NetId1: yaoxiao9 NetId2: xiaoxin2

Group81-Project Report
CS425-MP4

Design

Our design is based on the sdfs system in MP3 and change the put function to an concatenate instead of overwriting, using two phases(maple and juice) to apply a certain application on some files. For maple phase, the input is maple, maple_exe, num_maple, sdfs_intermediate_filename_prefix, sdfs_src_directory. Maple is to invoke the maple phase, the last parameter specifies the location of the files, sdfs_intermediate_filename_prefix provides the location of intermediate files of maple phase and the intermediate files are stored according to the key, num_maple is equal the number of VM, which is the number of machines that are responsible for doing the maple task together, maple_exe is the application that we want to apply. For juice phase, the input is juice, juice_exe, num_juice, sdfs_dest_filename, delete, partition. Juice is to invoke the juice phase, juice_exe is the application to do the juice task, sdfs_dest_filename is the location to store the final results, delete is allowed to have two values: 0 or 1, 0 means to delete the intermediate files, and 1 means to reserve the intermediate files, partition is allowed to have two values: hash or range, which is the shuffle method to assign the juice tasks to VMs when the juice phase starts.

Architecture and Programming Framework

For maple phase, master is responsible for split the target file into small pieces and evenly assigning the maple tasks to every VMs. Each VM maintains some lines of the original file and applies the application on them, accordingly, a file with the key in the intermediate result would be generated, and the intermediate result would be sent to the file by using “put” function. Since we have changed the “put” function into concatenate, after all the VMs finish their tasks, the final output would be stored over the sdfs system.

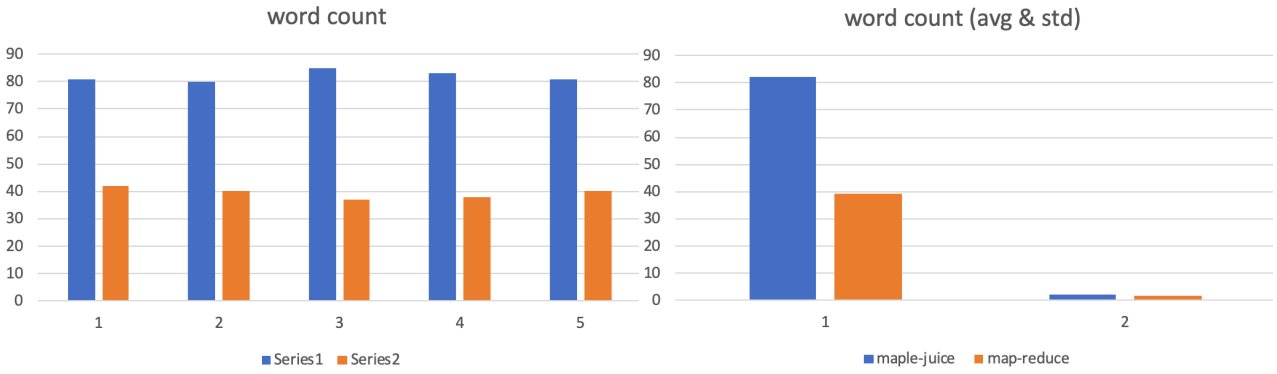
For juice phase, master would use the partition method that we provide to shuffle the key-targeted files and assign them to every VMs. Each VM would first use “get” command to fetch those key-targeted files that they are responsible for, and then apply the juice application on these files and use “put” command to append the result to the destination file that is stored on sdfs. In this way, after all the VMs finish their juice task, the destination file in sdfs system would maintain all the result information.

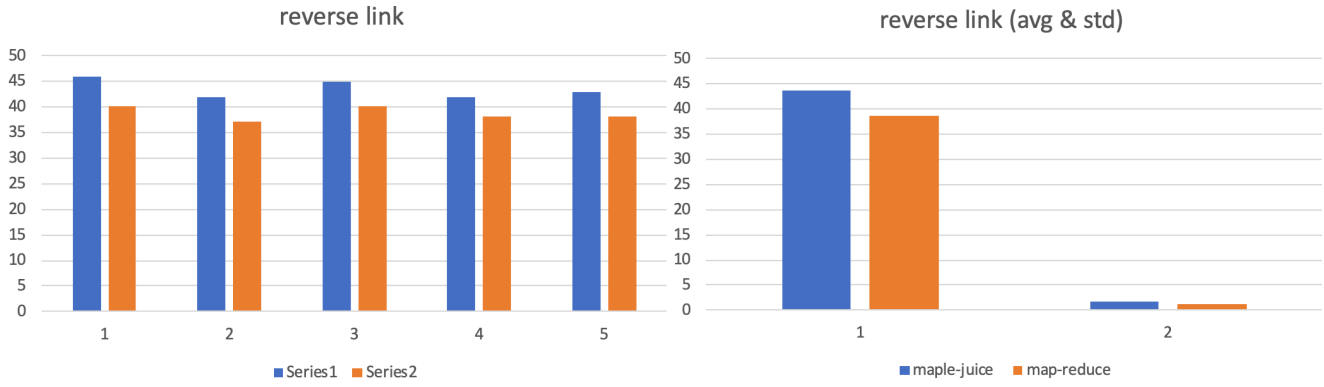
What makes this maple-juice task more simply is the modified “put” and “get” function in sdfs system. We change the “put” function to concatenate instead of overwrite, these two commands enable us to avoid extra concatenating after each phase, but just invoke these two command, and each piece of result would automatically go into the sdfs system.

NetId1: yaoxiao9 NetId2: xiaoxin2

Measurement

run on 5 VMs, caculated by seconds								
		1	2	3	4	5	avg	std
maple-juice	word count(46 lines)	81	80	85	83	81	82	2
	reverse link(106 lines	46	42	45	42	43	43.6	1.81659021
map-reduce	word count(46 lines)	42	40	37	38	40	39.4	1.94935887
	reverse link(106 lines	40	37	40	38	38	38.6	1.34164079





Analysis

1. For maple-juice, reverse link application takes shorter time than word count application. It meets our expectation, because though reverse link application has longer lines, its application is much simplier than word count. And for map-reduce, the time almost has no difference, it is because the size of these two files does not differs a lot and has almost no influence on the operation time of hadoop.
2. Maple-juice always runs longer than map-reduce. It meets our expectation, because the maple-juice is based on our sdfs system and whenever we put or get, we use 4 replicas. But for the reverse link application, these two systems just have slight difference on running time, which shows that the complexity of application matter a lot.

Conclusion

What matters the operation time more is the complexity of application, and when the file size is relatively small, it just has slight influence on the running time for map-reduce system.