CME 323: Distributed Algorithms and Optimization

Instructor: Reza Zadeh (rezab@stanford.edu)
TA: Anuj Nagpal (anujnag@stanford.edu)

HW#4 - Due 1st June

1. **PageRank Computation:** Consider a network of websites given to you as a directed graph $G = (V, E)$ with adjacency matrix A and a out-degree matrix D such that you can define $Q = D^{-1}A$:

   (a) Power Iteration: Assume that you have a $n \times n$ matrix M with all eigenvalues distinct. For any vector r_0, prove that the iterated product $M^k r_0$ converges to the eigenvector corresponding to largest eigenvalue of M as $k \to \infty$. Make sure to state any constraint assumed for proving this convergence.

   (b) In the pagerank lecture notes, it is stated that: *The stationary distribution specifies what proportion of time on average is spent at specific node during an infinitely long random walk.*

   With this infinitely long random walk analogy with transition probability of the random walker given by matrix Q, describe two different scenarios where the random walker can 'get stuck' despite every node in the graph having atleast one incoming edge. Draw an example graph for each scenario and explain how adding some teleportation probability from each node to every other node will help.

   (c) Prove that the stationary distribution r of node probabilities for matrix

   $$P = \alpha Q + (1 - \alpha) \left[\frac{1}{N}\right]_{N \times N}$$

   satisfies the pagerank equation:

   $$r_j = \sum_{i \to j} \alpha \frac{r_i}{deg_i} + (1 - \alpha)\frac{1}{N}$$

   (d) Write a Spark program to compute pageranks for the following graph given as *(source, destination)* edge list. Report your rank value for each URL after 20 iterations. Assume you start with an r_0 having all 1-s and you compute rank vector iteratively as:

   $$(r_j)_{t+1} = \sum_{i \to j} 0.85 \frac{(r_i)_t}{deg_i} + 0.15$$

```
google.com wikipedia.org
stanford.edu google.com
youtube.com stanford.edu
youtube.com google.com
wikipedia.org youtube.com
wikipedia.org google.com
```

2. **Singular Value Decomposition:**

   (a) Write a Spark program to compute the Singular Value Decomposition of the following 10×3 matrix M:

```
-0.5529181 -0.5465480 0.009519836
-0.5428579 -1.5623879 0.982464609
-1.3038629  0.5715549 0.499441144
 0.6564096  1.1806877 0.495705999
-1.2061171  1.3430651 0.153477135
 0.2938439 -1.7966043 0.914381381
-0.2578953  0.2596407 0.815623895
 0.9659582  2.3697927 0.320880634
-0.4038109  0.9846071 0.488856619
 0.6029003 -0.3202214 0.380347546
```

   The matrix M should be stored in a row matrix format i.e. the rows should be split up and inserted into an RDD. Report all singular vectors and values and submit your Spark program. You can use Spark's MLlib library for this problem.

   (b) The eigenvalue decomposition of a real, symmetric, and square matrix A (of size $d \times d$) can be written as the following product: $A = Q\Lambda Q^T$ where $\Lambda = diag(\lambda_1, \ldots, \lambda_d)$ contains the eigenvalues of A (which are always real) along its main diagonal and Q is an orthogonal matrix containing the eigenvectors of A as its columns.
   Can we do eigenvalue decomposition of $M^T M$?

   (c) Prove that the nonzero eigenvalues of MM^T are the same as the nonzero eigenvalues of $M^T M$. You may ignore multiplicity of eigenvalues. Are their eigenvectors the same?

   (d) Compute the eigenvalue decomposition of $M^T M$. What is the relationship (if any) between the eigenvalues of $M^T M$ and the singular values of M?

3. Given a matrix M in row format as an RDD[Array[Double]] and a local vector x given as an Array[Double], give Spark code to compute the matrix vector multiply Mx.

4. In class we saw how to compute highly similar pairs of m-dimensional vectors x, y via sampling in the mappers, where the similarity was defined by cosine similarity: $\frac{x^T y}{|x|_2 |y|_2}$. Show how to modify the sampling scheme to work with overlap similarity, defined as

$$\text{overlap}(x, y) = \frac{x^T y}{\min(|x|_2^2, |y|_2^2)}$$

   (a) Prove shuffle size is still independent of m, the dimension of x and y.

   (b) Assuming combiners are used with B mapper machines, analyze the shuffle size.

5. **Shallow Graphs** For an undirected graph $G = (V, E)$ with n vertices and m edges ($m \geq n$), we say that G is shallow if for every pair of vertices $u, v \in V$, there is a path from u to v of length at most 2 (i.e. using at most two edges).

   (a) Give an algorithm that can decide whether G is shallow in $O(n^{2.376})$ time.

   (b) Given an $n \times r$ matrix A and an $r \times n$ matrix B where $r \leq n$, show that we can multiply A and B in $O\left((n/r)^2 r^{2.376}\right)$ time. Hint: use the fact that we can multiply two $r \times r$ matrices in $O(r^{2.376})$ time.

   (c) Give an algorithm that can decide whether G is shallow in $O(m^{0.55} n^{1.45})$ time. Hint: consider length-2 paths that go from low-degree vertices and length-2 paths that go through high-degree vertices separately. Use result from part (b).