CS 425 Distributed System MP4 Report

Group 69: Yash Saboo (ysaboo2), Hess Parker I (pihess)

I. Design

In this MP, we have two components – master and worker. There is one Master (each for Maple and Juice) which allocates tasks to workers (single worker program which performs Maple or Juice, based on the request from Master) running on multiple machines. As the number of workers increases, the distributed nature of MapleJuice also increases. Master and Workers are connected over TCP connection, for reliability, for message transfers.

The design is built in such a smart manner that if one needs to change the type of application that needs to be performed using MapleJuice, all one needs to do is add a simple function in worker program using the already available utilities. Master program for both Maple and Juice remains the same no matter what application is run on it. This helped us design the second application quite easily, since we just had to focus on the application logic. We use Hashmaps to keep track of the (key, value) pairs which are used by Maple and Juice. We have already implemented (string, integer) pair for Word Count and (string, string) pair for Reverse Web Link, which can be further utilised for other applications too. Thus, MapleJuice is an easy to use model.

During Maple, Master partitions the file based on the number of workers available, which is identified using MP2 code. The partitioned files are acted upon by worker nodes upon receiving the request from Master. This request comes along with the operation that needs to be performed -Maple or Juice- and also, what kind of application needs to be run - in our case, Word Count and Reverse Web Link. Upon receiving the request, worker checks that the message received is of type “Map”, so it invokes the maple functionality of the given application it is supposed to perform. Once it finishes, it puts the maple output file on distributed file system, using the MP3 code, and also sends an indicator to Master whether the map task alloted has been completed or not. Meanwhile, the Master waits for the worker to finish the task. After a certain amount of time, it reflects whether all the maple tasks were successful or not. If not, then it assigns the failed maple tasks to available workers. Considering the previous MP codes, it is only able to sustain three worker failures.

During Juice, the sequence of steps remains quite similar to that of Maple, but with few added functionality. Here, the shuffling is done using two methods - Hash partitioning and Range partitioning - which is a user’s choice as expressed in the terminal command. After juice sends the message to workers in the same format as Maple does, but with the operation tagged as “Reduce”. Upon receiving the request, worker performs the task and securely transfers the file, which is then combined (simple operation of append) by the Master to generate the final MapleJuice output. We give user a choice to whether the files generated by map needs to be deleted or not, which is taken care of by Master. Also, we have added sorting functionality so that we can compare our output with the Hadoop output in a better manner. The last functionality of the master juice is to combine all the reducer outputs into one single file when all the workers have successfully completed the task, and store it in the file path specified by the user.

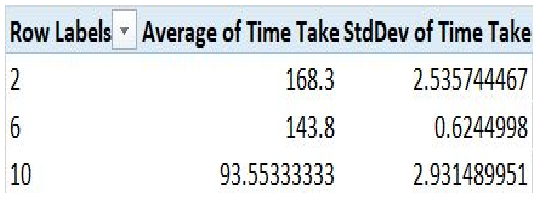
MP1 was used to debug the loggings done by masters and workers to identify where the code was breaking, why we were getting too many files open error and whether hashmap was working as expected.

II. Applications

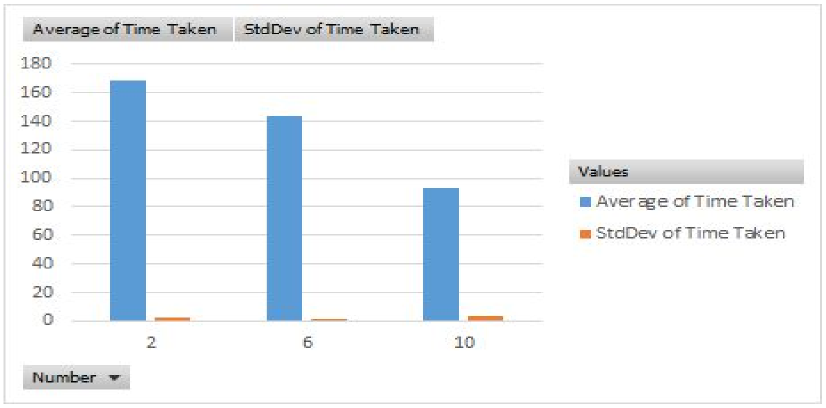
Both the applications have plot which discusses the time (in seconds) taken for the MapleJuice to run on 2, 6 and 10 VMs. 3 readings were taken, with mean and standard deviation measured along with it.

A. Word Count

Dataset Used: MIT words Dataset iterated randomly for 300 different keys to get 100MB file.
Row Labels refers to number of VMs used for both Maple and Juice.

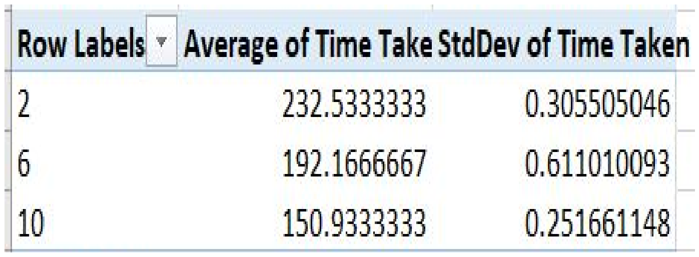


Row Labels	Average of Time Take	StdDev of Time Take
2	168.3	2.535744467
6	143.8	0.6244998
10	93.55333333	2.931489951

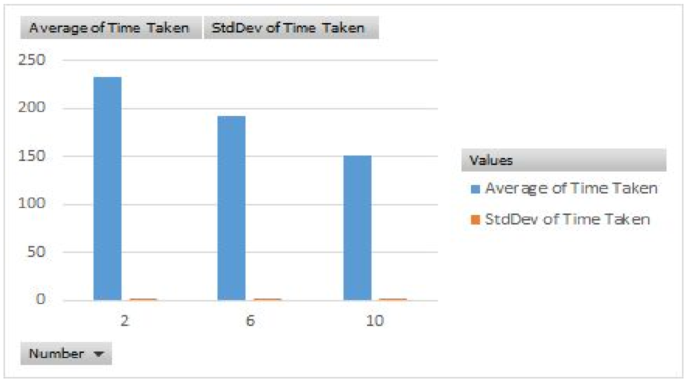


B. Reverse Web Link Graph

Dataset Used: Dataset Stanford SNAP Repository with more than 500 different keys.
Row Labels refers to number of VMs used for both Maple and Juice.



Row Labels	Average of Time Take	StdDev of Time Taken
2	232.5333333	0.305505046
6	192.1666667	0.611010093
10	150.9333333	0.251661148

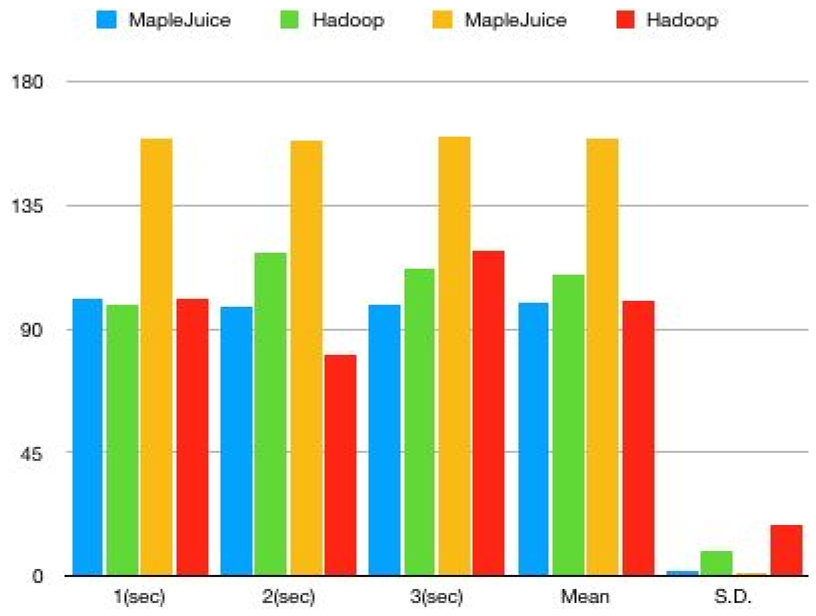


III. Hadoop vs MapleJuice

The performance of Hadoop and Maple Juice was compared keeping the same topology and dataset. We used 9 VMs for both. The time is in seconds. Row Labels are the readings.
First column is the readin number

	WordCount		Reverse Web Link	
	MapleJuice	Hadoop	MapleJuice	Hadoop
1(sec)	101.2	99.119	159.2	101.4
2(sec)	98.36	117.57	158.4	80.967
3(sec)	98.52	112.17	160.4	118.67
Mean	99.36	109.62	159.33333	100.34
S.D.	1.5954937	9.4855	1.0066446	18.872

WordCount: Colors - Blue and Green
Reverse Web LinkColors - Yellow and Red



IV. Conclusion/Inference

MapleJuice runs better as the number of VMs are increased. It's quite obvious since, the task gets distributed into multiple machines which leads to less run time and higher performance. Also, one thing we noticed was that juice takes less time than maple, and the reason would be that the hashmap generated for maple would be massive compared to juice's hashmap, since the keys for maple are in random order and may not be repetitive in nature which causes the hashmap to be massive, while the keys which the juice are meant to be repetitive since the same keys are supposed to go at the same VM (key ‘a’ can go to VM 1 and 2 for maple, but key ‘a’ will only go to VM 1 for juice), that is the whole logic of map-reduce.

While comparing MapleJuice and Hadoop, the factors which can play role is the number of distinct keys and the way to handle them. In case of Word Count, MapleJuice runs faster than Hadoop because we have significantly less keys to handle compared to Reverse Web Link. MapleJuice takes significant time in assigning the files/keys to the workers, so as the keys increases, the amount of time MapleJuice takes also increases, and by a significant amount. This is why MapleJuice works well for WordCount, but not for Reverse Web Link.