EDITORS
MRUTYUNJAYA PANDA
ABOUL-ELLA HASSANIEN
AJITH ABRAHAM

BIG DATA ANALYTICS

A SOCIAL NETWORK APPROACH

CRC Press
Taylor & Francis Group
A SCIENCE PUBLISHERS BOOK

Big Data Analytics
A Social Network Approach

Editors

Mrutyunjaya Panda
Computer Science Department
Utkal University
Bhubaneswar, India

Aboul-Ella Hassanien
University of Cairo
Cairo, Egypt

Ajith Abraham
Director, Machine Intelligence Research Labs
Auburn, Washington, USA

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
A SCIENCE PUBLISHERS BOOK

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed on acid-free paper
Version Date: 20180803

International Standard Book Number-13: 978-1-138-08216-8 (Hardback)

Library of Congress Cataloging-in-Publication Data

Names: Panda, Mrutyunjaya, editor. | Abraham, Ajith, 1968- editor. | Hassanien, Aboul Ella, editor.
Title: Big data analytics. A social network approach / editors, Mrutyunjaya Panda, Computer Science Department, Utkal University, Bhubaneswar, India, Ajith Abraham, Director, Machine Intelligence Research Labs, Auburn, Washington, USA, Aboul-Ella Hassanien, University of Cairo, Cairo, Egypt.
Description: Boca Raton, FL : Taylor & Francis Group, [2018] | "A science publishers book." | Includes bibliographical references and index.
Identifiers: LCCN 2018029675 | ISBN 9781138082168 (hardback : acid-free paper)
Subjects: LCSH: Big data. | Discourse analysis, Narrative. | Truthfulness and falsehood. | Online social networks.
Classification: LCC QA76.9.B45 B547 2018 | DDC 005.7--dc23
LC record available at https://lccn.loc.gov/2018029675

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the CRC Press Web site at
http://www.crcpress.com

Content

CHAPTER 1

Linkage-based Social Network Analysis in Distributed Platform

Ranjan Kumar Behera,[1,*] *Monalisa Jena*,[2] *Debadatta Naik*,[1] *Bibudatta Sahoo*[1] and *Santanu Kumar Rath*[1]

Introduction

The social network is a platform where a large number of social entities can communicate with each other by sharing their views on a number of topics, posting a bunch of multimedia files or exchanging a number of messages. It is the structure, where the group of users is connected through their social relationships. The social network can be modeled as a non-linear data structure, like the graph, where each node represents a user and the edges between them depict the relationships between the users. Facebook, Twitter, YouTube, LinkedIn, Wikipedia are some of the most popular social network platforms where billions of users interact every day. In these social network websites, along with the interaction, exabyte of structured, unstructured and semi-structured data is generated at every instance of time (Hey et al. 2009). This is the reason where the term 'big data' is associated with the social network. Research in the social network has the beginning with the social scientist analyzing human social behavior,

[1] National Institute of Technology, Rourkela, 769008, India.
[2] Fakir Mohan University, Balasore, 756019, India.
Emails: bmonalisa.26@gmail.com; deba.uce03@gmail.com; bdsahu@nitrkl.ac.in; skrath@nitrkl.ac.in
* Corresponding author: jranjanb.19@gmail.com

mathematicians analyzing complex network theory and now computer scientists analysing the generated data for extracting quite a number of useful information. Social scientists, physicists, and mathematicians are basically dealing with the structural analysis of social network while computer scientists are dealing with data analysis. Social networks are the most important sources of data analytics, assessment, sentiment analysis, online interactions and content sharing (Pang et al. 2008). At the beginning stage of a social network, information was posted on the homepages and only a few internet users were able to interact through the homepages. However, nowadays an unimaginable number of activities are carried out through the social network which leads to a huge amount of data deposition. Social network enables the users to exchange messages within a fraction of time, regardless of their geographical location. Many individuals, organizations and even government officials now follow the social network structure and media data to extract useful information for their benefit. Since the data generated from the social network are huge, complex and heterogeneous in nature, it proves a highly computationally challenging task. However, big data technology allows analysts to sift accurate and useful information from the vast amount of data.

Social network analysis is one of the emerging research areas in the era of big data analytics. Social network consists of linkage and content data (Knoke and Yang 2008). Linkage data can be modeled through graph structure which depicts the relationship between the nodes whereas the content data is in the form of structured, semi-structured or unstructured data (Aggarwal 2011). They basically consist of text, images, numerical data, video, audio, etc. Basically social network analysis can be broadly classified into two categories: Social Media Analysis (He et al. 2013) and Social Structure Analysis (Gulati 1995).

It has been observed that a huge amount of data is generated from social network at every fraction of time and the size of generated data is increasing at an exponential rate. Storing, processing, analyzing the huge, complex and heterogeneous data is one of the most challenging tasks in the field of data science. All the data which is being created on the social network website can be considered as social media. The availability of massive amounts of online media data provides a new impetus to statistical and analytical study in the field of data science. It also leads to several directions of research where statistical and computational power play a major role in analyzing the large-scale data. The structure-based analysis is found to be more challenging as it is a more complex structure than the media data. A number of real-time applications are based on structural analysis of the network where linkage information in the network plays a vital role in analysis. Social network is the network of relationships between the nodes where each node corresponds to the

user and the link between them corresponds to the interaction between them. The interaction may be friendship, kinship, liking, collaboration, co-authorship or any other kind of relationship (Zhang et al. 2007). The basic idea behind each social network is the information network where groups of users either post and share common information or exchange information between them. The concept of social network is not restricted to particular types of information network; it could be any kind of network between the social entities where information is generated continuously. A number analysis can be carried out using the structural information of the network to identify the importance of each node or reveal the hidden relationships between the users. Before discussing the details of research in social network analysis, it is better to point out certain kinds of structural properties that real-world social network follows. Some of them are small-world phenomenon, power-law degree distribution or scale-free network, etc., which were devised much before the advent of computer and internet technology. Small-world phenomenon was proposed by Jeffery and Stanley Milgram in 1967 which says that most of the people in the world are connected through a small number of acquaintances which further leads to a theory known as six-degree of separation (Kleinfeld 2002, Shu and Chuang 2011). According to the theory of six-degree of separation, any pair of actor in the planet are separated by at most six degree of acquaintances. This theory is now the inherent principle of today's large-scale social network (Watts and Strogatz 1998). A number of experiments were carried out by social scientists to prove the six-degree of separation principle. One of the experiments is reflected in MSN messenger data. It shows that the average path length between any two MSN messenger users is 6.6. The real-world social network is observed to follow power-law degree distribution, which implies that most of the nodes in the network are having less degree and few of the nodes are having a larger degree. The fraction of nodes having k connection with other nodes in the network depends on the value of k and a constant parameter. It can be mathematically defined as follows (Barabsi and Albert 1999):

$$F(k) = k^{-\gamma} \tag{1}$$

where F(k) is the fraction of node having out-degree k and y is the scale parameter typically ranging from 2 to 3. Scale-free network is the network which follows the power-law degree distribution. We can say that the real-world social network is scale-free in nature rather than following a random network where degree distribution among the nodes is random. Traditional tools are inefficient in handling a huge amount of unstructured data that are generated from the large-scale social network. Apart from the generated online social media data, the structure of the social network is quite complex in nature, being difficult to analyze. A distributed platform

like Spark (Gonzalez et al. 2014), Hadoop (White 2012) may be a suitable platform for analyzing large-scale social network efficiently.

Research Issues in Social Network

Nowadays the social network is observed to be an inevitable part of human life. The fundamental aspect of social networks is that they are rich in content and linkage data. The size of the social network is increasing rapidly as millions of users along with their relationships are added dynamically at every instance of time (Borgatti and Everett 2006). A huge amount of data is generated exponentially through the social network. The social network analysis is based on either linkage data or the content data of the network. A number of data mining and artificial intelligence techniques can be applied to these data for extracting useful pattern and knowledge. As the size of the data is huge, complex and heterogeneous in nature, traditional tools are inefficient in handling such data. Big data techniques may be helpful in analyzing the ever-increasing content data of the social network. However, in this chapter, we mainly focus on the structural analysis of the social network. Structural analysis can be carried out on either static or dynamic network. Static analysis of the network is possible only when the structure of the network does not change frequently, like in bibliographic network or co-authorship network where the author collaboration or citation count increases slowly over time. In static analysis, the entire network is processed in batch mode. Static analysis is easier in comparison to dynamic analysis where the structure of the networks is changing at every instance of time.

A large number of research problem may evolve in the context of structural analysis of the network. As we know, the structure of the social network changes at an exponential rate in the era of the Internet, the attributes involved in dynamics of the network also change. Modeling the dynamic structure of the social network is found to be quite a challenging task as the network parameters are changing more rapidly than expected. For example, as per small-world phenomena, any two entities are supposed to be separated by a small number of acquaintances but the actuality of this phenomenon may be fickle over the structural changes of the network. Verifying several structural properties in the dynamic network is found to be of great interest in recent years.

Linkage-based Social Network Analysis

Centrality Analysis

Centrality analysis is the process of identifying the important or most influential node in the network (Borgatti and Everett 2006). The meaning of importance may differ from application to application. In one context, a specified node is found to be the most influential node while in another context, the same node might not have a higher influence factor in the network, for example, in a social network website, it can be inferred that at one instance of time, Obama is the most powerful person in the world while in another instance of time, he might not be that powerful. Centrality analysis could be helpful in finding the relative importance of the person in the network. The word importance has a wide number of meanings that lead to different kinds of centrality measures. Evaluating centrality values for each node in a complex network is found to be quite a challenging task and considered as one of the important applications in the large-scale social network where data size is large. The value of centrality of a node might differ in the variety of centrality measure. The importance of the nodes can be analyzed by the following centrality measures:

Degree centrality

Degree centrality is the simplest form of centrality measure which counts the number of direct relationships of a node with all nodes in the network. The generalization of the degree centrality can be *k*-degree centrality, where *k* is the length of the path or connection from one node to all other nodes. The intuition behind this centrality measure is that people with a greater number of relationships or connections tend to be more powerful in the network. Measuring the degree centrality in the small network may be an easier task, where it is a computationally challenging for a large-scale network with millions of nodes connected with billions of edges. It can be mathematically represented as follows (Freeman 1978):

$$DC(i) = \frac{\sum_{j=1}^{n} a_{ij}}{n-1} \qquad a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{if } (i,j) \notin E \end{cases} \tag{2}$$

where *DC(i)* is the degree centrality of the node *i*; *n* is the total number of nodes in the network; a_{ij} represents the element of adjacency matrix for the network; the value of a_{ij} is 1, if there is an edge between node *i* and node *j*; otherwise, its value is 0. Degree centrality can be extended to the network level. It can be used to measure the heterogeneity of the network.

The measure of the degree variation in the network can be specified by the degree of centrality of the network which can be mathematically defined as follows:

$$DC(network) = \frac{\sum_{i=1}^{n} [DC(\max) - DC(i)]}{(n-1)(n-2)} \tag{3}$$

Here, *DC(network)* is the degree of centrality for the network; *DC(max)* is the maximum degree of the network.

Betweenness centrality

Betweenness centrality of a node is used to measure the amount of flow information that passes through the node. It is defined as the number of the shortest path between any pair of nodes that passes through the node in the network (Barthelemy 2004, Newman 2005). It is the most useful centrality measure that has a number of real-time applications. For example, in telecommunication cation network, a node with high betweenness centrality value may have better control over information flow because more information can pass through the node in community detection. Node with higher betweenness value may act as the cut node between two communities, in software-defined network. It can be used to find the appropriate position of the controller. Betweenness centrality of a node can be mathematically defined as follows (Brandes 2001):

$$BC(i) = \sum_{j \neq i \neq k} \frac{nsp_{jk}(i)}{nsp_{jk}} \tag{4}$$

where $BC(i)$ is the betweenness centrality for the node i; $nsp_{jk}(i)$ is the number of shortest paths between node j and k and which pass through node i; and nsp_{jk} is the total number of shortest paths between j and k.

Eigen-vector centrality

Eigen-vector centrality of the nodes is based on the principle of liner algebra which has a number of theoretical applications in the social network (Ruhnau 2000). The intuition behind eigenvector centrality is that a node is said to be more central if it is connected to more important nodes rather than having the connection with a greater number of unimportant nodes. According to the Perron Frobenius theorem, values of the eigenvector corresponding to highest eigen value of adjacency matrix represent the centrality value of each node. It can be defined as (Newman 2008):

$$EC(i) = \sum_{j=1}^{n} [a_{ij} * EC(j)] \quad where\ a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{if } (i,j) \notin E \end{cases} \tag{5}$$

where $EC(i)$ is the eigenvector centrality of node i; a_{ij} represents the element of the adjacency matrix. From equation 5, it can be observed that eigen centrality of node i depends on the eigen centrality of neighbouring node (j).

Closeness centrality

Closeness centrality is the measure of closeness of a node to all other nodes. A node which is more close to all other nodes may have a high influence factor in the network (Okamoto et al. 2008). In the social network, people who are more close to all other people may have a higher influence on others. Closeness centrality is nothing but the reciprocal of distance of a node from the rest of the nodes. It can be expressed as follows (Rahwan 2016):

$$CC(i) = \frac{1}{\sum_{j=1}^{n} d_{ij}} \tag{6}$$

where $CC(i)$ is the closeness centrality of node i and d_{ij} is the distance between nodes i and j.

Community Detection

Communities are found to be one of the most important features of the large-scale social network. Uncovering such hidden features enables the analysts to explore the functionalities of the social network. There exists quite a good number of definitions of community depending on the contexts pertaining to different applications. However, as per a number of commonly accepted definitions, they are considered to be a group of nodes which have a dense connection among themselves as compared to sparsity outside the group (Tang and Liu 2010, Newman and Girvan 2004). Communities in the social network represent a group of people who share common ideas and knowledge in the network. Their identification helps in getting insight into social and functional behavior in the social network. However, due to unprecedented growth in the size of the social network, it is quite a challenging task to discover subgroups in the network within a specified time limit. A number research in the direction of community detection was done recently. Community detection algorithms were basically classified into the following types (Fortunato 2010, Day and Edelsbrunner 1984):

- Node-based community detection
- Group-based community detection
- Network-based community detection
- Hierarchy-based community detection

In node-based community detection algorithms, group of nodes are said to form a community, if each node in the group satisfies certain network properties, like *k*-clique, where each node has connection with at least *k* number other nodes; *k*-plex where each node in the group has at least *n-k* number of connections. In group-based community detection algorithm, the group of nodes must meet the required constraints without zooming into the node level. A node within the community may not satisfy the criteria but as a group, if it satisfies the network properties, it can be treated as a community. Network-based community detection algorithm depends on certain criteria which must be satisfied by the network as a whole. Modularity and clustering coefficient are some of the well-known parameters that are used in network-based community detection algorithm. Communities are detected in a recursive manner in hierarchy-based community detection algorithms. It is further divided into two types—one is the agglomerative approach (Day and Edelsbrunner 1984) and the second is the divisive approach (Reichardt and Bornholdt 2006). In agglomerative approach, initially each node is considered as a community. At each step, the set of nodes are grouped into the community in a way that improves certain network parameters, like modularity. Grouping of nodes is considered to be a community at a certain point, where no further improvement of the parameter is possible. Unlike agglomerative approach, divisive algorithms constitute the top-down approach where the whole network is considered as a community at the initial stage. Consequently, nodes are partitioned into communities by removal of edges from the network in a way that improves the network parameter. Similar to the first approach set of nodes is grouping into communities where no further improvement of the parameter is possible.

Community detection in the large-scale network is an extremely challenging task as it involves high computational complexity. In the real-world social network, the structure of the network changes frequently as a number of nodes are added and/or removed at every instance of time. Structure of communities may change over time which the analyst has difficulty in identifying.

Link Prediction

Link prediction is found to be the most interesting research problem in social network analysis. Here the analyst tries to predict the chances of formation of hidden links in the social network. It has a wide number of applications, such as suggesting friends in the online social network, recommending e-commerce product to users, detecting links between terrorists so as to avoid future unwanted circumstances. The basic objective of this problem is to determine the important future linkage information in

the network (Liben-Nowell and Kleinberg 2007). By utilizing the linkage information, one can predict the future potential relationships and hidden links in the network. Link prediction in the large-scale network is found to be an NP-complete problem. Most of the researchers are utilizing structural information to make the prediction. However, it may also be possible to predict the link using node attributes. Structural link prediction can be classified as node-based or the path-based. Node-based link prediction is based on the similarity score between the nodes computed by utilizing the local topological structure around the node. A number of similarity indices have been proposed by several researchers in literature. Path-based link prediction score is based on the global structural information of the network. The intuition behind the path-based link prediction is that if two nodes are going to be connected in future, then there must be a path between the two nodes in current topological information. Less is the path length the higher the chance of them getting connected in future.

Modelling Network Evolution

Probably social network is the fastest-growing dynamic entity in the real-world network. It is important to study the structure of network dynamics and mechanisms behind the network evolution (Grindrod et al. 2011). The structure of the social network is inherently dynamic in nature as a large number of entities are dynamically added along with their relationships. A few of the entities may also want to be disconnected from the network over time. This leads to change in several network parameters, like the number and structure of communities, node centrality, degree distribution, clustering coefficient, modularity, etc. Modelling of the dynamics can be used in generating the synthetic networks whose properties resemble a real-world social network. Scale-free network model (Holme and Kim 2002), random graph model (Aiello et al. 2000), exponential random graph model (Robins et al. 2007) and small world model (Newman and Watts 1999) are some of the widely accepted models which generate the real-world social network. A network is said to be scale-free if its degree distribution is observed to follow the power law which is already discussed in equation 1. A few nodes in the network are observed to have an unusually high degree as compared to other nodes and the degree of most of the nodes is quite less. A number of complex network systems, like citation network, social network, communication network, internet, the worldwide web are said to follow scale-free networks. Unlike scale-free network, the random model is a generative model where the degree distribution in the network is random and most importantly, it follows the small-world network. Small-world network is a network model where any two nodes in the network can be reachable by a certain number of

acquaintances. Exponential random graph model (ERGM) is one kind of statistical model that tries to analyze the statistical properties of the social network (Robins et al. 2007). The statistical metric, like density, assortative, centralities capture the structural properties of the network at a specific instance of time. The ERGM model is considered to be the most suitable model for capturing the differences in statistical metrics of the network at two different instances of time. Capturing and presenting the accurate network dynamics in the large-scale network with the help of different generative network models may need a huge amount of computational time. Some big data tools, like GraphX component in Spark framework may be helpful in capturing and analyzing the huge data in order to model the real-world network (Xin et al. 2013).

Social Influence Analysis

Social influence is the behavioural change of a social entity due to relationships with other people, organizations, community, etc. Social influence analysis has attracted a lot of attention over the past few years. By analyzing the strength of influences among the people in society, a number of applications for advertising, recommendations, marketing may be developed. The strength of influence depends on many factors, like the distance between people, network size, clustering coefficient and other parameters. Since the social network is a collection of relationships between social entities, influence of one entity may affect the activity of other. The influencing factor of the entity depends on the importance of that entity in the network which can be analyzed through centrality analysis. In social influence analysis, researchers are trying to model the spread of influence, and its impact. Identifying most influential persons allows the researcher to seed the information at the proper location in order to propagate it effectively. The size of social network services like Facebook, Twitter, LinkedIn is increasing at an exponential, allowing the researchers to analyze the influence of large-scale network.

Big Data Analytics in Social Network

In the past decade, the amount of data has been created very steeply. More than 30,000 GB of data are generated every second with a great rate of acceleration. These data may be structured, unstructured or semi-structured. The sources of these data are blog posts, social network data, medical, business data, digital data, research data, scientific data, internet data, etc. The internet is the ultimate as the source of data, which is almost indecipherable. The volume of data available at present is indeed huge, but it may not be the most relevant characteristic of data. The extreme growth of

data severely influences business. Business executives, scientists, medical practitioners, business executives, governments and advertising alike regularly face problems with huge data-sets in areas including finance, Internet search, business informatics and urban informatics. The main objective of today's era is simply to extract value from data by means of user behavior analytics, predictive analytics, or any other advanced data analysis method that rarely works on a specific size of data set. Traditional database systems, such as relational database management systems, are inadequate to deal with a large volume of data. These database systems face the challenges of storage, data curation, capture, search, analysis, updating, sharing, visualization, transfer, querying and information privacy (Fig. 1).

Big data mainly addresses the scalability and complexity issues that appear in a traditional dramatic fashion within a reasonable elapsed time. It mainly refers to the large and complex data sets that are not conventionally analyzed by traditional database systems. It needs a set of new techniques and technologies in order to extract useful information from datasets that are heterogeneous in nature.

Big data can also be defined in terms of 'three V's' that occur due to data management challenges (Zikopoulos et al. 2011). Three V's are presented in Fig. 2. Traditional database systems are unable to solve the issues related to these three V's. Here the 'three V's' of big data is defined as volume that ranges from MB (megabytes) to PB (petabytes) of data. The velocity defines the rate at which data are generated and delivered to the end user and variety includes data from a different variety of formats and sources (e.g., financial transactions, e-commerce, online transactions, social media interactions and weblogs, bioinformatics data, medical data, etc.).

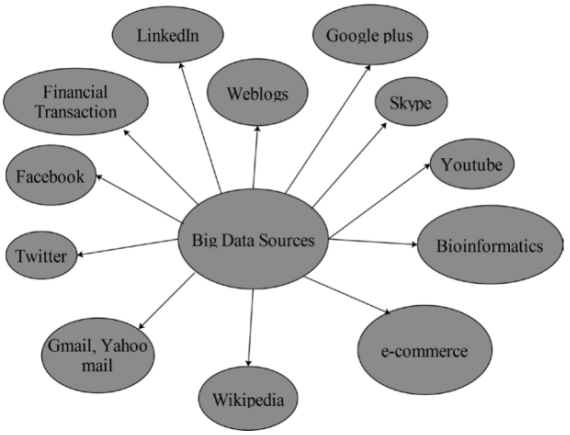

Fig. 1. Sources of big data in social network.

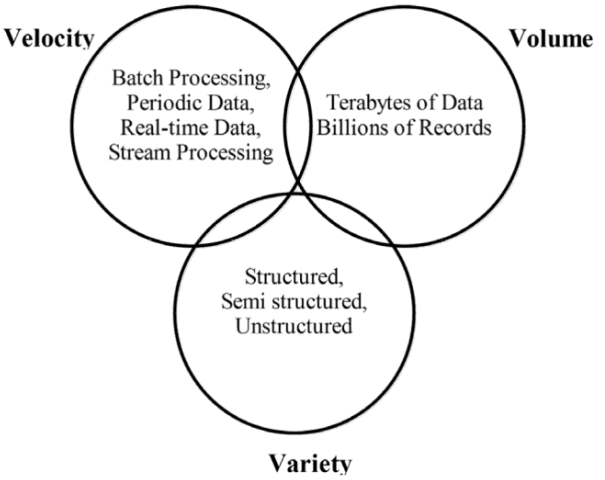

Fig. 2. 3 V's of big data analytics.

Application of Big Data Analytics

- Marketing agencies predict the future strategies by studying the information kept in the social networks like Twitter, Instagram, WhatsApp, Facebook, etc.
- Business organizations learn the reaction of their consumer regarding their products, using the information in the social media, such as product perception, preferences, etc.
- By using the historical data of the patient, analysis can be done to provide better and quick services in the healthcare centre.
- Organizations can identify the errors within a quick span of time. Real-time error detection helps organizations to react quickly in order to minimize the effects due to any operational problem. This prevents consumers from stopping use of products and save the operation from failing completely or falling behind.
- Big data analytic tools eventually save a lot of money even though expensive. They help to reduce the extra burdens of business leaders and analysts.
- Fraud detection is one of the most useful applications of big data analytics where criminals are easily identified and damage can be minimized by taking prompt measures.
- Organizations can increase their revenue by improving the quality of services by monitoring the products used by their customers.

Big Data System

Following are few of the desirable properties of the big data systems:

- *Robustness and fault tolerance*: In all circumstances, the system must behave correctly. It is essential that the system must be human-fault tolerant.
- *Low latency reads and updates*: System must achieve low latency reads and updates without compromising the robustness of the system.
- *Scalability*: Maintaining performance by adding resources to the existing system when load increases.
- *Generalization*: It must be useful in a wide range of application.
- *Extensibility*: Extra functional features can be added without changing the internal logic of the system.
- *Ad hoc queries*: Ability to mine unanticipated values from large dataset arbitrarily.
- *Minimal maintenance*: System must be maintainable (keep running smoothly) at minimum cost. One way of achieving this goal is by choosing the component with less implementation complexity.
- *Debuggability*: System must provide all essential information for debugging the system if any-thing goes wrong.

How Does Big Data Work?

With the development of modern tools and technologies, it is possible to perform different operations on larger datasets and valuable information can be extracted by analyzing the datasets. Big data technologies make it technically and economically feasible. Processing of big data involves different steps that begin from collection of raw information to implementation of actionable information.

Collection of data: Many organizations face different challenges in the beginning while dealing with big data. Such challenges are mobile devices, logs, raw data transactions, social sites, etc., but these are overcome by an efficient big data platform and developers are allowed to ingest the data from different varieties.

Storing: Preprocessing and post-processing data are stored in a secure, durable and scalable repository system. This is an essential step for any big data platform. Depending on the particular requirements, the user can opt for temporary storage of data.

Processing & Analyzing: Transformation of raw data into user consumable data through various stages, like sorting, joining, aggregating, etc. The result data sets are stored and used for further processing tasks.

Consuming & Visualizing: Finally the end user gets valuable insights from the processed data sets and according to requirements the user plan for future strategies.

Evolution of Big Data Processing

Currently basically three different styles of analysis are done on data sets to carry out multiple functions within an organization. These are as follows:

Descriptive Analysis: It helps the user to get the answer to the question: "Why it happened and what happened?"

Predictive Analysis: Probability of certain events in the feature data sets are calculated by the user, like forecasting, preventive maintenance applications, fraud detection, early alert systems, etc.

Prescriptive Analysis: User gets recommended for different events and gets the answer to the question: "What should I do if 'x' happens"?

Originally, Hadoop is the most familiar framework of big data analytics that mainly supports batch processing, where datasets of huge size are processed in bulk. However, due to time constraints new frameworks, such as Apache Spark, Amazon Kinesis, Apache Kafka, etc., have been developed to support streaming data and real-time data processing.

Traditional Approach for Data Analytics

In the traditional approach, data was stored in an RDBMS like DB2, MS SQL Server or Oracle databases and this software interacts with the sophisticated software to process the data for analysis purposes. Figure 3 presents the interaction among the major components in the traditional processing system.

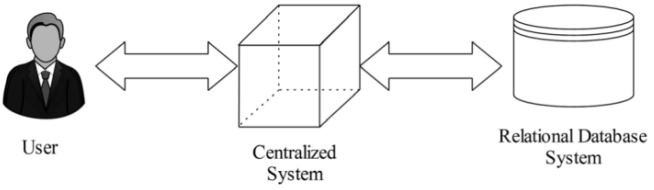

Fig. 3. Interaction in the traditional processing system.

Limitations

The traditional approach is best suited for works where the user deals with a less volume of data. While dealing with large volume of data, it was observed that the traditional database server is unable to process such data efficiently. In the traditional system, storing and retrieving volumes of data faced three major issues—as cost, speed and reliability.

Hadoop Framework

Map-Reduce programming which is found to be a most suitable approach for analyzing big data was first adopted by Doug Cutting, Mike Cafarella in their open source project named Hadoop in 2005 (Bialecki 2005). It was named after his boy's toy elephant. Hadoop is a distributed platform where data can be processed in multiple nodes in a cluster. Statistical analysis of a huge amount of data was made possible using Hadoop in a reasonable amount of time. The computational time also depends on the dependencies that exist among the data. Hadoop was written in java programming language; however, it is also compatible with python, R, and Ruby. Hadoop provides scalable and fault-tolerant platform by keeping the replica of data in multiple nodes which also increases the reliability of the programming environment. As we know, the social network data is very large and traditional programming tools are quite incapable of handling such enormous data. Hadoop is considered as a suitable programming environment for big data analytics.

Hadoop Architecture

The basic modules for Hadoop distributed framework are as follows:

- *Hadoop Map Reduce*: A programming module where data is processed in two phases—map and reduce.
- *Hadoop Common*: Common Java library files reside in this module which can be used by other modules of Hadoop. It also provides OS level abstraction and library file to start and stop programming platform.
- *Hadoop YARN*: This module contains scripts for task scheduling and resource management in the cluster of computing nodes (Vavilapalli 2013).
- *Hadoop Distributed File System (HDFS)*: It is the dedicated file system for Hadoop framework where files can be distributed across thousands of nodes. One cannot access the files of HDFS in the local file system (Shvachko et al. 2010).

Map Reduce Programming Model

Map Reduce is a programming model suitable for distributed and process the huge amount of data in several commodity nodes in the cluster simultaneously. Reliability, scalability and fault tolerant are the key features of the map reduce programming model. Each data processing in the map-reduce environment must pass through two phases, i.e., mapper and reducer. Map-reduce programming model is presented in Fig. 4.

Mapper Phase: It is the initial phase of data processing, where input data is first transformed into the set of key-value pair and passes as the argument to the mapper module. The output of the mapper phase is another set of key-value pair.

Reducer Phase: This phase takes the output of mapper as input key-value pair and reduces the size of tuple set by combining the value of each key. The output of the reducer is also a key-value pair. The reducer phase always performs after the mapper phase.

HDFS is the file system where data are distributed across multiple computing nodes. All the inputs and outputs are communicated with HDFS file systems throughout the processing steps. The framework has the responsibility for both scheduling and monitoring the task. If any task fails, it is automatically handled in a fault-tolerant manner. In the cluster of nodes, one node is considered as master and the others are treated as slaves. The framework consists of single Job Tracker for master and a Task Tracker for each slave in the cluster. The master is responsible for distributing and monitoring the task across several nodes. Job Tracker keeps track of resource consumption by all the nodes. Slave's Task Tracker executes the assigned task to the slave and provides the status of execution to the master periodically. The master is the single-point failure in Hadoop environment. If the master fails to execute, all the tasks of the slaves come to a halt.

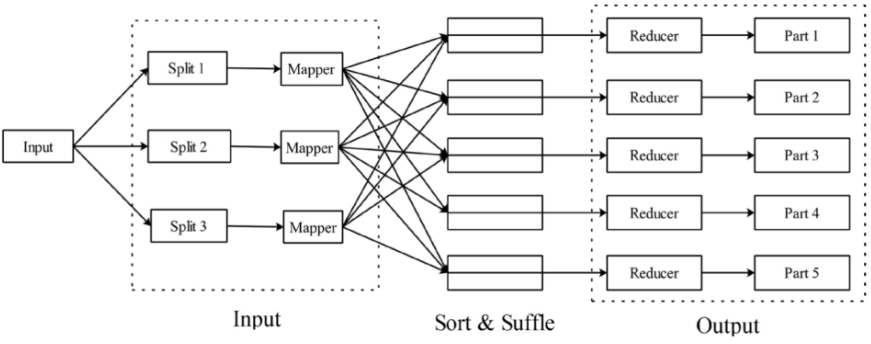

Fig. 4. Map-reduce programming model in Hadoop platform.

Advantages of Hadoop

- Hadoop is one of the most suitable tools for quickly distributing the dataset in multiple computing nodes for processing simultaneously.
- Hadoop can handle the data in a fault-tolerant and reliable manner automatically by keeping multiple blocks of the dataset in several nodes. It has a dedicated library for handling any failure at the application layer.
- The data node, i.e., one of the slave nodes can be added or deleted from the system without affecting the work in progress.
- One of the major advantages of Hadoop is that it is compatible with all the platforms. Apart from Java, it also supports a number of other languages, like Python, R, Ruby, etc.

Limitation of Hadoop

- For a better reliability and fault tolerant, multiple copies of datasets are stored in HDFS file system. It is inefficient for storage space where data size is very big.
- Hadoop is based on NoSQL database where a limited query of SQL is supported. Some open source software could make use of Hadoop framework but due to limited SQL query, they might not be that much useful.
- The execution environment is not efficient due to lack of query optimizer; size of the cluster must be large enough to handle the similar kind of database of moderate size.
- For execution in Hadoop framework, dataset must be transformed into key-value pair pattern; however, it is not possible to transform all datasets into the specified key-value pair. Therefore, it is found to be a challenging execution environment for many big data problems.
- Writing efficient program requires skill set for both data mining and distributed programming where knowledge of parallel execution environment is necessary.
- Hadoop is based on map-reduce programming model where data has to land between map and reduce phase which requires a lot of IO operation.
- Hadoop is not suitable for real-time data streaming analysis.

Challenges Faces in Hadoop Framework

- Hadoop is the distributed framework, where data are distributed across several computing nodes. The most challenging task in Hadoop is to manage the execution environment. The security model for Hadoop is by default unable to perform efficient execution. The one who wants to manage Hadoop requires to have knowledge about the security model. Encryption level at storage and network level is less tight, so huge data may be at risk in Hadoop framework.
- Hadoop is written entirely in Java language. Although Java is platform independent and widely used programming language, it is heavily vulnerable to cybercriminal activities that may lead to security breaches in Hadoop.
- Hadoop is not at all suited to small sized data. It takes more computational time as compared to traditional systems for small size data. It is not recommended for organizations where data size is not heavy.

Spark Programming Framework

Apache Spark was designed as a computing platform to be fast, general purpose and easy to use (Shoro 2015). The question is why should the user want to use Spark? As we know, Spark is closely related to map reduce in a sense that it extends on its capabilities. For batch processing, Hadoop is best, but overall, Spark is faster than others. So Spark is an open source processing engine built around speed, ease of use and analytics. If we have a large amount of data that requires low latency processing that a typical map reduces program cannot provide, then Spark is the way to go. Generally, computations are of two types—one is I/O intensive and another is CPU intensive. Spark performance seems to be much better for I/O intensive computation. In Spark, data is loaded in a distributed fashion in memory and is free from data locality concept. Spark is completely in-memory processing; hence it is faster than Hadoop.

Like map reduces, Spark provides parallel distributed processing, fault tolerance on commodity hardware, scalability, etc. Spark tool is time- and cost-efficient due to the concept called cached in memory distributed computing, high-level APIs, low latency and stack of high-level tools. Mainly two groups of people use Spark—one is data scientist and another is engineer.

Data scientists analyze and model the data to obtain information. They use some techniques to transform the data into some suitable form so that they can easily analyze. They use Spark to get the results immediately. Once the hidden information is extracted from the data, later the persons

who work on this hidden information are called engineers. Engineers use Spark's programming APIs to develop business applications while hiding the complexities of distributed system programming and fault tolerance across the clusters. Engineers can also use Spark for monitoring and inspecting applications.

Component of Spark Framework

Spark core is a general purpose system that provides scheduling, distributing and monitoring of the applications in a cluster. On top of the core, components are designed to interoperate closely, so that users can combine them as a library in a software project. Top-level components will inherit the improvements made at the lower layers, e.g., by optimizing the Spark core speed of SQL, machine learning, the streaming and graph processing libraries increase. Spark core can even run with its own built-in scheduler. It also can run over a different cluster manager, like apache mesos and Hadoop YARN. Spark components reside above the Spark core are shown in Fig. 5.

Spark SQL: It works with unified data access which means the user can have any kind of data. Since it is compatible with Hive, the user can run hive queries on top of the Spark SQL. As a result, time is saved on migrating hive queries to Spark queries and no conversion is required. Queries are embedded in Python, Scala and Java. It gives a lot of tight integration with SQL while providing standard connectivity for all popular tools, like ODBC, JDBC.

Spark Streaming: Streaming task means immediately processing data and generating the result at runtime. It is a real-time language integrated APIs to do stream processing by using Java and Scala. This stream processing framework is fault tolerant because Spark has got an excellent recurring mechanism. The user can also combine stream processing with another component, like MLib, GraphX, etc. (Meng et al. 2016, Gonzalez et al. 2014). User can run streaming with both batch and interactive mode. Streaming

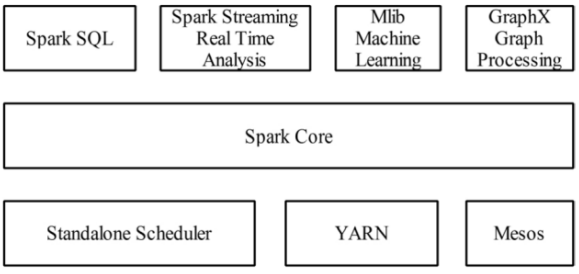

Fig. 5. Spark components.

also integrates with lots of existing standards like HDFS, FLUME, KAFKA (Hoffman 2013, Garg 2013). Kafka is distributed message queue which is used for data processing in streaming work. It was written by Twitter. Flume framework works on top of Hadoop. Hadoop performs poor in streaming data and does not have provision for processing streaming data.

Spark MLib: It is a collection of machine learning library functions and runs 100 times faster than Hadoop because of Spark storing data in memory. It has a lot of machine learning algorithm like *k*-means clustering, linear regression, logistic regression, etc. Spark optimizes the algorithms to run top of Spark platform. One main feature of this component is that it can deploy existing Hadoop clusters. In Hadoop, there is no provision for built-in machine learning algorithm. The user has to use Mahout Platform to build an application in machine learning (Lyubimov 2016).

RDD: RDD stands for Resilient Distributed Data which is a basic abstraction unit of Spark. Spark provides APIs to interact directly with RDD, which has a higher partition tolerance as compared to HDFS. In HDFS the user gets partition by doing replication. But in RDD the replication factor is lot less as compared to HDFS, so it requires less space. RDD is immutable and Spark stores the data in RDD.

GraphX: Spark provides an optimized framework for graph processing. An optimized engine supports the general execution of graphs called DAGs. A lot of built-in algorithms are present for graph processing. GraphX is a unique feature in Spark (Gonzalez et al. 2014).

Functional features of Spark

- It provides powerful caching and disk persistence capabilities.
- Faster batch processing as compared to Hadoop.
- Real-time stream processing.
- Faster decision making, because data resides in memory. Hence the user can definitely go for iterative algorithms. This is a severe limitation in Hadoop.
- It supports iterative algorithms. In Hadoop, sharing of data is not possible which leads to non-iterative algorithms.
- Spark is interactive data analysis, whereas Hadoop is non-interactive.

Non-functional Features of Spark

- A fully apache hive compatible data warehousing system that can run 100x faster than Hive.

- Spark framework is polyglot, or it can be programmed in several programming languages (currently in Java, Python and Scala).
- Spark is not really dependent on Hadoop for its implementation as it has got its own manager.

What is Streaming Data?

Because of the dynamic nature of social network, data is generated continuously at an exponential rate. Real-time data that stream in the era of big data can be considered as streaming data. They can include a variety of data in different sizes. This homogenous nature of data makes the analysis more complex. The major source for streaming data comes from the log files that are generated from e-commerce website, online gaming activity, message exchanges by the user in the online social network, weather forecasting, information generated from health care systems, etc. Streaming data must be analyzed on a record-by-record basis over a time-frame. A wide range of analyses like regression analysis, correlation, aggregation, sampling, filtering can be done on real-time streaming data. Visualization of information gathered from these analyses can be utilized to improve the marketing business by responding to the upcoming situation effectively. For example, companies can analyze the sentiment data over their e-commerce product for building a better recommendation-engine for their product.

Benefits of Streaming Data

Streaming data processing is helpful in many situations where new, dynamic information is produced consistently. It applies to a majority of the business portions and huge information utilize cases. Organizations, for the most part, start with basic applications, like gathering framework logs and simple handling, like moving min-max calculations. Hence, these applications develop to more complex real-time processing. At first, applications may handle information streams to deliver basic reports and perform basic activities accordingly, for example, discharging cautions when key measures surpass a particular threshold. In the end, these applications perform more refined types of information examination, such as machine learning algorithm applications and gather further bits of knowledge from information. Further, stream, complex and algorithms involving event processing, such as decaying time windows to locate the latest prominent motion pictures, are connected, additionally enhancing the knowledge.

Examples of Streaming Data

Sensors in vehicles used for transportation, modern hardware and machines used for farming send information to a streaming application. The application screens execution performance and identifies any potential errors ahead of time, and puts in an extra part order request that prevents equipment downtime.

- Changes in the share trading system can be tracked progressively by a financial institution, which naturally, rebalances portfolios in view of stock value developments.
- A subset of information from customers' cell phones is tracked by the real-estate websites and continuous property suggestions of properties to visit in light of their geolocation is made.
- A company using solar energy to replenish needs to keep up power throughput for its clients or pay penalty. It thus implements an application based on streaming data analysis which scrutinizes all the panels in the field, scheduling the service in real-time, which lessens the duration of throughput from each and every panel along with its associated penalty payouts.
- Humongous numbers of clickstream records are streamed by media publishers taken from their online contents, thus aggregating and enriching the data with certain location-related information about its users. They also optimise the placement of the content on its site, which delivers a better and relevant experience to its users.
- Streaming data that involves the interaction between a player and a game can be collected by an online gaming company. This collected data is fed into their platform which analyses real-time data and offers new experiences to the players who play it which gives them a better liking for the game.

Comparison between Batch Processing and Stream Processing

Before diving into the arena of streaming data, we can have a look at the differences between two major areas of processing techniques, namely, batch processing and stream processing. Batch processing can be utilized to figure random inquiries over various information. It computes results which are computed from all the already encompassed data, thus enabling a deep and close analysis of big datasets. Amazon EMR, which is a map-reduce-based system is an example of a platform which supports batch jobs. To the contrast, ingesting a sequence of data is something required by stream processing. Incremental updating metrics, summary statistics and reports in response to each arriving data record are also sometimes a necessity for stream processing which makes it more efficient for real-time processing and other responsive functions.

Table 1. Comparison between batch processing and stream processing.

	Batch Processing	**Stream Processing**
Data scope	Uses all or most of the data in the dataset for querying and processing.	Uses data within a very small time window or on just the most recent data record for processing and querying.
Data size	Huge amount of data, in the form of batches of data, is used.	Single records or very small batches called 'micro-batches' of records are used.
Performance	Offers more latency which last in minutes to hours.	Provides less latency which is in the order of few seconds or milliseconds.
Analyses	Analyses of these data are quite complex.	Analysis is done through simple response functions, aggregates, and rolling metrics.

Challenges in Analyzing Streaming Data

There are two layers for streaming data processing—a processing layer and a storage layer. There lies a need to support ordering of records and strong consistency to have a faster, less costly and reads and writes of large streams of data which are repayable by the storage layer. The second layer, that is the processing layer, is responsible for consumption of data from the storage layer, executing various actions on that data and finally, notifying the storage layer to remove the data that is no longer required. Along with all the already available features, we also need to have a plan which ensures fault tolerance, data durability and scalability in both streaming data processing layers namely, the storage and processing layers. There have been many platforms that have been developed as a result of this, which provide the platform and the framework required to build streaming data applications. Some of these are Apache Storm, Apache Flume, Apache Spark Streaming and Apache Kafka.

Conclusion

Social network analysis is a typical example of an idea that can be applied in many fields. With mathematical graph theory as its basis, it has become a multidisciplinary approach with applications in sociology, the information sciences, computer sciences, geography, etc. In this chapter, different research direction in social network analysis has been explained with relevant application. As the data generated from social network is increasing at an exponential rate, the traditional system fails to process and analyze such huge heterogeneous and complex data. Hadoop or Spark may be considered as an alternative approach for analyzing such data as they have an individual speciality in analyzing the social structure

and social media data. Hadoop is basically used to analyze batch mode social data, whereas Spark may be considered as suitable for real-time streaming data.

References

Aggarwal, Charu C. (2011). An Introduction to Social Network Data Analytics. In Social Network Data Analytics. Springer, Boston, MA, 1–15.

Aiello, William, Fan Chung and Linyuan, Lu. (2000). A random graph model for massive graphs. Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing 171–180.

Barabsi, Albertszl and Rka Albert. (1999). Emergence of scaling in random networks. Science (American Association for the Advancement of Science) 286(5439): 509–512.

Barthelemy, Marc. (2004). Betweenness centrality in large complex networks. The European Physical Journal B-Condensed Matter and Complex Systems 38(2): 163–168.

Bialecki, Andrzej. (2005). Hadoop: a framework for running applications on large clusters built of commodity hardware. http://lucene. apache. org/hadoop.

Borgatti, Stephen, P. and Martin G. Everett. (2006). A graph-theoretic perspective on centrality. Social Networks 28(4): 466–484.

Brandes, Ulrik. (2001). A faster algorithm for betweenness centrality. Journal of Mathematical Sociology 25(2): 163–177.

Day, William H.E. and Herbert Edelsbrunner. (1984). Efficient algorithms for agglomerative hierarchical clustering methods. Journal of Classification 1(1): 7–24.

Fortunato, Santo. (2010). Community detection in graphs. Physics Reports 486(3): 75–174.

Freeman, Linton C. (1978). Centrality in social networks conceptual clarification. Social Networks 1(3): 215–239.

Garg, Nishant. (2013). Apache Kafka. Packt Publishing Ltd. Birmingham, United Kingdom.

Gonzalez, Joseph E., Reynold S. Xin, Ankur Dave, Daniel Crankshaw, Michael J. Franklin and Ion Stoica. (2014). GraphX: Graph processing in a distributed dataflow framework. OSDI 599–613.

Grindrod, Peter, Mark C. Parsons, Desmond J. Higham and Ernesto Estrada. (2011). Communicability across evolving networks. Physical Review E 83(4): 046120.

Gulati, Ranjay. (1995). Social structure and alliance formation patterns: A longitudinal analysis. Administrative Science Quarterly 619–652.

He, Wu, Shenghua Zha and Ling Li. (2013). Social media competitive analysis and text mining: A case study in the pizza industry. International Journal of Information Management 33(3): 464–472.

Hey, Tony, Stewart Tansley, Kristin M. Tolle and others. 2009. The Fourth Paradigm: Data-intensive Scientific Discovery. Vol. 1. Microsoft Research Redmond, WA.

Hoffman, Steve. (2013). Apache Flume: Distributed Log Collection for Hadoop. Packt Publishing Ltd.

Holme, Petter and Beom Jun Kim. (2002). Growing scale-free networks with tunable clustering. Physical Review E 65(2): 026107.

Kleinfeld, Judith S. (2002). The small world problem. Society 39(2): 61–66.

Knoke, David and Song Yang. (2008). Social Network Analysis. Vol. 154. Sage.

Liben-Nowell, David and Jon Kleinberg. (2007). The link-prediction problem for social networks. Journal of the Association for Information Science and Technology 58(7): 1019–1031.

Lyubimov, Dmitriy and Palumbo, Andrew. (2016). Apache Mahout: Beyond MapReduce. CreateSpace Independent Publishing Platform.

Meng, Xiangrui, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman et al. 2016. Mllib: Machine learning in apache spark. The Journal of Machine Learning Research 17(1): 1235–1241.

Index