# Lab 3

Blake Smith

May 30, 2020

## CONTENTS

CHAPTER

ONE

# SEQUENTIAL MAP REDUCE, A BASELINE

Before proceeding with testing my implementation from Lab 2 I will first gather some metrics from the sequential implementation given in *mrseqential.go*. This baseline will be used to determine at what point the multi-process implementation becomes worthwhile and what the trade offs are in regards to space & time complexity.

## 1.1 The Phases of Map Reduce (In the Sequential Case)

In order to estimate the performance of a sequential map-reduce I will be collecting the following information at each step in the execution.

1. Read input files and pass into the map function, producing a collection of intermediate values.
    - Time taken to read in the input files and produce the intermediate collection.
    - Space required to store the intermediate values
    - Time taken to sort the intermediate values
2. Group the intermediate values by key, producing a list of values for every key.
    - Time taken to group the values by key
    - Amount of memory used in that process
3. Run Reduce on each key and create a single output file
    - Time taken to complete all reduce jobs and produce the full output

## 1.2 Modifications to *mrsequential.go*