A Cloud System for Community Extraction from Super-Large Scale Social Networks

Zhiang Wu, Haicheng Tao, Youquan Wang, Changjian Fang, and Jie Cao*

Jiangsu Provincial Key Laboratory of E-Business,
Nanjing University of Finance and Economics, Nanjing, China
{zawuster,haicheng.tao,youq.wang}@gmail.com,
{Changjian.Fang,Jie.Cao}@njue.edu.cn

Abstract. This demo showcase the Community Extraction Cloud (CEC) system. The key idea is to drop weak-tie nodes by efficiently extracting core nodes based on the novel concept of asymptotically equivalent structures (AES) and parallel AES mining algorithm. Meanwhile, to facilitate storing and processing of massive networks, several cloud computing technologies including HDFS, Katta, and Hama are seamlessly integrated into CEC system.

1 Introduction

As real-life social networks evolving into super-large scales, community detection becomes increasingly challenging, even though a sea of research efforts have been devoted. Since there exist a large number of weak-tie or overlapping nodes in super-large scale network, the network often cannot be partitioned into several crisp communities. To remedy this, *community extraction* has been proposed [4] to extract tight and meaningful communities with only core nodes from massive social networks. Along this line, this paper showcase a cloud system designed for extracting communities from super-large scale networks (a.k.a. CEC, Community Extraction Cloud). As shown in Fig. 1, the CEC system consists of three layers, i.e., data layer, processing layer, and presentation layer. We proceed to introduce these three layers as follows.

Data layer is responsible for storing and managing network data. Each network is represented as a *market basket transaction* file, where each line corresponds to a node and items in this line are neighbor nodes of that node. Since our target is super-large scale networks, Hadoop Distributed File System (HDFS) is employed, and then a big file is split into many small files. To speed up retrieving a specific node and its neighbors, we utilize Katta to create distributed indexes on each small file.

Processing layer executes the core task of CEC, that is, to extract closely-knit communities from super-large social networks. Our COSCOM (COSine-pattern based COMmunity extraction) framework is equipped on this layer. Generally, COSCOM consists of three main phases: (1) to extract asymptotically equivalent structures (AES) using CoPaMï, which will be elaborated in

X. Lin et al. (Eds.): WISE 2013, Part II, LNCS 8181, pp. 499–502, 2013.

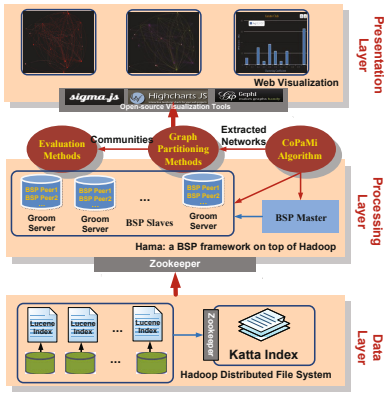

Fig. 1. The system architecture of CEC

Sect. 2; (2) to assemble all the nodes from the extracted AESs and partition them into communities using existing community detection methods such as Kmeans, METIS, Fast Newman, etc; (3) to evaluate the quality of the detected communities using various validation measures. To facilitate the parallel mining, Bulk Synchronous Parallel (BSP) model and one of its implementation framework called Hama are adopted in CEC.

Presentation layer provides user interaction and visualization results according to Internet explorer. The user can upload or select networks to be processed. Two kinds of settings are provided for extracting the core network derived from AESs: the first one for professional users is to set two thresholds τ_G, τ_F , and another one for common users is simply to set the percentage of extracted nodes. Then, before starting partitioning, CEC also provides two kinds of settings: the first one is to set the number of communities C, and another one is to select automatically determine C in which CEC will return several communities with maximal *modularity*. During this process, original network, extracted sub-networks, and communities will be visualized to provide intuitive view for users. Meanwhile, the quality of extracted communities in terms of modularity, and some topological statistics both in original and extracted network such as degree, clustering coefficient, eigenvector centrality, betweenness centrality, etc will be exhibited.

2 Method Details

In this section, we briefly introduce the key idea of CEC, i.e., AES and the parallel AES mining algorithm.

2.1 Asymptotically Equivalent Structures

The nodes in *structural equivalence* must have exactly the same friends, and thus tend to form a very tight community. This idea, however, confront one

problem. That is, structural equivalence is often too restrictive for real-life social networks. To meet this challenge, we first reformulate the concept of structural equivalence. As we know, a node set $S = \{i_1, \cdots, i_{|S|}\}$ is structurally equivalent if $N_1 = \cdots = N_{|S|}$, or equivalently, $r_p = 1$. Let $r_p = |\bigcap_{q=1}^{|S|} N_q|/|N_p|$ characterize the ratio of common friends for node i_p, $1 \leq p \leq |S|$. To relax the concept of structural equivalence, we can set r_p as a measure but lower its threshold from one to a proper level. Moreover, to filter out weak-tie nodes, we should further demand that a node set should have a certain number of common friends. Therefore, we can derive the following two measures:

$$G(S) = \sqrt[|S|]{\prod_{p=1}^{|S|} r_p}, F(S) = \frac{|N_1 \bigcap \cdots \bigcap N_{|S|}|}{n}. \tag{1}$$

Based on G and F, we have the following definition:

Definition 1. *A node set S is an asymptotically equivalent structure if $G(S) \geq \tau_G$ and $F(S) \geq \tau_F$, where $\tau_G, \tau_F \in [0, 1]$ are given thresholds.*

So our task here is to extract all the AES from a large-scale network. The most beautiful part of an AES, however, is that it is a *cosine pattern* in essence. To understand this, let us consider the adjacency matrix of an undirected network (denoted as A), where $A_{pq} = 1$ $(p \neq q)$ if there is an edge between node i_p and node i_q and 0 otherwise. Accordingly, $\sum_{q=1}^{n} A_{pq} = |N_p| = d_p$, $1 \leq p \leq n$. Let $\mathcal{T}_A$ be the transaction data set transformed from A, we now reformulate G and F from a pattern mining perspective as follows. Given a node set S, $|N_1 \bigcap \cdots \bigcap N_{|S|}| = |\{t_p | S \subseteq t_p, 1 \leq p \leq n\}| = \sigma(S)$, where $t_p = \{i_q | A_{pq} = 1, 1 \leq q \leq n\}$ is the pth transaction in $\mathcal{T}_A$, and $\sigma(S)$ is the support count of S in $\mathcal{T}_A$. As a result, we have $F(S) = s(S)$, i.e., the support of S. Moreover, it is easy to show $G(S) = \sigma(S)/\sqrt[|S|]{\prod_{p=1}^{|S|} \sigma(\{i_p\})} = s(S)/\sqrt[|S|]{\prod_{p=1}^{|S|} s(\{i_p\})} = \cos(S)$, i.e., the cosine value of S. To sum up, we have the following proposition:

Proposition 1. *Given the thresholds τ_G and τ_F, to extract all the asymptotically equivalent structures from a network $\mathcal{G}$ is equivalent to mine all the cosine patterns from the corresponding adjacency matrix A, with $\tau_c = \tau_G$ and $\tau_s = \tau_F$.*

2.2 Parallel Cosine Pattern Mining Algorithm

Since mining AESs is equivalent to mining cosine patterns, we present a novel Cosine Pattern Mining (CoPaMï) algorithm which employs a FP-growth-like [2] procedure. However, the key difference between CoPaMï and FP-growth lies in that CoPaMï employs cosine similarity to prune sub-trees, but retains the pruning effect of support. Cosine similarity can be used for pruning, since it holds the conditional anti-monotone property (CAMP), an extension of the anti-monotone property. More details about CAMP can be found in [3].

To cope the challenge led by the drastically increase of the data scale, CoPaMï should be extended to support parallelized mining from disk. Fortunately, FP-tree is apt to be decomposed into several smaller sub-trees. In light of the

aggressive decomposition of FP-tree [1] and BSP model, we present parallelized implementation of CoPaMï, which works as follows:

1. A big file is split into several small files of which the size is 64M according to the setting of HDFS. Then, the first round of BSP peers are started, and each peer works on a small file. Each peer creates Lucene index for each line and obtains tuples $(i_k, \sigma_l(i_k))$ indicating the node i_k and its support in the local file.
2. After BSP master collects all outputs of peers, it aggregates all tuples and sort nodes by support, so that F_1 is obtained. Note that since the range of support is between 0 and n, counting sort can be applied to obtain F_1 with the time complexity $O(n)$.
3. Before starting the second round of BSP, the master should partition F_1 into K groups, where K is the number of BSP peers. Each group corresponds to aggregated frequent items in [1], and then projection can be done to obtain sub-FP-trees. Note that the index on each node is essential to this step, since it needs to repeatedly retrieve a specific line for constructing sub-trees.
4. Every BSP peer invokes CoPaMï on its sub-FP-tree for mining AES. BSP master finally collect the outputs of all BSP peers to get AESs of the original large-scale network.

3 Conclusions

This demo showcase a cloud system called CEC for community extraction from super-large scale social networks. CEC has two notable features: (1) the novel concept of AES and its parallel mining algorithm are employed to isolate weakly connected nodes; (2) multiple cloud computing technologies are integrated for storing and processing massive social networks.

Acknowledgments. We thank all of teachers and students in JSELAB who have put their efforts to CEC. This research was partially supported by NSFC (Nos. 71072172, 61103229), National Key Technologies R&D Program of China (Nos. 2013BAH16F01, 2013BAH16F04), Industry Projects in Jiangsu S&T Pillar Program (No. BE2012185), and the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

1. Grahne, G., Zhu, J.: Fast algorithms for frequent itemset mining using fp-trees. IEEE Trans. Knowl. Data Eng. 17(10), 1347–1362 (2005)
2. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proceedings ACM SIGMOD, pp. 1–12. ACM Press, Dallas (2000)
3. Wu, J., Zhu, S., Liu, H., Xia, G.: Cosine interesting pattern discovery. Information Sciences 184(1), 176–195 (2012)
4. Zhao, Y., Levina, E., Zhu, J.: Community extraction for social networks. Proceedings of the National Academy of Sciences of the USA 108(18), 7321–7326 (2011)