Easwaran Ramamurthy
eramamur@andrew.cmu.edu

TREXQUANT TAKE HOME PROJECT REPORT

NQTVITCH PARSING

RUNNING THE CODE:

Compile the three source files CalculateVWAP.java, Util.java and StockAndPrice. java using javac in UNIX.

Run the java code using the following UNIX command:

java CalculateVWAP <in_file> | sort -s -k1,1 > <output_file>

CODE DESCRIPTION:

The first part of the pipe using CalculateVWAP prints the running VWAP scores as and when a trade is executed to System.out. However, the output of this, although sorted in terms of time stamp is not grouped by the stock name. Therefore, it is essential to sort the output by stock name to group the output by stock and we use the Unix sort function that runs parallely and is much more efficient than alternative approaches. A detailed discussion of this is given below in the section titled Efficiency.

The output buffer of CalculateVWAP is flushed after every 5000 lines and piping of this output to the Unix sort function allows it to run parallely (saving lot of time and memory) and give the output that is sorted by the stock name. The Unix sort is suitable for this purpose since it sorts input as and when it gets new input. It then puts the sorted output into temporary spill files which are then read later and merged with output of new incoming input. This allows it to run parallely and makes full use of the hard disk and in the process, saving us lot of memory.

DESCRIPTION OF THE OUTPUT:

This particular version of the sort function sorts only by the first key which is the stock name. It does not sort by the second key and so it preserves the original time ordering of the trades.

An example output file is supplied in the file final_output.zip. You can extract the file and view the running VWAPs for all stocks.

The output file is a tab separated file formatted as follows:

Stock Symbol	P	V	$\sum V$	$V * P$	$\sum (V * P)$	Running VWAP $\sum(V * P) / \sum V$

where P and V represent the price and volume of the stock at that particular trade. The sort step in the code allows the VWAP scores for a particular stock to be grouped together in the output for easy viewing.

Alternatively, to only view the running VWAPs for a particular stock, you can use the following UNIX command:

cat final_out.txt | grep <stock_name>

Easwaran Ramamurthy
eramamur@andrew.cmu.edu

EFFICIENCY:

Although the big-O time complexity of this approach is technically $O(nlogn)$ where n is the number of trades, the parallel sorting and the use of less memory makes it much more efficient than other approaches.

Some alternative less efficient $O(n)$ time approaches include approaches that calculate and store all the VWAPs in memory and then finally output to disk (highly inefficient in terms of memory) or else open multiple output files for each stock and output as and when a new trade comes in (which is very slow because of too much disk I/O).

My code takes 17m to run on a Intel Core i7-4700MQ @ 2.40GHz CPU with 8 GB ram and is much faster and memory efficient than these alternative approaches that I implemented and tried out. All commands were run using the Cygwin environment for Windows that mimics Unix functionality.

ANOTHER ALTERNATIVE APPROACH (Using MapReduce):

A possibly more efficient approach could be the use of MapReduce. However, there are some complications to this approach since the sharding of files could mean that entries for the same order reference go to different Map workers.

Therefore, we would need to do two MapReduces sequentially. The first Map could map each trade message to a <orderRef, message> key value pair. This would allow each record for a particular order reference to go to the same reduce worker. Once we get to the reduce worker, we can iterate through the messages for a particular order Reference in time sorted order and output <stockName, timeStamp, volume, price> with stockName as the key.

The second MapReduce would then have an identity Mapper which just outputs the <stockName, timeStamp, volume, price>. The Reduce step would involve iterating through the entries for a particular stockName in timestamp sorted order. We can then keep track of the running VWAPs and output them to a file.

This approach would require access to a MapReduce framework and a distributed computing cluster and is something that can be attempted in the future.