ASSIGNMENT 3

Map Reduce

DESIGN DETAILS:

MASTER: It is a TCP server built using python3 which handles connections from clients and perform map reduce operations on the basis of input provided by the client. At the moment Master only supports "Word Count" and "Inverted Index" operations on the given files. The Master also provides support to handle and serve multiple clients at once based on the capacity of hardware it is being run on. Master instantiates an object of MapReduce class which is implemented as a library that higher level applications can use for data processing, with the data provided from the user and performs the task requested using concepts of Map and Reduce. To start the master, a script can be run as follows:

./start_server.sh

MapReduce: This is a library implementation of Map Reduce concept that can be used by higher level applications for data processing. It is instantiated by passing the number of mappers and reducers as arguments to the object. It is called as:

cluster = MapReduce(num_of_mappers, num_of_reducers)

The library can then be run by calling the runMapReduce() method and passing input file location, map function, reduce function and output location as parameters like:

status = runMapReduce(input_file, map_func, reduce_func,output_file)

This method returns the status of actions performed by the file, it returns "Completed" if task is successful. The method starts with splitting the input file depending on the number of mappers and storing it in a temporary file. Then all mappers are started as parallel processes with each mapper being fed different chunk of the input file. The output of each mapper is combined once all mappers complete their process in a single file. Next, we find all the unique keys and start all the reducers as parallel processes. Each reducer gets a single key and performs reduce operation for that key. Once a reducer finishes operation for one key, another key is fed to it to perform the same operation for that one. Once all the keys are processed among the reducers we stop the reducers and the results of each reducer are stored in a separate temporary file. As a last operation results of all reducers are combined as a single file and stored in the output file provided by the client with all temporary data deleted.

<Bonus Component> Fault Tolerance: As a bonus component, fault tolerance is implemented in the system. When the master fails for any reason, it sends response to the client to retry and resend all the input parameters for the master to run again from the beginning. If a mapper fails, it will simply restart and read from the chunk of data assigned to it from the input file. If a reducer fails, all the reducers will restart without any problem as it reads from a temporary middleware file which stores results of all the mappers and is deleted only when all the reducers have done processing.

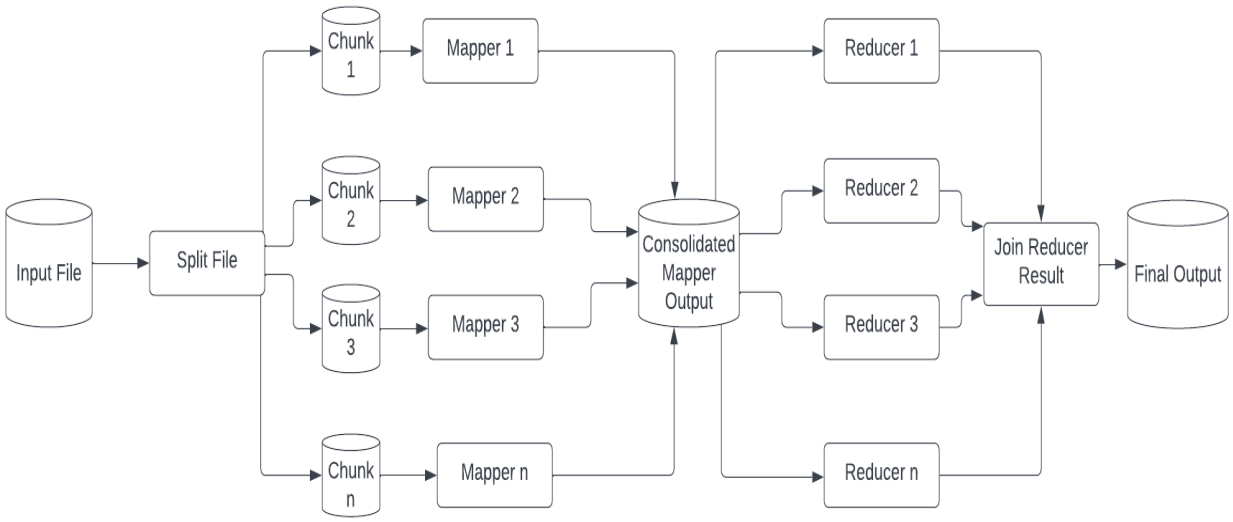
<Bonus Component> Protocol Buffers: As a bonus component, all the results of the mappers follow a same protocol with which to pass the intermediate data to reducers. This helps in consolidating the results of mappers for easier running of reducers and work as a temporary storage for reducer input to also support fault tolerance for reducer failures.

ConfigFile: It is a configuration file which holds parameters to run the server and client in any environment easily. Since I am using Socket programming and not using a separate implementation for storing the data in a separate process, the only common parameters are server name and server port on which the master server will be running. To change the parameters, simply change the values in ConfigFile.py.

src > ConfigFile.py > ...

```
1  # Change the port number where you want to run the master server
2  server_port_num = 9999
3
4  # Change the server name where you want to run the master server
5  server_name = 'localhost'
```

FLOW CHART:



TESTS:

All the test cases are included in the tests folder which holds 3 test files. To run all the test files, we simply run a script as follows:

./run_tests.sh

There are three test files which tests the application as follows:

1. *client_inverted_index*: This file passes input parameters to the library to run inverted index application on the input file. The purpose of this test is to check if inverted index runs correctly on a file.
2. *client_word_count*: This file passes input parameters to the library to run word count application on the input file. The purpose of this test is to check if word count runs correctly on a file.
3. *Multiple_clients*: This file tests if the master is able to serve multiple clients at once while performing different applications for each client.

The results of this script are as follows:

```
**********************************************************************************************************
TEST FILE TO PERFORM INVERTED INDEX
**********************************************************************************************************

Status of Inverted Index operation:
Completed
**********************************************************************************************************


**********************************************************************************************************
TEST FILE TO PERFORM WORD COUNT
**********************************************************************************************************

Status of Word Count operation:
Completed
**********************************************************************************************************
```

```
**********************************************************************************************************
TEST FILE TO CONNECT MULTIPLE CLIENTS TO THE MASTER SERVER
**********************************************************************************************************

Connecting client 1
Connecting client 2
Response from client 1:
Completed
Response from client 2:
Completed
Closeing client 1
Closeing client 2
**********************************************************************************************************
```

Sample Output File for Inverted Index:


```
{"abated": [[3, 6]], "abbott": [[0, 12]], "abide": [[4, 6]], "able": [[2, 12], [3, 6]], "about": [[0,
168], [1, 180], [2, 96], [3, 90], [4, 66]], "above": [[0, 6], [1, 12]], "absolute": [[0, 6]],
"accented": [[4, 6]], "accept": [[3, 6], [4, 6]], "accepted": [[4, 12]], "accepting": [[4, 6]],
"access": [[0, 6], [4, 60]], "accessed": [[4, 6]], "accessible": [[4, 6]], "accompanied": [[3, 6]],
"accompany": [[0, 6]], "accordance": [[4, 12]], "according": [[0, 12]], "accordingly": [[0, 24], [1, 18]
, [2, 6], [3, 30], [4, 6]], "accosted": [[0, 6]], "account": [[0, 6], [1, 12]], "accurately": [[2, 6]],
"accustomed": [[0, 30], [1, 6], [3, 6]], "acid": [[4, 18]], "acorn": [[1, 6]], "acorns": [[1, 6]],
"acquainted": [[1, 6], [3, 6]], "acquired": [[2, 6]], "across": [[0, 12], [1, 18], [2, 12], [3, 66]],
"act": [[0, 6], [1, 6], [3, 12]], "acted": [[3, 6]], "acting": [[3, 6]], "active": [[4, 12]], "acts": [
[2, 6]], "actual": [[4, 6]], "actually": [[3, 6]], "add": [[1, 6]], "added": [[0, 18], [1, 36], [2, 12],
[3, 12], [4, 6]], "addition": [[4, 6]], "additional": [[4, 18]], "additions": [[4, 6]], "address": [[4,
6]], "addresses": [[4, 6]], "adjust": [[1, 6]], "adjusting": [[1, 6]], "admiring": [[2, 6]], "admit": [
[1, 6]], "adopted": [[0, 6], [3, 6]], "advance": [[3, 6]], "advanced": [[1, 12], [2, 12], [3, 12]],
"advancing": [[0, 6], [2, 6], [3, 12]], "advantage": [[1, 6], [3, 6]], "adventure": [[2, 6], [3, 6]],
"advice": [[1, 6], [2, 12]], "affair": [[4, 6]], "afford": [[0, 6], [3, 6]], "affording": [[3, 6]],
"afraid": [[0, 6], [1, 12], [2, 18], [3, 66]], "after": [[0, 108], [1, 84], [2, 96], [3, 96], [4, 6]],
"afternoon": [[0, 6], [2, 12], [3, 54]], "afternoons": [[0, 6]], "afterwards": [[1, 12], [3, 6]],
"again": [[0, 72], [1, 54], [2, 126], [3, 102], [4, 18]], "against": [[0, 12], [1, 24], [2, 12], [3, 18]
, [4, 6]], "age": [[0, 6], [3, 6]], "aged": [[4, 6]], "agent": [[4, 6]], "agitated": [[1, 6]],
"agitation": [[4, 6]], "agree": [[4, 54]], "agreeable": [[0, 6]], "agreed": [[1, 12], [2, 12], [4, 6]],
"agreement": [[4, 108]], "air": [[3, 6]], "alarm": [[2, 6], [4, 6]], "alarmed": [[3, 18]], "all": [[0,
222], [1, 330], [2, 438], [3, 282], [4, 126]], "allow": [[0, 12], [2, 6], [4, 6]], "allowed": [[0, 6]],
"almost": [[0, 18], [1, 18], [2, 6], [3, 6], [4, 6]], "alone": [[0, 30], [2, 30], [3, 42], [4, 6]],
"along": [[0, 114], [1, 60], [2, 174], [3, 120]], "aloud": [[0, 6], [2, 6]], "already": [[0, 6], [4, 6]]
, "also": [[0, 18], [1, 12], [2, 12], [4, 12]], "alteration": [[4, 6]], "altered": [[3, 6], [4, 6]],
"alternate": [[4, 6]], "although": [[0, 12], [1, 6]], "altogether": [[0, 6]], "always": [[0, 48], [1, 12]
, [2, 36], [3, 6], [4, 6]], "amends": [[3, 12]], "american": [[0, 6]], "among": [[1, 18], [2, 18], [3,
54]], "amuse": [[0, 12], [2, 6]], "amused": [[0, 6], [1, 6], [3, 6]], "amusement": [[3, 12]],
"amusements": [[0, 6]], "amusing": [[0, 12]], "and": [[0, 1404], [1, 2046], [2, 1914], [3, 2052], [4,
630]], "anew": [[3, 12]], "ann": [[0, 6]], "anna": [[2, 12]], "anne": [[0, 174], [1, 174], [2, 114], [3,
```

Sample Output File for Word Count:


```
{"abdicate": 6, "abide": 6, "able": 42, "ably": 6, "about": 564, "above": 78, "abroad": 12, "abruptly":
6, "abruptness": 6, "absence": 18, "absolute": 6, "absolutely": 6, "absurd": 18, "absurdity": 6,
"absurdly": 12, "abuse": 6, "abused": 6, "abusing": 6, "accept": 6, "accepted": 24, "accepting": 12,
"access": 60, "accessed": 6, "accessible": 6, "accidental": 12, "accompanied": 6, "accompaniment": 6,
"accompany": 6, "accomplished": 6, "accordance": 12, "according": 6, "accordingly": 18, "accordion": 6,
"account": 48, "accounts": 6, "accumulated": 6, "accustomed": 6, "achievement": 6, "acquaintance": 18,
"across": 96, "act": 66, "acted": 6, "acting": 12, "active": 12, "actual": 6, "actually": 6, "adams":
12, "add": 6, "added": 102, "addition": 6, "additional": 18, "additions": 6, "address": 6, "addressed":
6, "addresses": 6, "admiration": 6, "admired": 12, "admirer": 6, "admirers": 12, "admiring": 6,
"admiringly": 6, "advance": 12, "advanced": 18, "advancing": 6, "advantage": 18, "adventure": 30,
"adventures": 36, "adversaries": 12, "advertise": 6, "advertisements": 6, "advertising": 12, "advice":
12, "advise": 6, "advisers": 6, "afar": 6, "afeard": 6, "affair": 6, "affairs": 12, "affected": 6,
"affixing": 6, "afford": 6, "afore": 6, "afraid": 78, "africa": 6, "african": 12, "after": 336,
"afternoon": 66, "afterward": 6, "again": 264, "against": 162, "age": 24, "aged": 6, "agent": 18,
"aggrieved": 6, "aglow": 6, "ago": 30, "agonizing": 6, "agree": 72, "agreed": 30, "agreement": 120,
"agrees": 6, "agricultural": 12, "ahead": 36, "aid": 6, "aim": 6, "aimed": 6, "aiming": 6, "air": 72,
"alarm": 6, "albert": 390, "alden": 6, "alexander": 6, "algerine": 6, "alike": 6, "alive": 12, "all":
1074, "alleys": 6, "allow": 30, "allowed": 42, "alma": 12, "almost": 114, "alone": 78, "along": 54,
"aloud": 12, "already": 102, "also": 66, "alteration": 6, "altered": 6, "alternate": 12, "alternatives":
6, "although": 24, "altogether": 12, "alumnus": 6, "always": 96, "amateur": 6, "amazement": 6,
"ambassadors": 6, "amend": 6, "america": 12, "american": 204, "among": 90, "amongst": 12, "amount": 24,
"amused": 6, "amusement": 6, "amusements": 12, "anchor": 18, "and": 10824, "anger": 6, "angry": 66,
"animal": 18, "animals": 30, "ankle": 84, "ankles": 18, "annapolis": 12, "anne": 12, "annex": 6,
"annexation": 6, "annexed": 6, "annexing": 12, "announce": 6, "announced": 18, "annoyed": 6, "annual":
6, "another": 162, "answer": 48, "answered": 84, "answering": 6, "answers": 12, "anticipated": 12,
"ants": 6, "anxiety": 18, "anxious": 66, "anxiously": 48, "any": 504, "anybody": 18, "anyhow": 6,
"anyone": 30, "anything": 156, "anyway": 6, "anywhere": 12, "apart": 6, "apiece": 6, "apollinaris": 6,
"apology": 6, "apparatus": 6, "apparent": 6, "apparently": 18, "appeal": 18, "appear": 36, "appearance":
36, "appeared": 30, "appearing": 6, "appears": 6, "appetites": 6, "applauding": 6, "applicable": 18,
"applications": 6, "apply": 6, "appoint": 18, "appointed": 24, "appreciate": 12, "appreciated": 12,
```


LIMITATIONS:

- The application does not support arbitrary map and reduce functions so at present it is currently limited to Word Count and Inverted Index.
- The datastore does not run on a separate node from the MapReduce library and is integrated into the library.
- Since the datastores are not running on a separate node, some file paths had to be hardcoded and all input files need to be present inside the “Data” folder.
- The number of clients that can be connected to the master node at once is limited to the configurations of the physical system on which the server will be hosted.

FUTURE SCOPE:

- The application can be upgraded to allow arbitrary map and reduce functions to run smoothly.
- A separate datastore as a node can be implemented so that the application can work on a more distributed setting and increase performance.