1. Compare the application's completion time.
   When I use the old application it takes 20 min. However, when i use Graphx it takes only 2.1 minutes.
2. Compute the amount of network/storage read/write bandwidth used during the application lifetimes.
   Disk usage after execution

1239555 1818 473533674 39886092 11039102 5462140 615622016 206995232 0 45066280 246880052

Disk usage before execution

1228438 1818 471420682 39773580 11029406 5458680 607518016 205126944 0 45011356 244899272

Readen Disk usage is (473533674 - 471420682)*512B = 1031.73 MB
Written Disk usage is (615622016 - 607518016)*512B = 3957.01 MB

Inter-| Receive | Transmit
face |bytes packets errs drop fifo frame compressed multicast|bytes packets errs drop fifo colls carrier compressed
eth0: 2259246247498 1952780758 0 0 0 0 0 0 3670008704332 749321870 0 0 0 0 0 0
lo: 1276754754580 94746917 0 0 0 0 0 0 1276754754580 94746917 0 0 0 0 0 0

Inter-| Receive | Transmit
face |bytes packets errs drop fifo frame compressed multicast|bytes packets errs drop fifo colls carrier compressed
eth0: 2256257917574 1950642709 0 0 0 0 0 0 3666770137588 749080585 0 0 0 0 0 0
lo: 1276190006074 94717713 0 0 0 0 0 0 1276190006074 94717713 0 0 0 0 0 0

Receive: 2259246247498 - 2256257917574= 2849.893/212=12mb/s
Transmit: 3670008704332 - 3666770137588 = 3088.5379/212 = 14.5686mb/s

3. Compute the number of tasks for every execution.

719 tasks

4. Does GraphX provide additional benefits while implementing the PageRank algorithm? Explain and reason out the difference in performance, if any.

Yes, Graphx improve the performance a lot. Based on cut schema pf pregel, which will use more storage for computation but much smaller bandwidth. Graphx also provide extra optimization base the design of Graphx based on RDD.

Application 2:

Find the number of edges where the number of words in the source vertex is strictly larger than the number of words in the destination vertex. Hint: to solve this question please refer to the Property Graph examples from here:

Used the triplets API, use reduction to get the result. Total number of edges:76.

Find the most popular vertex. A vertex is the most popular if it has the most number of edges to its neighbors and it has the maximum number of words. If there are many satisfying the above criteria, pick any one you desire. Hint: to solve this question please refer to the Neighborhood Aggregation examples from here. You can ignore the frequency of the words for this question.

First need to sum up the count of neighbors at each vertex and using a custom max function.

Then we can get the most popular vertex : The id of the most popular vertex is: 1694

Find the average number of words in every neighbor of a vertex. Hint: to solve this question please refer to the Neighborhood Aggregation examples from here.

First we calculate the number of words and the count of the words. Then we use a reduce function to compute the average.

Extra Credit

Find the most popular word across all the tweets:

For each vertex using the triplets API to sum up comma separated string. Then we used the standard word count program to determine the maximum popular word.

Find the size of the largest subgraph which connects any two vertices:

We used the graphX connectedComponents construct. after get this output, we used mapReduce to get the size of the maximum connected component :

Find the number of time intervals which have the most popular word.

First we determine the most popular word. Then we use the triplet api to get total number for each vertex. Then we used map and reduce to get the number of vertices containing this popular word.

The Number of time intervals with most popular word: 80