Detecting Genuine Communities from Large-Scale Social Networks: A Pattern-Based Method

ZHIANG WU¹, JIE CAO¹,², JUNJIE WU³, YOUQUAN WANG² and CHUNYANG LIU⁴,*

¹Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing 210003, China
²College of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China
³Department of Information Systems, School of Economics and Management, Beihang University, Beijing 100191, China
⁴National Computer Network Emergency Response Technical Team, Coordination Center of China, Beijing, China
*Corresponding author: lcy@isc.org.cn

Community detection is a long-standing yet very difficult task in social network analysis. It becomes more challenging as many online social networking sites are evolving into super-large scales. Numerous methods have been proposed for community detection from massive networks, but how to reconcile the partitioning efficiency and the community quality remains an open problem. In this paper, we attempt to address this challenge by introducing a COSine-pattern-based COMmunity extraction framework: COSCOM. The COSCOM adopts an extracting view of community detection. It first extracts the so-called asymptotically equivalent structures (AESs) from networks, from which the nodes are further partitioned into crisp communities using any of the existing methods. Specifically, we prove that an AES is a very tight group of nodes, and is actually a cosine pattern defined by the extended cosine similarity. A novel cosine-pattern mining algorithm based on the ordered anti-monotone of cosine similarity is thus proposed for the efficient extraction of AESs. Experiments on various real-world social networks demonstrate the advantage of the extracting view of community detection. In particular, COSCOM shows merits in detecting genuine communities by either internal or external validity.

Keywords: community detection; social network; asymptotically equivalent structure; cosine pattern; ordered anti-monotone

Received 12 January 2013; revised 28 August 2013
Handling editor: Guandong Xu

1. INTRODUCTION

Recent years have witnessed an increasing interest in detecting closely knit groups in large-scale social networks of various kinds. The groups are also called communities, clusters, cohesive subgroups or modules in different research fields [1]. In general, actors in a same community tend to interact with each other more frequently than with those outside the community. Since massive online social networks have been deeply integrated into our daily life, detecting meaningful communities from them has become a critical task for research and applications of various purposes.

Although extensively studied [2–6], community detection remains a core problem in social network analysis. The existing methods in the literature can be roughly divided into two categories, i.e. the one with global models and the one without. The methods with global models include mixture models (or simply K-means), latent space models [7], stochastic block models [8], spectral clustering [9] and modularity optimization [2], among others. These methods typically consider the global topology of a network, and aim to optimize a criterion defined over a network partition. The criteria are often designed carefully so that the optimizations could be achieved via some
efficient yet robust heuristics. The problem is that a real-life network might probably contain nodes that have weak connections to any communities. In such cases, the global models typically split up weakly connected nodes and group them together with tight communities, which actually impedes us from finding the genuine communities.

The methods without global models typically employ a bottom-up strategy to find communities. They often start by defining the properties of a node, a pair of nodes or a group of nodes in a same community, and then search within the whole network for the communities with the proposed properties [10]. Since having clear pictures about genuine communities, they might avoid the negative influences from weakly connected nodes. However, they often suffer from the computational inefficiency—the exhaustive search of a defined structure is often prohibitively expensive, especially when dealing with big data emerging from ever-growing social media.

In this paper, we propose a novel framework named COSine-pattern based COMmunity (COSCOM) extraction for community detection. COSCOM is better regarded as a hybrid method that seeks the compromise between the global and local methods so that we can find truly meaningful communities in an efficient way. The key features of COSCOM lie in the following two folds. First, COSCOM looks for tight structures with nodes in asymptotical equivalence. These asymptotically equivalent structures (AESs) are then proved to be the so-called cosine patterns defined on an extended cosine similarity for node sets. A novel method named cosine-pattern mining (CoPaMi) algorithm can thus be called to mine cosine patterns (i.e. AESs) from large-scale social networks in an efficient manner. Second, COSCOM can incorporate any of existing global methods for the partitioning of extracted nodes assembled from AESs. This helps to find crisp communities from overlapped AESs, and guarantees the efficiency and flexibility of COSCOM for real-world applications. Owing to the above two features, it is more precise to describe COSCOM as a method for community extraction; that is, COSCOM attempts to extract tight communities from massive social networks.

Experiments on four real-world social networks demonstrate that an extracting view of community detection is indeed more appropriate than a partitioning view for networks of large scales. COSCOM improves the performances of three existing partitioning methods by extracting AESs first from the networks. The important role of AESs in community extraction is further explored, from which it turns out that AESs can effectively filter out weakly connected nodes yet maintain the diversity of community members. This endows COSCOM with the capability to extract truly meaningful communities hidden inside massive networks.

The remainder of this paper is organized as follows. In Section 2, we introduce the concept of cosine patterns and the mining algorithm CoPaMi. In Section 3, we introduce AESs and the community extraction framework COSCOM. Experimental results will be given in Section 4. We present the related work in Section 5, and finally conclude this paper in Section 6.

2. COSINE-PATTERN MINING

In this section, we introduce the concept of cosine pattern and its ordered anti-monotone (OAM), upon which an FP-growth-like algorithm is presented for efficient CoPaMi.

2.1. Cosine pattern

In real-world applications, we often meet transaction data sets (denoted as \( T \)), in which a row represents a transaction (denoted as \( t \)) that contains only the items (denoted as \( i \)) bought in this transaction. The relationship between transactions and items can go far beyond ‘buying’. For instance, text data sets are actually a type of transaction data sets, where a document can be viewed as a transaction, each item indicating the particular word appeared in this document.

A transaction data set can be transformed equivalently to a binary matrix (denoted as \( D \)). The \((p, q)\)th entry of \( D \) has a value 1 if transaction \( t_p \) contains item \( i_q \), and 0 otherwise. As the total number of items is often very large, \( D \) is typically a highly sparse matrix that contains a huge volume of zero elements.

To differentiate \( D \) from \( T \), we hereinafter use \( \tilde{t}_p \) to denote the \( p\)th row-vector in \( D \), and \( \tilde{i}_q \) the \( q\)th column-vector in \( D \). They correspond to the transaction \( t_p \) and item \( i_q \), respectively, in \( T \).

Given a transaction data set, we would like to find some hidden interesting patterns, i.e. the special combinations/sets of items or shortly itemsets, that appear simultaneously in some transactions. To this end, an interestingness measure (denoted as \( M \)) as well as a threshold (denoted as \( \tau \)) must be specified to verify whether an itemset (denoted as \( S \)) is truly interesting. That is, \( S \) is interesting with respect to \( M \), iff \( M(S) \geq \tau \). The task of interesting pattern mining is then to discover the set of interesting patterns \( S \) that such that (1) \( \forall S \in S \), \( M(S) \geq \tau \) and (2) \( \forall S \in S \), \( S \subseteq S \). This means all patterns must be included into \( S \), nothing more and nothing less.

Apparently, different \( M \) will result in different \( S \). Therefore, the selection of a proper \( M \) is very important for interesting pattern mining. In the literature, many interestingness measures have been proposed for mining truly interesting patterns [11]. Among them, the cosine similarity gains particular interests. Indeed, cosine similarity has been widely used as a popular proximity measure in text mining [12], information retrieval [13] and bio-informatics [14] to avoid the ‘curse of dimensionality’ [15]. Moreover, it is very simple and has a clear physical meaning—the cosine value of the angle of two vectors.

We therefore focus on cosine interesting pattern discovery, or shortly CoPaMi, from transaction data sets. Given any item pair \( S = \{t_p, t_p'\} \), by the definition of cosine similarity, we have \( \cos(S) = \frac{\tilde{t}_p \cdot \tilde{t}_p'}{\|\tilde{t}_p\| \|\tilde{t}_p'\|} \). Further, let \( \sigma(S) = \frac{|\{t_p | S \subseteq t_p, 1 \leq p \leq n\}|}{\|S\|} \) and \( s(S) = \sigma(S)/n \), with \( n = |T| = |D| \); then
we have
\[
\cos(S) = \frac{s(S)}{\sqrt{s([i_p])s([i_p])}},
\]
where \(s(S)\) is the support of \(S\) from an association rule mining perspective [16]. It is natural to extend Equation (1) to the multi-itemset case. Let \(S\) be a \(P\)-itemset, i.e. \(S = \{i_1, \ldots, i_P\}, P \geq 2\); we define
\[
\cos(S) = \frac{s([i_1, \ldots, i_P])}{\sqrt{s([i_1]) \cdots s([i_P])}}.
\]

It could be argued that the cosine similarity might prefer to change itemsets; that is, if all the itemsets in \(S\) only occur once in some transaction \(t_p\), we still have \(\cos(S) = 1\). To correct this, we can make use of the support measure \(s\), and require that a cosine pattern have a large enough support value. We finally formulate the definition of a cosine pattern as follows.

**Definition 2.1 (Cosine pattern).** \(S\) is a cosine pattern with respect to \(\tau_c\) and \(\tau_s\), if \(\cos(S) \geq \tau_c\) and \(s(S) \geq \tau_s\), where \(\tau_c, \tau_s \in [0, 1]\) are given thresholds.

**Remark 2.1.** In fact, if we remove the cosine constraint or equivalently set \(\tau_c = 0\), the cosine pattern defined in Definition 2.1 reduces to the well-known frequent pattern, with the support \(s\) being the only interestingness measure. This implies that cosine patterns can be interesting and frequent patterns, by setting relatively high \(\tau_c\) and \(\tau_s\) values, such as the widely known \{beer, diapers\} in market-basket analysis. But a cosine pattern means more than a frequent pattern, because cosine patterns can be interesting and infrequent patterns, like \{earrings, gold ring, bracelet\} with rare occurrences but large business values, by setting a relatively high \(\tau_c\) value and a relatively low \(\tau_s\) value. This will not result in the explosion of infrequent patterns for the thresholding effect of \(\tau_c\).

### 2.2. Ordered anti-monotone

To mine cosine patterns, the brute-force method is to examine all the itemsets one by one by Definition 2.1, which is prohibitively expensive due to the complexity \(O(d \cdot 2^{d-1})\), where \(d\) is the total number of items in \(D\).

One possible solution is to take an ‘evaluating’ rather than ‘mining’ strategy. That is, first we find the set of itemsets \(S'\) such that \(\forall S \in S', s(S) \geq \tau_c\); we then refine \(S'\) to \(S\) such that \(\forall S \in S \subseteq S', s(S) \geq \tau_c\). This solution is much better than the brute-force one, since the first step to find \(S'\) is actually the classic frequent pattern mining process, which can be greatly accelerated by using the AM of \(s\) to prune infrequent itemsets before examining them. The second step to refine \(S'\) to \(S\) is more like using cosine similarity to evaluate frequent itemsets, which however does not tap into the full potential of the cosine measure. Ideally, if the cosine measure can work as \(s\) to prune uninteresting itemsets in advance, we can have a much smaller \(S'\) and avoid the refinement step for \(S = S'\). This is what we call the mining strategy.

Unfortunately, the cosine measure does not hold AM. That is, for any two itemsets \(S \subseteq S'\), if \(s(S) < \tau_c\), we have \(s(S') < \tau_c\), which is the well-known Apriori Principle; but if \(\cos(S) < \tau_c\), we cannot guarantee \(\cos(S') < \tau_c\). In other words, when we detect that \(S\) is not a cosine pattern, we cannot claim that all its supersets are definitely not cosine patterns, which will result in huge additional computational costs. Therefore, our primary task here is to find an alternative to AM for efficient CoPaMi. We first present the OAM as follows.

**Definition 2.2 (OAM).** Let \(I\) be a universal itemset. A measure \(M\) holds the OAM, if \(\forall S, S' \subseteq I\), given that (1) \(S \subseteq S'\) and (2) if \(S \setminus S' \neq \emptyset\), \(\forall i_p \in S\) and \(i_p' \in S' \setminus S, s([i_p]) \leq s([i_p'])\), we have \(M(S) \geq M(S')\).

**Remark 2.2.** Definition 2.2 implies that OAM can be regarded as a special case of AM. That is, a measure possessing AM certainly holds OAM, such as the support measure; but the reverse is not true—cosine similarity is just an example. Compared with the well-known AM, OAM demands an extra condition that all the items in the difference set \((S' \setminus S)\) must have higher supports than the items in the subset \((S)\).

We then have the following theorem.

**Theorem 2.1.** Cosine similarity holds the OAM.

**Proof.** Without loss of generality, we assume \(S = \{i_1, \ldots, i_P\}\) is a \(P\)-itemset \((P \geq 1)\), and \(S' = S \cup \{i_{P+1}, \ldots, i_{P+L}\}\) is a \(P\)-itemset \((L \geq 0)\), with \(s([i_{P+1}]) \geq s([i_{P+1}])\) for \(1 \leq l \leq L\), \(1 \leq p \leq P\). Now it remains to show \(\cos(S) \geq \cos(S')\).

Note that if \(L = 0\), we have \(S = S'\); thus \(\cos(S) = \cos(S')\). Now assume \(L \neq 0\), i.e. \(S' \setminus S \neq \emptyset\). It is easy to know
\[
s(S) \geq s(S').
\]

Moreover, according to the definition of geometric mean, we have
\[
\frac{1}{P} \prod_{p=1}^{P} s([i_p]) \leq \frac{1}{P+L} \prod_{p=1}^{P+L} s([i_p]),
\]
for \(s([i_{P+1}]) \geq s([i_{P+1}])\), \(1 \leq l \leq L, 1 \leq p \leq P\). So we finally have
\[
\cos(S) = \sqrt{\frac{1}{P} \prod_{p=1}^{P} s([i_p])} \geq \sqrt{\frac{1}{P+L} \prod_{p=1}^{P+L} s([i_p])} = \cos(S'),
\]
which completes the proof. \(\square\)

Theorem 2.1 indicates that the cosine measure holds the OAM, which lays the foundation for efficient CoPaMi. Before we go into the algorithmic details, however, we would explore the AM of an upper bound of cosine similarity, which can further improve the mining efficiency.
2.3. Upper bound of cosine similarity and its AM
Given an uninteresting itemset, the OAM of cosine similarity can help to prune its uninteresting supersets. Interestingly, by using an upper bound of cosine similarity (or cosine upper bound for short) and its AM, we might further prune its uninteresting siblings. To illustrate this, we first have the following lemma.

**Lemma 2.1.** For any 2-itemset $S = \{i_p, i_p\}$,

$$\cos(S) \leq u_c(S) = \frac{\min(s(|i_p\rangle), s(|i_p\rangle))}{\max(s(|i_p\rangle), s(|i_p\rangle))}$$  \hspace{1cm} (6)

Therefore, $u_c$ in Equation (6) is a cosine upper bound for 2-itemsets. More importantly, $u_c$ holds the following property.

**Theorem 2.2.** Given any 1-itemset $S = \{i_p\}$ and its immediate supersets $S' = S \cup \{i_p\}$ and $S'' = S \cup \{i_p\}$, if $s(|i_p\rangle) \leq s(|i_p\rangle) \leq s(|i_p\rangle)$, we have

$$u_c(S') \leq \tau_c \Rightarrow \cos(S'') \leq \tau_c.$$ \hspace{1cm} (7)

**Remark 2.3.** The proof is straightforward by noting that $u_c(S') \geq u_c(S'')$. Since $S'$ and $S''$ are two siblings with the same parent set $S$, Theorem 2.2 indeed indicates the AM of $u_c$ on sibling itemsets. It is worth noting that the computation of $u_c(S)$ is more cost-efficient than the computation of $\cos(S)$, so we can use the AM of $u_c$ to accelerate the mining of cosine patterns with two items.

It is natural to extend the 2-itemset case to the multi-itemset case. We have the following lemma.

**Lemma 2.2.** For any $P$-itemset $S = \{i_1, \ldots, i_P\}$,

$$\cos(S) \leq u_c(S) = P^{-1} s(S \setminus |i_P\rangle) \sum_{p=1}^{P} s(|i_p\rangle)^{-1}.$$ \hspace{1cm} (8)

We then analogously have the following theorem.

**Theorem 2.3.** Given any $P$-itemset $S$ and its immediate supersets $S' = S \cup \{i_p\}$ and $S'' = S \cup \{i_p\}$, if $s(|i_p\rangle) \leq s(|i_p\rangle)$, we have

$$u_c(S') \leq \tau_c \Rightarrow \cos(S'') \leq \tau_c.$$ \hspace{1cm} (9)

**Remark 2.4.** Note that the cosine upper bound in Equation (8) is looser than the one in Equation (6) for the purpose of lowering the computational cost. Therefore, although Theorem 2.3 is also applicable to 2-itemsets, we prefer using Theorem 2.2 for 2-itemsets. Considering that $u_c$ in Equation (8) is often too loose and long cosine patterns are usually very few, Theorem 2.3 indeed has more theoretical rather than practical meaning.

2.4. CoPaMi: the algorithm
In this section, we propose CoPaMi, a novel CoPaMi algorithm based on the OAM of cosine similarity.

In general, CoPaMi employs two steps to mine cosine patterns. The first step is to construct an FP-tree, a special structure for fast support computation in frequent pattern mining [17]. The second step is to search the FP-tree for cosine patterns using an FP-growth-like [17] procedure: CP-growth. The pseudocodes of CoPaMi are given in Algorithm 1.

**Algorithm 1 CoPaMi**

1: Create the FP-tree $Tree$ for transaction data $D$;
2: Let $S$ store the current suffix, $S \leftarrow \emptyset$;
3: Let $S$ store the cosine patterns, $S \leftarrow \emptyset$;
4: **procedure** CP-GROWTH($Tree\; S\; \tau_s\; \tau_c\; S$)
5: for each item $i_k$ from bottom to top in $Tree$’s head table
6: generate candidate pattern $S \leftarrow \{i_k\};$
7: if $u_c(S') < \tau$, then break;
8: $S \leftarrow S \cup \{S'\}$;
9: **end if**
10: **end for**
11: **procedure** CP-GROWTH($Tree\; S\; \tau_s\; \tau_c\; S$);
12: **end if**
13: **end procedure**
14: return $S$;

Note first that to utilize the OAM of the cosine measure, CoPaMi demands the FP-tree be built on transactions with items sorted in a support-descending order. This is, however, not mandatory for the FP-tree used in frequent pattern mining, where the items can be sorted in any order due to the more powerful AM of the support measure. Let us take the toy data in Fig. 1 as an example. As can be seen, given the transaction data set in the upper-left part of the figure, we have $s(G) > s(F) > s(E) > s(D) > s(C) > s(B)$ $\geq s(A)$. As a result, we must first sort the items in each transaction from $G$ to $A$ (A and B are interchangeable). The FP-tree for CoPaMi can then be built in the bottom-left part of Fig. 1. In this way, CP-growth can traverse itemsets ended by $A$ to $G$ orderly, no matter in the FP-tree or a conditional FP-tree. This will tap the potentials of the OAM of the cosine measure and the AM of the cosine upper bound. We will detail this by the example below.

It is also noteworthy that CP-growth no longer distinguishes between single-path trees and multi-path trees, which were, however, treated differently in the classic FP-growth procedure. In FP-growth, when a conditional FP-tree is a single-path tree, FP-growth can simply enumerate all the node combinations as frequent itemsets without further sub-tree projections.

---

**The Computer Journal, Vol. 57 No. 9, 2014**

---

**Downloaded from http://comjnl.oxfordjournals.org/ at Southeast University on September 14, 2014**
Detecting Genuine Communities from Large-Scale Social Networks

should be attributed to the powerful AM. However, since the cosine measure only possesses OAM, CP-growth cannot just enumerate the node combinations in a single-path tree as cosine patterns. Instead, it has to continue the sub-tree projection as for the multi-path trees, although in a much simpler manner—a single-path tree will be always projected to a single-path sub-tree. The pseudocodes in Line 12 of Algorithm 1 are for the sub-tree projections, for both single-path and multi-path trees.

Now we illustrate how CP-growth uses the three measures, i.e. support \( s \), cosine \( \cos \) and cosine upper bound \( u_c \), simultaneously to reduce the search space in Algorithm 1. Actually, \( s \) serves as the first filter when executing the projection operation in Line 12, which guarantees that all the candidate itemsets are frequent. Then \( u_c \) acts as the second filter in Line 7; that is, if a candidate itemset \( S \) has \( u_c(S) < \tau_c \), the loop in Line 5 will be broken—the remaining siblings of \( S \) will have \( u_c \) definitely lower than \( \tau_c \) due to the AM of \( u_c \). If the candidate passes the \( u_c \) examination, CP-growth finally checks its cosine value in Line 10. If the cosine value is lower than \( \tau_c \), it means the candidate and its supersets are definitely not cosine patterns, and the loop in Line 5 will jump to the next sibling and continues. To sum up, \( s \), \( u_c \) and \( \cos \) together form a stratified valve that ensures the efficiency of CP-growth.

**Example 2.1.** Here, we illustrate CP-growth by the toy example in Fig. 1. Let \( \tau_c = 0.65 \) and \( \tau_s = 0 \). Assume that we want to mine cosine patterns ended by item \( C \). To this end,

(i) CP-growth first generated the conditional FP-tree based on the suffix \( S = \{ C \} \), as shown in the upper-right part of Fig. 1. Note that since \( \tau_s = 0 \), \( s \) did not take the filtering effect here.

(ii) CP-growth then proceeded to check the cosine upper bound of \( \{ D, C \} \). Since \( u_c(\{ D, C \}) = 0.707 \), CP-growth further examined the cosine value of \( \{ D, C \} \), which was \( \cos(\{ D, C \}) = 0.364 < \tau_c \). As a result, CP-growth stopped the sub-tree projection for potential cosine patterns ended by \( \{ D, C \} \), based on the OAM of the cosine measure. This means the pruning of the whole sub-tree circled by the red dotted-line in the bottom-right part of Fig. 1.

(iii) CP-growth then jumped to the itemset \( \{ E, C \} \), the immediate sibling of \( \{ D, C \} \). Since \( u_c(\{ E, C \}) = 0.632 < \tau_c \), CP-growth suspended the loop for the supersets of \( \{ E, C \} \), as well as its siblings including \( \{ F, C \} \) and \( \{ G, C \} \) and their supersets. In other words, the whole sub-trees circled by the blue dashed-line in the bottom-right part of Fig. 1 were pruned.

As a result, only \( \{ C \} \) will be returned from the 16 itemsets. This case well demonstrates the high efficiency of CoPaMf rooted in the OAM of the cosine measure, the AM of its upper bound and the FP-tree structure with items sorted in a support descending order.

3. COMMUNITY DETECTION BASED ON COSINE PATTERNS

Community detection traditionally implies the partitioning of the entire network into various sub-networks. As many real-world social networks are getting unprecedentedly large, say far beyond thousands of nodes, this ‘partitioning’ notion of community detection seems getting less meaningful—the
resulting sub-networks are often so complex that few clues can be found to trace genuine communities.

We therefore argue that an ‘extracting’ view of community detection might be more appropriate than a ‘partitioning’ view for networks of large scale. Indeed, a genuine community is considered to be a closely knit group consisting of members in frequent communication with each other. These communities are usually relatively small such that they seem to be hidden in the massive networks, disguised by a huge number of weakly connected nodes. In this sense, community detection is the task of extracting tight communities from large-scale social networks, which is thus more precise to be called community extraction.

Some math notations are given as follows. An undirected network is often defined as $G = (V, E)$, where $V$ is the set of nodes in $G$, and $E$ is the set of edges that connect the nodes in $V$. Community detection is formulated as finding a $K$-way partition $\mathcal{P} = \{C_1, \ldots, C_K\}$, where $C_k$ is the $k$th community, and $C_1 \cup \cdots \cup C_K \subseteq V$. For a crisp partition, we have an additional requirement: $C_k \cap C_{k'} = \emptyset \ \forall k \neq k'$, although there is some recent work on detecting overlapping communities [18, 19]. The set of nodes (or called neighbors or friends interchangeably) that link to node $i_p$ is denoted as $N_{i_p}$. Apparently, $|N_{i_p}| = d_p$ for networks without self-loops, where $d_p$ is the degree of node $i_p$.

### 3.1. From AESs to cosine patterns

In the literature, a key concept related to communities is structural equivalence. That is, two nodes $i_p$ and $i_q$ are structurally equivalent if $N_{p} = N_{q}$. In other words, the nodes in structural equivalence must have exactly the same friends, and thus tend to form a very tight community. This idea, however, poses one problem. That is, structural equivalence is often too restrictive for real-life social networks—we can only find very few equivalent structures inside a network. Other existing definitions of equivalence, such as automorphic equivalence and regular equivalence [20], are looser than structural equivalence but suffer from the very high computational cost.

To meet this challenge, we first reformulate the concept of structural equivalence. As we know, a node set $S = \{i_1, \ldots, i_{|S|}\}$ is structurally equivalent if $N_{i_1} = \cdots = N_{i_{|S|}}$, or equivalently,

$$\frac{|\bigcap_{q=1}^{|S|} N_{i_q}|}{|N_{i_1}|} = \cdots = \frac{|\bigcap_{q=1}^{|S|} N_{i_q}|}{|N_{i_{|S|}}|} = 1. \quad (10)$$

Let $r_p = |\bigcap_{q=1}^{|S|} N_{i_q}|/|N_{i_p}|$ characterize the ratio of common friends for node $i_p$, $1 \leq p \leq |S|$. Equation (10) actually indicates that all the ratios of common friends must be one in an equivalent structure. Following this idea, to relax the concept of structural equivalence, we can set the ‘ratio of common friends’ as a measure but lower its threshold from one to a proper level. This gives birth to the following statistic: the averaged ratio of common friends, $G$.

### Definition 3.1

Given a node set $S$ with $|S| \geq 2$, the average ratio of common friends of $S$ is defined by

$$G(S) = \frac{\sum_{p=1}^{|S|} r_p}{|S|}, \quad (11)$$

where

$$r_p = \frac{|N_{i_1} \cap \cdots \cap N_{i_{|S|}}|}{|N_{i_p}|}, \quad 1 \leq p \leq |S|. \quad (12)$$

According to the definition, $G$ is essentially the geometric mean of the ratios of common friends for all the nodes in a same set. Apparently, $G \in [0, 1]$, and a larger $G$ indicates a generally higher percentage of common friends, and thus implies a tighter community.

The community extraction task therefore turns into the search for the node sets that have $G$ values beyond some given threshold. Moreover, to filter out node sets weakly connected to the network, we should further demand that a node set should have a certain number of common friends. To this end, we further employ the statistic: $F(S) = |N_{i_1} \cap \cdots \cap N_{i_{|S|}}|/n$, where $n$ is the total number of nodes in a network. Based on $G$ and $F$, we have the following definition.

### Definition 3.2

A node set $S$ is an AES 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 AESs from a large-scale network. We use the term ‘asymptotically’ here because of the following fact.

### Proposition 3.1

An AES turns into an equivalent structure iff $\tau_G = 1$.

The proof is straightforward by Equations (11) and (10). Proposition 3.1 implies that an AES can be viewed as a ‘relaxed’ equivalent structure, and the gap reduces gradually with the increase of the threshold $\tau_G$.

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 $A$), which is actually a binary matrix, where $A_{pq} = 1 \ (p \neq q)$ if there is an edge between node $i_p$ and node $i_q$ and 0 otherwise. We let $A_{pp} = 0 \ \forall p$ to exclude self-loops. Accordingly, $\Sigma_{q=1}^n A_{pq} = \Sigma_{q=1}^n A_{qp} = |N_{i_p}| = d_p$, $1 \leq p \leq n$. Let $T_A$ 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 $S$, $|N_{i_1} \cap \cdots \cap N_{i_{|S|}}| = |T_p \mid S \subseteq T_p, 1 \leq p \leq n| = \sigma(S)$, where $t_p = |i_p|A_{pq} = 1, 1 \leq q \leq n$ is the $p$th transaction in $T_A$, and $\sigma(S)$ is the support count of $S$ in $T_A$. As a result, we have $F(S) = |N_{i_1} \cap \cdots \cap N_{i_{|S|}}|/n = s(S)$, i.e. the support of $S$. Moreover, it is easy to show $G(S) = |N_{i_1} \cap \cdots \cap N_{i_{|S|}}|/\sqrt{\prod_{p=1}^{|S|} |N_{i_p}|} = \sigma(S)/\sqrt{\prod_{p=1}^{|S|} \sigma(i_p)}$.
Proposition 3.2. Given the thresholds $\tau_G$ and $\tau_F$, to extract all the AESs from a network $G$ is equivalent to mining all the cosine patterns from the corresponding adjacency matrix $A$, with $\tau_c = \tau_G$ and $\tau_s = \tau_F$.

3.2. Community detection framework

In this section, we propose a general framework called COSCOM extraction framework for community detection from massive networks.

COSCOM consists of three main phases as follows:

(i) Extracting: To extract AESs from the whole network using the CoPaMi algorithm: CoPaMi.
(ii) Partitioning: To assemble all the nodes from the extracted AESs and partition them into communities using existing community detection methods.
(iii) Evaluating: To evaluate the quality of the detected communities using various validation measures or node semantics.

Figure 2 illustrates the main phases of COSCOM, among which Phase (i) is of particular importance. During the extracting process, we use the two thresholds $\tau_G$ (i.e. $\tau_c$) and $\tau_F$ (i.e. $\tau_s$) to flexibly control the results. In general, we would like to set a moderate value to $\tau_G$ first, e.g. let $\tau_G = 0.5$, to guarantee the quality of the extracted AESs. $\tau_F$ is then set from a relatively large value to a small value to make the balance between the scale of the extracted nodes and the efficiency of the mining process. If the results are yet not satisfactory, we can further adjust $\tau_G$ to either a smaller value for more extracted nodes or a larger value for higher efficiency. In the experimental section below, we give the information about the settings of the two thresholds in each experiment.

Since the nodes that are not contained by any AES can be regarded as weak-tie nodes, we drop these nodes, and thus extract all the edges that connect nodes included in AESs. In this way, we assemble the nodes from AESs and form a smaller sub-network that may or may not contain multiple connected components.

It might be questioned as to why we need to further partition the nodes collected from extracted AESs in Phase (ii). To understand this, we should note that the extracted AESs are generally small and overlapped. Therefore, by assembling and partitioning the nodes, we could finally obtain crisp communities in moderate sizes. It is also noteworthy that one obvious advantage of COSCOM is the capability of incorporating any of the existing partitioning tools. This provides great flexibility to COSCOM in dealing with networks generated from various domains. As the extracted nodes are expected to have a much clearer tribalized tendency than the original network, the difficulty of partitioning is actually lowered to a great extent. We will demonstrate this in the experimental section.

The community evaluation, as to be conducted in Phase (iii) of COSCOM, is actually non-trivial or even controversial. This should be largely attributed to the unavailability of the community labels for the nodes in a large-scale network. As a result, many external measures widely used for cluster validity generally cannot be adapted to community evaluation. Nevertheless, there yet exist some alternative ways for this purpose. First, some small real-world networks with known community labels, e.g. Karate-Club [21], can be used as the benchmarks for algorithm testing. Synthetic networks generated on specific parameters can also serve this purpose. In both cases, external measures can be adopted since the node labels are known. Second, some internal measures using the network topological information only can be used for community evaluation. For instance, the Modularity $Q$ [3] is such a popular internal measure, although some recent studies have pointed out its limitations [22]. Third, some real-world networks are provided with the semantic information of the nodes, which can be used to infer the cohesion of a detected community. All these three methods can be used in Phase (iii) of COSCOM, as will be demonstrated in the sections to follow.

3.3. Some examples

Here, we apply COSCOM to two small social networks with ground truths. The purpose is to gain a direct understanding of community extraction by network visualization.
Karate-Club. This is a classic social network with 34 members in a Karate club [21]. This club was split into two parties following a disagreement between an instructor (node 1) and an administrator (node 34), which serves as the ground-truth about the communities in Fig. 3a. We set $\tau_G = 0.3$ and $\tau_F = 7\%$, and employed COSCOM with FastNewman (FN) (a partitioning tool introduced in Section 4) to extract communities from the network. The result is shown in Fig. 3b, which supplements the division of the club with more information about the 'cores' of each fraction. More interestingly, according to the Modularity $Q$ measure (specified in Section 4.2.1), COSCOM actually tends to partition this network into three rather than two communities, as indicated by the nodes in three colors in Fig. 3b. This implies that there exits a latent sub-party (including nodes 5, 6, 7, 11) inside the party led by node 1, whose members are absolutely loyal to node 1.

Political-Books. This network contains 105 nodes representing books about US politics sold by Amazon.com [23]. Edges indicate pairs of books frequently bought together, and it is assumed that customers with political inclinations would tend to buy books in the same political positions. The ground truth labeled nodes with ‘liberal’, ‘neutral’ or ‘conservative’, corresponding to ‘blue diamond’, ‘green circle’ and ‘red triangle’, respectively, in Fig. 3c. By setting $\tau_G = 0.5$ and $\tau_F = 5\%$, 42 nodes were extracted by COSCOM and partitioned into two communities using FN, as shown in Fig. 3d. As can be seen, COSCOM generally well captured the ‘sharp-cut’ books taking the liberal and conservative positions, respectively, although there yet existed two books (highlighted by dashed circles) assigned mistakenly. Note that nearly all the neutral books (the green nodes in Fig. 3c) were kept out by COSCOM. This is indeed reasonable since they have very

![FIGURE 3. COSCOM on small social networks. (a) Karate-Club: ground truth, (b) Karate-Club: our result, (c) Political-Books: ground truth and (d) Political-Books: our result.](http://comjnl.oxfordjournals.org/DownloadedFrom Southeast University on September 14, 2014)
few internal connections but link heavily to the books with inclinations.

4. EXPERIMENTAL RESULTS

In this section, we showcase the effectiveness of COSCOM on detecting communities from various real-world social networks. Three existing tools, i.e. K-means in CLUTO [24], METIS [25] and FN [2], were exploited to partition the extracted nodes in Phase (ii) of COSCOM. K-means is a classic clustering method, whose CLUTO implementation (http://glaros.dtc.unm.edu/gkhome/views/cluto) has been widely used in text clustering. METIS (http://glaros.dtc.unm.edu/gkhome/views/metis) is a famous tool based on the graph partitioning method. FN (http://cs.unm.edu/~aaron/research/fastmodularity.htm) is a tool based on a modularity optimization method, which adopts an agglomerative hierarchical clustering strategy. Note that we here have no intention of comparing the three tools but rather attempt to demonstrate the flexibility of COSCOM in using the existing tools.

4.1. Data sets

Four real-world social networks: Oklahoma, Enron, Gowalla and CMC’05 were used for experiments. Some characteristics of these data sets are shown in Table 1, where |V| and |E| indicate the numbers of nodes and edges, respectively, in the network, \( \langle k \rangle = 2|E|/|V| \) indicates the average degree, and \( C \) indicates the average clustering coefficient.

Oklahoma is a Facebook network whose ties are within the University of Oklahoma [26]. In the network, each node represents a user and each tie means that there exists a mutually permitted friendship between two users.

Gowalla is also a friendship network collected from a location-based social networking site called Gowalla, where users share their locations by checking-in [27]. The network is undirected and was collected using the public API.

Enron is a communication network covering all the email communications within a data set of half a million emails [28]. Nodes of the network are email addresses. If an address \( i \) sends at least one email to address \( j \), there will be an undirected edge from \( i \) to \( j \), and vice versa.

CMC’05 is a co-authorship network including all preprints posted between 1 January 1995 and 31 March 2005 on Cornell University Library [29]. After removing the isolated nodes, we finally obtained a graph with 39,577 nodes and 175,693 edges, where a node represents a scientist (author) in the area of computer network, and an edge indicates that two authors have co-authored at least one paper.

As Oklahoma, Enron and Gowalla contain only topological information, the detected communities will be evaluated by the internal measure: Modularity \( Q \) (see Section 4.2.1 for details). For CMC’05, since more external information about the authors can be accessed online, we will evaluate the communities based on their semantics (see Section 4.2.2 for details).

4.2. Overall performances of COSCOM

Here, we present the performances of COSCOM on four real-world networks. The results by directly partitioning the original networks are also presented as the benchmarks.

4.2.1. Performances in terms of modularity \( Q \)

We first validate the effectiveness of COSCOM via a widely adopted internal measure: Modularity \( Q \). It is computed as follows:

\[
Q = \frac{1}{2|E|} \sum_{i=1}^{K} \left( \frac{|E_i|}{|E|} - \frac{d_i^C}{2|E|} \right)^2, \tag{13}
\]

where \( K \) is the number of communities, \( E_i \) is the set of edges within community \( C_i \), and \( d_i^C = \sum_{p \in C_i} d_p \) is the sum of node degrees of \( C_i \). The value of \( Q \) is in the interval: \((-1, 1)\), and a larger value indicates a better performance.

First of all, we investigate the impact of the parameter \( K \) by fixing \( \tau_G = 0.5 \) and taking \( \tau_F = 0, 0.01, 0.015\% \), respectively, for Oklahoma, Enron and Gowalla. As can be seen from Fig. 4, the \( Q \) value changes slightly with the increase of \( K \) for nearly all the data sets, no matter what partitioning method is used. So we set \( K = 10 \) as the default value in the following experiments.

We then extensively explore the performances of COSCOM. Figure 5 shows the results of COSCOM on Oklahoma, Enron and Gowalla. In each experiment, we set \( \tau_G = 0.5 \), and increased \( \tau_F \) gradually so that the number of extracted nodes dropped accordingly, as illustrated by the leftward x-axis in each sub-figure. The left-most points highlighted by solid lines in the sub-figures are different from other points; they are the results by employing CLUTO, METIS and FN directly on the original networks, which are used for the comparison purpose here.

Two observations are noteworthy from Fig. 5. First, compared with the direct community detection (indicated by the left-most points), COSCOM indeed showed significant improvements in terms of \( Q \), no matter what partitioning method was used. For instance, FN obtained a \( Q \) value lower than 0.4 on the original Gowalla network; but after using CoPaMi in COSCOM to extract nodes, the \( Q \) value increased rapidly, and finally reached...
0.85 on a set of highly tribalized nodes. This implies that the extracting phase is critically important for the success of COSCOM in community detection.

Another important observation is that less extracted nodes might not indicate a clearer communitization. As can be seen from Fig. 5b, when the number of extracted nodes reduced to around 7000, all the three partitioning methods approximated to the highest $Q$ values. As $\tau_F$ further went up and less nodes were thus extracted, COSCOM however showed poorer performances instead. This implies that if we set a too high $\tau_F$ for CoPaMi, we might miss some meaningful yet hidden AESs, which will eventually degrade the performance of COSCOM.

We then take a direct look at the extracted communities. The Enron network is used here as an example. As can be seen from Fig. 6a, the whole graph of Enron consists of a gigantic intertwined cluster and lots of peripheral nodes. We then employed CLUTO directly on this network, and obtained ten obscured communities that could not be identified even with great efforts, as shown in Fig. 6b. However, if COSCOM is used instead with $\tau_G = 0.5$ and $\tau_F = 0.02\%$, only 1766 nodes will be extracted from the original network, which display a clear structure of 10 communities in Fig. 6c.

In summary, as indicated by Modularity $Q$, COSCOM indeed can improve the community detection performance by mining AESs in the extracting phase.

### 4.2.2. Performances in terms of node semantics

The modularity $Q$ only gives us a general feeling about the performance of community detection. For some complex networks with high clustering coefficients, like CMC’05 to be explored in this section, a moderate $Q$ value cannot tell us more detailed information about the detected communities. Neither can we know whether the detected communities are truly meaningful, for extracted nodes might still form a seriously intertwined graph. In such situations, our focus should shift to checking whether there does exist one or several genuine communities that contain nodes with consistent semantics.

In the co-authorship network CMC’05, there are usually several co-authors in one paper, which makes it a very dense network with a high clustering coefficient. Therefore, after employing COSCOM on CMC’05, we attempt to seek genuine communities from the detected communities. The semantics of the nodes, such as the affiliations, research interests and biographies of the authors retrieved from Cornell University Library, the homepages of the authors and the websites of their affiliations, were jointly used for this purpose.

We set $\tau_G = 0.5$ and $\tau_F = 0.06\%$ to obtain more AESs yet avoid running CoPaMi out of memory. METIS was used here for partitioning the extracted nodes, with the community number $K = 30$. Figure 7 shows the results with $Q = 0.570$, where the
FIGURE 6. Community detection from Enron. (a) Original network, (b) direct partitioning and (c) using COSCOM.

FIGURE 7. Community detection from CMC’05 using COSCOM and node semantics.

left part illustrates the extracted nodes in 30 communities. As can be seen, even for the extracted 930 nodes, the subgraph is still very complex and show highly obscured structures. So we turn to find some genuine communities with clear semantics. The right part of Fig. 7 shows one such community we found.

This is a closely knit community containing 29 nodes in total, isolated from the subgraph. After carefully examining the co-authored papers and the homepages of these co-authors, we identified a genuine research group consisting of 16 members, called Stanford for short. Among them, five members are from the Physics Department, and another seven from the SLAC National Accelerator Laboratory, both in Stanford University. By further investigating the biography of C. Hall from University of Maryland and A. Piepke from University of Alabama, we found that both of them even worked in Stanford University for some time; that is, C. Hall conducted postdoctoral research during 2002–2006, and A. Piepke was a visiting physicist in 2006. In this sense, they can be also viewed as part of the research group of Stanford University. Finally, K. Hall from Colorado State University and Z. Djuric from Argonne National Laboratory are deemed to be two members...
TABLE 2. Stanford community by direct partitioning.

<table>
<thead>
<tr>
<th>Community</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>R. DeVoe</td>
</tr>
<tr>
<td>C2</td>
<td>J. Wodin</td>
</tr>
<tr>
<td>C3</td>
<td>C. Hall</td>
</tr>
<tr>
<td>C4</td>
<td>A. Odian</td>
</tr>
<tr>
<td>C5</td>
<td>K. Hall</td>
</tr>
<tr>
<td>C6</td>
<td>R. Neilson</td>
</tr>
<tr>
<td>C7</td>
<td>G. Gratta, T. Koffas, M. Breidenbach, R. Conley, Z. Djurcic</td>
</tr>
<tr>
<td>C8</td>
<td>A. Piepke, C.Y. Prescott, P.C. Rowson, K. Skarpaas, K. Wamba</td>
</tr>
</tbody>
</table>

of this research group, since they co-authored multiple papers with the scholars at Stanford University.

Detecting the Stanford community using COSCOM is meaningful, since it will be hard to find it by simply partitioning the original network. To illustrate this, we have employed METIS directly on CMC ‘05 to generate 30 communities. The result, however, is not satisfactory. The 16 members of the group were partitioned into 8 different communities, as shown in Table 2. For instance, K. Hall was assigned to community C5, although he co-authored 11 papers with C. Hall in C3, and 4 papers with G. Gratta and 2 papers with Z. Djurcic in C7.

4.3. Inside communities

Here, we look inside the extracted communities to see whether the nodes in a same community show some structural properties. We took as an example a community with 443 nodes extracted from Enron (for COSCOM: \( \tau_G = 0.5, \tau_F = 0.01\% \), METIS, \( K = 10 \)), and selected four topological indices including degree, clustering coefficient, eigenvector centrality and betweenness centrality [30, 31] for the explorative purpose. The last two indices are often used to measure the node importance to the network than just connectivity, and a larger value implies higher importance.

Figure 8 shows the comparative results of the indices. In addition to the community extracted by COSCOM using CoPaMi (titled ‘Comm. with CoPaMi’), we added a community obtained by employing METIS directly on Enron without using CoPaMi (titled ‘Comm. without CoPaMi’). This community was selected purposefully so that it has the biggest overlap (263 nodes) with the former community. Finally, the entire graph was regarded as a largest community and thus also included for the comparison purpose. For all the indices, the right-most (left-most) bars include all the nodes with index values no less (greater) than the labeled ones.

As can be seen from Fig. 8, the percentages of the nodes with the lowest index values were reduced greatly for the
extracted community. For instance, in Fig. 8b, only 2 out of 12,411 nodes that have zero clustering coefficients were selected into the extracted community. This implies that COSCOM has the ability to filter out weakly connected nodes (indicated by the degree and clustering coefficient indices) and unimportant nodes (indicated by the two centrality indices) in the extracting phase. However, this does not mean COSCOM is biased to the most important nodes only, which will not agree with our intuition about a genuine community—members should keep in touch yet have good diversities. As can be seen from each sub-plot in Fig. 8, compared with the other two communities, the extracted community generally contained more nodes in percentage with moderate index values. This reflects that the AESs indeed capture well the normal community members.

One may still wonder why the right-most red bar is lower than the blue bar for the clustering coefficient index (denoted as $c$), which does not agree with the situations of the other three indices. To understand this, we looked inside the nodes with $c = 1$. It is interesting to find that the Enron network actually contains many triangle structures, i.e. the cliques consisting of only three nodes, which result in a large portion of nodes with $c = 1$, as indicated by the highest right-most blue bar in Fig. 8b. Such structures, however, might contain the nodes weakly connected to the networks, e.g. the nodes #7 to #11 in two cliques in Fig. 9, and thus be discarded by COSCOM with a non-zero $\tau_F$. Indeed, this is why we introduce the $F$ measure to define an AES in Definition 3.2.

### 4.4. Further analysis on AESs

Here, we further explore the critical factor that contributes to the success of COSCOM. As discussed in Section 3.1, while an AES is defined on both the $G$ and $F$ measures, $G$ is particularly responsible for the quality of an AES. To validate this, we design a comparative experiment here by using AESs with and without $G$, respectively. That is, in the ‘COSCOMwoG’ method, we employ COSCOM on networks using AESs defined on both the $G$ and $F$ measures; in the ‘COSCOMwoG’ method; however, we use AESs defined only on the $F$ measure. If COSCOMwoG shows better performances than COSCOMwoG, we would tend to believe that $G$ plays a vital role in defining an AES.

The Oklahoma network was selected for this comparative study, and METIS was adopted for both methods to partition the extracted nodes into 10 communities. In COSCOMwoG, $\tau_F$ was first set to a relatively high value, and then decreased gradually until CoPaMi ran out of memory. In COSCOMwoG, however, $\tau_F$ could be lowered to even zero for the existence of $\tau_G$. Therefore, we tuned the two thresholds carefully so that the scales of the generated AESs were roughly comparable with the ones by COSCOMwoG.

Figure 10 shows the extraction results under five threshold configurations for both methods (specified in Table 3). As can be seen, COSCOMwoG can only extract 2000 nodes at most, since a $\tau_F$ smaller than 0.76% will lead to memory failure of CoPaMi. For COSCOMwoG, however, $\tau_F$ can be set as small as possible for the existence of $\tau_G$, which did help to extract much

![Figure 9. Illustration of triangle structures.](image)

![Figure 10. Comparison of #nodes extracted from Oklahoma.](image)

<table>
<thead>
<tr>
<th>$\tau_F$ (%)</th>
<th>$\tau_G$</th>
<th>$Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>0.20</td>
<td>0.50</td>
</tr>
<tr>
<td>$P_2$</td>
<td>0.00</td>
<td>0.45</td>
</tr>
<tr>
<td>$P_3$</td>
<td>0.10</td>
<td>0.40</td>
</tr>
<tr>
<td>$P_4$</td>
<td>0.10</td>
<td>0.38</td>
</tr>
<tr>
<td>$P_5$</td>
<td>0.10</td>
<td>0.35</td>
</tr>
<tr>
<td>$P_6$</td>
<td>0.95</td>
<td>N/A</td>
</tr>
<tr>
<td>$P_7$</td>
<td>0.85</td>
<td>N/A</td>
</tr>
<tr>
<td>$P_8$</td>
<td>0.80</td>
<td>N/A</td>
</tr>
<tr>
<td>$P_9$</td>
<td>0.78</td>
<td>N/A</td>
</tr>
<tr>
<td>$P_{10}$</td>
<td>0.76</td>
<td>N/A</td>
</tr>
</tbody>
</table>
The differences between these methods ultimately come down to the fact that it is not possible to replace the valve effect of G. As can be seen, COSCOMwG exhibited significantly better performance than COSCOMwoG under any configuration, although there were actually more extracted nodes to be partitioned by COSCOMwG. This reveals that the G measure indeed plays a vital role for the success of COSCOM; that is, it is G rather than F that is responsible for the extraction of high-quality AESs.

Table 3 shows the community detection results measured by Modularity Q. As can be seen, COSCOMwG exhibited significantly better performance than COSCOMwoG under any configuration, although there were actually more extracted nodes to be partitioned by COSCOMwG. This reveals that the G measure indeed plays a vital role for the success of COSCOM; that is, it is G rather than F that is responsible for the extraction of high-quality AESs.

5. RELATED WORK

Community detection has become a fundamental problem ever since network science came into vogue. We here briefly review some basic methods, including the recent work on community extraction.

In the literature, the existing community detection methods can fall into two categories, one with global models and the other without. The methods with global models typically consider the global topology of a network, and aim to optimize a criterion defined over a network partition. Some methods along this line include the Kernighan–Lin algorithm [32], latent space models [7], stochastic block models [8], modularity optimization [2] and traditional clustering techniques [33] such as K-means, multi-dimensional scaling and spectral clustering. The differences between these methods ultimately come down to the precise definition of a ‘denser’ community, i.e. the global criterion and the algorithmic heuristic followed to identify such sets.

The methods without global models typically employ a bottom-up strategy to find communities. They often start by defining the properties of a node, a pair of nodes or a group of nodes in a same community, and then search within a whole network for the communities that hold the proposed properties [10]. A network’s global community structure is detected by considering the ensemble of communities obtained by looping over all of these local structures. For example, the method of k-clique percolation [18] is based on the concept of k-clique, and a k-clique community is then defined as the union of all ‘adjacent’ k-cliques, which by definition share k − 1 nodes. Besides k-clique, a community could be regarded as a clique, a k-club [34], a quasi-clique [35], an equivalent structure or the combination of node pairs that have nodes similar to each other, as measured by, for example, Jaccard coefficient or cosine similarity [1].

To elaborate all the related work on community detection is definitely impossible. Readers with this interest may refer to some excellent books and survey papers [1, 3]. In recent years, researchers realize that genuine communities are actually hidden inside the massive networks, and thus propose to extract communities from the networks. Borgatti and Everett [36] tried

6. CONCLUDING REMARKS

This paper proposed a novel framework named COSCOM for community extraction from massive social networks. COSCOM can efficiently extract tight groups called AESs from networks using a novel CoPaMi algorithm: CoPaMi. In addition, any existing methods can be incorporated into COSCOM for the partitioning of extracted nodes, which makes COSCOM flexible in dealing with networks of various domains. Finally, the effectiveness of COSCOM was validated by experiments on four real-world social networks, on both internal and external measures.

FUNDING

This research was partially supported by NSFC under Grants 71072172, 61103229 and 71372188, National Key Technologies R&D Program of China under Grant 2013BAH16F00, Key Project of Natural Science Research in Jiangsu Provincial Colleges and Universities under Grant 12KJA520001 and the Natural Science Foundation of Jiangsu Province of China under Grant BK2012863. Dr J. Wu was supported in part by NSFC under Grants 71171007, 70901002 and 71031001, by the National Information Security Research Plan of China under Grant No. 2012A137, by the Foundation for the Author of National Excellent Doctoral Dissertation of PR China under Grant 201132 and by the Program for New Century Excellent Talents in University under Grant NCET-11-0778.

REFERENCES