W261

Office Hours:

Saturday 3:00 - 4:00 (Kyle)

Sunday 12:00 - 1:00 (James)

Monday 8:30 - 9:30 (Maya)

Week 1

Questions:

1. Difference between MapReduce and Hadoop
2. Functional programming and MapReduce (2 things)
   1. Functions can be taken as parameters and functions are stateless (have no side effects)
3. What is the data structure for MapReduce?
   1. Data Structure for Map-Reduce is Key-Value Pairs. We don’t want to use dictionaries because it might be larger than memory.
4. How is that helpful in a
   1. Keep data and or Code together so you don’t have to move data around
5. Race Condition (In a distributed process)
   1. Order matters between which sub processes finish first
6. How does MapReduce handle syncrhonization - what stage and what’s it called?
   1. MapReduce uses barrier method. Won’t makes processes wait for each other
7. Where does sort happen and what is the default sort behavior?
   1. Before reduce
8. What is the most expensive?
   1. Shuffle/Sort is most expensive because you actually have to move the data (into order). The moving of the data is what’s most expensive.

Week3

Broadcasting vs. encapsulating

Sending broadcast variable you send 1 copy to each node

Encapsulate you send the data 1000 times to the worker nodes (1 for each task in the node) Every time the node does a task it gets the data again.

Week 5

Use accumulators in actions. Move accumulator to action so that if a node fails, you don’t get the transform to process the accumulation twice.

A priori - filters out a lot of data. Only looks at frequent item sets and not single item sets.

Properties of well behaved graph - Aperiodic and Irreducible (There is a path from every node to every other node)

Period of a cycle is how long it takes to get back to where we start. (If our graph is periodic, the idea that where we start doesn’t affect the probability is violated).

Inverted index is the data structure we use to parallelize. Queries as keys and our documents are our postings ordered by page rank.

Termination criteria (when the top ranks stop changing, or when the change in the page ranks is smaller than some epsilon, or use an arbitrary number of iterations)

content and collaborative filtering two main approaches?

Content filtering: demographic information or answers provided on a suitable questionnaire. The profiles allow programs to associate users with matching products

Collaborative filtering analyzes relationships between users and interdependencies among products to identify new user-item associations.

What is the cold start problem?: Collaborative filtering

Content filtering: You are your demongraphic i.e. profiles users

collaborative filtering: assumptions: you are your actions

User based vs Item based approach.

Use cosine similarity or Pearson approach

Dimensionality Reduction is a good solution to issue of too many features.

A priori algorithm - Most frequent matches between movie A and movie B. The movies that are most watched together. Not great approach though

Latent Factor: breaks down similarities between two users or two movies on specific factors to group them in terms of similarities. A vector of numbers that describes a user or item based on certain features.

Implicit feedback = feedback we determine from user behavior. Explicit feedback is given specifically by user. Explicit feedback - not many people would take a survey or give feedback rating for movies so this would be a sparse ratings matrix. The implicit feedback would result in a dense ratings matrix. We would have a lot of entries into the matrix and a lot of it might not be very useful.

SVD - singular value decomposition decompose our matrix into 3 sections, single value right, left and center. SVD won’t work if matrix is sparse. Must be a complete matrix. Let’s say we use a implicit feedback and have a dense matrix. This will be computationally very challenging to solve this problem at scale.

Collaborative Filtering:

ALS is a dimensionality reduction. Factor a large matrix into 2 smaller ones. Optimizing the values within U and V so that their product best matches the values in R. Mean squared error between R and U\*V.

Why don’t we use gradient descent to solve that loss function? Not a convex algorithm because we have an infinite number of solutions.