Approach to Implement

SON is implemented w/ Apriori administered within each chunk (Corresponding to the number of partitions for the RDD). mapPartition is used in both steps 1 and 2 to act on each chunk rather than the entire dataset. For step 1 of SON, a reduced threshold value of s_thres/num_partitions is used to filter candidate itemsets while the entirety or s_thresh is used for step 2. For each iteration, broadcasting is used to cache the results of step 1 of SON within each worker node. A listbuffer stores the result of the filtered itemsets derived from the previous iteration. It is cleared after candidates can be calculated and is filled with the output of SON step 2. The program stops when the output of the previous iteration produces a result such that there cannot be any further frequent itemsets.

Command Line Command

For test/small datasets:

```
./spark-submit --class Pradeep_Lam_SON ./Pradeep_Lam_SON.jar <input_path>
<support_thresh> <output_path>
```

For large dataset:

```
./spark-submit --driver-memory 4g --class Pradeep_Lam_SON ./Pradeep_Lam_SON.jar
<input_path> <support thresh> <output_path>
```

Scala Version: 2.11
Spark Version: 2.3.1

Problem 2 Execution Table

Support Threshold	Execution Time
500	9.858043596
1000	6.689766506

Problem 3 Execution Table

Support Threshold	Execution Time
100,000	452.81005826
120,000	404.241586058

Bonus Question

Without such a large support thresh, there would be much too many combinations of itemsets to consider and, most likely, the machine would hang/run out of memory.

Empirically, for the large input, I still had to increase the memory given to spark from the default amount (Even with these high thresholds). The main problem with larger files is there is a chance that you'll produce a lot more singletons. And pairs and onward, grow dramatically with the number of singletons.

A bottleneck can be identified with the use of broadcasting to store the output of SON part 1 locally in every worker. A worker has limited memory and if the thresh is too small (Or the dataset is too large), then the worker will have to pull/push to/from disk.