Lab 3

Smart grid technologies to manage energy generation

Kiran Prabhakar | UBIT: kprabhak | Person No: 50287403

Shishir Suvarna | UBIT:shishirs | Person No: 50290573

Abstract

Since the invention of energy generation, there was just one way transmission of energy from the sources to consumers, which doesn't provide any streaming information about the energy consumption. Smart Grid technology provides two way power and data transmission, to provide balance between real time production, regulation and transmission of energy sources. The generation is monitored using Apache Spark leveraging the polled data from smart meters which form an Advanced Metering Infrastructure (AMI). The connected utilities in smart grid network forms a graph, which allows us to use page-rank technique to determine optimal transmission with minimum cost. To solve real time transmission failure, based on existing data, a Machine Learning model is used to determine the next optimal path.

How Spark is used?

- As per the analysis from government organization, using smart meters and power can collect around three petabytes of data a month for about one million households. Apache spark is used to load and transform large-scaled utility data.
- The GraphX and MLLib libraries in Spark framework is helpful for performing page rank and decision tree.
- The streaming API from Apache can be used to ingest real time data from various utilities present in the Grid.

Problem statement

Minimizing the cost of transmission

Importance: The initial setup of the grid requires a mechanism to identify the optimal path from the energy sources to the consumers, thus minimizing the cost and loss during transmission.

Resolution: By using the **pagerank** algorithm from **GraphX** library in Spark, the best optimal path can be obtained to initially setup the Grid network.

Impact: By leveraging the streaming data, real time update of transmission path is done to attain optimal transmission.

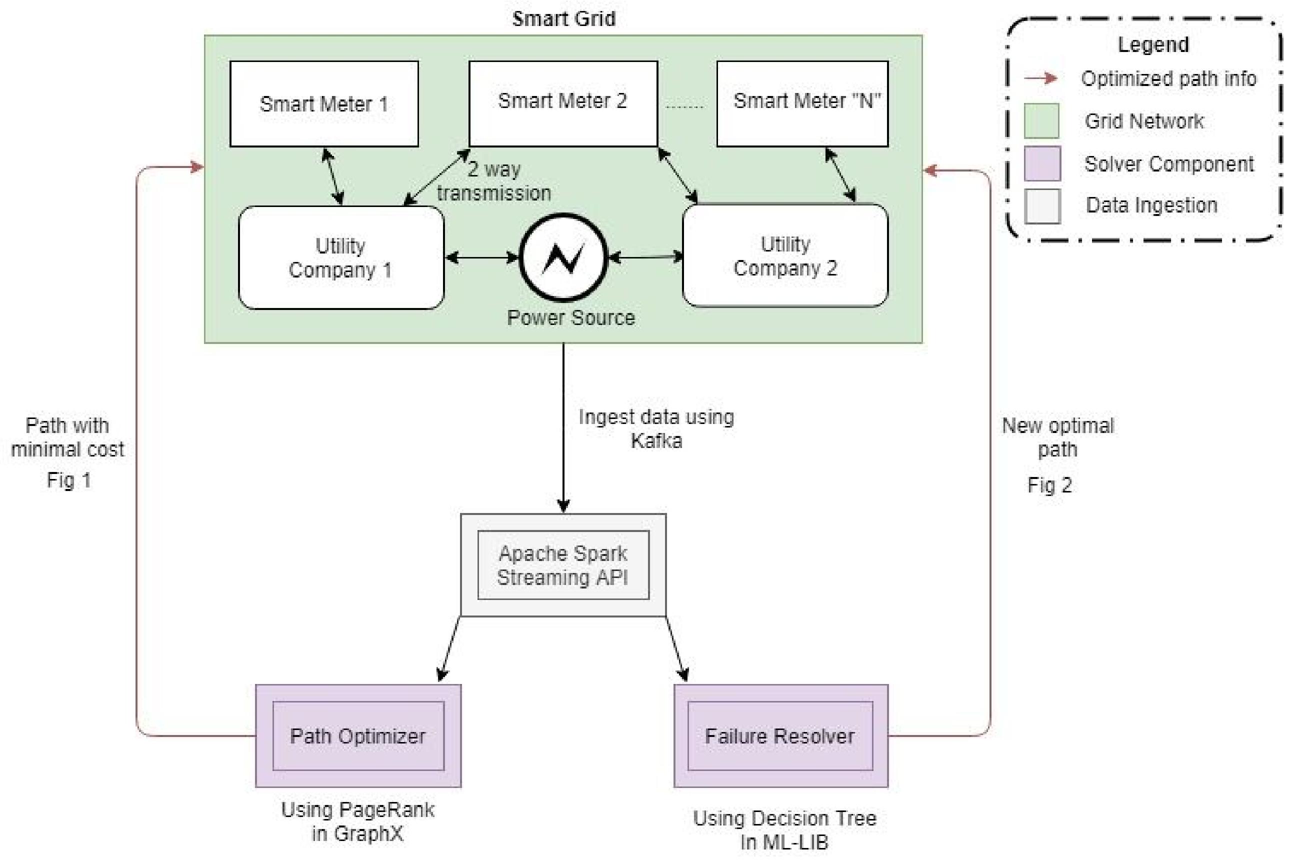
Identifying the optimal path during Power Plant failure

Importance: The key feature of smart grid network is to ensure availability.

Resolution: To achieve the optimal path for transmission during failure, a **Decision tree** model is trained using Spark's **MLLib** on all available paths. The input for the model will be from the real time data, which will provide instant resolution during failure.

Impact: As a result of this the energy transmission, the path for few of the utilities in the grid will be updated, by simultaneously ensuring high availability.

Architecture Diagram



Data and Metadata

Where is it?

For the current project, in order to work with small dataset, the data is taken from the publicly available data sources and API endpoints, from the below websites.

- SmartGrid.gov

- Open Energy Information (OEI)
- Advanced Metering Infrastructure (AMI)

What is it about?

The common data points, which serve as metadata from the above data sources, are processed to form a consolidated data by reducing the dimensionality. This will be later used as features in pagerank and Random forest algorithms. The data points are described as below.

Advanced Metering Infrastructure (AMI)

The AMI data set contains the information about the following, the dataset provides information about the data ingested from the meters in the infracture

- UtilityType - Which specifies about the type meter installed at the customer unit.
- CustomerType- ResidentialCustomer, CommercialCustomer, IndustrialCustomer with counts.
- EnergyConsumed - The KW units of energy consumed by the utility, provided by meter.
- NercRegion - The energy supply division of the grid system.

Transmission

The transmission data points provides information about the **TransLineMiles** and **TransmissioinCost** which can be used while calculating pagerank algorithm.

Distributed Energy Resources (DER)

The DER provides information about the type of energy resource, the cost associated with smart meters: **EnergyProduced**, **AmiBackOfficeCost**, **AmiCommEquipCost**. Using the cost information the optimal path is derived.

Size of Dataset and API details

For analysing the project feasibility, a subset of dataset from Open Energy Information (OEI), which provides **22,685** records of the information about energy produced and consumed in the smart grid. However, as described earlier, the real time streaming data from the meters counts to **three**

petabytes of data a month for about one million households. This proves the necessity for using big data pipeline to handle the huge data.

The Rest API endpoint, that gives the information about the meta data for getting insights about the energy production and utilization in the grid system.

Sample Json

```
{
        “utilityType”: “Cooperative Utility”
        “consumerType”: “Commercial consumer“
        “transLineMiles”: “140”
        “transmissioinCost”: “248”
        “energyConsumed”:”534”
        “nercRegion”: ”MRO”
}
```

Solution

Minimizing the cost of transmission

The initial step is to convert the grid network in to a graph where nodes are the utilities and the meters in the grid and the edges are the connection between them. The weight of each edge can be calculated as the net amount of flow of energy though the edge. The flow is calculated as below.

Cost= TransLineMiles +TransmissioinCost - EnergyConsumed

The data from the grid is processed in the below format before it is processed for page rank to identify the optimal path. The data for each utility type can be grouped to improve **parallelism**. The can The sample processed dataset is in the below format.

Utility Type	Source Utility(KW)	Destination Meter (KW)	TransLineMiles	TransCost	Energy Consumed
U1	100	2	200	450	240
U2	1000	10	670	830	300

The sample algorithm for the finding optimal path is as below. The implementation can be done by using the **runUntilConvergence** GraphXLib in spark.

```
public static <VD,ED> Graph<java.lang.Object,java.lang.Object>
runUntilConvergenceWithOptions(Graph<VD,ED> graph,
                                              double tol,
                                              double resetProb,
                                              scala.Option<java.lang.Object> srcId,
                                              scala.reflect.ClassTag<VD> evidence$7,
                                              scala.reflect.ClassTag<ED> evidence$8)
Algorithm:
# initialize spark conext
 sc = SparkSession.builder.getOrCreate()
# load all utility data from
stream_data = getStreamData() # Performing streaming using Kafka
 utilities= stream_data .load()
# calculate initial cost based on data available
 netCosts = utilities.map(lamdba {
        Cost = TransLineMiles +TransmissioinCost - EnergyConsumed
        Return(utility, cost)
}
#compute pagerank until convergence.
while(convergence)
Do
        Contributions = utilities.join(cost).flatMap( lambda (utility, cost/total_utilities))
        updatedCost = contributions.reduceByKey(utility).map(lambda cost)
# update the ranks with new cost.
For each path
do
Cost [i] = updatedCost[i]
```

After running the above algorithm, the updated weights give the optimal path for transmission. The algorithm is applied continuously for streaming data, which includes inclusion and exclusion of meters in the grid. So, the optimal path for the energy transmission is obtained in real time.

Identifying the optimal path during failure

The MLLib provides Decision tree package, which follows a greedy strategy at every hop in the grid to identify the optimal path during transfer. A Sample algorithm is as below, **MLLib.DecisionTree** should be used for actual implementation.

```
# initialize spark conext
 sc = SparkSession.builder.getOrCreate()
# load all utility data from
stream_data = sc.getStreamData('sourceAPI') # Performing streaming using Kafka
 utilities= stream_data .load() # the data is loaded as [(source_utility, des_meter), cost]
train_data,test_data = data.randomSplit(Array(0.7, 0.3)) # train and test splits
# (source_utility, des_meter) is independent variable and cost is dependent variable.
Categorical_features = Map[int,int] # Yes or No decision
Model = DecisionTree.train_classifier(train_data)
model.predict(path) # gives yes or no decision for the path.
```

By greedily relying on predicted decisions from the classifier, the optimal path during failure is found.

Visualizations

Figure 1: Page Rank

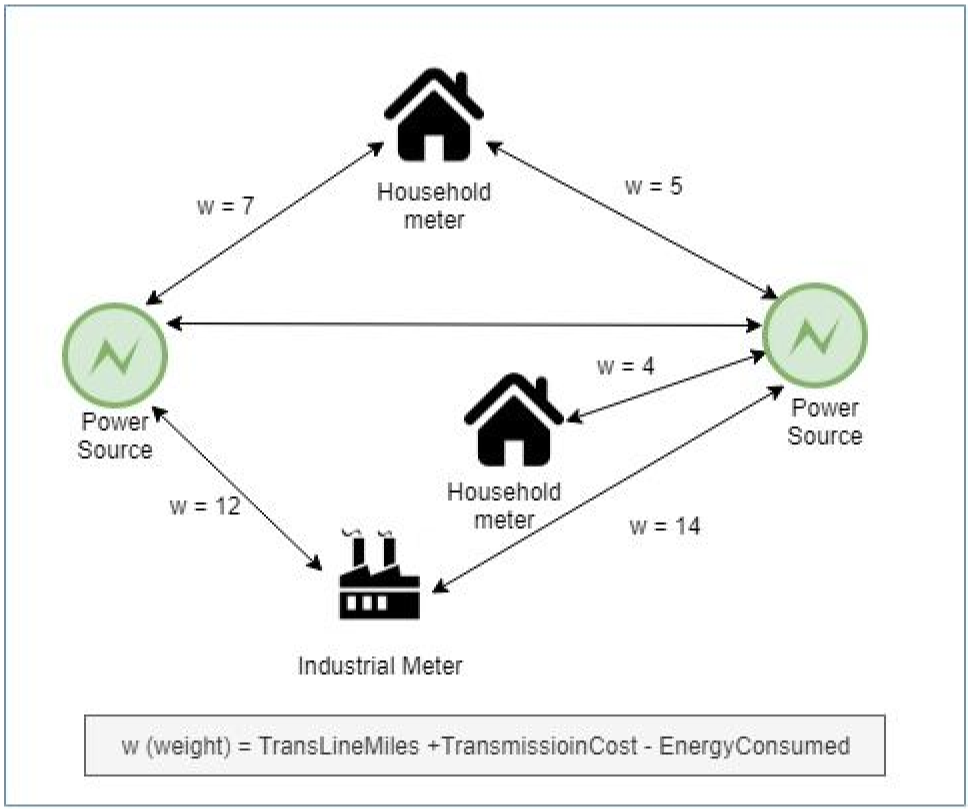
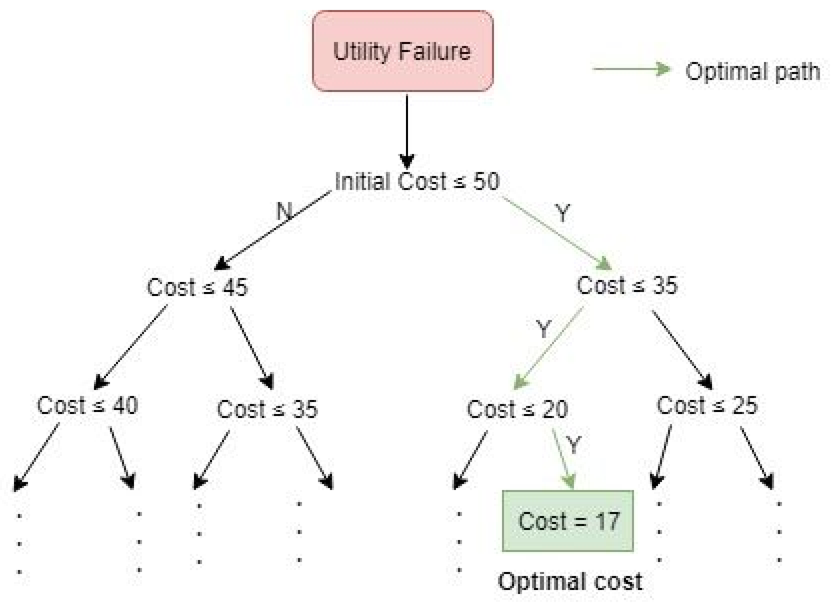
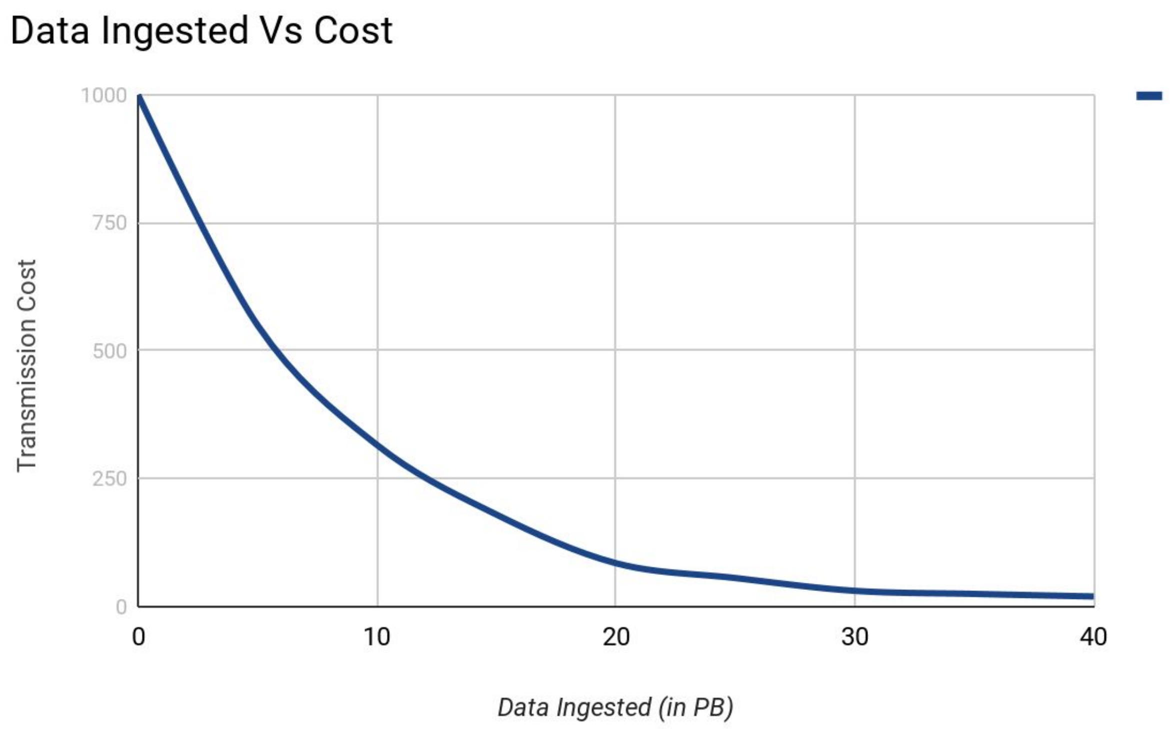


Figure 2: Decision Tree



Data Vs Transmission Cost



From the above graph, it is evident that the transmission cost can be optimized if a large amount of data is ingested into the system. Using the high volumes of data, the Decision tree classifier will be **less susceptible to bias**. This provides best optimal path.

Summary

Thus, by utilizing the spark framework and by setting up a big data pipeline for streaming, loading and processing the real time data from the smart meters, the two main goals of availability and efficiency can be achieved. The current approach can also be extended to other energy sources to provide optimal solutions for transmission related issues.

References

- Science Redirect
- Streaming Data In Smart Grid
- Spark Documentation
- SmartGrid.gov