CS 571 Mini Exam 3
Spring 2022

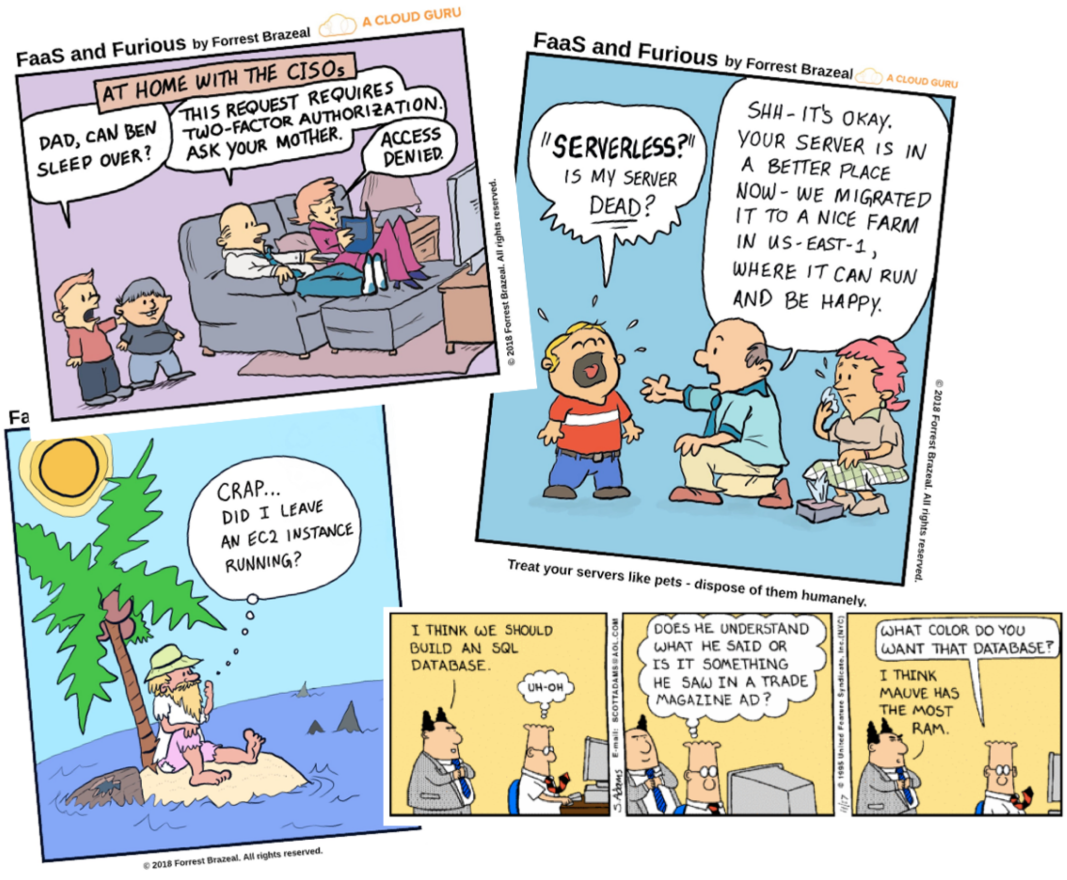
NAME:_B SAI SAHANA BHARGAVI______________
G#:_________G01330358__________________

Directions

Welcome to the CS 571 Mini Exam 3!

You have 3 days to take this exam. It is divided into three parts (Part A, B, and C). There are 7 pages.

This exam is **open book**, **open notes**. You may use the class notes during the exam.


FaaS and Furious by Forrest Brazeal
A CLOUD GURU
AT HOME WITH THE CISOs
DAD, CAN BEN SLEEP OVER?
THIS REQUEST REQUIRES TWO-FACTOR AUTHORIZATION. ASK YOUR MOTHER.
ACCESS DENIED.
© 2018 Forrest Brazeal. All rights reserved.
FaaS and Furious by Forrest Brazeal
A CLOUD GURU
"SERVERLESS?" IS MY SERVER DEAD?
SHH - IT'S OKAY. YOUR SERVER IS IN A BETTER PLACE NOW - WE MIGRATED IT TO A NICE FARM IN US-EAST-1, WHERE IT CAN RUN AND BE HAPPY.
© 2018 Forrest Brazeal. All rights reserved.
Treat your servers like pets - dispose of them humanely.
CRAP... DID I LEAVE AN EC2 INSTANCE RUNNING?
© 2018 Forrest Brazeal. All rights reserved.
I THINK WE SHOULD BUILD AN SQL DATABASE.
UH-OH
DOES HE UNDERSTAND WHAT HE SAID OR IS IT SOMETHING HE SAW IN A TRADE MAGAZINE AD?
WHAT COLOR DO YOU WANT THAT DATABASE?
I THINK MAUVE HAS THE MOST RAM.

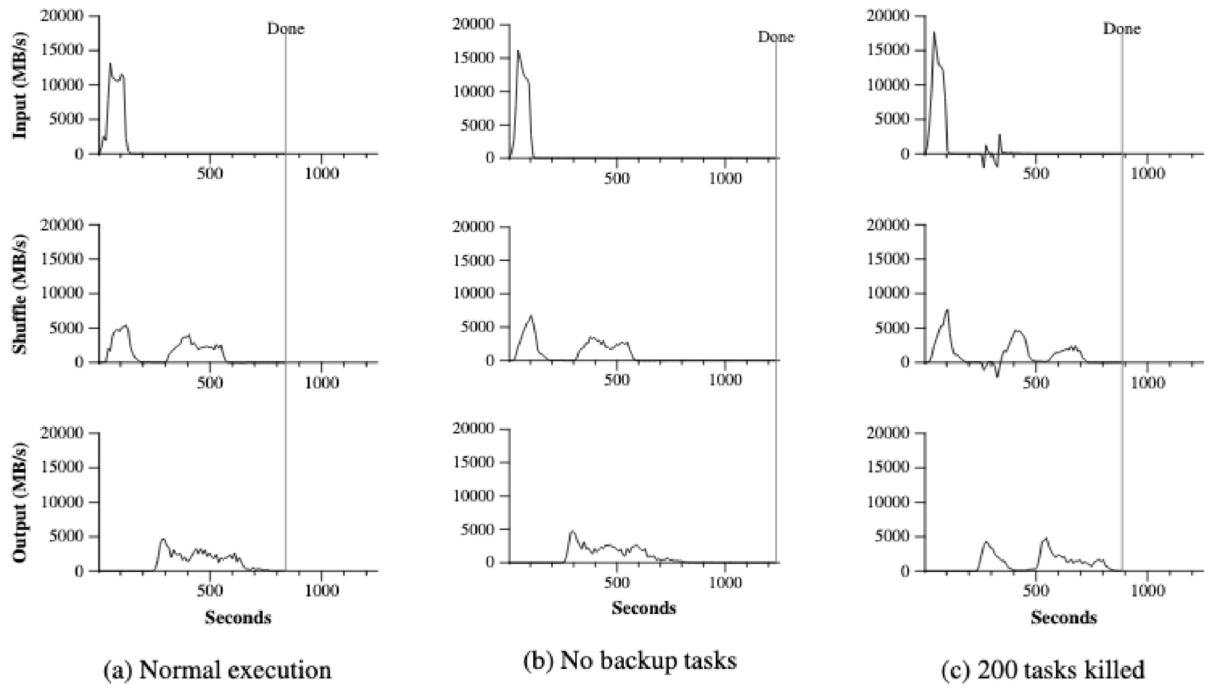
Usually, exams have something fun that ties them together. But not always. In this case, there is one thing that units this exam: it's just several questions about map and reduce, some small, some bigger, and hopefully, if we're lucky, unusual enough to take you somewhere new in your thoughts. Think, then answer. Good luck!

Grading Page

	Points	Total Possible
Part A		10
Part B		10
Part C		20
Total		40

Part A [10 points]: MapReduce

MapReduce (MapReduce: Simplified Data Processing on Large Clusters, by Dean and Ghemawat) stores the Map output on the local disk of the Map worker. A Reduce worker that needs a Map worker's output fetches the intermediate data over the network directly from the Map worker. In contrast, the Map inputs and Reduce outputs are stored in the GFS (Google File System) replicated distributed file system.



The figure above shows the three cases of executing a (Tera)Sort MapReduce job in the MapReduce paper.

A1. [5 points]: Under normal execution (a), explain why the shuffle phase and output (reduce) phase observed much lower aggregate throughput.

shuffle rate and the output rate has much lower aggregate throughput compared to input rate because of the locality optimization – most data is read from a local disk and bypasses the relatively bandwidth constrained network. And also, the shuffle rate is higher than the output rate because the output phase writes two copies of the sorted data (two replicas of the output are made for reliability and availability reasons provided by the underlying file systems).Network bandwidth requirements for writing data would be reduced if the underlying file system used erasure coding rather than replication.

A2. [5 points]: Under the execution with no backup tasks (b), explain why the overall execution duration is much longer than that of normal execution.

there is a very long tail where hardly any write activity occurs. After 960 seconds, all except 5 of the reduce tasks are completed. However these last few stragglers don't finish until 300 seconds later. The entire computation takes 1283 seconds, an increase of 44% in elapsed time.

Part B [10 points]: Spark

A developer wants to do interactive debugging using Spark. He implemented a Spark program shown as below:

```
1. lines = textFile("hdfs://foo.log")
2. errors = lines.filter(_.startWith("ERROR"))
3. errors.persist()
4. errors.count()
5. errors.filter(_.contains("MySQL")).count()
6. errors.filter(_.contains("HDFS"))
       _.map(_.split("\t")(3))
       .collect()
```

B3. [5 points]: Mark the lines in the above program, during whose execution, an RDD is computed and consumed. (Briefly explain your answer.)

Until the third point, no work is performed on the cluster.
From fourth point the user can use the RDD in actions, here, to count the number of messages:

```
4. errors.count()
```

user can also perform further transformations on the RDD and use their results, here,to count the errors mentioning MySQL:

```
5.errors.filter(_.contains("MySQL")).count()
```

and, also,return the time fields of errors mentioning // HDFS as an array (assuming time is field // number 3 in a tab-separated format):

```
6. errors.filter(_.contains("HDFS"))
       _.map(_.split("\t")(3))
       .collect()
```

B4. [5 points]: Consider the following scenario. The input file is partitioned into 256 pieces which are all stored and replicated in HDFS. This program is executing line 6 on a cluster of 64 worker machines (assuming those Spark worker machines are different than the HDFS servers). One worker machine loses power and does not restart. Explain what data (if any) must be re-computed to recover. (Briefly explain your answer.)

<u>PageRanking:</u>
The algorithm iteratively updates a rank for each document by adding up contributions from documents that link to it. On each iteration, each document sends a contribution of r n to its neighbors, where r is its rank and n is its number of neighbors. It then updates its rank to α/N + (1 – α)∑ci , where the sum is over the contributions it received and N is the total number of documents.
This program leads to the RDD lineage graph and On each iteration, we create a new ranks dataset based on the contribs and ranks from the previous iteration and the static links dataset. in a job with many iterations, it may be necessary to reliably replicate some of the versions of ranks to reduce fault recovery times.
The user can call persist with a RELIABLE flag to do this. However, note that the links dataset does not need to be replicated, because partitions of it can be rebuilt efficiently by rerunning a map on blocks of the input file.
This dataset will typically be much larger than ranks, because each document has many links but only one number as its rank, so recovering it using lineage saves time over systems that checkpoint a program's entire in-memory state.
the join operation between links and ranks will automatically aggregate the contributions for each URL to the machine that its link lists is on, calculate its new rank there, and join it with its links.
<u>Representing RDDs:(Resilient Distributed Datasets)</u>
dependencies in RDD are classified into two types: narrow dependencies, where each partition of the parent RDD is used by at most one partition of the child RDD, wide dependencies, where multiple child partitions may depend on it. For example, map leads to a narrow dependency, while join leads to to wide dependencies (unless the parents are hash-partitioned).
recovery after a node failure is more efficient with a narrow dependency, as only the lost parent partitions need to be recomputed, and they can be recomputed in parallel on different nodes.. In contrast, in a lineage graph with wide dependencies, a single failed node might cause the loss of some partition from all the ancestors of an RDD, requiring a complete re-execution.

Part C [20 points]: Project & Feedback

C5. [5 points]: Given the current status of your project, describe how you would like to expand on it in the future.

> Since this is an *open-ended* question, you can answer it from any perspective, e.g., more sophisticated features that you'd like to incorporate to the existing implementation, performance optimizations to improve the efficiency of the current prototype, or any interesting insight that you discovered during the process.

Findings:
Importance of data distribution for Indexes and for efficiency of data prediction.
The usage of neural networks CNN in machine learning models helps us train the model and predict accurately. It also allows us assess the false positive rate for both approaches.
how the trained bloom filter distinguishes between keys and non-keys, as well as how it performs in classification and hash model techniques, when compared to the baseline classical bloom filter.
Future work:
Ideally, the traditional bloom filters should have zero false negative rate, but the current value is 0.569 the project and the query latency is 0.95s for bloom filters using Hash Model approach, so need to explore and fine tune the current implementation in order to achieve more optimized value.
I would like to explore more methodologies and languages like C++ in order to reduce the memory overhead caused by tensor flow deep learning framework.
To test the real world datasets like MNIST and LETOR and to check for the performance metrics for the model.

C6. [5 points]: Which papers did you find most useful?

[ ] Serverless SFS scheduler
[ ] ghOSt
[ ] ARC caching policy
[ ] InfiniCache
[ ] Google MapReduce
[ ] RDD and Spark

C7. [5 points]: Check the box if you think that would better help improve the learning experience.

[ ] Refresher on fundamentals about processes, fork, and threads

[ ] More paper readings about computer systems research
[ ] More paper readings about OS and less on distributed systems/datacenters
[ ] More weight on exams / homework assignments
[ ] More weight on OS programming labs and less weight on open-ended projects

fundamentals ,research papers ,exams and the project work provided has helped me to a larger extent in learning and improving my knowledge in OS.

C8. [5 points]: Consider the transition of CS 571 from "***pure OS textbook oriented + closed-ended OS lab assignments***" to "***a mix of OS fundamentals + systems paper readings/discussion + open-ended project***", what went well and what could be improved?

"a mix of OS fundamentals + systems paper readings/discussion + open-ended project" *requires constant strong efforts and lot of patience in reading and understanding the concepts of the research papers and while implementing the open-ended project mind gets to think from various perspectives of a problem/study for improving the efficiency and optimization of certain metrics and exposed to the real-world domains and added to it are the exams that enlightened on more clarity on the concepts ranging from rudimentary to complicated level. and the class discussions were fun and engaging.*
On the whole, the curriculum is quite challenging but was really worth the experience especially under the guidance of the professor.!!.

[Don't forget to fill the online teaching evaluation form at https://gmu.bluera.com/gmu/ , which is due at 11:59pm, 05/08]- Filled!

End of Exam.

This page is intentionally left blank.