International Journal of Computer Trends and Technology
ISSN: 2231-2803 / https://doi.org/10.14445/22312803/IJCTT-V68I6P101.

Volume 68 Issue 6, 1-9, May 2020

Review Article

Analysis of Big Data Tools and Algorithms

Kundan Kumar[1], Franklin Valcin[2]
[1,2]*Phd, Dean of University Atlantic International University, Honolulu-United State.*

Received Date: 25 April 2020
Revised Date: 05 June 2020
Accepted Date: 06 June 2020

Abstract - *At present social networks are becoming more popular and increasing the use of people all over the world. This social network generates big data which differ in nature. To manage those big data of social networks, different algorithms and big data models are used. This paper will explore and analyze the big data tools and algorithms in order to know their advantages and drawback.*

Keywords - *Analysis, big data model, and algorithms*

I. INTRODUCTION

This paper includes an analysis of big data tools and algorithms.

II. BIG DATA

Big data is a collection of structured, semi-structured and unstructured large data sets, which are difficult to process using traditional data processing applications.

The architecture of big data visualizes the three main strategies; they are Things, People and Concepts[1].

Social media (e.g., Facebook, Twitter, Google+ etc.)[7] are not the only source of big data. There are various IoT devices that generate big data, which are wireless sensors, network-connected weather stations, cameras etc. [15].

A) 5 V's in Big Data

5 V's have been described [7] in order to identify the Big Data problem.

- ***Volume:*** Due to the vast use of social network, huge amounts of data is being generated.
- ***Variety:*** It includes Structured, Semi-Structured and Unstructured data.
- ***Velocity:*** Data is being generated at an alarming rate.
- ***Value:*** Worthiness of data.
- **Veracity:** Uncertainty and Inconsistent in the data

Fig. 1 5 V's in Big Data

There are different tools available in the Hadoop ecosystem to manage the big data, such as Hadoop Map-Reduce, Spark, Pig, Hive etc.

A) Hadoop Map Reduce Framework

Hadoop is a framework that allows us to store and process large data sets in a parallel and distributed fashion. Basically, Hadoop has two components one is HDFS, and another is Map-Reduce [9].

a) HDFS

In Traditional Data Model, it is not possible for storing a huge amount of data on a single node; therefore, Hadoop offers a new file system is known as HDFS. HDFS stands for Hadoop distributed File System. HDFS divide the data into smaller chunks and distribute each chunk of data across multiple nodes [9].HDFS, which works on Master-Slave architecture theory. HDFS have three types of nodes: the Name Node as Master Node, Data Node as Slave Node and Secondary Name Node. Name Node maintains and manages Data Nodes, records metadata and receives heartbeat and block reports from all the data nodes. Data nodes store actual data and serve read and write requests from clients [10].

b) Map Reduce

The Map-Reduce consists of two components: Map and Reduce. The Map-Reduce has two basic tasks to perform the data. i.e Mapper and Reducer [14]

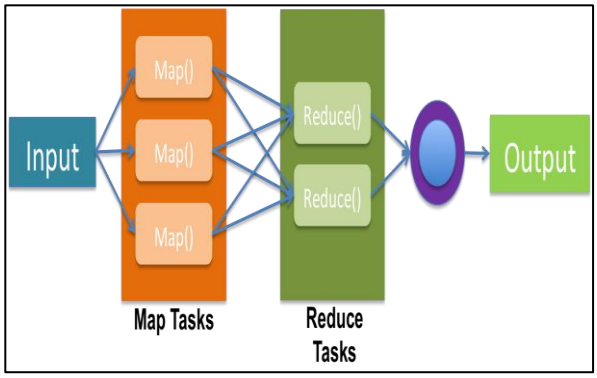

Fig. 2 Map & Reduce Tasks

1.Mapper

It is applied to all input key-value pairs that generate a number of intermediate key-value pairs. Then Mapper sorts the set of key-value pairs by their keys [11]&[12]

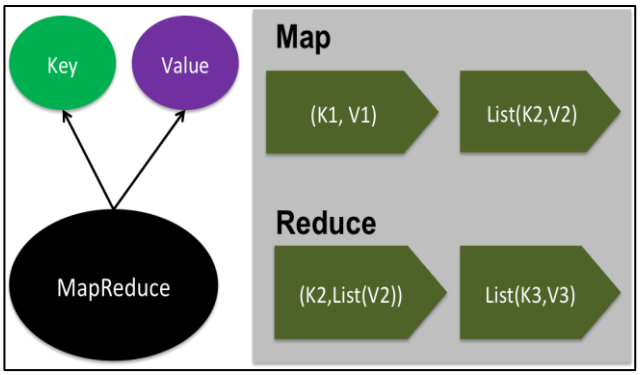

Fig. 3 Map Reduce Key-Value Pair

2.Reducer

The Reducer's job is to gather all of its input from the various mapper output and shuffles before reducing the mapper output, and the reducer's output is returned on the file system.

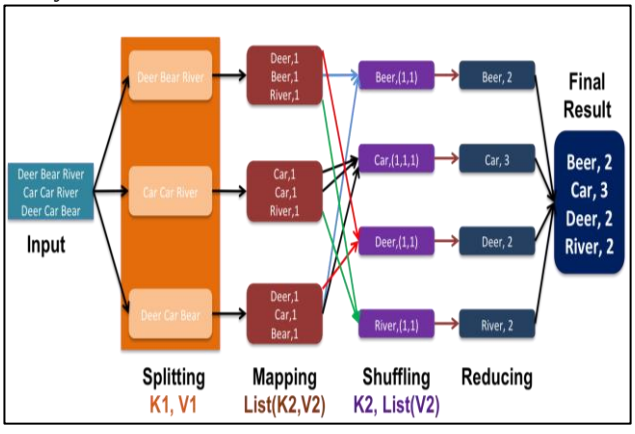

Fig. 4 Map Reduce's task illustration

c) Traditional versus Map Reduce

In traditional processing, data moves to process units that are expensive. In Map Reduce, processing instructions move to the Data node. Hence data is processed in parallel, and processing becomes faster.

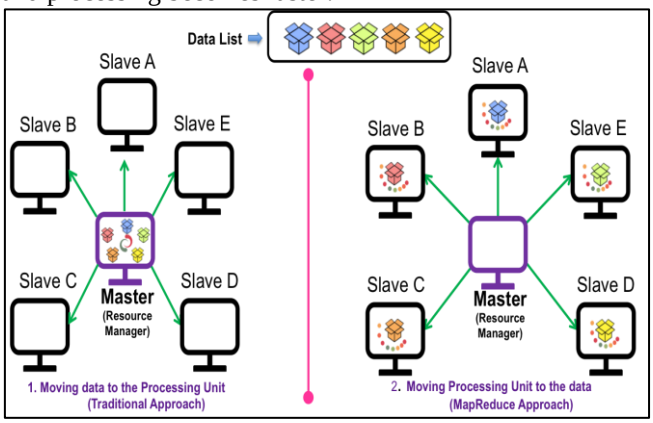

Fig. 5 Traditional versus Map Reduce Processing

B) Spark

Spark is a free clustering computing framework for real-time processing. Spark is an alternative to the traditional map and reduces model. It is designed to run on top of Hadoop [13]. It is its in-memory clustering computing that speeds up the processing speed of an application. It provides implicit data parallelism and fault tolerance.

a) Features of Spark

Spark includes in its features are Speed, Powerful caching, Deployment, Real-Time, Polyglot and scalable.

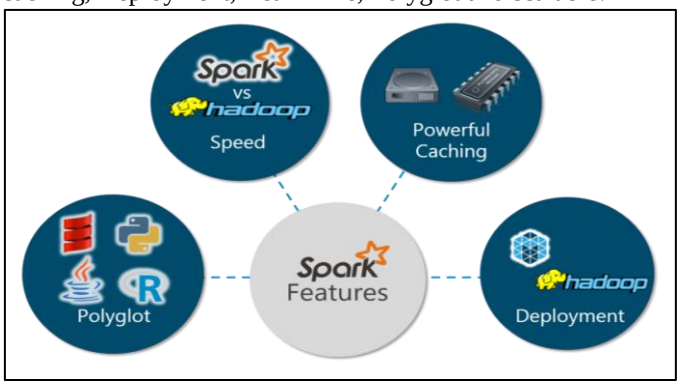

Fig. 6 Features of Spark, Source- Edureka

b) Spark Architecture Overview

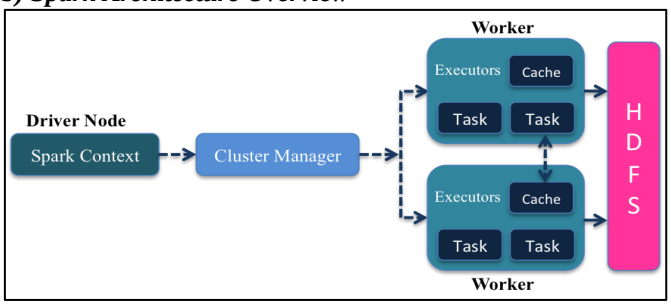

Fig. 7(a) Spark Architecture

Spark consists of a driver node (Spark Context), Cluster Manager, workers also called executors, and the HDFS as cited by[9]. See Fig 7(a).

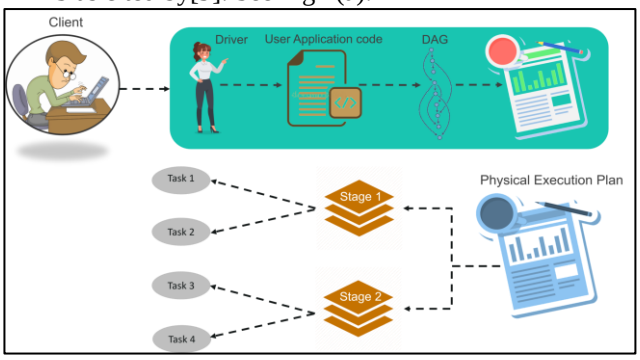

Fig. 7(b) Spark Architecture Workflow, Source- Edureka

To understand the flow of work of spark architecture, you can see at the infographic Fig 7(b) above[14].

Phase 1. The client/user submits spark user application code. On submission of an application code, the driver indirectly translates user code that contains change and activities into a logically directed acyclic graph called DAG**.** At this phase, it also executes optimizations. For example, pipelining transformations.

Phase 2. Then, it changes the logical graph called DAG into a physical execution plan with various phases. After changing into a physical execution plan, it produces physical execution units called tasks under each stage. Afterwards, the tasks are bundled and directed to the cluster.

Phase 3. Now the driver speaks to the cluster manager and discusses the resources. Cluster manager starts executors in worker nodes on behalf of the driver. At this stage, the driver will direct the tasks to the executors based on data assignment. When executors begin, they register themselves with drivers. Therefore, the driver will have a complete view of the executors that are executing the task.

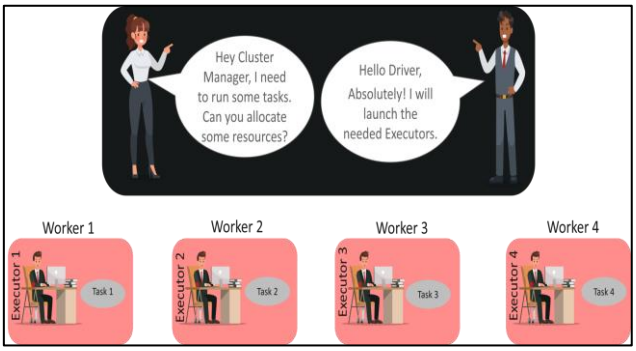

Fig. 7(c) Spark Architecture workers' node, Source- Edureka

Phase 4. Throughout the sequence of execution of tasks, the driver program will invigilate the set of executors that runs. Driver node also plans future tasks based on data placement.

c) Hadoop versus Spark

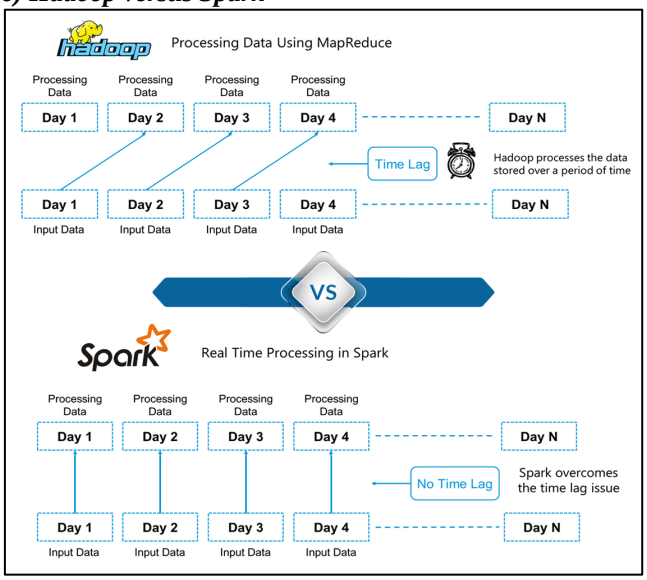

Fig. 8 Hadoop versus Spark, Source- Edureka

Big data management is handled by Hadoop map-reduce to cope with the huge amount of data. To handle real-time tasks, map-reduce was not the effective choice [9], [14]. Map-reduce was not able to maintain a higher level of security. Map-reduce is not efficient to be used for applications in which the data are repeatedly used.

III. ALGORITHMS TO MANGE THE BIG DATA OVER SOCIAL NETWORKS

An algorithm is a step-by-step mathematical instruction to solve a problem.

According to [2][3]&[5], there are many algorithms being used to manage the big data in social networks. Few are described below:

- Directed graph
- Bi-directed graph
- Undirected graph
- Big parties Graph
- Traditional K-means algorithms
- Improved K-means algorithms
- H-PRE algorithm

A) Directed Graph Algorithm

In Social networking sites like Twitter, Weibo, and Google+, users are connected by the "following relationship such that a user X (i.e. follower) follows another user Y (i.e. followee), which can be denoted as X->Y.

In the current age of social networks such as Twitter, Facebook, LinkedIn, Google+**,** which is the prime source of big data [2]. For Instance, Facebook users can create their own profile, add users as friends, exchange chats, and join common-interest user groups. The number of mutual friends may vary from one user to another user. It is not uncommon for a Facebook user X to have hundreds of friends. Note that, although many of the Facebook users are connected to some other Facebook users via mutual friendship (i.e., if a user X is a friend of another Y, then

user Y is also a friend of user X), there are situations in which such a relationship is not mutual. To handle these situations, Facebook provided the feature of "follow", which allows a user to follow the public posting of other Facebook users without the need of adding them as friends. Hence, users in Facebook can also be linked by the "following relationships too," and these "following" relationships are directional. Consider Case 1.

Case 1: For a demonstrative purpose, let us consider a small portion of a social network. Here, there are |V|=8 users (Kundan, Sima, Kabera, Aanya, Lucy, Raj, Jean, Miriam). Each user is following other another user as stated below:

- Kundan is following Sima.
- Sima is following Kundan and Kabera.
- Kabera is following Kundan
- Aanya is followingLucy, Kundan, Sima, and Kabera.
- Lucy is following Kundan, Aanya, Sima, and Kabera.
- Raj is following Lucy, Jean.
- Jean is following Raj.
- Miriam is following Jean.

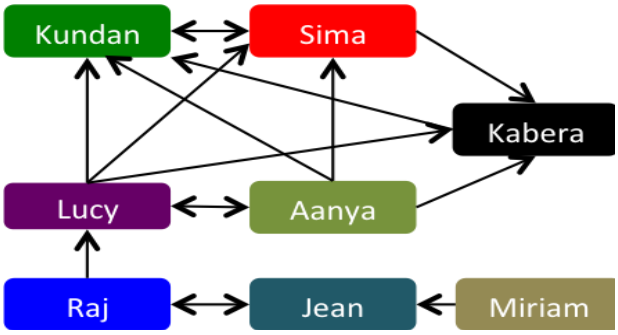

Fig. 9 Illustration of social network users' data in the directed graph

We represent the social network users data in Case 1 by the directed graph

G = (V,E), ----------(1)

Where,

- each node/vertex $v \in V$ represents a user in the social network by nodes, and
- each directed edge $e = (u, v) \in E$represents the follow relationship between a pair of users $u, v \in V$ such that user u follow user v.

The arrow on the edge represents the "following" direction. For instance, a directed arrow "$Kundan \rightarrow Sima$" $represents$ that Kundan follows Sima. Similarly, a di-directional arrow "$Kundan \leftrightarrow Sima$" represents Kundan and Sima following each other. See Fig 9.

From above Fig 9, we noted that the "following" relationships are directional. As such, a user Y may be following another user Z while Z is not following Y (i.e., $Lucy \rightarrow Kundan\,, but\, Kundan \nrightarrow Lucy$). To a further extent, the group of users followed by user Y may not be the same group of users as those who are following user Y $(e.g., Sima \rightarrow \{Kundan, Kabera\}, but\, \{Kundan, Aanya, Lucy\} \rightarrow Sima)$.

Let us consider the space requirement for this directed graph representation of social network data. Theoretically, given |V| social entities, there are potentially $|V|(|V|-1)$ unless for the extreme case where everyone follows everyone in a social network. In Fig 9, There are only $|E| = 16$directed edges (bi-directed arrow counted as two edges), (If N nodes in a given graph, there are N – 1 directed edges that can lead from it. Therefore, the maximum number of edges are N*(N-1)) [8]. Hence 8*(8-1) = 56 edges for$|V| = 8\, users)$, where E= {(Kundan, Sima),(Sima, Kundan), (Kabera, Sima), (Sima, Kabera), (Kabera, Aanya), (Lucy, Aanya),(Aanya, Lucy),… (Miriam, Jean)}.

The advantages of this directed graph of social network data are that we can easily perform the following task.

Question-1: Who are the most popular followers?

Answer: By counting the number of incoming edges of every node and picking the social entities with the highest number of incoming edges.

Let NumOfInEdges(v)=NumOfFollowers(v)= $|\{u \in V\ (u, v) \in E\}|$. Hence, the most popular followers can be found by $argMax_{v \in V}$

NumInEdges(v)= $argMax_{v \in V}$ NumOfFollowers(v).

Example: For case 1, Kundan is most the most followee. Who is followed by Sima, Kabera, Aanya, and Lucy? See in Fig 9

Question 2:Who are the most active followers?

Answer: By counting the number of outgoing edges of every node and picking the social entities with the highest number of outgoing edges. See in Fig 9.

Let NumOfOutEdges(v) = NumOfFollowees (v) = $|\{u \in V\ (u, v) \in E\}|$. Hence, most active followers can be found by $argMax_{v \in V}$

NumOfOutEdges(v)= $argMax_{v \in V}$ NumOfFollowees(v).

Example 2: For Case 1. Lucy is the most active follower. Lucy is following Kundan, Soma, Kabera, Aanya, Raj.

B) Bi-directed Graph

In social networking sites like Twitter, LinkedIn, and Facebook, users are usually connected by mutual friendship such that user X is a friend of another user Y, meaning that user Y is a friend of user X. Such a mutual friendship is denoted as$X \leftrightarrow Y$. Unlike those directional "following" relationships described in **Part III (A)**, the mutual friendships described here are bidirectional. Consider Case 2.

Case 2: For a demonstrative purpose, let us consider another small portion of the big social network. here, there are $|V| = 8\, users$ (Kundan, Sima, Kabera, Aanya, Lucy, Raj, Jean, Miriam). Each user is a friend of some others, as described below:

- Kundan is a friend of Sima, Aanya, Lucy, and Kabera.

- Sima is a friend of Kundan, Kabera, Lucy, and Aanya.
- Kabera is a friend of Sima, Kundan, Lucy, Aanya.
- Aanya is a friend ofKundan, Sima, Lucy, and Kabera.
- Lucy is a friend of Kundan,Sima, Aanya, Raj, and Kabera.
- Raj is a friend of Lucy and Jean.
- Jean is a friend of Raj and Miriam.
- Miriam is a friend of Raj.

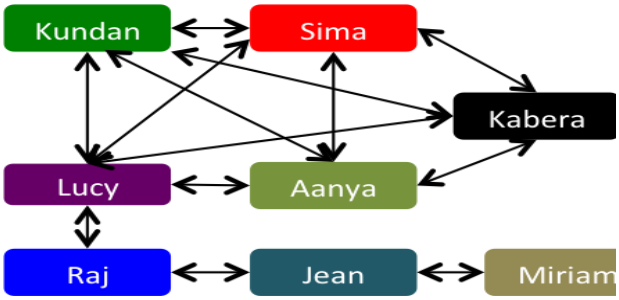

Fig.10 Illustration of social network users' data in bi-directed graph

We represent the social network users' data in Case 2 by the bi-directed graph

G = (V, E), ----------(2)

Where,

- each node/vertex $v \in V$ represents a user in the social network by nodes, and
- each directed edge $e = (u, v) \in E$ represents the follow relationship between a pair of users $u, v \in V$ such that user u follow user v, a pair of directed edges (u, v) and (v, u) represents the mutual friendship between a pair of users $u, v \in V$.

A bidirectional edge represents mutual friendships. For instance, a bi-directed arrow "$Kundan \leftrightarrow Sima$" represents that Kundan and Sima are mutual friends. See Fig 10.

Let us consider the space requirement for this directed graph representation of social network data. Theoretically, given $|V|$ social entities, there are potentially $|V|(|V|-1)$ directed edges for mutual friendships. Practically, the number of edges is usually lower than its maximum $|V|(|V|-1)$unless for the extreme case where everyone follows everyone in a social network. In Fig 10, There are only $|E| = 26$ directed edges(possibly 56 edges for $|V| = 8\ users$ as definedin Part III-A-Case1), where E= {(Kundan, Sima),(Sima, Kundan), (Kabera, Sima), (Sima, Kabera),(Kabera, Aanya),(Aanya, Kabera) ,…(Jean, Miriam),(Miriam, Jean)}.

The advantages of this bi-directed graph representation of social network data are that we can easily perform the following tasks.

Question 3: Who is the most popular user?***Answer:*** By counting the number of incoming and outgoing edges of every node and picking the social entity with the highest number of edges.

Let NumOfEdges(v) =NumOfFriends(v) = $|\{u \in V | (u,v) \in E \cup (v,u) \in E\}|$. Hence, the most popular user can be found by $argMax_{v \in V}$ NumOfEdges(v)= $argMax_{v \in V}$ NumOfFriends(v).

Example 3:For case 2, Lucy is the most popular user. Lucy is friend ofKundan, Sima, Kabera, Aanya, Jean. See in Fig10.

Question 4: Who are the most isolated users?

Answer: By finding all k-degree connections of every node and picking the social users with the highest k.We use $\kappa^{\circ} Connections(v)\ for\ \kappa \geq 1$. The user with the highest k is the most isolated user, meaning that it takes a $\kappa - edge$ path in the bi-directed graph to reach some users in the social network.

Example 4: For case 2, Sima, Kabera and Miriam are the most isolated users. This can be answered by the existence of a six-degree connection of Sima, Kabera and Miriam.

Miriam is an isolated user because there exists $4^{\circ} Connections(Miriam) = \{Sima, Kabera\}$. It takes four edges from Miriam to reach Sima or Kabera via Miriam->Jean->Raj->Lucy->Sima or Kabera.

Sima is another isolated user because there exists $4^{\circ} Connections(Sima) = \{Miriam\}$. It takes four edges fromSima->Lucy->Raj->Jean->Miriam.

Kabera is another isolated user because there exists $4^{\circ} Connections(Kabera) = \{Miriam\}$.

It also takes 4 edges from Kabera->Lucy->Raj->Jean->Miriam.

C) Undirected Graph

In section III-B, we managed social network data in the bi-directed graph. Observing that the mutual friendships are symmetric in nature, we could manage these social network data in a space-efficient matter. Specifically, we do not need to use bi-directed edges. Instead, we use undirected edges so that an edge $X - Y$ indicates a user X is a mutual friend of another user Y and vice-versa.

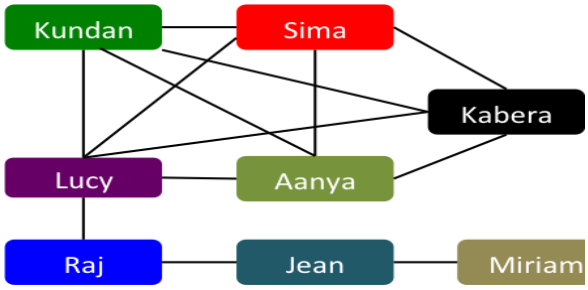

Fig. 11 Illustration of social network users' data in an undirected graph

Let is recall Case 2 and represent the social network data in the scenario by using an undirected graph

G = (V, E), ----------(3)

Where,

- each node/vertex $v \in V$ represents a user in the social network by nodes, and
- each undirected edge $e = (u, v) \in E$ represents the mutual friendship between a pair of users

$u, v \in V$ such that user u is a mutual friend of user$v$.

Case 3: Observed the Fig11 and recalled that the number of edges was $|\mathbf{V}|(|\mathbf{V}|-\mathbf{1})$ For bi-directed graphs. With undirected graphs, we reduce this potential number by half. So, in terms of space requirements, given $|\mathbf{V}|$social network users, the potential number of edges are reduced by half from $|V|(|V|-1)$ required by bi-directed graphs to $\frac{|V|(|V|-1)}{2}$ required by undirected graphs. In fig 11, there are only $|E| = 13$ undirected edges while $|E| = 26$ directed edges in the bi-directed graph in Fig 10. Note that, due to the symmetric nature of mutual friendships, we do not need to store (Sima, Kundan) when storing (Kundan, Sima).

Advantages of this undirected graph representation of social network data are that we can reduce the space requirements while being able to easily perform Questions 3&4.

D. Bipartite Graph

So far, we have shown how we manage a social network in which a user follows another user or add another user as a friend. However, in some social networking sites like Twitter, LinkedIn, and Facebook, users are linked by participation relationships such that users u and ω both join a common-interest group v. For instance, some users may join the common-interest groups on two Canadian national sports (ice hockey and lacrosse). Consider Scenario 3. As another real-life instance, participants of CCGrid 2016 may join the common-interest group on CCGrid.

Case 4:For an illustrative purpose, let us consider a small portion of a social network. Here, there are $|U| = 12$ users (Kundan, Sima, Kabera, Aanya, Lucy, Raj, Jean, Miriam, Booker, Valcin, Michel, Leah). Each user joins some of the $|V| = 12$ common-interest sports groups as described below:

- Kundan joins a common-interest group on baseball.
- Sima joins common-interest groups on archery and cycling.
- Kabera joins the common-interest group on archery and ice hockey
- Aanya joins a common-interest group on archery, baseball, cycling and equestrian.
- Lucy joins common-interest groups on archery, baseball, cycling and diving.
- Raj joins common-interest groups on equestrian and golf.
- Jean joins a common-interest group on fencing.
- Miriam joins a common-interest group on golf.
- Booker joins a common-interest group on lacrosse.
- Valcin joins a common-interest group on ice hockey.
- Michel does not join any common-interest group.
- Leah joins common-interest groups on cycling, ice hockey, and karate.

We represent the social network data in Case 4 by using a bipartite graph

$$G = (U, V, E) \text{ ----------(4)}$$

Where,

- each node/vertex $v \in U$ represents a user in the social network by nodes,
- each node/vertex $v \in V$ represents a common-interest group (e.g., like ice hockey, cycling), and
- each edge $e = (u, v) \in E$ represents the follow/participate relationship of users $u \in U$ on a common-interest group $v \in V$ such that user u joins the group v.

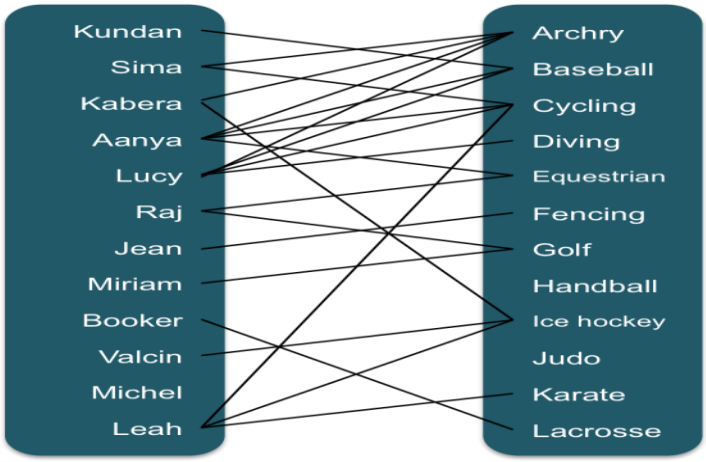

Fig. 12 Illustration of social network users' data in a bipartite graph

An undirected edge represents the group participants. For instance, an edge "$Kudna - Baseball$" represents that Kundan joins group Baseball. See Fig 12.

Let us consider the space requirements for this bipartite graph representation of social network data. Theoretically, given $|U|$ social entities and $|V|$ Edges for group participation. Practically, the number of edges is usually lower than its maximum $|U| \times |V|$ Unless for the extreme case where everyone joins every group in a social network. In Fig 12, there are only $|E| = 22$ edges (cf. possibly 144 edges for $|U| = 12\ users\ and\ |V| = 12\ groups$), where E = {(Kundan, Baseball),(Sima, Archery), (Sima, Cycling),(Kabera, Archery), (Kabera, Ice hockey), …, (Leah, Cycling),(Leah, Ice hockey),(Leah, Lacrosse)}.

The advantages of this bipartite graph representation of big social network data are that we can easily perform the following tasks.

Question 5: Which is the most popular common-interest group?

Answer: By counting the number of edges of every common-interest group and picking the groups with the highest number of edges.

Let

Num Of Members $(v) = |\{u \in U | (u, v) \in E\}|$. Hence, the most popular groups can be found by $argMax_{v \in V}$ NumOfMembers(v).

Example 5: For Case 4: the common-interest groups on archery and on cycling are the most popular. This is because of the four edges connecting the group. See Fig12.

Question 6: Who are the most active users?
Answer: By counting the number of edges of every social user with the highest number of edges.
Let
NumOfGroups $(\upsilon) = |\{u \epsilon V|(u,\upsilon)\epsilon E\}|$. Hence, the most active users can be found by $argMax_{\upsilon \epsilon V}$ NumOfGroups(υ).
Example 6: For case 4, both Aanya and Lucy are the most active users. Where Aanya is in the group on archery, baseball, cycling, and equestrian, while Lucy is grouped on archery, baseball, cycling and diving, this is because of the four edges connecting Aanya and Lucy. See in Fig 12.

E) Traditional K-means algorithms

J.B MacQueen is the father of the K-means clustering algorithm. The core idea of the K-means algorithm is that the data is divided into a number of classes and the distance between each cluster and the cluster centre is the smallest [3].
Algorithm steps are:

- Choosing k objects from n data objects as the initial cluster centres.
- According to the cluster centre of each cluster object, the distance between each object and the centre of the object is calculated; the object can be reclassified according to the smallest distance.
- Recalculate the centroid point of each cluster.
- Calculation of the standard measurement function, if certain conditions are met, the algorithm will be terminated; if the condition is not met, then back to step ii.

F) The Improved K-Means Algorithm

The improved algorithm is mainly based on the average of the data record to continuously ensure cluster centres and uses the improved Euclidean distance formula to calculate the distance between data and cluster centre by judging whether the distance is greater than the threshold to automatically adjust the number of clusters. [3]. See Fig13.
The main steps of the algorithm are as below:
Step 1: To sum continuous data and find the average.

$$\overline{\text{Continuous}} = \frac{\sum(X_1,X_2,X_3\ldots+X_N)}{N} \text{ ------- (5)}$$

In the above equation (5), Continuous is the average of a continuous feature. The algorithm needs to do the same calculation for all continuous features. X_N represents the N^{th} data. N is a number of data in the list.

Step 2: Characteristics Standardization

$$Standardization = \frac{\sqrt{\frac{(X-\overline{Continuous})^2}{N-1}} - \overline{Continuous}}{\sqrt{\frac{(X-\overline{Continuous})^2}{N-1}}} \text{---------(6)}$$

In the above equation (6), Standardization is the standard value of a continuous feature in all records. The algorithm needs to do the same calculation for all continuous features. X is data. When(3) are satisfied, the formula can be used.

$$\sqrt{\frac{(X-\overline{Continuous})^2}{N-1}} \neq 0\text{--------(7)}$$

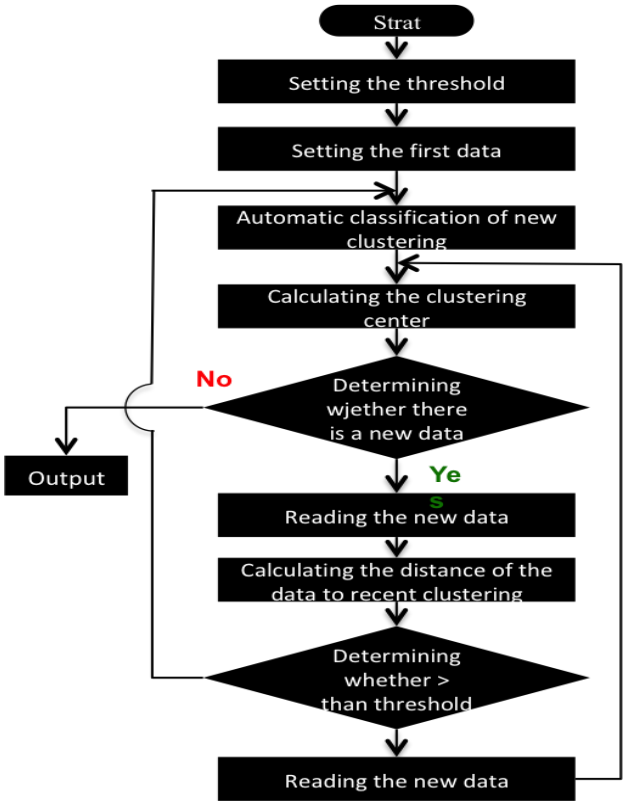

Fig. 13 K-Means algorithm for dynamically adjusting the number of clusters

The main aim of the Standardization of data is to reduce the difference between the training data set and the test data without affecting the accuracy of the test data that can make the data of the test data close to the training data.
Step 3: Automatic clustering, calculate the distance between unclassified data and the cluster centre of the nearest cluster. If the minimum distance is greater than the threshold, the data will be divided into a new cluster. On the contrary, the data will be divided into a cluster of minimum distance.
The calculation method of the characteristics of continuous type:

$$D_{CalcNumDiff} = (A - \bar{A})^2 + (B - \bar{B})^2 + \cdots + (C - \bar{C})^2 \text{ -----(7)}$$

In the above equation (7), A, B, C, respectively, which indicates the different characteristics of continuous data in unclassified data, $\bar{A}, \bar{B}, \bar{C}$ Respectively, which indicates the characteristics of cluster centres in a cluster.
The calculation method of the characteristics of discrete type:

$$D_{CalcFeaDiff} = \left(1 - \frac{\alpha}{n}\right)^2 + \left(1 - \frac{\beta}{n}\right)^2 + \cdots + \left(1 - \frac{\chi}{n}\right)^2 \quad \text{-----(8)}$$

In the above equation (8), n represents the number of data in a cluster. α, β, χ respectively, which indicates the number of the same features in the cluster and unclassified data.

The final distance:

$$D_{TotalDiff} = \sqrt{\frac{D_{CalcNumDiff} + D_{CalcFeaDiff}}{N}} \text{-----(9)}$$

In the above equation (9), N is a number of features.

Step 4: To update the cluster centre for the newly added records.

$$\bar{C} = \frac{\sum(X_1 + X_2 + \cdots + X_n)}{n} \quad \text{--------(10)}$$

In the above equation (10), n is the number of data in the cluster. X_n Represents the N[th]data is a cluster. $\bar{C}$ It is a continuous feature in the cluster. The algorithm needs to do the same calculation for all continuous features.

Step 5: Repeat the above steps until all the data in the test set are completed.

During the testing process, all the data in the test set need to be standardized.

$$Std = X - \frac{\overline{Continuous}}{Standardization} \text{--------(11)}$$

In the above equation (11), X is a continuous feature. The algorithm needs to do the same calculation for all continuous features. $\overline{Continuous}$ And Standardization is the result of the training process.

Locking for the nearest cluster and dividing the data into the nearest cluster. The way to calculate the distance is as same as that of Step 3 in the training process.

G) H-PRE Algorithm

To ensure secure storage, [5] designed Heterogeneous Proxy Re-Encryption (H-PRE) that supports diverse transformation from Identity-Based Encryption (IBE) to Public-Key Encryption (PKE). H-PRE is well-suited with traditional cryptography.

H-PRE includes three types of the algorithm, traditional identity-based encryption, re-encryption and traditional public-key cryptosystems.

In the basic H-PRE process, the owner of data encrypts delicate data using a local security plug-in and then uploads the encrypted data to a big data platform. The data are transformed into the cypher text that can be decrypted by a specified user after PRE services. If a SESP is the stated user, then the SESP can decrypt the data using its own private key to obtain the related clear text.

The steps of H-PRE algorithms [5] are as follow:

Step1: $\boldsymbol{Setup_{IBE}(k)}$**:** Input security parameters *k*, produce a primary security parameter mk randomly, compute the system parameter set *parameters* using a bilinear map and hash function.

Step2: $\boldsymbol{KeyGen_{IBE}(mk, parameters, id)}$ **:** On the request of the user, the private key from the key creation centre, the key creation centre obtains the legal identity of the user and creates the public and private keys (pk_{id}, sk_{id}) for the user using *parameters* and *mk.*

Step3: $\boldsymbol{KeyGen_{PKE}(parameters)}$**:** On submission of user's demand, the key controlling centre not only creates the identity-based public and private keys but also produces the public and private keys of the traditional public-key system ($\overline{pk}_{id}$, $\overline{sk}_{id}$).

Step4: $\mathrm{Enc_{IBE}}(\mathrm{pk_{id}}, \mathrm{sk_{id}}, \mathrm{parameters}, \mathrm{m})$: When a user crypts data, the owner of data encrypts the cleartext (m) into the ciphertext $(\mathrm{c} = (\mathrm{c}_1, \mathrm{c}_2))$ using the user's personal(pk_{id}, sk_{id})and a random number$r \in RZ_p^*$

Step5: $\boldsymbol{KeyGen_{IBE}(sk_{id}, \overline{sk}\mathbf{id}, \overline{pk}\mathbf{id}, parameters)}$**:** When the owner of data (user *i*) permits user *j* permissions, using sk_{id}, $\overline{sk}_{id}$, and $\overline{pk}_{id}$, user *i* computes the PRE key ($rk_{id_i - id_j}$), finalizing the change from user *i* to user *j*.

Step6: $\boldsymbol{ReEnc(c_i, rk_{id_i - id_j}, parameters)}$**:** This activity is implemented transparently on the big data platform. The function re-encrypts the ciphertext that user *i* encrypted into ciphertext that user *j* can decrypt. It inputs$(c = (c_1, c_2))$.

Step7: $\boldsymbol{Dec_{PKE}(c_j, \overline{sk}_{id_j}, parameters)}$**:**

This is a function for decrypting the PRE ciphertext. After receiving the PRE ciphertext ($C_j = (C_{j_1}, C_{j_2})$) from the proxy server of the big data platform, the user determines the clear text($\bar{m} = m$) of the data using his or her own $\overline{sk}_{id_j}$.

The advantages of the H-PRE algorithm are protecting the security of users' sensitive data. The data owners have complete control of their own data.

IV. DRAWBACK OF THE ALGORITHMS TO MANAGE BIG DATA IN SOCIAL NETWORKS

A) Drawback of Directed, Bi-directed, Undirected and Bipartite Graph

In the case of managing data, the construction of a graph consumes a higher amount of time and is complex when the number of data increases.

B) Drawback of Improved K-Means clustering algorithms

The computation of the Improved K-Means clustering algorithm is complex as it uses Euclidean distance that could not tolerate huge data as the dataset scale up.

C) Drawback of H-PRE Algorithm

Using the H-PRE algorithm to secure sensitive data sharing on a Big Data Platform consumes higher time for encrypting the data twice.

V. CONCLUSION

Big data is complex data that is being managed by different tools and algorithms. The Hadoop and Spark big data tools have been explored and compared and found that Hadoop map-reduce is not able to handle real-time tasks, and it is not efficient to be used for applications in

which the data is repeatedly used. Big data management algorithms have also been analyzed and found that they use different graphs to manage the users, mutual friends, followers and their groups but consumes a higher amount of time in case of managing data construction of graphs. K-means clustering technique uses Euclidean distance computation that may not tolerate huge data as the dataset scale up. H-PRE algorithm to secure sensitive data sharing on a Big Data Platform consumes higher time for encrypting the data twice.

Future work should be dedicated to implementing the data managing algorithms (e.g., Directed Graph), K-Means algorithm and H-PRE security algorithm using Java programming language and evaluate proposed novel phi-so clustering algorithm of data management & T&F based security technique.

REFERENCES

[1] Shuntaro Hitomi, Keiro Muro Social Innovation through Utilization of Big Data, Hitachi Review 62(7)(2013) 384 – 388.

[2] Carson K. Leung, Hao Zhang, S Management of Distributed Big Data for Social Networks, IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, (2016) 639 – 648.

[3] Li Jun Tao, Liu Yin Hong, Hao Yan, The Improvement and Application of a K-Means Clustering Algorithm, IEEE International Conference on Cloud Computing and Big Data Analysis, (2016) 93 – 96.

[4] Smt. Shashi Rekha .H., Dr. Chetana Prakash , Smt. Kavitha G., Understanding Trust and Privacy of Big Data in Social Networks: A Brief Review, 3rd International Conference on Eco-friendly Computing and Communication Systems, IEEE Computer Society, (2014) 138 – 143.

[5] Xinhua Dong, Ruixuan Li, Heng He, Wanwan Zhou, Zhengyuan Xue, and Hao Wu, Secure Sensitive Data Sharing on a Big Data Platform,Tsinghua Science and Technology, 20(1) (2015) 72 – 80.

[6] SHUI YU, Big Privacy: Challenges and Opportunities of Privacy Study in the Age of Big Data, IEEE Access,(2016) 2751 – 2763.

[7] Duygu Sinanc Terzi, Ramazan Terzi, Seref Sagiroglu, A Survey on Security and Privacy Issues in Big Data, 10th International Conference for Internet Technology and Secured Transactions, (2015) 202 – 207.

[8] Robin J. Wilson, Introduction to Graph Theory, Fourth Edition, Addison Wesley Longman Limited, ISBN 0-582-24993-7, (1996)

[9] Ankush Verma, Ashish Huassin Mansuri, Dr Neelesh Jain, Big Data management Processing with Hadoop MapReduce and Saprk Technology: A Comparision IEEE Symposium on Colossal Data Analysis and Networking (CDAN), (2016).

[10] Dr Siddaraju, C.L. Sowmya, K.Rashmi and M. Rahul, Efficient Analysis of Big Data using map Reduce Framework, International Journal of Recent Development in Engineering and Technology, 2 (2014).

[11] Jimmy Lin and Chris Dyer Data-Intensive Text Processing with Map-Reduce , (2010) 18-38.

[12] Donald Miner and Adam Shook, MapReduce Design Patterns, 2(6).

[13] Amol Bansod, Efficient Big Data Analysis with Apache Spark in HDFS, IJEAT 4 (2015).

[14] Edureka Blog (2020) https://www.edureka.co/blog/spark-architecture/ accessed on

[15] Kundan Kumar, Benson Murimi, Ishimwe Romalice IoT Based Farmers' Smart Technology, Case Study: Rwanda Meteorology Agency, Kayonza District International Journal of Computer Trends and Technology 67(9) (2019)1-6.