Evaluating Spark on Azure and AWS by K-Means Algorithm

Andrew Zhou
Department of Data Science
University of Washington
Seattle, WA 98195
ajz55@uw.edu

1 Proposal

1.1 Introduction

Today there are many competitors in the cloud computing world. One of the most popular analytical engines for big data is Spark. For my proposal, I would like to try the k-means algorithm on a pre-determined set of centroids on Azure Spark and AWS Spark. My goal is to compare and contrast the running times of Spark on Azure and AWS. I will not be looking at k-means clustering performance, but the running time of the algorithm on different sizes of data and number of worker clusters.

1.2 Objectives

The two systems that will be under analysis is Spark on Azure and AWS. They will be performing a k-means clustering task on a data set of different sizes. I will also be running the k-means clustering algorithm on different number of worker clusters.

The goal is to compare and contrast the systems and their spark computing power.

1.3 Data set

To conduct the analysis for this project, I will be using large-scale astronomical imaging surveys found at Sloan Digital Sky Survey. The schema of the data is as follows:

Column	Value
ID	Unique Identifier for each source
X	Attribute for source
Y	Attribute for source
Z	Attribute for source
W	Attribute for source

This data set includes 15M sources of data which we will use the K-Means classifier to cluster the data points. This data set can be found as a CSV file in S3. The file can be loaded directly from S3 using Spark. If it fails to load, the csv file can also be imported locally and used directly. More details of the data can be found at Sloan Digital Sky Survey

1.4 Algorithms and Techniques

1.4.1 K-Means Algorithm

For this experiment, I will be running the K-Means clustering algorithm. The K-Means algorithm aims to partition data X to k clusters $P = \{P_1, ..., P_k\}$, by minimizing the sum of squared L^2 distances between every data point and the centroid of associated with the cluster.

$$\min_{C,P} \Phi(C,P) = \min_{C,P} \sum_{i=1}^{k} \sum_{x \in P_i} ||x - c^{(i)}||_2^2$$

The cost function can be calculated below:

$$\phi(C) = min_P \Phi(C,P) = \sum_{x \in X} \min_{c \in C} ||x - c||_2^2$$

The algorithm for K-Means clustering can be found below:

Algorithm 1 k-means Algorithm

```
1:  procedure K-MEANS
2:      Select k points as initial centroids of the k clusters.
3:      for iteration := 1 to MAX_ITER do
4:          for each point x in the data set do
5:              Cluster of x ← the cluster with the closest centroid to x
6:          end for
7:          for each cluster P do
8:              Centroid of P ← the mean of all the data points assigned to P
9:          end for
10:         Calculate the cost for this iteration.
11:     end for
12: end procedure
```

1.4.2 Techniques

I will be testing Azure and AWS spark sessions with different sizes of data. The dataset I will be using has 15M data points. I plan on running the k-means algorithm with sizes 1M, 3M, 5M, 10M, 15M. I'll also be varying the number of worker clusters to see how the number of clusters (perhaps of sizes: 2, 4, 8, 16) for Azure and AWS affect the running time of the k-means cluster algorithm.

1.5 Evaluation

My main evaluation metric is the amount of time it takes to run the k-means algorithm on different sizes of data and number of worker clusters. I plan on plotting multiple line graphs of Azure and AWS running times on different sizes of datasets. The graph will have the running time on the y-axis and the size of the dataset on the x-axis. There will be multiple graphs for each number of cluster workers. I also plan to graph all plots into one graph to compare all sizes of data sets and number of clusters run, separated by color.

I will also be leaving raw running time output as a table.