ENSC 351 - Lab 3: MapReduce

Nic Klaassen, Galen Elfert, Diane Wolf

October 17, 2018

1 Explanation of workload

The workload invented to fit the MapReduce framework better than word counts was [matrix multiplication | distributed sort problem | merge sort problem]

1.1 Conception

[Matrix multiplication | distributed sort problem | merge sort problem] was chosen as a better alternative to word counts because...

1.2 Speed: Single-threaded implementation

1.3 Speed: MapReduce implementation

1.4 Comparison

2 Word count efficiency

Both implementations of the program counted the instances of words in fifty paragraphs (with a total length of 2261 words) of Lorem Ipsum. They were run on the same machine with hardware to support twelve threads. Ten executions of each implementation were conducted, with the duration measured by the built-in Linux `time` command. A call graph for both implementations was generated using Valgrind's Callgrind tool.

2.1 Single-threaded implementation

2.2 MapReduce implementation

2.3 Comparison

3 Most appropriate workload for MapReduce

Data that needs sorting?

Execution time: single-threaded word count			
run #	user (s)	system (s)	wall (s)
1	0	0.004	0
2	0	0.008	0.01
3	0.004	0	0.02
4	0	0.004	0.01
5	0	0.004	0.01
6	0.004	0	0.01
7	0	0.004	0.02
8	0.004	0	0.01
9	0.004	0	0.04
10	0	0.004	0.01
mean (s)	0.0016	0.0028	0.0140
std. dev. (s)	0.0021	0.0027	0.0107

Table 1: Duration of single-threaded implementation measured by `time`

4 Impact of using multiple machines on execution speed

Execution time: MapReduce word count			
run #	user (s)	system (s)	wall (s)
1	-	-	-
2	-	-	-
3	-	-	-
4	-	-	-
5	-	-	-
6	-	-	-
7	-	-	-
8	-	-	-
9	-	-	-
10	-	-	-
mean (s)	-	-	-
std. dev. (s)	-	-	-

Table 2: Duration of MapReduce implementation measured by `time`