Developing an Efficient Spectral Clustering Algorithm on Large Scale Graphs in Spark

Ahmed I.Taloba*, Marwan R. Riad[x] and Taysir Hassan A. Soliman**
*Associate Professor
Email: Taloba@aun.edu.eg
[x]Teaching Assistant
Email: marwan-radwan-is@aun.edu.eg
**Professor
Email: taysser.soliman@fci.au.edu.eg
*,[x],** Information System Department, Faculty Computer and Information, Assiut University, Egypt

***Abstract*—Recently, most of the data can be represented by graph structures, such as social media, Protein-Protein Interaction, transportation system, systems biology,..., etc. Many researches have been achieved to cluster very large graphs but more efficient algorithms are required since such a process takes a long time and requires more memory. In this paper, we propose an Efficient Spectral Clustering Algorithm on Large Scale Graphs in Spark (ESCALG), using map reduce function and shuffling phases in Dijkstra's algorithm. In addition, ESCALG depends mainly on a sparse matrix as a data structure, which less time in execution. Then, GraphX is applied to deal with graph data processing and in GraphX used Pregel in computing shortest path . To test the performance of ESCALG, it is compared with Large-Scale Spectral Clustering on Graphs and Standard Spectral Clustering Algorithms using seven datasets, where ESCALG proved high efciency in terms of memory and time performance.**

keywords: Spectral Clustering , Apache Spark, Large scale Graph Clustering

I. Introduction

One of the most current vital research problems is to develop and implement efficient and scalable algorithms for mining large graph data. A graph can be represented by $G = (V, E)$ where V is a vertex, node or object. E is an edge, a relation or link. $|V|$ is the number of nodes in a graph and $|E|$ is a number of edges in graph. A graph can be represented as a directed or undirected, weighted and unweighted graph. In this paper, an undirected weighted graph is used. A graph can be represented by two types: an adjacent matrix or a sparse matrix. The first type of representation is an adjacent matrix W as NXN matrix, where$W_{i,j} = 1$ when there is link between node i and node j, otherwise$W_{i,j} = 0$. The second type of representation is a sparse matrix, which saves the nonzero only object represented source node, destination node ,and weight. This type of matrix will reduce the memory taken for execution.

Clustering assigns each object into groups, which depends on similarity. Each cluster must contain closely related objects. Graph clustering partitions the original graph to a set of densely subgraphs, where in each cluster there is a densely connected graph. In each cluster, nodes are more connected to each other rather than connecting another node in another cluster.

Spectral Clustering [1] techniques perform dimensionality reduction by using the eigenvalues of the similarity matrix of the data before clustering in fewer dimensions. An example of a real-world application of spectral clustering is social networks. Spectral Clustering depends on the eigenvectors, where eigenvalues and can be derived from the affinity matrix is an NXN matrix. N is represented as the number of persons in a social network. These eigenvectors motivate to merge all networks and convert it in a low-dimensional subspace, wherein a simple clustering method (such as k-means) can then be used to do the final partitioning.

Applying spectral clustering algorithms on large scale graphs data requires more attention. One of the recent frameworks that can be used to make spectral clustering algorithms more efficient on large scale graph data is Spark. Spark [2] is an open source distributed computing platform using Resilient Distributed Datasets (RDD). RDD is a fundamental data structures, allowing computation on datasets to be distributed among different nodes in the cluster to be memory efficient. Data storage usage must be reduced, which transforms time taken from storage to memory in order to perform different kinds of operations such as clustering, shortest path,...,etc and return the data to storage again. This process consumes very large time. RDD uses memory of all clusters to perform the operation with costless trip to storage.

The advantages of using spark for hadoop [3] include:

- When data are lost on any worker, spark re-execute automatically the data again on another worker.
- Support the map and reduce function.
- Use in-memory data storage parallel called Resilient Distributed Datasets (RDD).
- User can partition the data.
- RDD can co-schedule tasks to avoid data movement.

To use Spark framework for graph data, GraphX [3] has been developed as a distributed graph processing framework

978-1-5386-0821-0/17/$31.00 ©2017 IEEE

that was built on top of Spark. Fig. 1 illustrates the architecture of Spark and where GraphX is located in this architecture. It extends RDD in Spark by introducing the Resilient Distributed Property Graph, on top of it. GraphX has many Built in fundamental operations on graph and contains a number of graph algorithms, such as (Triangle Counting, Connected Components and Page Rank algorithms) as well as some fundamental operations on graphs. In addition, It has an optimized variant of Pregel API, which is a popular graph processing architecture developed to solve some of the problem faced in large graph processing. Pregel [3], [4] is a bulk synchronous message passing graph-parallel abstraction in which all vertex programs run concurrently in a sequence of super-steps. Using super-steps, vertices receive all message sent from its neighbors and calculate the minimum value short paths before starting the next super-steps.

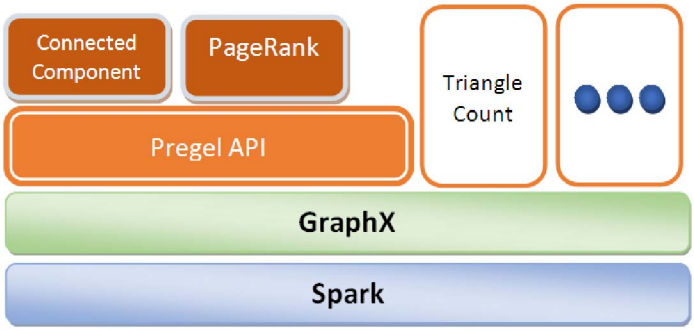

Figure 1: Spark Layers' Architecture

In this paper, an efficient algorithm (ESCALG) is developed in order to cluster very large graphs in a high processing time, utilizing reduced usage of memory. The main contributions of ESCALG are:

- Using map reduce function in Spark to reduce time and memory when used memory not storage.
- Utilizing Sparse Matrix to reduce usage memory.

This paper is organized as follows: section two summarizes related work. Section three describes the main idea of Spectral Clustering algorithm. Section four concentrates on the proposed framework preprocessing and ESCALG algorithm. Section five explains the comparisons among ESCG, SC, and ESCALG algorithms. Finally, section six explains concluding remarks and gives more insights into proposed future work.

II. Related Work

Most of research on Spectral clustering can be dependent on eigenvalues and eigenvectors and decomposition of the graph for example,
Y. Ng, Jordan and Weiss [5] calculated the Eigen-decomposition of the laplacian matrix,which is the biggest problem because it takes time $O(N^3)$. When using graph structure, we need to input adjacent matrix NxN taking time more than $O(N^2)$. Another problem here the need to improve the time and memory because it is difficult to use it with large-scale graph data.

Fowlkes, Belongie , Chung and Malik [6] provided a technique to partition image and video segmentation based on Nystrom extension, which reduce the number of groups in an image compare to the number of pixels and apply k-means to clustering.

Chen, et al. [7] proposed this paper to reduce time and reduce the lost of accuracy data, and accelerate the spectral clustering on large-scale application by applying sequentially depress the size of Laplacian matrix in each iteration step applied in spectral clustering.

Shinnou, and Sasaki [8] reduced similarity matrix size, applying clustering algorithm k-means to partition data and selecting some of data closest to central and represented this data in represented data to reduce number of data point these data called committee, formation the similarity matrix from these data. When used the committee, it will be reduce the similarity matrix size and finally applying spectral clustering on the reduced data.

Yan, Huang and Jordan [9] introduced a small number of representative points by to perform a distortion minimizing local transformation on the data by with two way first k-means used as a local data reduction, and the second way the Random Projection tree (RP tree) used in first step used affinity graph to collapse neighbors of the data point into a represented data point.The second step was running spectral clustering on the represented data point from last step and final step assign each original point to the cluster based on the represented point of it.

Chen, and Cai [10] provided scalable spectral clustering method called Landmark-based Spectral Clustering (LSC). Here they selected a sample from data point as the landmark and represented the remaining data point as the linear combinations of these landmark and encoding all data point to codebook. Acceleration could be achieved using the new representation.

Liu, Wang, Danilevsky and Han [11] generated supernodes to decrease the size of input graph and applying the spectral clustering on the new graph, which depress by supernode to decrease the time and input to the algorithm adjacent matrix but here the adjacent matrix take large size of memory when data be large.

Other research papers focus on multiple machine; for example,
Chen, et al. [12] developed an algorithm in this paper called Parallel Spectral Clustering (PSC), which distributed all data instance on number of distributed machine nodes. On each

node, PSC calculated the similarity between the original data and local data on node, and used minimal I/O disk. PSC stored the eigenvector matrix on distributed node to reduce memory used. Finally, PSC improved speedup on large data set when used parallel eigensolver and k-mean together.

Recent research used GraphX in Spark to parallel data and multiplication.
Xin ,Gonzalez, Franklin [3] illustrated the use and improved of GraphX when combining between data-parallel systems and graph-parallel systems.They provided programming abstraction called Resident Distributed Graphs (RDGs) and implemented Pregel in few lines. GraphXs can minimize movement of data during graph computation by the internal data representation using a vertex cut partitioning scheme.

Zadeh , et al. [13] added matrix operations such as addition or multiplication there is problem when using with large scale matrices sparks distributed linear algebra and optimization a library,called LINALG. LINALG has many advantages; one of them is working on row, column, entry, or block sparsity to operate and store matrices local or distributed. In addition, LINALG support Scala, java ,and python API. Furthemore, LINALG produce a sparse matrix, which provide an efficient storage in Compressed Column Storage (CCS) that save the nonzero element the index of row and index of column and the value.

III. Spectral Clustering Algorithm

Given data points, let A be an symmetric matrix, where the similarity measures between the i and j points can be represented as the entry (i, j). The goal of spectral clustering is to partition the points into disjoint clusters, and different spectral clustering methods formulate the partitions in different ways such as Normalized cut.

Spectral Clustering [1] consists of three stages: preprocessing data, spectral representation and clustering (such as k-means) illustrate algorithm here:

- Preprocessing data:
  The objective of data preprocessing is very important because effect in result choosing similarities function. The Most cases use Gaussian Kernal or cosine similarities.
  There are more problem in data such as outlier ,missing value,..., etc we must repair it. Finally represented the dataset by similarity matrix.
- Spectral Representation:
  - Extract Laplacian matrix.
    Laplacian matrices can be calculated by more than type example,

    *

    $$L = D - W \tag{1}$$

    , where W is adjacent matrix and D is Diagonal Matrices calculate by

    $$d_i = \sum_{j=1}^{n} w_{ij} \tag{2}$$

    *

    $$L_{sy} = D^{-1/2} L D^{-1/2} \tag{3}$$

    where L_{sy} is Laplacian symmetric
  - Compute eigenvectors and eigenvalues of the Laplacian matrix.
  - Assign each point in dataset to lower-dimensional representation based on one or more eigenvectors.
- Clustering:
  The objective of this is assign points to closest classes. Applying simple algorithms such as k-means or other than k-means can be used in the this stage such as simple linkage, k-median, klines, elongated k-means.

The most famous algorithm in spectral clustering algorithm [5] ,illustrated in Algorithm 1.

Algorithm 1 Standard Spectral

1: Input : Given a data set where $S = \{S_1, S_2, \ldots, S_N\}$.
2: Output: clustered data set
3: Calculate the affinity matrix $A_{ij} = exp(-||s_i - s_j||^2/2\sigma^2)$, if $i \neq j$ and $A_{ii} = 0$ where σ^2 is the scaling parameter
4: Define D to be the diagonal matrix whose (i,i)-element is the sum of $A's$ i-th row, and construct the matrix $L = D^{-1/2} A D^{-1/2}$
5: Find k largest eigenvectors of L and form the matrix $X = \{x_1, x_2, \ldots, x_k\}$
6: Form the matrix Y from X by normalizing each of $X's$ rows to have unit length, $Y_{ij} = X_{ij}/(\sum_j X_{ij}^2)^{1/2}$
7: Treating each row of Y as a point, cluster them into k clusters via k-means or any other algorithm
8: Assign the original point s_i to cluster j if and only if row i of the matrix Y was assigned to cluster j

IV. Proposed Algorithm

In this paper, an Efficient Spectral Clustering Algorithm on Large scale Graph in Spark (ESCALG) is proposed. The principle idea of ESCALG is to reduce the size of the original graph by choose random nodes represented the group. IF we have n nodes represent the new graph by (Represented Node or Super Node) SN. SN must be equal or less than the number of original nodes and connected each node in the original graph to closest data.

A. Data Preprocessing

First step in our algorithm is preprocessing of data to remove the outliers and reorder the id of each node to be in continuous ordering. Example, if we have nodes with id $\{0, 1, 2, 4, 6\}$ must be reordered to $\{0, 1, 2, 3, 4\}$. Old node id 4 is changed to 3 in the new node id and old node id 6 is changed to 4 in the new node id and then make link between source and destination not repeated. An example in and the result of preprocessing are illustrated in Fig. 2 and Fig. 3 respectively.

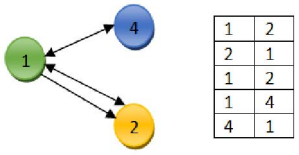

Figure 2: An Example of data not clean.

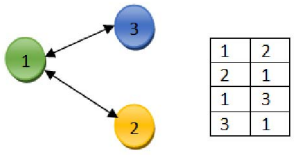

Figure 3: After Preprocessing

The steps of the preprocessing algorithm are as follows in Algorithm 2:

Algorithm 2 Data Preprocessing

```
Input: Edge File before preprocessing.
Output: Edge File after preprocessing
Function Conv(int k, int m, int [][] l)
For i=0 to l.length - 1
    For j=0 to 2
      if[i][j] == m)
       l[i][j]=k
      End if
    End for
End for
End Function
# edge file col1 : source, col2 : destination
Read unique edge file in array 2 Dim =¿ a
get the unique number in edge in array =¿ h
for p=0 to b.length-1
for p=0 to b.length-1
    if p!=h[p]
     Conv(p,h[p],a)
    end if
end for
```

B. Efficient Spectral Clustering Algorithm On Large Scale Graphs

First, Given the initial graph with n nodes ,Then generated a set of random d supernodes (SN) where number of SN must be less than number of nodes in the initial graph.second develop a simple algorithm to apply initial clustering by applying Dijkstras algorithm to compute the shortest path using pregel. In pregel, API messages are computed in parallel as a function of the edge triplet and the message computation has access to both the source and destination vertex attributes. Vertices's that do not receive a message are skipped within a super step. The Pregel operators terminate iteration and return the final graph when there are no messages remaining, after we will apply Dijkstra's on random SN, partition the data to d disjoint subset, assign each node to closest represented supernode, generate the matrix between the original node and SN if there is relation between nodes and closest SN set to 1 otherwise set to 0, in $R \in R_{dxn}$. At this point, SP describes the spares matrix of G and R describes relation between nodes in G and SN. We therefore absorb SP into R to finish transforming G into the bipartite graph. Denote the edge weight matrix $W1 \in R_{dxn}$ of the bipartite graph is as follows:

$$W1 = R \quad . \quad SP \tag{4}$$

after that we compute Z as:

$$Z = D_2^{-1/2}. \quad W1 \quad . \quad D_1^{-1/2} \tag{5}$$

where D_1 and D_2 are Diagonal Matrix of column and row sums of $W1$ in equation (2).

then extract the largest k eigenvalues and eigenvectors from ZZ^T. we get the right singular values as all nonnegative real numbers and Y is an unitary matrix, we can get X as follows:

$$X^T = \sum^{-1} Y^T Z \tag{6}$$

As the final step, use k-means cluster to pick the "top" k column vectors of normalized X. All multiplication applied in this algorithm are in parallel.
All the steps of the ESCALG algorithm are explained in Algorithm 3.

V. Experiment and Result

To evaluate the performance of ESCALG algorithm, experiments are compiled using Apache Spark on an Intel R Core(TM) i7-3770 3.4 GHz processor, and memory size of 32 GB. In the following subsection, Datasets used will be explained in details.

A. Data sets

Real world data sets used, as illustrated in Table. I,which are as follows :

- **DBLP** [14] : is computer science bibliography provides a comprehensive list of research papers in computer

Algorithm 3 Efficient Spectral Clustering Algorithm on Large scale Graph in Spark

1: Input: SP: Sparse Matrix.
2: k: Number of Cluster.
3: d: Number of Supernode.
4: Output : Clustering Large Graph.
5: First select random supernodes from nodes in graph and run parallel Dijkstra's algorithm to compute shortest path between nodes to each closest supernodes.
6: Partition node to disjoint supernodes according to Dijkstra's algorithm.
7: Create matrix between all nodes and supernodes $R \in R_{dxn}$.
8: Compute matrix W_1 from $W_1 = R \quad x \quad SP$ by parallel multiplication to transformed bipartite graph as in Eq 4.
9: Compute $Z = D_2^{-1/2} W_1 D_1^{-1/2}$ apples the parallel multiplication where D_1 and D_2 are Diagonal matrices of column and row sums of W_1 in Eq 5.
10: Extract the largest k eigenvalues and eigenvectors from ZZ^T.
11: Generate right singular vectors X of Z as in Eq 6.
12: From the matrix $U = D_1^{-1/2} X$.
13: apply parallel k-means algorithm on nodes and each row represented.

science. We construct a co-authorship network where two authors are connected if they publish at least one paper together.

- **Youtube** [15] : is a video-sharing web site that includes a social network. In the Youtube social network, users form friendship each other and users can create groups which other users can join. We consider such user-defined groups as ground-truth communities
- **Amazon** [16] : Network was collected by crawling Amazon website. It is based on Customers Who Bought This Item Also Bought feature of the Amazon website. If a product i is frequently co-purchased with product j, the graph contains an undirected edge from i to j.
- **Amazon 2003** [17] : The data was collected in June 01 2003
- **twitter** [18] : This dataset consists of 'circles' (or 'lists') from Twitter. Twitter data was crawled from public sources. The dataset includes node features (profiles), circles, and ego networks.
- **Cond Mat 2003, Cond Mat 2005** [19] : Collaboration network of scientists posting preprints on the condensed matter archive at www.arxiv.org, 1995-1999, as compiled by M. Newman.

B. Algorithms for Comparisons

ESCALG algorithm will prove that it is more efficiency compare to another method of clustering on graph data

- **Efficient Spectral Clustering on Graphs (ESCG):** [11] the method replaces the original node by super nodes

Table I: Data Set

Data set	Nodes	Edges
DBLP [14]	317080	1049866
Youtube [15]	1134890	2987624
Amazon [16]	334863	925872
Amazon 2003 [17]	403394	3387388
Twitter [18]	81306	1768149
Cond Mat 2003 [19]	31163	120029
Cond Mat 2005 [19]	40421	175692

the parameter number of supernode ,number of cluster, adjacent matrix represented graph ,try two Experiment: First Experiment no. of supernodes are 30 and no. of cluster are 10,second experiment no. of supernode are three and no. of cluster are two

- **Standard Spectral clustering (SC):** [5] implemented the algorithm by Asad Ali ,on matlab in [Ng et al., 2001] and the input adjacent matrix.

C. Experimental Results

Table II shows the performance when run on number of supernode three and number of cluster is two when run on Cond Mat 2003 and Cond Mat 2005 the performance of the ESCG algorithm is better than ESCALG algorithm and SC algorithm show in Fig. 4a. However the performance of ESCALG algorithm is better than SC algorithm and ESCG algorithm, when apply on large data sets DBLP, Youtube, Amazon, Amazon 2003, Twitter as shown in Fig. 4b.

Table III also proves the performance when run number of supernode 30 and number of cluster is 10 when applying on Cond Mat 2003 and Cond Mat 2005 the performance of the ESCG algorithm is better than ESCALG algorithm and SC algorithm show in Fig. 4c. However, the performance of ESCALG algorithm is better than SC algorithm and ESCG algorithm, when apply on large data sets DBLP, Youtube, Amazon, Amazon 2003, Twitter, as shown in Fig. 4d.
In the previous result prove that using sparse matrix ,in addition Spark, Graphx and Pregel improve the memory and time used in some datasets.

VI. Conclusion and Future work

In summary, this paper gave a general and systematic study of developing an efficient spectral clustering on large scale graph, most of previous research are not efficient in time and memory. This paper used sparse matrix instead of using adjacent matrix. In addition to used map reduce function and RDD to partition data and operate function on data in parallel.

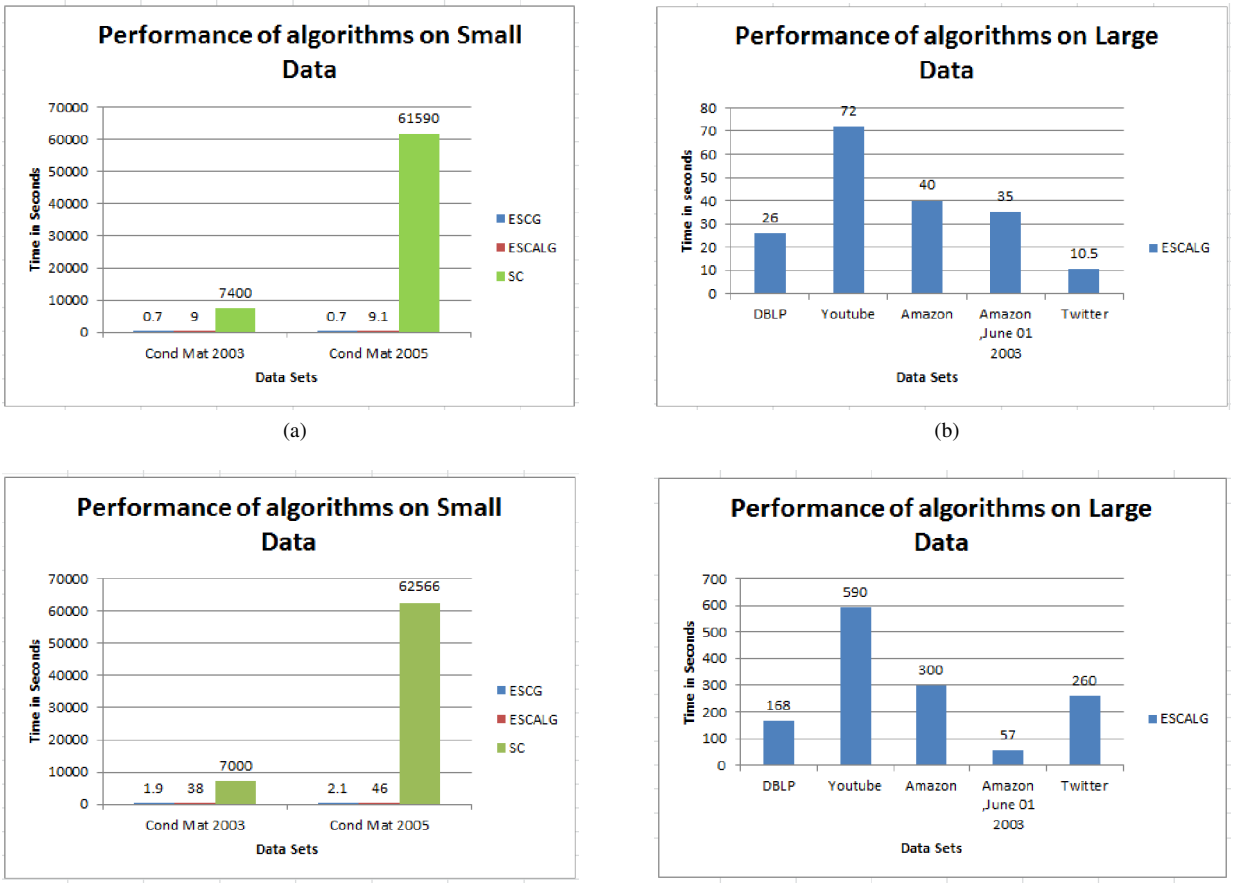

Figure 4: Results when apply ESCG,SC,ESCALG algorithm on seven data sets in (a) and (c) on small data Sets and (b) and (d) on Large Data Sets. run algorithm in (a) and (b) with number of super node are three and number of cluster are two.and run algorithm in (c) and (d) with number of supernode are 30 and number of cluster are 10 .

Table II: Running time in second on data set when no. of Supernode are three and no. of cluster are two

Data set	SC	ESCG	ESCALG
DBLP	-	-	26
Youtube	-	-	72
Amazon	-	-	40
Amazon 2003	-	-	10.5
Twitter	-	-	35
Cond Mat 2003	7400	0.7	9
Cond Mat 2005	61590	0.7	9.1

Table III: Running time in second on data set when no. of Supernode are 30 and no. of cluster are 10

Data set	SC	ESCG	ESCALG
DBLP	-	-	168
Youtube	-	-	590
Amazon	-	-	300
Amazon 2003	-	-	57
Twitter	-	-	260
Cond Mat 2003	7000	1.9	38
Cond Mat 2005	62566	2.1	46

Applying all matrix multiplication in parallel has reduce size and time used, Finally, the system effectively cluster large scale graphs in low memory.

In future work, can be improve in time processing by reduce the data size with lost-less in data. Applying ESCALG on directed graph. Finally can be improve the clustering method

with improving k-means or select another cluster method.

REFERENCES

[1] Hamad D, Biela P, "Introduction to spectral clustering". In: 3rd International conference on information and communication technologies: from theory to applications, 2008, 15, pp 490495.

[2] Xiangrui Meng , Joseph Bradley , Burak Yavuz , Evan Sparks , Shivaram Venkataraman , Davies Liu , Jeremy Freeman , DB Tsai , Manish Amde , Sean Owen , Doris Xin , Reynold Xin , Michael J. Franklin , Reza Zadeh , Matei Zaharia , Ameet Talwalkar, "MLlib: machine learning in apache spark", The Journal of Machine Learning Research,January 2016, v.17 n.1, p.1235-1241.

[3] Reynold S. Xin , Joseph E. Gonzalez , Michael J. Franklin , Ion Stoica, "GraphX: a resilient distributed graph system on Spark", First International Workshop on Graph Data Management Experiences and Systems, June 23-23, 2013, p.1-6, New York, New York

[4] Grzegorz Malewicz , Matthew H. Austern , Aart J.C Bik , James C. Dehnert , Ilan Horn , Naty Leiser , Grzegorz Czajkowski,"Pregel: a system for large-scale graph processing", Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, June 06-10, 2010, Indianapolis, Indiana, USA

[5] A.Y. Ng, M.I. Jordan, Y. Weiss, "On spectral clustering: analysis and an algorithm", Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge, MA, 2002

[6] Fowlkes, C., Belongie, S., Chung, F., and Malik, J. ,"Spectral grouping using the Nystrom method". IEEE Transactions on Pattern Analysis and Machine Intelligence,2004 , 26, pp 214225.

[7] Chen, B.; Gao, B.; Liu, T.-Y.; Chen, Y.-F.; and Ma, W.-Y. . "Fast spectral clustering of data using sequential matrix compression". In Proceedings of the 17th European Conference on Machine Learning (ECML06), 2006.

[8] Shinnou, H., and Sasaki, M. ,"Spectral clustering for a large data set by reducing the similarity matrix size". In Proceedings of the Sixth International Language Resources and Evaluation (LREC08), 2008.

[9] Yan, D.; Huang, L.; and Jordan, M. I. ,"Fast approximate spectral clustering". In Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD09) ,2009.

[10] Chen, X., and Cai, D. ,"Large scale spectral clustering with landmark-based representation". In Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI, San Francisco, California, USA, August 7-11, 2011

[11] Liu, J.; Wang, C.; Danilevsky, M.; and Han, J. ,"Large-scale spectral clustering on graphs". In IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013.

[12] Chen W-Y, Song Y, Bai H, Lin C-J, Chang EY , "Parallel spectral clustering in distributed systems". IEEE Trans Pattern Anal Mach Intell 99,2010.

[13] R. B. Zadeh, X. Meng, B. Yavuz, A. Staple, L. Pu, S. Venkataraman, E. Sparks, A. Ulanov, and M. Zaharia. "Matrix computations and optimization in apache spark". Proceedings of the Conference on Knowledge Discovery and Data Mining 2016 (KDD 2016), 2016.

[14] https://snap.stanford.edu/data/com-DBLP.html, visited in : 1-2-2017

[15] https://snap.stanford.edu/data/com-Youtube.html, visited in : 1-2-2017

[16] https://snap.stanford.edu/data/com-Amazon.html, visited in : 1-2-2017

[17] https://snap.stanford.edu/data/amazon0601.html, visited in : 1-2-2017

[18] https://snap.stanford.edu/data/egonets-Twitter.html, visited in : 1-2-2017

[19] http://www-personal.umich.edu/ mejn/netdata/, visited in : 1-3-2017