# EVALUATION OF TRUST PATH AMONG USERS IN ONLINE SOCIAL NETWORKS USING HADOOP MAP REDUCE

A. Satish Kumar

Research Scholar, School of Computing,
Sathyabama Institute of Science & Technology,
Chennai, India
satishathmakuri@gmail.com

Dr. S. Revathy

Associate Professor, Department of IT,
Sathyabama Institute of Science & Technology,
Chennai, India
revathy.it@sathyabama.ac.in

**Abstract**

**Online Social Networks have become vital part of our daily life. These social networks have steered into many new notions in social media, thereby providing an opportunity and to extend their mutual relationships among users though they are physically not connected. Now-a-days, there are many available social network platforms like Instagram, Twitter and Facebook. The users of these social networks are usually unknown with each other in person, but they share a lot of information through various posts, messages and other activities. If both sender and receiver are physically unknown, but connected via social media as friends, then trustworthiness among them is compulsory. Establishing a trust path between the end users is a very trivial task to be accomplished and many researchers have proposed various algorithms in their research work. This paper presents an algorithm using Hadoop Map Reduce to establish a trust path among various individuals in online social networks.**

***Key Words:*** **Online Social Network, Trustworthiness, Hadoop, Map Reduce.**

## 1. Introduction

Ever since the "Internet Time" has begun by the World Wide Web, the main goal of it is to provide an opportunity for all the users a platform to share their thoughts, exchange messages. Online social networks have become a part of our day-today life and experienced a tremendous growth in terms of usage and stay connected even though they are globally separated. Before any individual is trying to access any social network through online, he/she has to create an account based upon their respective credentials and make themselves authenticated. Only then, the users are given a provision to post or share any kind of information to all the other users whom are connected via the same network.

The online social network maintains a database of millions and trillions of users, huge amount of data they share on a regular daily basis. These transactions should be as smooth enough as possible, as the users expect the seamless connectivity of the online social network. In all this process, a factor which plays a crucial role is trust. All the users who are connected in the online social network have to ensure that they are having a trusted sort of information exchange between them. This paper take "Trust" as a primary attribute of any social network and our paper ensures that any exchange of message between the users connected in the network is properly verified and got authenticated. Also we assume that a person's decision making would be greatly affected the opinions shared by his companions. Social network sites are used to express thoughts, emotions and opinion of users to all other users who are part of the said network on a regular basis and thereby generating huge amount of online data.

As the name implies, social networks are completely online i.e. web based. Every user who has to access social network is to be compulsory online. This leads to the tremendous increase in the size of data getting stored electronically. As the years getting passed, the data that is getting stored electronically is increasing at greater speeds. To handle this huge amount of data, organizations and researchers have found the traditional techniques like RDBMS and SQL Server are inadequate. So, to handle this huge data, later named as "Big Data", Google

was the one who came up with a solution to process this big data called "Google File system (GFS)". The google file system is comprised of multiple nodes located globally across the world. These nodes are divided into mainly two categories: A single Master Node and multiple Chunk Servers. These chunk servers used to store all the distributed data on their local disks and are controlled by the master node. That is how the big data was processed using google file system. Later in 2005, Hadoop framework was launched to perform processing of big data that is distributed across the nodes for a given cluster. Hadoop is built based on Google File system.

In this paper, we have proposed few algorithms that establish trustworthiness among all the users and evaluate a trust path using which; any two individuals can share their views or opinions. The main framework we used in this paper is "Hadoop". Hadoop is used for distributed storage and parallel data processing. Our proposed algorithm is able to create a combined evaluation of Map Reduce and Trustworthiness in online social networks. It is very useful in the distributed environment for huge number data and performs classification faster than standalone system (Dr.S.Revathy).

## 2. Related Work

### 2.1 *Big Data*

Data is the collection of raw facts that can be stored, processed and analysed. When the internet usage growth has tremendously increased across the globe, the size of the data has got increased at larger rate and traditional techniques have failed to handle such enormous huge data. Thus generated huge data was termed as "Big Data". There are multiple data sources which were pouring data online at regular periods like data generating through social media, data generated via stock exchange, search engine data and etc. Though the size of data is getting larger, users and analysts expect the data access speed to be faster. Thereby "Big Data" has evolved as a challenging task to handle. There are three main characteristics included in big data, they are: Huge Volume, High Velocity and Variety of data. Volume implies the size of data (which has to be very huge), Velocity implies the rate of speed at which the data is processed and stored. Variety implies the types of data that can be included in big data. The traditional data used to store only structured data like in the form of tables, using RDBMS (Relational Database Management systems). These relations (tables) were processed using any query languages like Sql Server, MySQL, Oracle, etc. In contrast to this, big data has to handle all kinds of data like Structured (tables), Semi-structured (XML data) and Unstructured (media files).

### 2.2 *Hadoop*

Hadoop was introduced by "Doug Cutting" in the year 2005. This is a framework designed to process big data that is distributed across the nodes. Hadoop was built in such a way that it can run efficiently on commodity hardware. Hadoop has two main components named "HDFS (Hadoop Distributed File system)" and "Map Reduce". The HDFS is used to store large amounts of data that is being generated among the nodes and has to be distributed among the cluster. Coming to our research, since we are dealing with the data that is getting generated at faster rates through various social media networks, HDFS is the perfect solution to store such vast amount of data. Again to process the data that is stored in HDFS, we need a specialized framework called "Map Reduce". Hadoop is robust, scalable and fault-tolerant. The main four components of hadoop that are needed to store big data could be diagrammatically represented as follows:

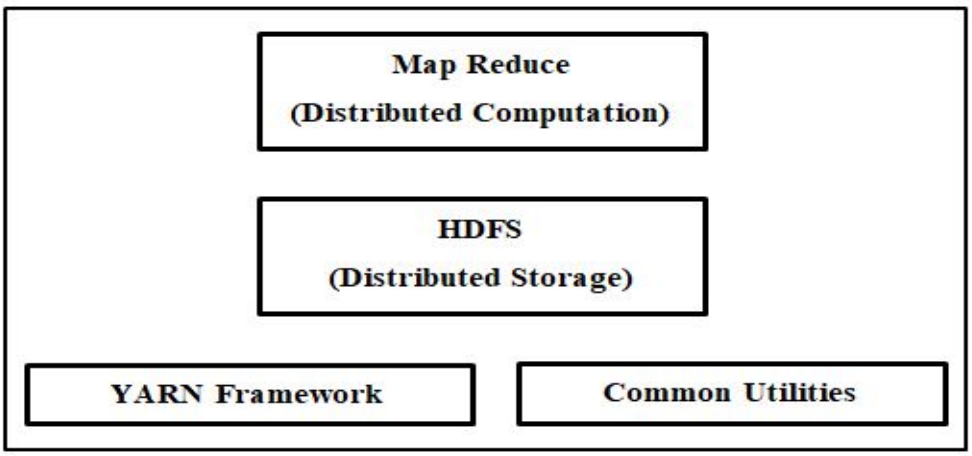



Fig 1: Hadoop Components

### 2.3 *Map Reduce*

Map Reduce is the other main component of Hadoop, whose job is to process the big data that is stored in HDFS. The programs used in map reduce are written in Java. The map reduce phase consists of two stages named "Map stage" and "Reduce stage". The map stage is the first step in data processing, where the given input data is processed with the help of multiple mappers and thereby produces <Key,Value> pairs for the given data. The number of mappers to be used is optional for the programmer or the analyst based upon the size and type of the data and depends on the context. There can be any number of mappers executed for the given data. Each mapper has to generate the entire possible <Key,Value> pairs. In the second stage i.e. reduce stage, all these generated <Key,Value> pairs are combined, portioned, shuffled and sorted using multiple routines written in a reducer code and thereby evaluates the expected result. In our case, the reducer has to evaluate the trust path between the users for a given network. The map reduce phase is diagrammatically represented as follows:

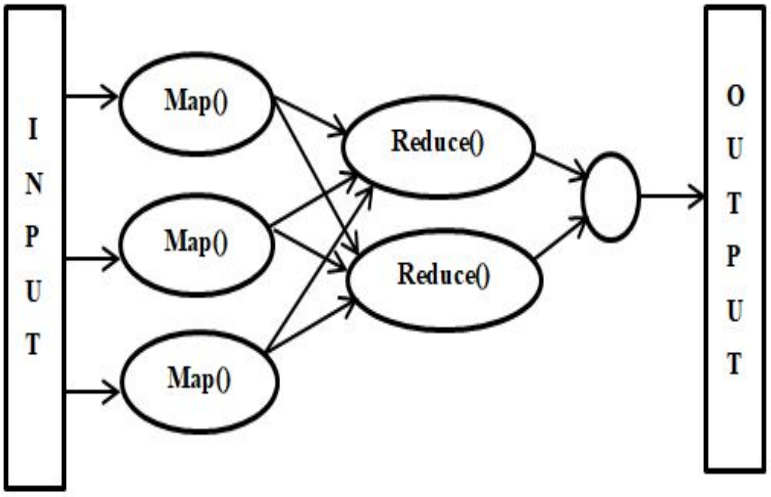



Fig 2: Hadoop Map Reduce

## 3. Implementation

In this paper, we propose two new algorithms, named 'Trust_OSN" and "Trust_MR". The algorithm Trust_OSN is used to divide the given social network graph into multiple sub graphs. The second algorithm Trust_MR is used to evaluate trust path between the users of each sub graph, obtained from the above step. Using Trust_OSN, the given social network is divided into sub communities based on many factors like relatedness between the users as a primary context. Later using the algorithm Trust_MR, we implement Map and Reduce phases as discussed above using Hadoop framework. In this paper, the online social networks are represented as a graph G with a set of nodes N and a set of edges E. The nodes in the graph considered represent the users in a real-world social network whereas edges can be considered as the relationship between the users. Here, the relationship between any two users doesn't mean that they are very well known to each other in person.

They might be friends of friends and so on. Thus, any exchange or sharing of information between these users is nevertheless treated as trusted. Trust plays a major role in decision making of any two individuals connected in a social network. Trust_OSN algorithm is aimed to divide the given graph G into sub graphs S1,S2,S3..Sn. In this process, the algorithm Trust_OSN has to estimate the proximity of any two given nodes in a network G based on many metrics like common interests, behavioral attributes etc. since all the users belong to same network, the algorithm's work may be a bit tedious to estimate and hence we have to define an algorithm in prior and make it implement on any given social network graph. Moreover, two individuals who are connected to each other need not necessarily to be in the same group. For example, consider an individual named A might be interested in movies, whereas the other user B with whom A is connected via. A social network might be interested in latest technologies. Although A and B are connected and remain as friends in the network, they have completely different areas of interest from each other. On the other side, users who are very much like minded and who share common areas of interests should not necessarily be grouped into same networks.

Based on the above discussion, it is pretty evident that to define and to divide any given social network graph G into different sub graphs S, apart from relatedness; we need to also consider many other factors like the shortest distance between the nodes and their interests in common or like-mindedness. This lies as the major idea behind the implementation of "Trust_OSN" algorithm, which tries to incorporate and involve both the common interest of the nodes and also the distance between them in the given graph. Precisely, any two nodes which are

sharing common interests and whom are in less proximity are divided into a sub graph. Whereas, the nodes having common interests but very distant from each other might be involved in two different sub graphs or sub networks. To implement this, Trust_OSN algorithm uses the weights associated with each node as the primary metric for dividing the given graph into multiple sub graphs.

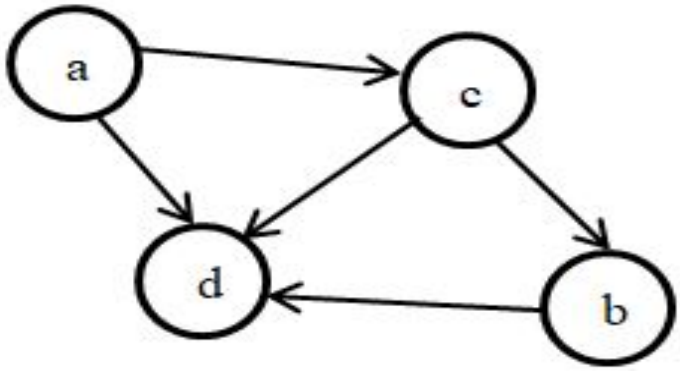



Fig 3: A sample graph G with four nodes five edges

### 3.1 *Trust_OSN*

As discussed earlier, the Trust_OSN algorithm uses both the metrics: relatedness and distance between the users to divide the given social network graph into sub graphs. To derive the proximity of any two given users (nodes n1,n2), the following equation is used in the algorithm Trust_OSN:

$$Prx(n_1, n_2) = £ \frac{d(n_1,n_2)}{l(G)} + (1 - £)(1 - sm(n_1, n_2)) \quad (1)$$

Prx(n1,n2) is the Trust_OSN measure of proximity between any given users denoted by nodes N in the given graph G. A variable quantity £ is used in the equation whose value ranges from any value between 0 and 1.The distance between any two given nodes is measured using the formula d(n1,n2). In a graph G, there may be many shortest paths between the nodes, let l(G) determines the shortest distance, which is the longest amongst all available shortest distances i.e. the maximum value. Let the function sm(n1,n2) measures the interests in common which are pre-determined during the formation of a network graph and are used appropriately. After substituting all the above said values, we then determine the value for the proximity between any two nodes denoted by Prx(n1,n2). This value indicates the closeness of any two given nodes, if the value is lower, then the two nodes n1 and n2 share a very good proximity and if the value is higher, then the given two nodes n1 and n2 are not close to each other.

The distance d(n1,n2) could be calculated using any one of the shortest path algorithms. In this paper, we have used Dijkstra's algorithm. To ensure that the distance between any two nodes in the graph should be a value in between 0 and 1, we have divided the distance between two nodes with the longest path among all the available shortest paths for any given two nodes. This division would act like a normalization factor to derive the value of Prx between zero and one for the nodes n1 and n2. Since the shortest distance between any two nodes in the given graph is calculated using Dijkstra's algorithm, now we have to calculate the similarity of common interests shared by these users. The following equation can be used for this:

$$sm(n_1, n_2) = \sum Max_{\mu(n_1,n_2)} - \log\cap (n_1, n_2) \quad (2)$$

Where μ(n1,n2) is the set of all the common interests between the nodes and ∩(n1,n2) is the whole interest dataset available in the graph. The value of μ should be always considered as maximum among all the available values and the value of ∩ should be a logarithmic one. After substituting the above values in the equation (2), we derive the similarity index of the two given nodes n1 and n2. If the sm value is equal to 1, then it determines that the said two nodes are very close to each other.

The Trust_OSN algorithm makes use of these equations (1) and (2) on the given graph G and splits the graph into multiple sub graphs S1,S2,..Sn. The implementation of this algorithm Trust_OSN can be divided into three phases namely : Selection of Centroids, Calculation of shortest distance between the given two nodes and the Similarity index of those nodes evaluated from the equations (1) and (2) respectively. Trust_OSN is an algorithm which is an iterative one, and it iterates between all the three above phases repeatedly until the membership of the chosen sub graphs become stable. The graph_variation is the difference between any two users like n1 and n2 of a given sub graph S1. This value has to be determined by definite successions of the loop. As shown in the following algorithm step 4, if this value i.e. graph_variation falls below a predefined threshold value,

the Trust_OSN algorithm execution stops and then gives the number of sub graphs detected till then, which is denoted by S.

The algorithm discussed in step 5 evaluates a mean node to represent each sub graph. When the algorithm enters into first iteration, since there are no sub graphs already present, the number of sub graphs initiated at the beginning, that is denoted by K are considered as the mean nodes, took in random. The degree of a node is defined as the number of connections a node possesses is used as criteria of selection at this stage. From each sub graph evaluated, the mean node would be the node which has the highest degree. To be simple, the no of friends a node have would be the degree of the node. To determine a node to be centroid, it should be most accessible to every other available node present in the graph G. Thus, the node which has most connections to all the remaining nodes of the given graph can be considered as the centroid for that particular graph. In the second phase, we have to calculate the shortest distance between all the nodes and also the similarity index between the nodes of the algorithm Trust_OSN. Here, all the distances which are needed are calculated by considering the distance between the nodes and all their mean values of nodes. As already mentioned, since we have many algorithms which can calculate the shortest path between any two nodes, here we are using Dijkstra's algorithm.

**Algorithm 1: *Trust_OSN***

**Input:** A social network with all its users represented as nodes V and their relationship as edges E. I be the set of all common interests of all the nodes.

**Output:** S – Set of sub graphs derived from the actual graph G.

1. procedure Trust_OSN(V,E,I,S)
2. while subgraph_created > threshold_value do
3. for all s € S do
    a. max_value = 0
    b. for all node € s do
        i. if node.deg > max_deg
        then
            1. mean[s] = node
            2. max_deg= node.deg
        ii. end if
    c. end for
4. end for
5. for all v € V do
    a. min_prx_value = ∞
    b. for mean_value € mean_nodes do
    c. sd = d(mean_node, v)
    d. calculate prx value using equation (1)
    e. if prx < minimum_prx then
        i. sid = mean_value.graph
        ii. min_prx_value = prx
    f. end if
    g. end for
    h. calculate sm value using equation (2)
    i. S[sid].addnode_vertex(v)
6. end for
7. end while
8. return S;
9. end procedure

In the following third phase i.e. clustering stage represented in step 7 uses the results which were obtained from the above phase to perform the clustering process based on the likeness of the nodes. The prx score value shall be determined from the equation (1). To derive a sub graph from the given values of graph G, thereby involving each node, the minimum score of all the nodes in a sub graph is evaluated and it is then added to the minimum value obtained among all the sub graphs.

### 3.2 *Trust_MR*

As discussed earlier in the above introduction, Hadoop has two main components: Hadoop Distributed File System (HDFS) and Map Reduce. HDFS is used to store all the given data in a distributed set of nodes across a cluster. There are five major daemons in HDFS named by NameNode, DataNode, JobTracker, TaskTracker and

secondary NameNode. The namenode acts a master and it allocates data to all the slave data nodes. Once the whole data is distributed across all the data nodes, then the other important component of Hadoop named Map Reduce comes into picture and then processes all the above data as shown in the below figure.

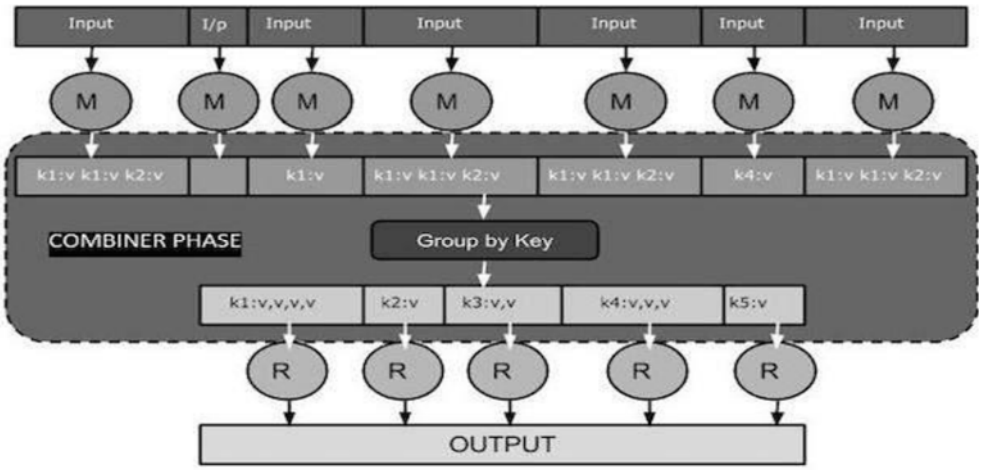



Fig 4: Map and Reduce phases

Trust_OSN algorithm have divided the given social network graph G into sub graphs. Trust_MR algorithm has to identify the trust paths within the sub graphs. The proposed algorithm Trust_MR implements the given job in two stages:

1. Map stage: Computation of all possible intermediate paths between the nodes in each sub graph as <Key,Value> pairs.
2. Reduce Stage: Concatenation of all those derived <Key,Value> pairs and finally deliver a trust path between the users of network.

The map reduce job would be initiated by the NameNode and it sends this source code to all the DataNodes respectively. There are few other daemons named JobTracker and TaskTracker, Jobtracker takes care of tracking the execution process of Trust_MR algorithm and whereas the actual execution is performed by the TaskTracker present at all the nodes in the cluster along with the DataNode (Dr.S.Revathy, Review on Social Network Trust With Respect To Big Data analytics, 2020). The total computation time required to execute Trust_MR algorithm (TMR) is equal to the sum of time required to process the map stage and reduce stage (TM and TR) respectively.

$$T_{MR} = T_M + T_R \qquad (3)$$

**Algorithm 2: *Trust_MR_$T_M$ (Map Phase $T_M$)***

**Input:** The sub graphs derived from Trust_OSN algorithm.
**Output:** Generating <key,value> pairs for all the sub graphs.

1. procedure Trust_MR_$T_M$
2. input
3. G(V,E) :Sub graphs with vertices and edges
4. s : starting vertex
5. e : ending vertex
6. output
7. <key,value> pairs for all the sub graphs
8. begin
9. d(v € V) := ∞
10. O : = □ // Open List
11. S : = □ // Closed List
12. c : =s
13. g(s) := 0
14. d(s) := Euclidean_Distance(s,e) // Calculated using Dijkstra's Algorithm
15. S = S + {s}
16. while O ≠ □ and c.key ≠ e.key do
    a. O := Extended_Vertex(O,G,c);

b. v := Proximed_Vertex(O,S);
c. c := v
17. end while
18. SP := Generate_Path(S);
19. intermediate values stored in mappers at local node;
20. wait until all mappers finish execution;
21. return(key:path_id,val:SP)
22. end procedure

The proposed algorithm Trust_MR_TM consists of three steps. The first step is the "Initialization Phase" described between the (steps 8-15) in the above algorithm 2. The second step is used for navigating the graph and then selecting the most proximity vertices until all the nodes and vertices are traversed. This step is included between the (steps 16-17) in the above algorithm 2. This step is achieved by implementing the functions Extended_Vertex() and Proximed_Vertex() functions. The third step is described between the (steps 18-21) in the above algorithm 2. In this step, the function Generate_Path() is used to extract the paths between all the nodes present in a sub graph. Later, the intermediate results i.e. all the <key,value> pairs are stored locally at the data nodes and then the reducer stage processes them.

**Algorithm 3 : *Trust_MR_$T_R$ (Reduce Phase $T_R$)***

**Input:** The array of Intermediate paths arrived from Map phase
**Output:** Evaluates Trust path

1. procedure Trust_MR_$T_R$
2. input
3. [SP] := All intermediate paths <Key,Value> pairs derived from Map phase of each sub graph
4. output
5. [TP] := The trust path between the nodes of each sub graph derived from G
6. begin
7. TP := empty;
8. for each identified <Key,Value> pair
9. k = <$k_1$,$v_1$> € SP do
10. TP := reduce(TP,k);
11. end
12. wait until all the reduce phase jobs get completed to evaluate a final path in each sub graph
13. return(key:pathid, value:TP)
14. end procedure

The algorithm Trust_MR_TR takes the array of intermediate paths delivered from the map phase described in algorithm 2 as input and evaluates the trust path by reducing all those derived paths between the nodes and finally return the trust path to the master node.

## 4. System Design

Our proposed system design for evaluating trust path between the users of a social network involves mainly three phases. In the first phase, the real world data that is downloaded from any network datasets are used. In our study, we are making use of "Facebook" data that is downloaded from Stanford University ego network datasets (McAuley). The facebook data that was available in this web site was the one which was collected from the participants as a part of survey using the facebook app. This dataset has around 4039 nodes along with their ego networks.

Once the dataset is successfully downloaded, the system enters into second phase, in this phase, our proposed algorithm "Trust_OSN" is used to divide the given facebook network graph G into several sub network graphs S1, S2, S3, ……Sn. Later, in the third phase, "Trust_MR" algorithm is used to process these sub graphs to evaluate a trust path between the users. The Trust_MR algorithm is divided into two phases named "Map stage" and "Reduce stage".

In the Map stage, our proposed algorithm Trust_MR_TM implements the map code, which processes each sub graph and generates all possible "Key-Value" pairs. These generated <key,value> pairs from the map stage are then subjected to "Reduce stage" in which all <key,value> pairs are processed and reduced to evaluate a trust path in each sub graph generated form the first phase. The system model is diagrammatically represented as follows:

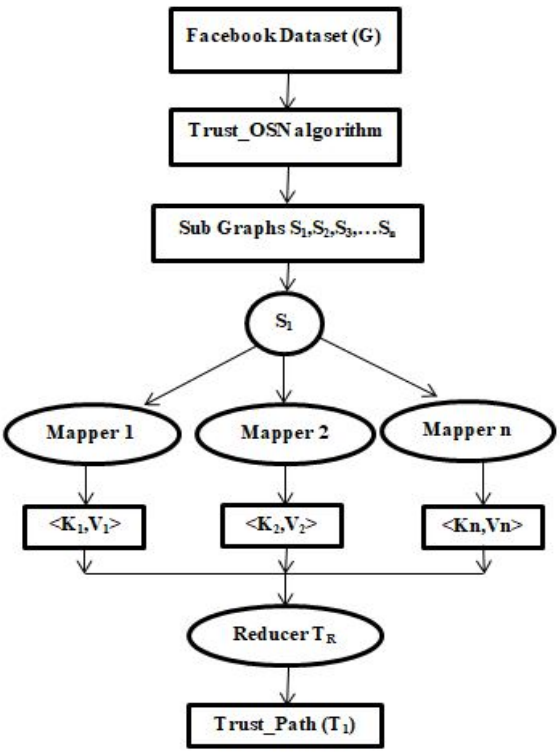



Fig 5: System Model

## 5. Experiment & Result

### 5.1 *Dataset*

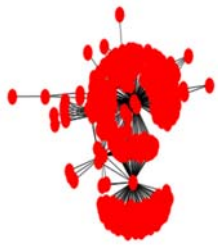
The dataset was downloaded from Stanford University ego network datasets named "Social Circles: Facebook". The details of the dataset are as follows:

| Type | Value(G) |
|---|---|
| Nodes (N) | 4039 |
| Edges (E) | 88234 |
| Average degree | 43.6910 |

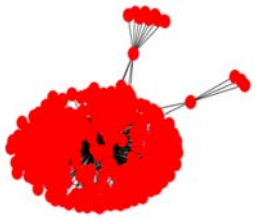
Fig 6: Social Network Graph G

### 5.2 *First Phase (Implementing Trust_OSN algorithm)*

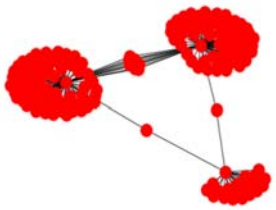
On implementing Trust_OSN algorithm on the facebook dataset, the network graph G is divided into four sub graphs S1, S2, S3 and S4. The details of sub graphs are as follows:



| Type | Value ($S_1$) |
|---|---|
| Nodes (N) | 252 |
| Edges (E) | 392 |
| Average degree | 3.111 |



| Type | Value ($S_2$) |
|---|---|
| Nodes (N) | 359 |
| Edges (E) | 1578 |
| Average degree | 8.7911 |



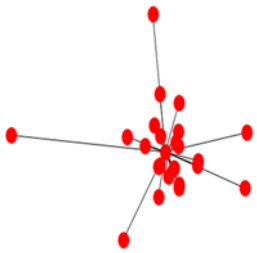
| Type | Value ($S_3$) |
|---|---|
| Nodes (N) | 160 |
| Edges (E) | 168 |
| Average degree | 2.100 |

| Type | Value ($S_4$) |
|---|---|
| Nodes (N) | 77 |
| Edges (E) | 254 |
| Average degree | 6.5974 |

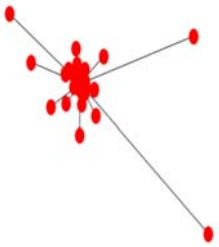
Fig 7: Generation of Sub Graph $S_1$, $S_2$, $S_3$, $S_4$

### 5.3 *Second Phase (Implementing Trust_MR algorithm)*

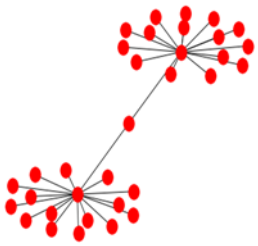
The second phase is divided into two stages “Map stage” and “Reduce stage”. The map stage is implemented using Trust_MR_TM algorithm, which divides each sub graph into multiple <key,value> pairs. Later, in the second stage “Reduce stage”, the algorithm Trust_MR_TR is used to evaluate trust path between all the nodes in each sub graph.



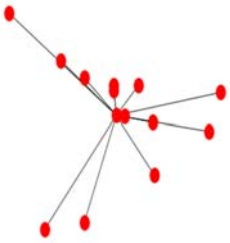
| Type | Value ($S_1$) |
|---|---|
| Nodes (N) | 28 |
| Edges (E) | 27 |
| Average degree | 1.928 |



| Type | Value ($S_2$) |
|---|---|
| Nodes (N) | 34 |
| Edges (E) | 23 |
| Average degree | 1.8379 |



| Type | Value ($S_3$) |
|---|---|
| Nodes (N) | 33 |
| Edges (E) | 32 |
| Average degree | 1.939 |



| Type | Value ($S_4$) |
|---|---|
| Nodes (N) | 14 |
| Edges (E) | 13 |
| Average degree | 1.8571 |

Fig 8: Generation of Trust Path $S_1$, $S_2$, $S_3$, $S_4$

## 6. Conclusion & Future Work

Evaluation of trust path in any given online social network is the most important aspect of every individual's day to day life. Since all the users are not physically connected in social network, deriving trust path among them is very much essential. In this paper, we have used "Trust_OSN" algorithm to divide the given network graph G into several sub graphs. Thus generated sub graphs were subjected to map reduce algorithms Trust_MR_TM and Trust_MR_TR. In this experiment, the whole study has been performed under "Pseudo-Mode" installation of Hadoop version 3.2.1, in which all the daemon nodes named "NameNode, DataNode, JobTracker, TaskTracker and Secondary Namenode" were executed on a single node i.e. on a single machine. In the future work, we are planning to evaluate the optimal trust path between all the users of a given online social network, derived from a large dataset and also implement the hadoop in a "Fully- Distributed" mode. In fully distributed mode of hadoop, multiple nodes have to be used to process the given task on several machines and all the daemons discussed earlier would run on different nodes located at different places.

## References

[1] A. Satish Kumar, Dr. S. Revathy, "Review on Social Network Trust With Respect To Big Data Analytics", Proceedings of the Fourth International Conference on Trends in Electronics and Informatics (ICOEI 2020) IEEE Xplore, Part Number: CFP20J32-ART; ISBN: 978-1-7281-5518-0.
[2] Debadatta Naik, Ranjan Kumar Behera, Dharavath Ramesh, Santanu Kumar Rath, "Map-Reduce-Based Centrality Detection in Social Networks: An Algorithmic Approach", Arabian Journal for Science and Engineering, Springer, June 2020.
[3] Guanfeng Liu, Lei Zhao, Kai Zheng, An Liu, Jiajie Xu, Zhixu Li and Athman Bouguettaya, "An Efficient Method to Find the Optimal Social Trust Path in Contextual Social Graphs", Springer International Publishing Switzerland 2015.
[4] Guanfeng Liu, Yan Wang, Mehmet Ali Orgun, Ee-Peng Lim, "Finding The Optimal Social Trust Path For The Selection Of Trustworthy Service Providers In Complex Social Networks", IEEE Transactions On Services Computing, 2013.
[5] Guohao Sun, Guanfeng Liu, Lei Zhao, Jiajie Xu, An Liu, and Xiaofang Zhou, "A Social Trust Path Recommendation System in Contextual Online Social Networks", Springer International Publishing Switzerland 2014.
[6] Haysam Selim and Justin Zhan, "Towards shortest path identification on large networks", Journal of Big Data, DOI 10.1186/s40537-016-0042-7, 2016.
[7] Imane Belkhadir, Elamine Didi Omar, Jaouad Boumhidi, "An intelligent recommender system using social trust path for recommendations in web-based social networks", Second International Conference on Intelligent Computing in Data Sciences, ICDS, Elsevier 2018.
[8] Maoguo Gong, Guanjun Li, Zhao Wang, Lijia Ma, Dayong Tian, "An efficient shortest path approach for social networks based on community Structure", ScienceDirect, CAAI Transactions on Intelligence Technology, 2016.
[9] Nikolaos Volakis, "Trust in Online Social Networks", thesis report, School of Informatics, University of Edinburgh, 2011.
[10] Nurul Naqqiah Binti Salman, "Achieving Trusted Data In Big Data By Using Social Network Platform", Thesis Report, Universiti Sultan Zainal Abidin, 2017.
[11] Ranjan Kumar Behera, Abhishek Sai Sukla, Sambit Mahapatra, Santanu Ku. Rath, Bibhudatta Sahoo, Swapan Bhattacharya, "Map-Reduce based Link Prediction for Large Scale Social Network", DOI reference number: 10.18293/SEKE2017-100.
[12] Sakshi Chauhan, Anandita Singh Thakur, Madhusudan, "Analysis of social networking data using Map Reduce and Hadoop", International Journal of Science and Applied Information Technology, Volume 5, No.3, May - June 2016.
[13] Simon Fong, Zhuang Yan, "Quantitative Analysis of Trust Factors on Social Network using Data Mining Approach", Conference Paper, ResearchGate, December 2012.
[14] Soumya, T. R., and S. Revathy. "Survey on threats in online social media" In 2018 International Conference on Communication and Signal Processing (ICCSP), pp. 0077-0081. IEEE, 2018.
[15] Sushant Khopkar, Rakesh Nagi, A.G. Nikolaev, "An Efficient Map-Reduce Algorithm for the Incremental Computation of All Pairs Shortest Paths in Social Networks", Conference Paper, ResearchGate, August 2012.
[16] Vikas Chauhan, Anupam Shukla, "Sentimental Analysis of Social Networks using MapReduce and Big Data Technologies", IJCSN International Journal of Computer Science and Network, Volume 6, Issue 2, April 2017.
[17] Wenjun Jiang, Guojun Wanga, Jie Wub, "Generating trusted graphs for trust evaluation in online social networks", Future Generation Computer Systems, Elsevier, 2012.

[18] Wilfried Yves Hamilton Adoni, Tarik Nahhal, Brahim Aghezzaf and Abdeltif Elbyed, "The MapReduce-based approach to improve the shortest path computation in large-scale road networks: the case of A* Algorithm", Journal of Big Data, Springer, 2018.
[19] Zaher Al Aghbari1, Mohammed Bahutair and Ibrahim Kamel, "GeoSimMR: A MapReduce Algorithm for Detecting Communities based on Distance and Interest in Social Networks", Data Science Journal, 18:13, pp. 1–20, April 2019.
[20] Fugkeaw, S., & Sato, H. (2015, November). "Privacy-preserving access control model for big data cloud." In Computer Science and Engineering Conference (ICSEC), 2015 International (pp. 1-6). IEEE
[21] Li, Peng, et al. (2016, September) "Privacy-Preserving Access to Big Data in the Cloud" IEEE Cloud Computing 3.5, pp.34-42.
[22] Shankarwar, Mahesh U., and Ambika V. Pawar. "Security and Privacy in Cloud Computing: A Survey." In FICTA (2), pp. 1-11. 2014.
[23] Shrivastava, Gulshan, et al. "New Age Analytics: Transforming the Internet through Machine Learning, IoT, and Trust Modeling." (2020).
[24] Raj, Ebin Deni, and LD Dhinesh Babu. "An enhanced trust prediction strategy for online social networks using probabilistic reputation features." Neurocomputing 219 (2017): 412-421.
[25] Praveena, A., and S. Smys. "Ensuring data security in cloud based social networks." In 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA), vol. 2, pp. 289-295. IEEE, 2017.
[26] Shakya, S. (2019). An Efficient Security Framework for Data Migration In A Cloud Computing Environment. Journal of Artificial Intelligence, 1(01), 45-53.

## Authors Profile

A Satish Kumar (Satish Kumar Athmakuri) obtained his Master's degree in Computer Science and Engineering from Sathyabama University, Chennai, India in the year 2009. He is currently pursuing the Ph.D. degree with the School of Computing, Sathyabama Institute of Science and Technology, Chennai, India. He is currently working as Assistant Professor in the Department of Computer Science & Engineering, VRS & YRN College of Engineering & Technology, Chirala, India. His current research interests include Big Data Analytics, Machine Learning, Operating Systems and Cyber Security.

Dr. S. Revathy (Subramanion Revathy) received Ph.D. in Computer Science and Engineering from Sathyabama University, Chennai, India in the year 2015. She is presently working as an Associate Professor in the Department of Information Technology, Sathyabama Institute of Science and Technology, Chennai India. Her research interest includes Machine Learning, Data Analytics and Big Data. She has published over twenty papers in refereed journals.