**Group 9 Members:**

Alexys Lamkin

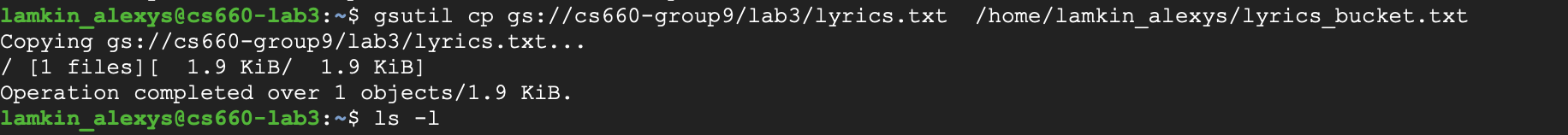
Michael Karaman

Yiyun Zhang

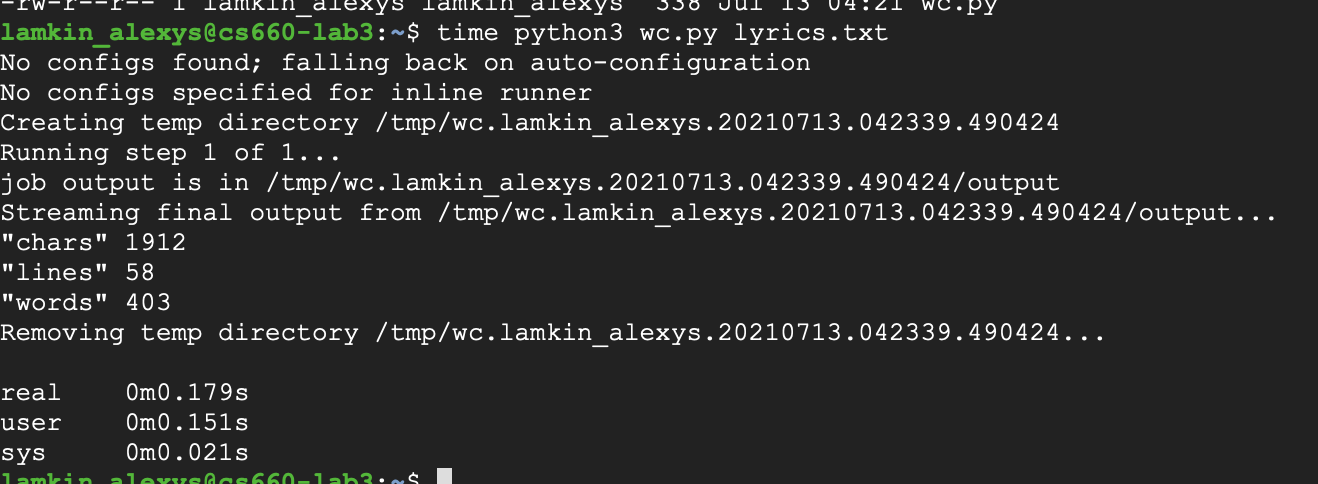
**Lab 3 - GCP and Hadoop**

Part 1:

* The time it took to upload the files to the VM versus uploading them to the Google bucket was comparable. Both were nearly instantaneous; however, the transfer rate when uploading the files via the VM wasn’t displayed for comparison. Below is the transfer rate when copying the input file over from the GCP bucket to the VM:



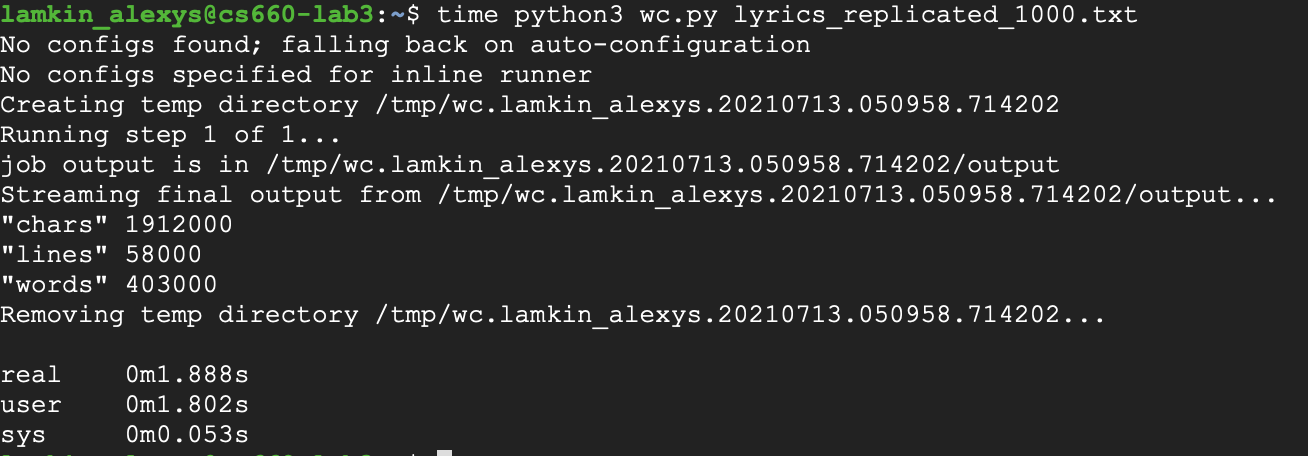
* Below is the runtime for word count MRjob program on a text file that contains the lyrics of a song:



* We then create a python script to replicate the content of the lyric time *n* times where *n* is specified by the user. Given the input file lyrics.txt that contains 58 lines, we replicated the content of the file 1,000 times to generate an output file with 58,000 lines:



* We then ran the same MRjob program on the larger input and timed the run:



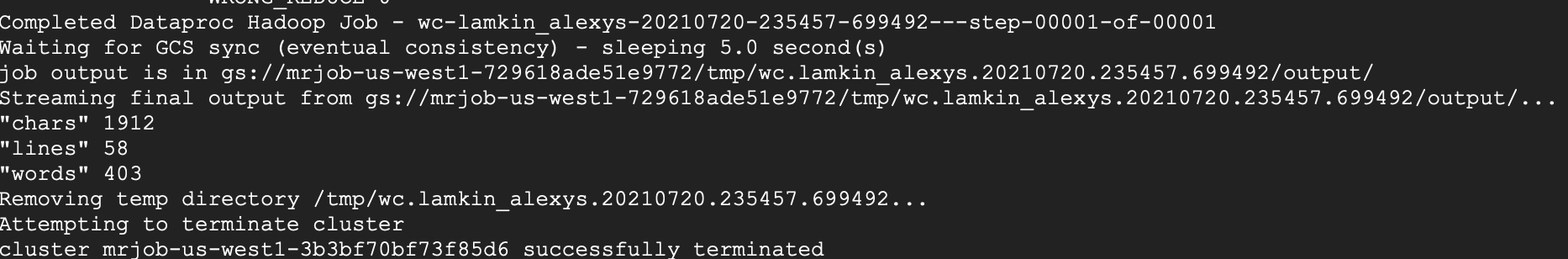
Part 2:

* We encountered several issues during this part of the lab. To remedy these issues, below details the steps we took to get our MRjob python script to run using GCP’s dataproc (assuming pip and mrjob are already installed on the VM and the service account key has been created):
  + Installing the following libraries with the correct versions:
    - pip3 install google-cloud-logging==1.15.1
    - pip3 install google-cloud-storage==1.31.0
    - pip3 install google-cloud-dataproc==1.1.1
  + Exporting the service account key:
    - export GOOGLE\_APPLICATION\_CREDENTIALS

="/home/lamkin\_alexys/cs6600-group9-8790352379d4.json”

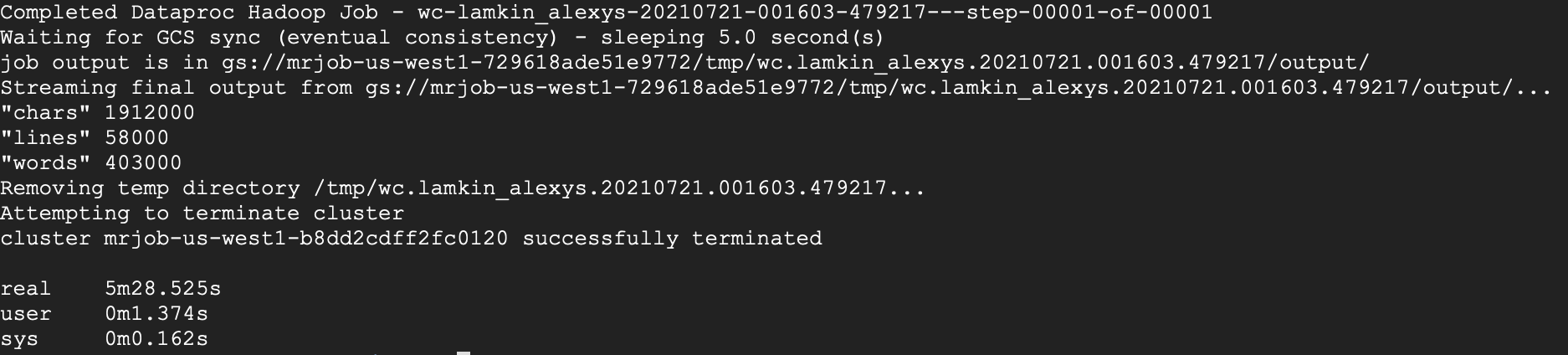
* After following the steps listed above, we were able to successfully run our python script using dataproc:

*python3 wc.py -r dataproc lyrics.txt*



* After our first run, we conducted another run using a larger output (i.e. 58,000 lines). The second run ran for 5 minutes:

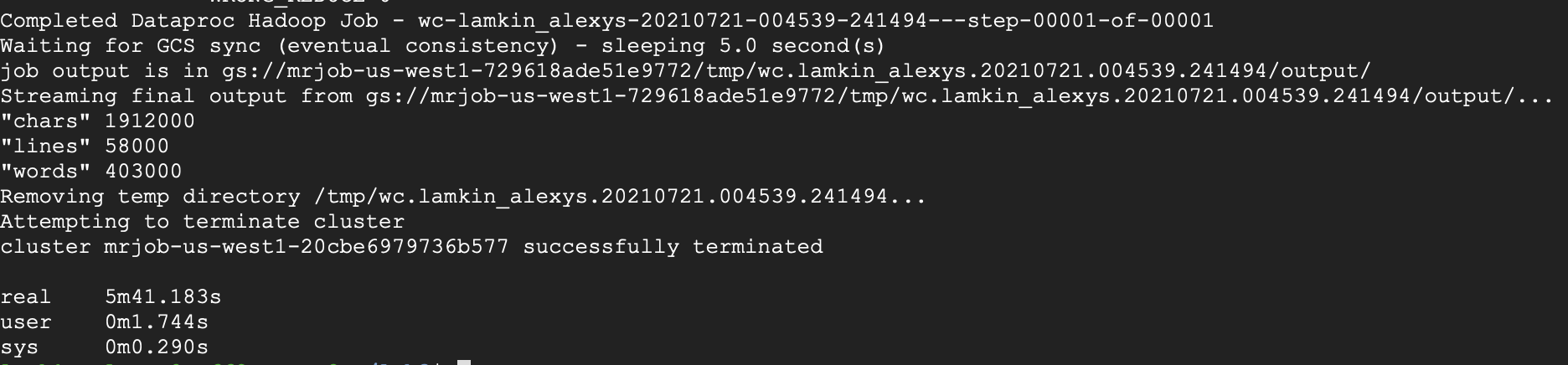
*time python3 wc.py -r dataproc lyrics\_replicated\_1000.txt*

****

* Lastly, we reran the script using the same input file, yet using different parameters and timed each run. When using different machine types, we had to request an increase in CPU quotas to avoid more errors:

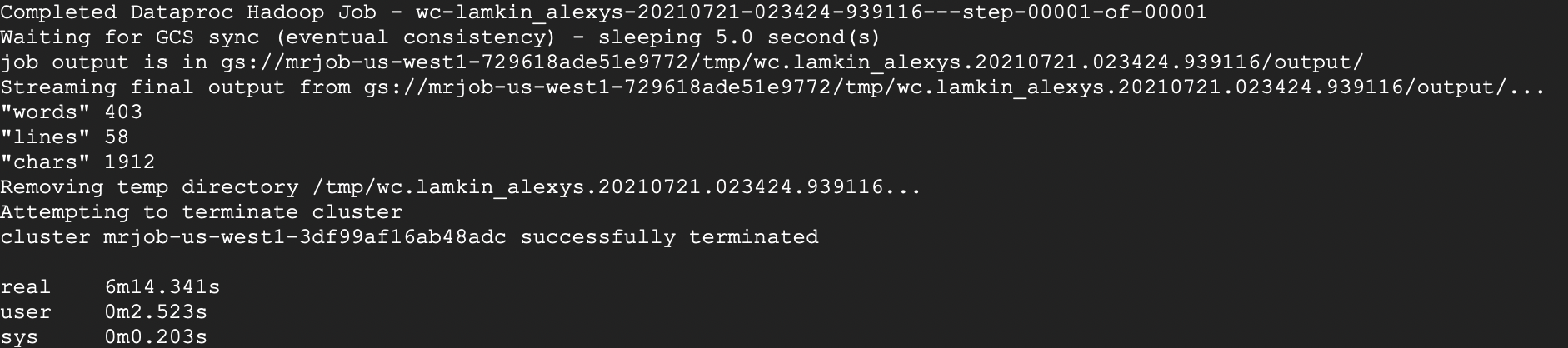
*time python3 wc.py -r dataproc lyrics\_replicated\_1000.txt --instance-type*

*e2-standard-16*

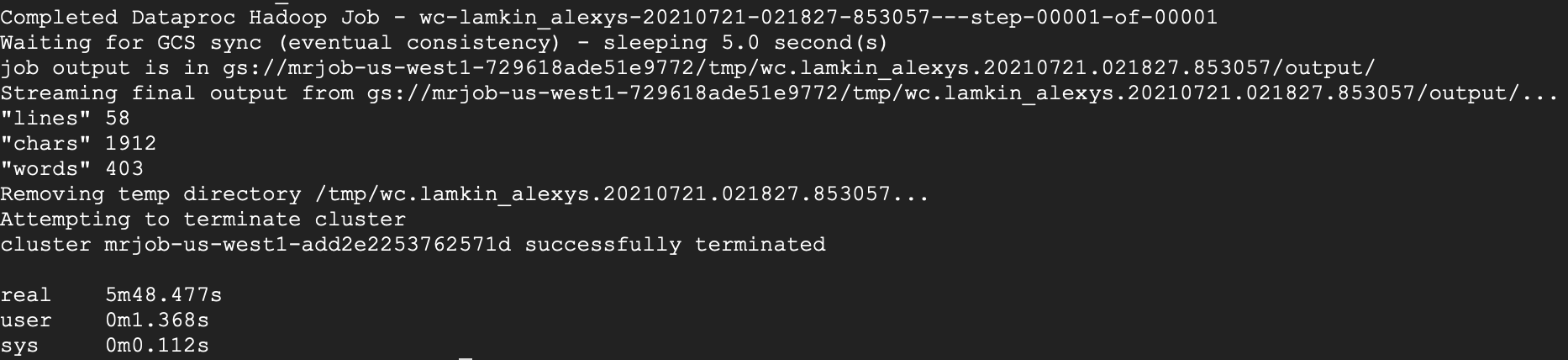


*time python3 wc.py -r dataproc lyrics\_replicated\_1000.txt --instance-type*

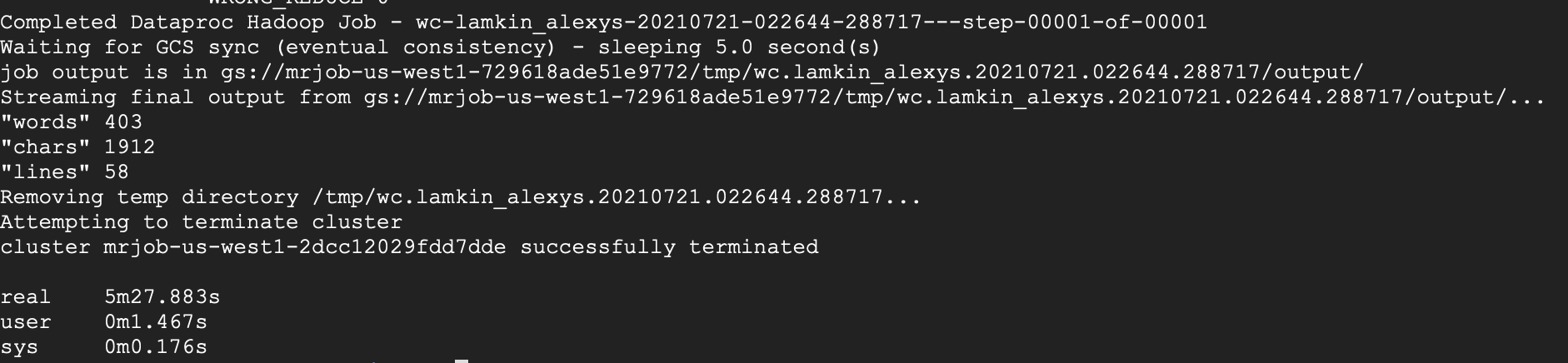
*e2-standard-32*

**

*time python3 wc.py -r dataproc lyrics\_replicated\_1000.txt --num-core-instances 4*

**

*time python3 wc.py -r dataproc lyrics\_replicated\_1000.txt --num-core-instances 7*

**