Harmonizing Communities: Exploring Deezer's Dataset for Distributive Analysis and Community Detection

Niharika R Pakhi Singhal Dhrithi K

***Abstract*—In the age of data, music streaming service platforms such as Deezer collect and analyze user inter-relationship data. From this data, we can find trends and clusters within the music world. This project focuses on studying music communities on Deezer using distributed computing methods. By utilizing Hadoop and Spark frameworks we Study the dataset over a range. Community detection algorithms help identify clusters of users who share music tastes and habits. Besides, we apply machine learning methods to explore further how these communities interact. The results are presented visually to offer insights, into the structure and arrangement of the music network. Through this Project, our goal is to enhance our understanding of how users engage with each other in the field of music streaming and also how distributive computing works.**
In the age of data, music streaming service platforms such as Deezer collect and analyze user inter-relationship data. From this data, we can find trends and clusters within the music world. This project focuses on studying music communities on Deezer using distributed computing methods. By utilizing Hadoop and Spark frameworks we Study the dataset over a range. Community detection algorithms help identify clusters of users who share music tastes and habits. Besides, we apply machine learning methods to explore further how these communities interact. The results are presented visually to offer insights, into the structure and arrangement of the music network. Through this Project, our goal is to enhance our understanding of how users engage with each other in the field of music streaming and also how distributive computing works.

Index Terms: Community detection, Distributed computing, Page Rank Algorithm, Connected Component algorithm and Triangle Count Algorithm.

INTRODUCTION

In this project, we used the Deezer dataset, which is a global music streaming service that includes user data and friendship networks of users from 3 European countries namely Romania, Croatia, and Hungary. In this dataset, nodes represent the users and edges are the mutual friendships. It contains 41,773 nodes and 125,826 edges for Romania,54,573 nodes and 498,202 edges for Hungary, and 47,538 nodes and 22,887 edges for Croatia.

We used this dataset for performing community detection in it.

b. Apache Spark is a distributed processing system and open- source analytics engine for faster processing and analysis of large datasets.

c. Hadoop played an important role in this project, it stores and processes data, integrates with other tools, and even provides fault tolerance.

d. In this project we used Spark-based Connected Component Algorithm and Triangle Count Algorithm for detecting communities or high modularity partitions of large networks and unveiling underlying structures because of its efficiency, simplicity, and scalability.

PAGE RANK ALGORITHM

The PageRank algorithm treats the web as a vast network of interconnected pages. Each page is represented on the web as a node with links between pages at the edges. The basic principle of PageRank is that a page is considered more important if other vital pages link it. The algorithm determines the initial PageRank value for each web page. This initial value can be uniform or based on certain factors, such as the number of incoming links to the page. The algorithm then repeatedly calculates the PageRank value of each page, taking into account the PageRank value of the pages that are related to the pages. During each iteration, the PageRank value of the page is updated based on the sum of the PageRank values of the incoming links. Pages with more inbound links have a more significant impact on the landing page's PageRank.

CONNECTED COMPONENT ALGORITHM

We used this algorithm to analyze dataset because it is fast, and provides high modularity output. This algorithm computes connected components for a given graph. Connected components are the set of its connected subgraphs. Two nodes belong to the same connected component when there exists a path (without considering the direction of the edges) between them. Therefore, the algorithm does not consider the direction of edges. The number of connected components of an undirected graph is equal to the number of connected components of the same directed graph. This algorithm tries to handle the dynamics of the graph, trying not to recompute all from scratch at each change (kind of re-optimization). In this way, each instance of the algorithm is registered as a graph sink. Each change in the graph topology may affect the algorithm. For the initial computation, let n be the number of nodes, then the complexity is 0(n). For the re-optimization steps, let k be the number of nodes concerned by the changes (k <= n), the complexity is O(k).

TRIANGLE COUNT ALGORITHM

Triangle counting algorithm is used to detect communities and measure the cohesiveness of those communities. It can also be used to determine the stability of a graph, and is often used as part of the computation of network indices, such as clustering coefficients. The Triangle Count

Algorithm is also used to compute the Local Clustering Coefficient.

HOW IS OUR WORK DIFFERENT FROM OTHERS?

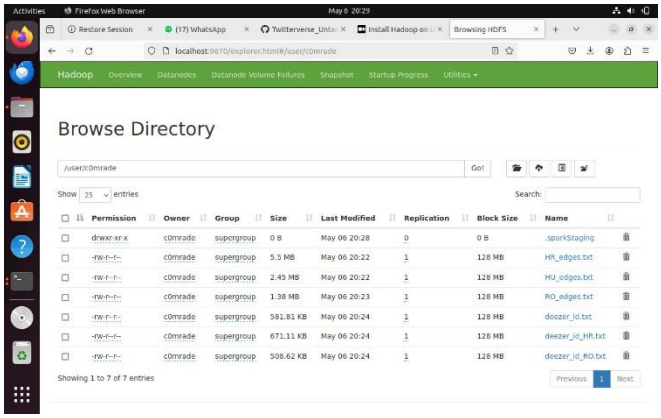
Our work can answer several questions like:
-How do user communities evolve over time on Deezer?
-What factors influence the formation of cross-genre communities on Deezer?
-How do social connections influence music discovery and recommendation on Deezer?
-What impact do user-generated playlists have on community engagement and cohesion within Deezer?
-How do cultural factors influence the formation of music communities within different regions or demographics on Deezer?
-What role do algorithmic recommendations play in shaping user communities and content consumption patterns on Deezer?
-How do user engagement metrics vary across different com- munity structures and user segments on Deezer?
So, by addressing these questions through empirical analysis and data-driven insights, our project can contribute valuable knowledge to the field of music streaming platform research and inform strategic decision-making for Deezer's platform development and user experience enhancements.

WHAT ARE WE DOING IN THIS PROJECT?

As mentioned above we are making use of Hadoop, an open-source framework that helps to process and store large amounts of data. Hadoop uses the MapReduce to process data. Hadoop has a distributed file system (HDFS), meaning that data files can be stored across multiple machines. The file system is scalable, since servers and machines can be added to accommodate increasing volumes of data. We are even making use of Apache Spark, a multi-language engine that uses machine learning nodes. Spark uses resilient distributed datasets (RDDs). We are using Spark mainly for computation. And we have even setup our computers for clustering that would show distributed computing on our machines that would make clusters of clusters.

What is HDFS?

HDFS stands for Hadoop Distributed File System. HDFS operates as a distributed file system designed to run on commodity hardware. HDFS is fault-tolerant and designed to be deployed on low-cost, commodity hardware. HDFS provides high throughput data access to application data and is suitable for applications that have large data sets and enables streaming access to file system data in Apache Hadoop.



This image shows HDFS used in our work

RESULTS

The results of our work for the country Hungary are as follows:

Using Connected Component Algorithm, our results in single node are this,

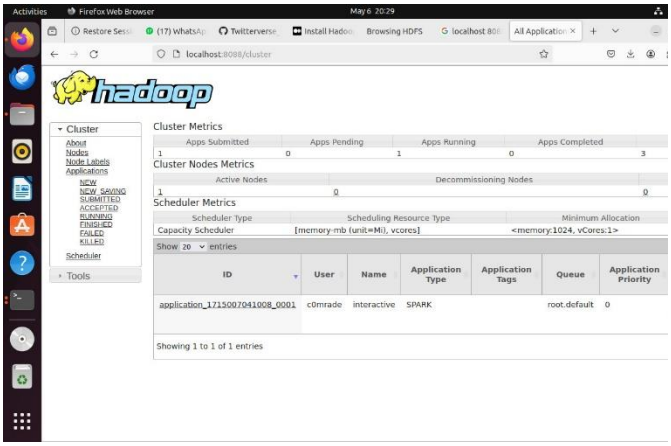
```
scala> val top10 = grp.top(10)(Ordering.by(_._2))
top10: Array[(org.apache.spark.graphx.VertexId, Int)] = Array((0,47538))

scala>

scala> top10 foreach {case (cc, size) => println(s"Component ID: $cc, Size: $size")}
Component ID: 0, Size: 47538
```

Page Rank Algorithm results in single node can be shown like this,

```
scala> top10 foreach {case (cc, size) => println(s"Component ID: $cc, Score: $size")}
Component ID: 19021, Score: 0.9999999999995721
Component ID: 41234, Score: 0.9999999999995721
Component ID: 34207, Score: 0.9999999999995721
Component ID: 28730, Score: 0.9999999999995721
Component ID: 31037, Score: 0.9999999999995721
Component ID: 23776, Score: 0.9999999999995721
Component ID: 39333, Score: 0.9999999999995721
Component ID: 32676, Score: 0.9999999999995721
Component ID: 29127, Score: 0.9999999999995721
Component ID: 53926, Score: 0.9999999999995721
```

And at last, we used Triangle counting Algorithm, and our results in single node were like this,

```
scala> top10.foreach { case (id, tri) => println(s"User Id: $id, Number of triangles: $tri") }
User Id: 12907, Number of triangles: 327
User Id: 42470, Number of triangles: 324
User Id: 31974, Number of triangles: 315
User Id: 8641, Number of triangles: 218
User Id: 15851, Number of triangles: 215
User Id: 8323, Number of triangles: 213
User Id: 26770, Number of triangles: 208
User Id: 19533, Number of triangles: 206
User Id: 40956, Number of triangles: 203
User Id: 32941, Number of triangles: 197
```



This image shows cluster done in Hadoop for a single node

CONCLUSION

In conclusion, our study delved deep into how people connect and content consumption patterns on Deezer. Leveraging some distributed computing techniques, like Spark, we processed large amount of data efficiently, which lead to comprehensive analysis. By applying the Connected Component, Page Rank and Triangle Counting algorithm for community detection and analysis, we explored the web of relationships, and structures that contributed to the community evolution within the platform. Our findings high- light not only the music people like, but also what genres they're into, who they know, and even the recommendations they receive and also the importance of the factors like genre preferences algorithmic recommendations and social connections. These findings have notable implications on Deezer—they can use them to develop their platform, make it more engaging, and recommend better content. But there are limits to what we can conclude from just one dataset. There's still much to explore, so future research could explore different approaches to uncover even more about music streaming platforms and the communities within them.

Overall, our study lays a solid foundation to- wards harnessing the power of distributed computing and understanding how data insights can shape a better user experience and communities on Deezer.

Acknowledgement

This research received support during the software engineering course instructed by professor Animesh Chaturvedi, PhD at King's College, London.

REFERENCES

[1] Benedek Rozemberczki and Ryan Davies and Rik Sarkar and Charles Sutton, GEMSEC: Graph Embedding with Self Clustering, arXiv preprint arXiv:1802.03997 (2018).