd, multiply N by three and add one. The result of this piecewise calculation will then become the next number N to be evaluated. When the number to be evaluated reaches a power of two, the number will continue to be divided by two until reaching one, at which point the sequence is complete. The time complexity of this reduction from a power of two down to one is logarithmic time.

Taking the Collatz process and breaking it down into multiple threads should ideally allow for a faster computation of the Collatz number sequence, and allow us to calculate the stopping time deltas between processes that kick off different numbers of threads. Each thread will take the next number in sequence as it finishes the task at hand and tally the number of times that stopping time occurs per number N. From this tally of stopping times ranging from one and one-thousand, the numbers can then be converted into a histogram which will show the frequency that each end time is reached. In addition, the process turnaround time is calculated which will indicate how much quicker the Collatz sequence can be calculated in parallel instead of sequentially using a single-threaded implementation.

# Implementation

TODO: Fill in Implementation details.

# Runtime Testing Results

TODO: Fill in the runtime testing results.

# Analysis

TODO: Fill in analysis

# Summary and Conclusion

TODO: Fill in Summary and Conclusion

# Acknowledgment

The authors would like to thank Dr. Thomas Reichherzer for his instruction in the course lectures about these topics. Dr. Reichherzer provided us with the knowledge necessary for us to setup and conduct the experiments presented in this document.

# References

*Books:*

[1] I. Asimov, *I, Robot*, New York.

[2] C. E. Harris, M. S. Pritchard, M. J. Rabins, *Engineering Ethics: Concepts and Cases.* 4th Edition.

[3] P. McCorduck, *Machines Who Think* (2nd ed.), Natick, MA: A. K. Peters, Ltd., pp. 356, 374–376

*Papers:*

[4] N. Bostrom, “Ethical Issues in Advanced Artificial Intelligence”, Oxford University. Available:

<http://www.nickbostrom.com/ethics/ai.html>

[5] Y. Sheliazhenko, "Artificial Personal Autonomy and Concept of Robot Rights",European journal of law and political sciences. Available: <https://cyberleninka.ru/article/v/artificial-personal-autonomy-and-concept-of-robot-rights>

*Web-Based Resources:*

[6] B. Lufkin, (March 2017). “Why the biggest challenge facing AI is an ethical one”. BBC.com

<http://www.bbc.com/future/story/20170307-the-ethical-challenge-facing-artificial-intelligence>

[7] B. Marr, (December 2016). “What Is The Difference Between Deep Learning, Machine Learning and AI?”. Forbes.com

[https://www.forbes.com/sites/bernardmarr/2016/12/08/what-is-the-difference-between-deep-learning-machine-learning-and-ai/#6fde9bc226cf](https://www.forbes.com/sites/bernardmarr/2016/12/08/what-is-the-difference-between-deep-learning-machine-learning-and-ai/%236fde9bc226cf%20)