Temporal motifs reveal collaboration patterns in online task-oriented networks

Qi Xuan,[1,2,*] Huiting Fang,[1] Chenbo Fu,[1,2,†] and Vladimir Filkov[2,‡]
[1]*Department of Automation, Zhejiang University of Technology, Hangzhou 310023, China*
[2]*Department of Computer Science, University of California, Davis, California 95616, USA*

(Received 10 October 2014; published 22 May 2015)

Real networks feature layers of interactions and complexity. In them, different types of nodes can interact with each other via a variety of events. Examples of this complexity are task-oriented social networks (TOSNs), where teams of people share tasks towards creating a quality artifact, such as academic research papers or software development in commercial or open source environments. Accomplishing those tasks involves both work, e.g., writing the papers or code, and communication, to discuss and coordinate. Taking into account the different types of activities and how they alternate over time can result in much more precise understanding of the TOSNs behaviors and outcomes. That calls for modeling techniques that can accommodate both node and link heterogeneity as well as temporal change. In this paper, we report on methodology for finding temporal motifs in TOSNs, limited to a system of two people and an artifact. We apply the methods to publicly available data of TOSNs from 31 Open Source Software projects. We find that these temporal motifs are enriched in the observed data. When applied to software development outcome, temporal motifs reveal a distinct dependency between collaboration and communication in the code writing process. Moreover, we show that models based on temporal motifs can be used to more precisely relate both individual developer centrality and team cohesion to programmer productivity than models based on aggregated TOSNs.

DOI: 10.1103/PhysRevE.91.052813 PACS number(s): 89.75.Hc, 89.75.Fb, 87.23.Ge, 89.75.Kd

I. INTRODUCTION

With the availability of electronic communication data on phone calls [1], emails [2], tweets [3], etc., social networks [4,5] have been extensively studied in recent years, and their significance in various social processes is now widely recognized [6–8]. More recently, the focus has been on task-oriented social networks (TOSNs), i.e., communities of people who are virtually organized around and working on a common goal; i.e., there is a typical wealth of other data recording their technical contribution. Examples of such communities are Open Source Software (OSS) projects [9], Wikipedia [10], Stack Overflow [11], etc., where people cooperate to create software, share knowledge, and provide quick and high-quality answers for different kinds of questions, respectively.

One of the more interesting phenomena observed in empirical network research is that social, biological, and technical networks share structural properties, such as small worldiness [12], scale freeness [13], and motifs richness [14,15]. Revealing global properties helps to capture a network as a whole, while identifying mesoscale motifs is important to understand its evolving mechanism in a bottom-up fashion [16,17]. Many efforts have been made to identify *motifs*, or small connected subgraphs, in complex networks. For example, Shen-Orr *et al.* [14] detected motifs in the transcriptional regulation network of *Escherichia coli* and found that each of the significant motifs has specific function in determining gene expression, while Valverde and Solé [18] found that the frequent motifs in software networks are more likely to be a consequence of network heterogeneity and size rather than software functionality.

Real complex networks are also dynamic or temporal. The nodes are connected via links representing discrete events [19,20], e.g., in social networks, individuals join and quit frequently and they communicate with each other at different times [21]; and in biological networks, there are sequences of activities to process gene regulation [22]. The sequence of these events have been shown to have important effects on many processes in the networks [23,24]. In fact, some sequences of links recur, forming temporal motifs. Identifying such temporal, or time-dependent, motifs in networks has been receiving attention lately. Braha and Bar-Yam [25] studied a series of static, or snapshot networks, by aggregating events over short time periods and counting observed temporal motifs in them. Bajardi and Dornhaus [26] considered temporal motifs as sequences of connected events belonging to adjacent time windows, and Kovanen *et al.* [27] considered two events Δt-connected if a sequence of events exists between them, satisfying that each pair of the consecutive events have at least one node in common and the time interval between them is no longer than Δt. Then they studied connected temporal motifs consisting of Δt-connected events. These studies address the time dimension of complex networks, and their results indicate that motifs in the overall aggregate networks are always overrepresented, leading to inflated results.

In this paper, we focus on identifying temporal motifs in TOSNs in order to reveal temporal collaboration between people working on the same artifact. In particular, we consider our networks as containing, in the simplest case, two types of nodes: people (P) and artifacts (A), and two types of links between them: people working on artifacts (P → A) and people communicating with others (P → P).[1] Then, for a

[*]crestxq@hotmail.com
[†]cbfu@zjut.edu.cn
[‡]filkov@cs.ucdavis.edu

[1] These two kinds of activities might not be independent; in fact, we have shown that communication is vital to make the temporal collaboration effective [28].

temporal collaboration to occur, it is necessary that two people work on the same artifact(s), e.g., $P_1 \rightarrow A$, and $P_2 \rightarrow A$, at close enough points in time. However, this is not sufficient. To observe the dependency between these two different kinds of activities and understand the evolution of a TOSN as a whole, the respective collaboration and communication networks should be considered simultaneously. This motivates our current work where the challenge is to develop novel methods to identify task-related temporal motifs, involving two people, an artifact, and the two activities of working on the artifact or communicating. With such methods we hope to reveal the specific roles of communication in temporal collaboration. Such temporal motifs can also be used to filter accompanying noisy information, such as noncollaborative activities and independent communication activities, so as to make the relationship between communication and collaboration more distinct and to suggest appropriate layered network models [29–33]. This approach can be further used to help reveal latent team structures with higher confidence, which is considered to be strongly associated with individual and group performance [34–37].

The rest of the paper is organized as follows. In Sec. II we introduce TOSNs and the data set collected from OSS projects. In Sec. III we propose a method to identify temporal motifs in TOSNs and use a null model to generate random networks for comparison. In Sec. IV we use the temporal motifs to filter structural noisy information so as to reveal the distinct relationship between collaboration and communication. We then apply these motifs to visualize the temporal interactions between individuals. In Sec. V we use these temporal motifs to reveal the team structure and adopt their structural properties to model individual and team productivity. Finally, the paper concludes in Sec. VI.

II. TASK-ORIENTED SOCIAL NETWORKS

A TOSN usually contains different types of nodes and links, i.e., people collaborate to produce different kinds of artifacts, such as movies [12], music [38], scientific papers [39,40], software [41–43], etc.; they may also communicate with each other to coordinate their work through different media. Such a network is more general than bipartite networks [1,44] and multiplex networks [29–33], where only the type of nodes and the type of links vary, respectively. For simplicity, here we mainly focus on the basic TOSN with two types of nodes; i.e., a group of people (P) work on artifacts (A), and they communicate with each other, as shown in Fig. 1. Such a basic TOSN is sufficient to provide the simplest framework to study interactions between communication and collaboration, thus enabling the quantitative study of sociotechnical systems [45–47]; a more complex TOSN can be considered as a natural collection of basic ones; i.e., more types of artifacts and more communication channels can be added into the framework to study interactions among them at finer resolutions.

Here a TOSN is denoted by $G = \{X, Y, E_X, E_Y, E_B\}$, where $X = \{x_1, x_2, \ldots, x_N\}$ and $Y = \{y_1, y_2, \ldots, y_M\}$ are the node sets containing two types of nodes, people and artifacts, respectively; E_X, E_Y, and E_B are the link (or edge) sets where E_X is the set of communication links between people (P → P), E_Y is the set of dependency links between artifacts (A → A), and E_B is the set of links from people to artifacts (P → A). Each link in the TOSN is represented by a sequence of time stamps at which the series of interacting events between the two incident nodes were recorded.

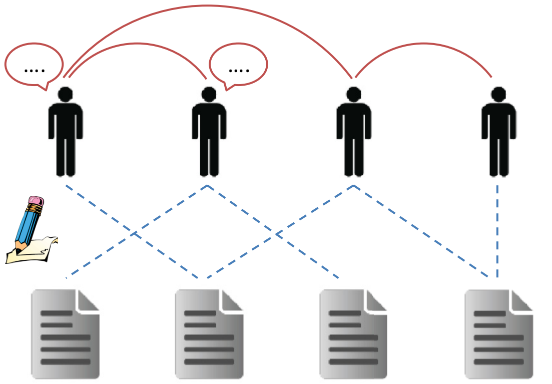


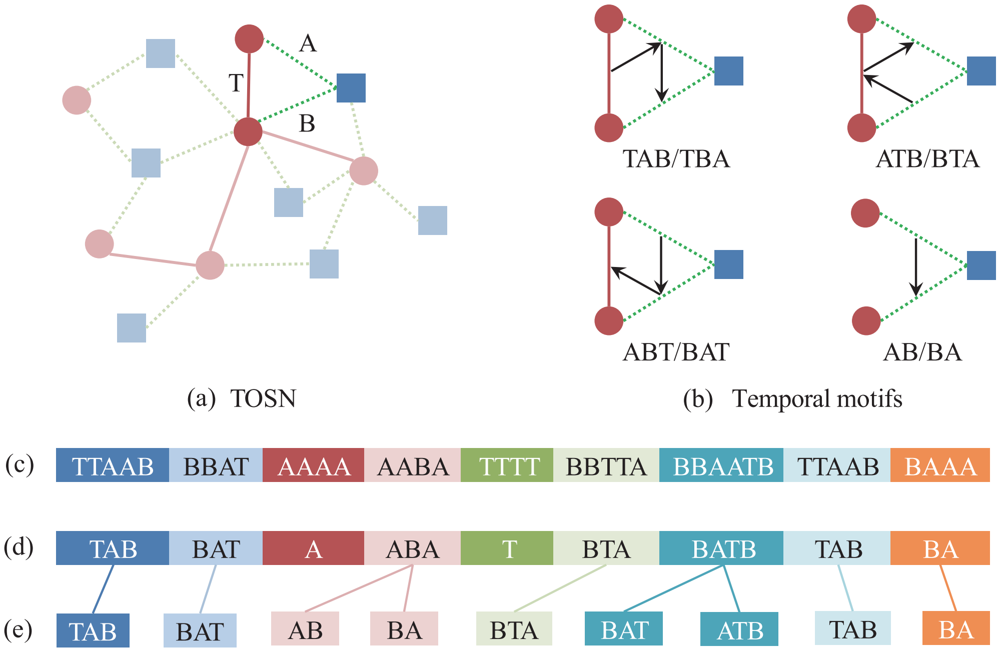
FIG. 1. (Color online) Sketch of a typical TOSN, with two types of nodes and links, i.e., a group of people collaborating to process a number of artifacts, while they communicate with each other to coordinate their work.

In this paper, we focus on Open Source Software (OSS) projects TOSNs, as examples on which to introduce and study temporal motifs. In each OSS project, there are a group of developers whose work activities are in large part committing code changes to the artifact files, and they communicate with each other through emails. We choose OSS because in addition to the availability of their work activities, the developer discussions via emails are also archived and often meaningfully related to the work activities [37].

We collected commit and email communication activities of developers from 31 OSS projects from the Apache Software Foundation on March 24, 2012, from which we obtain the TOSNs, one for each project. For each project, email communications were mined from the developer mailing lists, while commit activities are gathered from the corresponding Git repository [48]. Note that messages may be automatically posted to a mailing list in an OSS community to inform others when some work is completed. To exclude such trivial communication activities, we consider only response emails [37]. We also use a semiautomatic approach to solve the problem of multiple aliases which some developers use [48].

III. TEMPORAL MOTIFS IN A TOSN

The simplest meaningful motif in a network is a triangle involving three nodes, the richness of which is used to quantify the local clustering of the network [49]. For every triple of nodes in a TOSN denoted by G, there are in total four combinations: $\{x_i, x_j, x_k\}$, $\{y_i, y_j, y_k\}$, $\{x_i, y_j, y_k\}$, $\{x_i, x_j, y_k\}$. For the first two cases, the motifs are the same as those in the networks containing only one type of node and thus will not be considered here. For the third case, since the dependencies between files are relatively stable when compared to the other two types of links, the temporal three-motifs mean that a developer tends to process related artifacts at successive time by following a random walk on the file dependency network, which has been extensively studied in both theory and practice [50,51] and thus will also be excluded in this



(a) TOSN (b) Temporal motifs

FIG. 2. (Color online) (a) A TOSN with six developers (red circles) and seven files (blue squares). (b) Four typical temporal motifs: plan before temporal collaboration (TAB/TBA), call for participation (ATB/BTA), discussion after temporal cocommits (ABT/BAT), and temporal cocommits without communications by email (AB/BA). (c) The ordered sequence of activities, which is divided into a subsequence at the interactivity intervals larger than a given threshold θ. (d) The subsequences after combining the same successive elements. (e) Counting temporal motifs in the resulting subsequence.

study. The fourth case will be the focus of this paper: temporal three-motifs indicating a coevolving pattern between collaboration and communication of two developers working on the same file. Since the motifs on the combination $\{x_i,x_j,y_k\}$ do not involve the dependency links between artifacts, we set $E_Y = \emptyset$ in the rest of the paper.

A. Mathematical definitions

Here we define activity sequences among the nodes $\{x_i,x_j,y_k\}$ in G, as defined above. We denote by “T” a communication activity between developers x_i and x_j, by “A” a commit activity of x_i, and by “B” a commit activity of x_j on file y_k, respectively, as shown in Fig. 2(a). Then, over time, the activities over these three nodes form an ordered sequence $S = (s_1,s_2,\ldots,s_H) \in \{\text{T,A,B}\}^H$, with element s_h occurring at time t_h, and satisfying $t_h < t_{h+1}$ for $h = 1,2,\ldots,H-1$.

There are six different temporal activity triples, corresponding to the three element permutations of {T,A,B}:TAB, TBA, ATB, ABT, BTA, and BAT. TAB, e.g., means the two developers x_i and x_j communicate with each other first, perhaps planning or coordinating future actions, after which x_i commits to the file y_k and then x_j follows suit. Since the commit activities A and B have no precedence in time over each other, i.e., are topologically equivalent, these six triangles can be combined into the following three temporal motifs: TAB and TBA, denoted by M_1, meaning planning followed by temporal collaboration; ATB and BTA, denoted by M_2, meaning call for participation; and ABT and BAT, denoted by M_3, meaning discussion after temporal cocommits. Such temporal motifs can then be used to quantify the coevolving patterns of collaboration and communication between developers in a TOSN. In order to investigate more comprehensively the role of email communication in a collaboration, we include a fourth motif, representing AB and BA, denoted by M_4, meaning that two developers might temporally cocommit to the same file without communicating by email. All these motifs are visualized in Fig. 2(b).

B. Identification of temporal motifs

Given a sequence of activities for a triple of nodes $\{x_i,x_j,y_k\}$, we consider long time intervals in the sequence to be natural separators between series of temporal motifs, since a discussion or code commits are more likely occur as bursts of communications or commits [28]. We reconstruct their temporal motifs as follows.

(1) *Divide.* Given a sequence $S = (s_1,s_2,\ldots,s_H)$ of activities occurring at successive times t_i, we divide them into subsequences defined by instances of s_i and s_{i+1} separated by large time intervals, i.e., $t_{i+1} - t_i > \theta$, some predefined threshold, as shown in Fig. 2(c).

(2) *Combine.* For each subsequence, we combine repeated successive activities, as shown in Fig. 2(d). We denote by $F = (f_1,f_2,\ldots,f_P)$ the resulting subsequence, satisfying $f_i \neq f_{i+1}$ for $i = 1,2,\ldots,P-1$.

(3) *Count.* We count the occurrence of our four motifs above, $M_1,\ldots,M_4$, by looking at consecutive, nonoverlapping tuples or triples in F, as shown in Fig. 2(e). The tuples AB or BA are counted as motif M_4 only if not preceded or followed by T, in which case they will be counted as M_1 or M_3.

We note that we only consider two- or three-motifs throughout the paper, but the method proposed above can be directly extended to identify longer motifs.

C. Null model for comparison

In a static complex network containing only one kind of nodes, the significance of a motif is always validated by comparing with the random networks generated by a null model, as conditionally randomized versions of the real network [14,15,18,52], e.g., random networks with the same degree sequence. A motif is considered structurally significant only when its count in the real network is statistically larger than those in the random networks. The situation is more complicated in temporal networks, where the time dimension is not projected out and thus the successive activities in a temporal motif must be near each other in time. In this case, the randomized reference can be obtained by time shuffling [27], i.e., randomly exchanging the time stamps of events but keeping the structure of the aggregate network the same, to observe the temporal correlations among events.

In temporal TOSNs, we focus on motifs that indicate temporal correlations among different kinds of individual activities. Such temporal correlations can be found in both the network structure and the time series of the activities. In reality, however, the network structure may depend on other factors, such as the order in which individuals and artifacts joined the system (e.g., developers do not commit to files that were created after they had left the project) and the dependency between artifacts (e.g., developers are more likely to commit to dependent files). The rewiring process that generates the null background can potentially break down such orders and dependencies and thus might make the null model inappropriate. Therefore, here we would like to create a null model by only shuffling the time intervals between activities under certain conditions, rather than change the network structure randomly, so that most statistical properties of the real network will be kept in the process and the resulting aggregate network will be the same as the observed one.

Since there are two kinds of activities, commits and communications, in a TOSN the temporal correlation between them then can be discharged by shuffling the time intervals in only one of these activities. Here we adopt this approach, proposed originally in our previous work [37], to shuffle the time intervals between commit activities for each developer, on each file, via the following three steps:

(1) *Initialization.* For each commit link, let there be U commit activities occurring at times $\tau_1,\tau_2,\ldots,\tau_U$, with $\tau_i < \tau_{i+1}$ for $i = 1,2,\ldots,U-1$. Then we obtain an ordered sequence of interactivity time intervals between them, denoted by $\Delta\tau_i = \tau_{i+1} - \tau_i, i = 1,2,\ldots,U-1$.

(2) *Shuffling.* We randomly rearrange (permute or shuffle)[2] the $U-1$ time intervals and obtain a new sequence of time intervals, denoted by $\Delta\pi_i$, $i = 1,2,\ldots,U-1$. This essentially generates random orderings of idling periods for the developer on the current file, but ensures that the distribution of these idling periods are exactly the same as actually observed.

(3) *Welding.* We weld these time intervals in the new order, one by one, to obtain a new sequence of commit activity occurrence time, denoted by $\pi_1,\pi_2,\ldots,\pi_U$, satisfying

$$\begin{aligned}\pi_i &= \tau_i, \quad i = 1,\\ \pi_i &= \pi_{i-1} + \Delta\pi_{i-1}, \quad i \geqslant 2.\end{aligned} \tag{1}$$

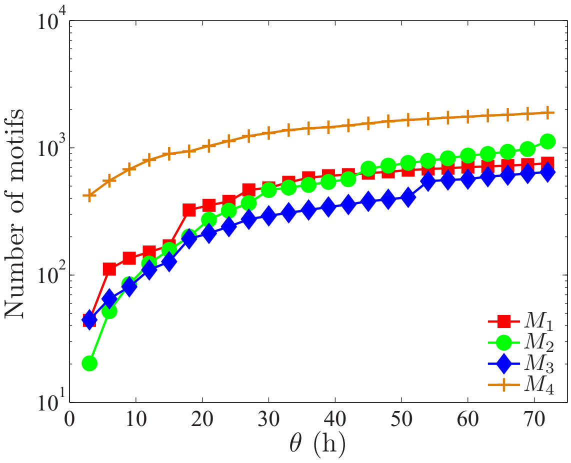


FIG. 3. (Color online) The average numbers of three-motifs M_1, M_2, and M_3 and two-motifs M_4 for the 31 TOSNs at different time threshold θ, varying from three hours to three days.

Note that this null model will not change the number of commit activities and the whole period of development, i.e., from τ_1 to τ_U, for each developer on each file as well.

We use the same method as above to identify the motifs in the temporal networks created by this null model. For each real network, we generate 100 random networks. Denote by λ_k the number of M_k motifs in the observed network and by λ_k^* the average number of M_k motifs and by σ the standard deviation among them in the random networks. Then the significance of a motif is measured by its Z-score [18], defined as

$$\chi_k = \frac{\lambda_k - \lambda_k^*}{\sigma}. \tag{2}$$

We also compare the occurrence of the four different motifs among themselves, in order to see which motif is preferred in the observed network, and thus address the role of communication in temporal collaboration.

D. Significance of temporal motifs

By applying these methods to the 31 observed projects collected from Apache Software Foundation, we get the numbers of three-motifs M_1, M_2, and M_3 and two-motifs M_4 in these networks. The average number of these motifs (over all 31 TOSNs) at different time thresholds θ are shown in Fig. 3. We find more temporal motifs M_1 than M_3 in most projects, indicating that developers prefer to talk prior to, rather than following, bursts of commit activity. Using Student's t test, we find the difference is quite significant ($p = 1.32\text{e}-11$), when considering all the cases with different time thresholds $\theta = 3,6,\ldots,72$ (h) together. It should be noted that here the number of the same motif across different projects has a relatively large standard deviation, e.g., about 795 for M_1 when $\theta = 24$ (h). This is mainly because these projects are of quite different size; e.g., there are 72 developers in Axis2_java, while there are only three developers in Bookkeeper.

[2] For example, by using the *sample*() function in R.

We generate random TOSNs as null model references and then calculate Z-scores for the three-motifs M_1, M_2, and M_3 with Eq. (2). We find that the Z-scores of the three-motifs in these networks under various time threshold are always positive, and most of them (87% for M_1, 90% for M_2, and 79% for M_3) are larger than two, indicating that the temporal three-motifs are more abundant than expected by chance. We also find that the counts of the two-motif M_4 in 28% of the cases are smaller than random,[3] and developers avoid temporally cocommitting with each other without coordinating. These results suggest that temporal collaboration is an important emergence in these TOSNs and confirm that communication plays an important role in synchronizing the work of developers [28].

Interestingly, although the count for each three-motif is much larger in most observed TOSNs than in the random ones, the fractions of each three-motif out of all three-motifs are close to each other in the observed and random networks. This is expected since the time shuffling process does not change most statistical properties of the TOSNs. For instance, the shuffling process does not change the total active period length and the total numbers of communication and commit activities of any developer, and thus at least the three-motifs involving their first and last commit activities are kept the same in the random shuffling process.

IV. TEMPORAL MOTIFS AS FILTERS

As we know, in many cases, communication is vital to making collaboration effective, i.e., more communication is always needed when coordinate additional collaborative work [28]. This can be coarsely validated by calculating the correlation between aggregate collaboration and social networks, both of which can be obtained from the TOSN.

In OSS projects, two developers are linked in the aggregate collaboration network if they have ever committed to the same files. To each link we give a weight representing the shared number of times that they have committed to the same files. That is, suppose they committed to the same K files, denoted by f_i, $i = 1,2,\ldots,K$, and for each file f_i, the first developer commits α_i times and the second β_i times. Then we define the collaborative weight between them as

$$w_C = \sum_{i=1}^{K} \min\{\alpha_i,\beta_i\}. \tag{3}$$

For a pair of developers linked in the aggregate collaboration network we also define social weight as the number of response email messages between them through the mailing list in Apache, denoted by w_E; it equals zero if no such email communication is observed. Then, for each TOSN, we have a list of collaborative weights of links and a list of corresponding social weights, denoted by W_C and W_E, respectively, and we calculate their similarity as

$$R = \frac{\langle W_C, W_E\rangle}{\|W_C\|\|W_E\|}. \tag{4}$$

While we indeed find W_C and W_E are positively correlated, $R = 0.46 \pm 0.33$ for the 31 TOSNs, we also find many outliers: i.e., pairs of developers who work on a large number of the same files may seldom contact each other through email, while developers who communicate frequently with each other may work alone most of the time and have little overlap between the files to which they contribute. This phenomenon is reasonable, since online communities are always highly dynamic; e.g., many OSS projects last for more than 10 years and the volunteer developers join and quit frequently. Thus, to assert a collaborative relationship between two developers we need more than just evidence that they simply contributed to the same files; i.e., we need also to know whether their activities occurred close together in time [28]. Moreover, for the same pair of developers, some communications may be independent from collaborative activities, since they may be just chatting or sharing knowledge, rather than discussing how to solve a current problem in the project. These may introduce noise and make it challenging to reveal a distinct relationship between communication and collaboration when only using aggregate TOSNs.

The temporal motifs we propose here capture the temporal collaborations between developers, and the communication links in the three-motifs can be considered strongly associated with the temporal collaborations, since they happen close in time. In other words, these temporal motifs can be used to filter structural noise and thus can help identify collaboration and the associated communications with higher confidence. We connect these two- and three-motifs to establish an aggregate motif network for each OSS project, where two nodes are linked if they are connected in at least one temporal motif, with the link weight representing the times a link appears in all the motifs. Such a network can be used to visualize various temporal interactions between individuals, based on which we can understand them as a system. As examples, the aggregate motif networks for the six largest OSS projects are shown in Fig. 4 when the time threshold is set to $\theta = 3$ (h). It can be seen that a large part of collaborations are indeed accompanied with email communications between developers. Note that these networks are quite different from the aggregate TOSNs, since a number of commit and communication activities have been removed in the process of identifying temporal motifs.

In addition, for each project, based on the aggregate motif network, we obtain another pair of aggregate collaboration and social networks, the correlation between which can be calculated by the same method, as described by Eq. (4), for comparison. As expected, for most OSS projects, collaborative and social weights are positively correlated, no matter whether they are obtained from the aggregate TOSN or the aggregate motif network. By comparison, we find that, on average, the correlation coefficients between the two based on the aggregate motif networks are much larger (with significance of $p = 0$, per Student's t test) than those based on the aggregate TOSN, and the former is even larger than 0.8 when $\theta \geqslant 24$ (h), as shown in Fig. 5. This validates the utility of the temporal motifs in information processing; i.e., they can be treated as

[3] This indicates that the large number of two-motifs shown in Fig. 3 is consistent with a random phenomenon, i.e., developers committed to the same files *independently* at close times.

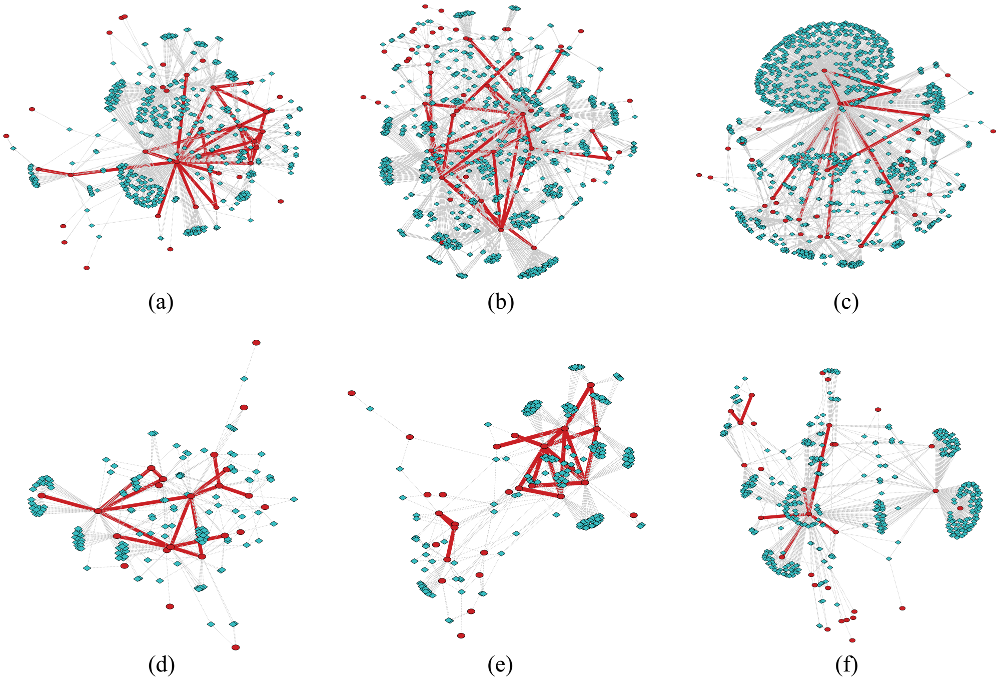


FIG. 4. (Color online) Aggregate motif networks for six OSS projects: (a) Ant, (b) Axis2_java, (c) Cxf, (d) Derby, (e) Lucene, and (f) Openejb, with time threshold set to $\theta = 3$ (h). Here the red circles represent developers and the blue diamonds represent files; the red solid lines represent the communications between developers, and the gray dashed lines represent the developer commits to files.

efficient information filters to identify temporal collaborations and further reveal the more distinct relationship between collaboration and communication.

Note that the collaborative activities of developers and the associated communications occur at close times, but almost never at exactly the same time; i.e., developers need some time to revise the code and reply to emails. As a result, the time threshold cannot be too small, since the counts of temporal motifs decrease very quickly as the time threshold θ approaches zero, resulting in the correlation R going to zero. On the other hand, the time threshold cannot be too large either, since it would result in temporally distant commits of two different developers being considered collaborative.

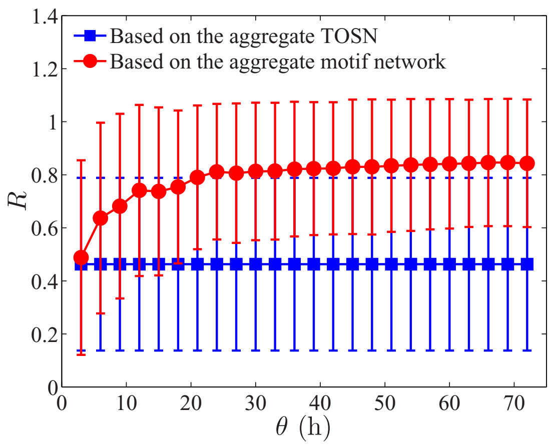


FIG. 5. (Color online) Correlation between collaborative and social weights, calculated by Eq. (4), obtained from the aggregate TOSN and the corresponding aggregate motif networks for all 31 OSS projects (the mean value and the error bars are shown), as functions of the time threshold θ.

V. IDENTIFYING TEAM STRUCTURE

Organizational properties of teams, such as their centralization and cohesion, may have significant effects on individual and group performance [34–37]. Teams in online TOSNs are typically self-organized, with volunteers who work on the projects remotely and come and go as they please. Identifying the structure of self-organized teams would, thus, be very valuable when studying their performance.

Due to the available trace data in OSS TOSNs, this can be done by reverse engineering the team compositions from the observations of peoples' activities. We proceed to do that here. To reveal a statistically meaningful relationship between network properties and individual and group performance, in the rest of the paper we consider only the most productive, or *top,* developers, having at least 100 commits, and keep only projects with at least five such developers. After this filtering, we were left with 21 projects and a total of 220 developers.

A conservative estimate of a team and their activities may be found in the overlap between their communication and collaboration networks. We consider that two developers collaborate if they commit to the same files and communicate with each other, around the same point in time. Given an aggregate TOSN and the temporal motifs for a group of developers, the following two networks over the top developers can capture their team structure:

TABLE I. The network properties, including average degree $\langle k\rangle$, average clustering coefficient $\langle CC\rangle$, and the heterogeneity H, of Γ_1 and Γ_2 for the six largest OSS projects. For Γ_2, the time threshold is set to $\theta = 12$ (h).

	$\langle k\rangle$		$\langle CC\rangle$		H	
Project	Γ_1	Γ_2	Γ_1	Γ_2	Γ_1	Γ_2
Ant	6.06	3.00	0.85	0.75	1.05	1.30
Axis2_java	7.88	2.23	0.85	0.73	1.16	2.02
Cxf	5.89	1.58	0.84	0.73	1.13	2.13
Derby	6.50	3.21	1.00	0.73	1.00	1.22
Lucene	4.73	2.45	0.95	0.73	1.01	1.10
Openejb	3.82	1.73	0.93	0.70	1.11	1.45

Network Γ_1: two developers are linked if they cocommit to at least one file and there is at least one email message between them.

Network Γ_2: two developers are linked if they form a temporal three-motif together with at least one file to which they have both committed code.

Note that network Γ_2 depends on the time threshold θ, and satisfies $\Gamma_2(\theta) \subseteq \Gamma_2(\theta + \Delta) \subseteq \Gamma_1$ for any positive θ and Δ, indicating that the links in Γ_2 with a smaller time threshold constitute a subset of the links in Γ_2 with a larger time threshold; also, the links in Γ_2 with any time threshold constitute a subset of the links in Γ_1.

For most of our projects, eventually almost every pair of top developers ends up communicating with each other and having cocontributed to same files. However, such nearly fully connected networks do not capture the real team structure of these projects, since, e.g., a pair of connected developers may have contributed to the same files and communicated with each other at very different times. The temporal motifs we propose here do not have that weakness and thus can better capture the collaboration between developers; hence, Γ_2 is more appropriate to describe the team structure. To address the difference between Γ_1 and Γ_2, and provide insights for future work in modeling team structure, we compare several of their global properties for the six largest OSS projects in Table I. We find that, after filtering those independent commit and communication activities, the team structure gets much sparser, while its clustering coefficient keeps relatively large for all the considered projects. This indicates that the developers in each of these projects tend to cluster together even when the average degree of the network is relatively low. Moreover, using the degree distribution, we define the heterogeneity[4] of a network as $H = \langle k^2\rangle/\langle k\rangle^2$. We find that Γ_2 is more heterogeneous than Γ_1, i.e., the individuals are more different in Γ_2, in terms of node degree.

These results are similar for varied time thresholds and suggest that both local and global topological properties of Γ_2 can be used to better characterize the individuals and the group, respectively, than those of Γ_1, and thus may be stronger predictors of their other properties, such as productivity.

A. Centrality and individual productivity

In a group, a person is central if he or she is the most popular and gets the most attention [36]. Centrality thus mirrors social status. A number of centrality measures have been devised, the most generic being the *degree* of a node in a social network [37]. More specific ones, e.g., *closeness* and *betweenness*, have been used to measure the centrality of a person in scientific collaboration networks [39]. There, the closeness of a node, defined as the average distance from the node to all other nodes, is a measure of information transmission from a person to all others, whereas betweenness is a measure of a person's control over information flowing between others. More generally, Borgatti and Everett [54] proposed a series of degree-like, closeness-like, and betweenness-like centralities by considering different kinds of network paths, or by expanding the definition of network distance. Rothenberg *et al.* [55] used eight such centrality measures to study the role of network structure in disease transmission. They found that although these measures differ in their theoretical formulation, they produce similar epidemiological results: noncentral persons are likely to be HIV positive in their low-prevalence social network. To measure centrality we use the normalized degree, $\widetilde{k}$, defined as the ratio of node degree to the maximum degree in a project network. We do this since we consider all top developers from all the projects together. In that setting, the normalized degree fixes the overall spread between node degrees across different projects. We choose node degree because it is easy to calculate and the definition is straight-forward in both connected and unconnected networks; also, in many cases, these measures of centrality are correlated with each other, and thus the use of alternatives may not influence the results very much. Note that, as suggested by Sarigöl *et al.* in their recent study on scientific coauthorship networks [56], although no single centrality measure could outperform the others, adopting a combination of many complementary notions of centrality has the potential to improve the precision of the model. This is partly because the social status in the collaboration network is multifaceted and thus can be reflected by different network measures.

To study the relationship between productivity [lines of codes (LoCs) per day] and centrality, with the number of commits, denoted by C, considered as a confound, we set up two multiple linear regression models for productivity of a developer as a function of the number of commits and of their centrality, one model for Γ_1 and another for Γ_2. We find that the models based on Γ_2 under various time thresholds are always better than those on Γ_1, although the multiple R^2 of the model is only slightly better, i.e., 0.2276 ± 0.0048 versus 0.2218. Note that this slight difference may still indicate quite different roles of centrality in the two models, since centrality may have totally different *relative importance* [we use the function *calc.relimp*() in R] in different cases; i.e., it explains different fractions of variance in the two models.

For example, the model and results for Γ_2 with the time threshold $\theta = 12$ (h) are shown in Table II [we used the function *lm*() in R]; the multiple R^2 equals 0.2333 and the residual standard error (RSE) equals 1.248. We can see that, when controlling for the number of commits, centrality $\widetilde{k}$ is a significant predictor in this model and its relative importance is 24.4%. By comparison, centrality is not significant in the

[4] This measurement is also important to determine the critical point to sustain reaction/epidemic activity on a network [53].

TABLE II. Multiple linear regression model for the individual productivity against the number of commits and the centrality in Γ_2 with $\theta = 12$ (h).

Variabes	Estimate	Std Error	z value	Pr(>|z|)
(Intercept)	1.3153	0.5530	2.379	0.0182
$\ln(C)$	0.6187	0.0995	6.215	2.59e-09
$\widetilde{k}$	0.6350	0.2899	2.191	0.0295

model based on Γ_1, and its relative importance is lower, 11.8%. In Table II the positive coefficient for $\widetilde{k}$ means that the effect of increased centrality is more LoCs per day, when the numbers of their commits are comparable. Note that here we log the productivity and the number of commits to stabilize the variance and improve the model fit.

B. Cohesion and team productivity

Next, we study the relationship between social connectivity, or cohesion, in a team and the team's productivity performance. To calculate a team's cohesion, we use a measure similar to that introduced by Yang and Tang [36]: *the ratio of positive mutual relationships to all possible ones*. Since most email communications in OSS are positive, e.g., encouraging others to do more extensive work [28], we approximate team cohesion by the link density in the network:

$$\phi = \frac{2}{n(n-1)} \sum_{i=1}^{n} k_i, \tag{5}$$

where n is the team size and k_i is the degree of node i. For each project, we also calculate the team productivity as the average productivity of the team members.

We use a multiple linear regression to model the average productivity of developers in a team as a function of cohesion, ϕ, in Γ_2,[5] while controlling for team size n and average number of commits C. Since developers have different commit and communication rhythms, to increase statistical power, we pooled all projects in Γ_2 with different time thresholds $\theta = 3,6,\ldots,72$ (h) into a single model. Note that varying time threshold only influences the cohesion but has no effect on team productivity; therefore, we expect pooling all data into a single model is appropriate, with little threat to the validation of the models. The results are shown in Table III, where the multiple R^2 equals 0.2681 and the RSE equals 0.56. We find that, while controlling for the average number of commits and team size, team cohesion has a significant positive effect on the average productivity of developers, indicating that developers in a more cohesive team will, statistically, have higher productivity.

These results about the relationship between network properties and individual and team productivity validate again the potential of the temporal motifs technology, i.e., the properties in the Γ_2 network can be used to better characterize individuals and groups in OSS projects, and, thus, might be appropriate to describe their team structures.

TABLE III. Multiple linear regression model for the average productivity in a team against the average number of commits, team size, and team cohesion, by considering all the cases in Γ_2 together.

Variable	Estimate	Std Error	z value	Pr(>|z|)
(Intercept)	2.4729	0.2969	8.330	7.80e-16
$\ln(C)$	0.4451	0.0478	9.317	<2e-16
n	0.0309	0.0054	5.684	2.24e-08
ϕ	1.0110	0.1417	7.137	3.37e-12

VI. CONCLUSION

In this study, we proposed a methodology for identifying temporal motifs composed of two people and an artifact in task-oriented social networks. Such temporal motifs can be used to filter out independent activities and thus help to identify temporal collaborations between people, find the associated communications, reveal the distinct dependency between the two, and further infer the latent team structure. The time threshold is always an important parameter in the studies of temporal networks. Here the appropriate time threshold is based on the Goldilocks approach: not too small, since people need some time to respond to an event; and not too large, since two events can hardly be considered as associated with each other if they are apart from each other for a long time. We thus vary it from three hours to three days.

We find that temporal motifs are an important emergence in OSS projects, i.e., the numbers of temporal motifs in real TOSN are always significantly larger than those in the randomized networks. More interestingly, based on the team structure inferred from the temporal motifs, we find that the more central individuals and the more cohesive teams are more productive. The results are less significant if we use the team structure derived from the aggregate TOSN instead. This is a validation for using temporal motifs, instead of an aggregate TOSN, to describe team structure.

Here we considered only the basic TOSN that contains two types of nodes and two types of links. However, people in general may use different communication tools, such as telephone, Twitter, MSN and so on, to chat or coordinate different kinds of work. More general nodes and links will certainly lead to more varied temporal motifs in TOSNs. More broadly, the proposed techniques can be easily generalized for use in other temporal networks, containing different types of nodes and different types of links, although the meaning of the temporal motifs might be domain specific. They can also be used to reason about the causal relationship between different kinds of activities, which can provide useful insights for the modeling of layered networks.

ACKNOWLEDGMENTS

The authors gratefully acknowledge support from the Air Force Office of Scientific Research, award FA955-11-1-0246, and the National Natural Science Foundation of China (Grants No. 61004097 and No. 61273212), the China Scholarship Council (CSC), and the China Postdoctoral Science Foundation (Grant No. 2014M551770).

[5] Model on Γ_1 had consistently poorer fit, i.e., none of the variables, including the number of commits, team size, and team cohesion, is significant in this case, so we omit it here.

[1] Q. Xuan, F. Du, and T. J. Wu, Chaos **19**, 023101 (2009).

[2] M. E. J. Newman, S. Forrest, and J. Balthrop, Phys. Rev. E **66**, 035101 (2002).

[3] A. Vespignani, Nat. Phys. **8**, 32 (2012).

[4] G. Kossinets and D. J. Watts, Science **311**, 88 (2006).

[5] S. P. Borgatti, A. Mehra, D. J. Brass, and G. Labianca, Science **323**, 892 (2009).

[6] J. M. Kleinberg, Nature (London) **406**, 845 (2000).

[7] P. S. Dodds, D. J. Watts, and C. F. Sabel, Proc. Natl. Acad. Sci. USA **100**, 12516 (2003).

[8] Q. Xuan and V. Filkov, in *Handbook of Human Computation* (Springer, New York, 2013), pp. 791–802.

[9] A. Mockus, R. T. Fielding, and J. D. Herbsleb, ACM Trans. Softw. Eng. Meth. **11**, 309 (2002).

[10] J. Giles, Nature (London) **438**, 900 (2005).

[11] B. Vasilescu, V. Filkov, and A. Serebrenik, in *Proceedings of 2013 International Conference on Social Computing* (IEEE, Alexandria, VA, 2013), pp. 188–195.

[12] D. J. Watts, *Small Worlds: The Dynamics of Networks between Order and Randomness* (Princeton University Press, Princeton, NJ, 1999).

[13] A.-L. Barabási, Science **325**, 412 (2009).

[14] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon, Nat. Genet. **31**, 64 (2002).

[15] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, Science **298**, 824 (2002).

[16] N. J. Guido, X. Wang, D. Adalsteinsson, D. McMillen, J. Hasty, C. R. Cantor, T. C. Elston, and J. J. Collins, Nature (London) **439**, 856 (2006).

[17] T. Kohonen, Biol. Cybern. **43**, 59 (1982).

[18] S. Valverde and R. V. Solé, Phys. Rev. E **72**, 026107 (2005).

[19] P. Holme and J. Saramäki, Phys. Rep. **519**, 97 (2012).

[20] R. K. Pan and J. Saramäki, Phys. Rev. E **84**, 016105 (2011).

[21] J.-P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, J. Kertész, and A.-L. Barabási, Proc. Natl. Acad. Sci. USA **104**, 7332 (2006).

[22] S. Lèbre, J. Becq, F. Devaux, M. P. H. Stumpf, and G. Lelandais, BMC Syst. Biol. **4**, 130 (2010).

[23] M. Starnini, A. Baronchelli, A. Barrat, and R. Pastor-Satorras, Phys. Rev. E **85**, 056115 (2012).

[24] R. Pfitzner, I. Scholtes, A. Garas, C. J. Tessone, and F. Schweitzer, Phys. Rev. Lett. **110**, 198701 (2013).

[25] D. Braha and Y. Bar-Yam, in *Adaptive Networks: Theory, Models and Applications*, ed. T. Gross and H. Sayama (Springer, New York, 2008), pp. 39–50.

[26] B. Blonder and A. Dornhaus, PLoS ONE **6**, e20298 (2011).

[27] L. Kovanen, M. Karsai, K. Kaski, J. Kertész, and J. Saramäki, J. Stat. Mech.-Theory Exp. (2011) P11005.

[28] Q. Xuan and V. Filkov, in *Proceedings of 36th International Conference on Software Engineering* (ACM, Hyderabad, India, 2014), pp. 222–233.

[29] Q. Xuan and T. J. Wu, Phys. Rev. E **80**, 026103 (2009).

[30] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin, Nature (London) **464**, 1025 (2010).

[31] J. Gómez-Gardeñes, I. Reinares, A. Arenas, and L. M. Floría, Sci. Rep. **2**, 620 (2012).

[32] S. Gómez, A. Díaz-Guilera, J. Gómez-Gardeñes, C. J. Pérez-Vicente, Y. Moreno, and A. Arenas, Phys. Rev. Lett. **110**, 028701 (2013).

[33] Q. Xuan, F. Du, L. Yu, and G. Chen, Phys. Rev. E **87**, 032809 (2013).

[34] S. Wuchty, B. F. Jones, and B. Uzzi, Science **316**, 1036 (2007).

[35] R. Guimerà, B. Uzzi, J. Spiro, and L. A. N. Amaral, Science **308**, 697 (2005).

[36] H. L. Yang and J. H. Tang, Inform. Manage. **41**, 335 (2004).

[37] Q. Xuan, M. Gharehyazie, P. Devanbu, and V. Filkov, in *Proceedings of 2012 International Conference on Social Informatics* (IEEE, Washington, DC, 2012), pp. 78–85.

[38] P. Gleiser and L. Danon, Adv. Complex Syst. **6**, 565 (2003).

[39] M. E. J. Newman, Phys. Rev. E **64**, 016131 (2001).

[40] M. E. J. Newman, Proc. Natl. Acad. Sci. USA **98**, 404 (2001).

[41] M. Pinzger and H. C. Gall, in *Collaborative Software Engineering* (Springer, Berlin, 2010), pp. 265–284.

[42] C. Bird, D. Pattison, R. D'Souza, V. Filkov, and P. Devanbu, in *Proceedings of 16th International Symposium on Foundations of Software Engineering* (ACM, Atlanta, 2008), pp. 24–35.

[43] C. Gutwin, R. Penner, and K. Schneider, in *Proceedings of 2004 Conference on Computer-Supported Cooperative Work* (ACM, Chicago, 2004), pp. 72–81.

[44] T. Zhou, J. Ren, M. Medo, and Y. C. Zhang, Phys. Rev. E **76**, 046115 (2007).

[45] D. H. Sonnenwald, Design Stud. **17**, 277 (1996).

[46] R. Kraut, C. Egido, and J. Galegher, in *Proceedings of 1988 Conference on Computer-Supported Cooperative Work* (ACM, Portland, 1988), pp. 1–12.

[47] D. Bertram, A. Voida, S. Greenberg, and R. Walker, In *Proceedings of 2010 Conference on Computer-Supported Cooperative Work* (ACM, Savannah, 2010), pp. 291–300.

[48] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan, in *Proceedings of 2006 International Working Conference on Mining Software Repositories* (ACM, Shanghai, 2006), pp. 137–143.

[49] L. D. F. Costa, F. A. Rodrigues, G. Travieso, and P. R. V. Boas, Adv. Phys. **56**, 167 (2007).

[50] Z. Zhang, Y. Qi, S. Zhou, W. Xie, and J. Guan, Phys. Rev. E **79**, 021127 (2009).

[51] Q. Xuan, A. Okano, P. Devanbu, and V. Filkov, in *Proceedings of 22nd International Symposium on the Foundations of Software Engineering* (ACM, Hong Kong, China, 2014), pp. 401–412.

[52] X. Liang, S. Yanchuk, and L. Zhao, Phys. Rev. E **88**, 012910 (2013).

[53] Q. Xuan, F. Du, T. J. Wu, and G. Chen, Phys. Rev. E **82**, 046116 (2010).

[54] S. P. Borgatti and M. G. Everett, Social Netw. **28**, 466 (2006).

[55] R. B. Rothenberg, J. J. Potterat, D. E. Woodhouse, W. W. Darrow, S. Q. Muth, and A. S. Klovdahl, Social Netw. **17**, 273 (1995).

[56] E. Sarigöl, R. Pfitzner, I. Scholtes, A. Garas, and F. Schweitzer, EPJ Data Sci. **3**, 9 (2014).