Finding multiple core-periphery pairs in networks

Sadamori Kojaku and Naoki Masuda[*]

*Department of Engineering Mathematics, Merchant Venturers Building, University of Bristol, Woodland Road,
Clifton, Bristol BS8 1UB, United Kingdom*

With a core-periphery structure of networks, core nodes are densely interconnected, peripheral nodes are connected to core nodes to different extents, and peripheral nodes are sparsely interconnected. Core-periphery structure composed of a single core and periphery has been identified for various networks. However, analogous to the observation that many empirical networks are composed of densely interconnected groups of nodes, i.e., communities, a network may be better regarded as a collection of multiple cores and peripheries. We propose a scalable algorithm to detect multiple nonoverlapping groups of core-periphery structure in a network. We illustrate our algorithm using synthesized and empirical networks. For example, we find distinct core-periphery pairs with different political leanings in a network of political blogs and separation between international and domestic subnetworks of airports in some single countries in a worldwide airport network.

I. INTRODUCTION

Many complex systems can be expressed as networks in which a node represents an object (e.g., person, web page, protein) and an edge represents the relationship between two objects (e.g., friendship, hyperlink, physical interaction). A network can be characterized by microscale, mesoscale, and macroscale structural patterns such as the degree (i.e., the number of edges that a node has), clustering coefficient, and diameter [1,2]. Among various structural properties of networks, community structure is a representative mesoscale structure of networks [3]. A community is a group of nodes that are densely interconnected and sparsely connected to nodes in different communities. Nodes in the same community often share a role [3–9] (for an exceptional case, see Ref. [10]), and therefore identifying communities aids classification of nodes and visualization of networks [3].

Core-periphery structure is another mesoscale structure of networks, with which we view a network as being composed of two groups of nodes called the core and periphery. Although the definition varies, a core is often defined as a group of densely interconnected nodes and a periphery as a group of nodes that are densely connected (i.e., adjacent) to the core nodes but not to other peripheral nodes [11–23]. A core and community are both groups of densely interconnected nodes but have a difference; a core connects densely to its periphery, whereas a community is not densely connected to other nodes outside it. Core-periphery structure has been found in various networks, including brain networks [24], metabolic networks [25], protein interaction networks [26], social networks [11,16,17,22], international trade networks [13,18,27], financial networks [15,28,29], and transportation networks [12,15,16]. For example, in a coauthorship network

among researchers, leading researchers often publish papers with other leading researchers, forming a core, while other researchers tend to publish papers with particular leading researchers such as those in the same research group, forming a periphery [16].

Borgatti and Everett introduced the first quantitative formulation of core-periphery structure [11]. In the discrete version of core-periphery structure, which we will focus on in this paper, they introduced an idealized core-periphery structure in which core nodes are adjacent to all other nodes, and peripheral nodes are adjacent to all core nodes but not to any peripheral nodes. Although it is also realistic to assume that the core-periphery connectivity is sparser than the core-core connectivity [11], we will focus only on the idealized core-periphery structure in the present study. Borgatti and Everett sought for the assignment of all nodes in a given network to a core or periphery such that the network is as close as possible to an idealized core-periphery structure. Following their study, many core-periphery detection algorithms have been developed [11–13,15–18,20–22,25,27]. These algorithms aim to identify a single core-periphery pair embedded in a network. However, a network may be better regarded as a collection of multiple core-periphery pairs [11,14,16,19–21], which is the focus of the present study. For example, coauthorship networks would be composed of multiple groups of researchers. Researchers would often collaborate with the leading researchers in the same group but not with other researchers in the same group, which may lead to core-periphery structure within the group [16]. Previous studies in this direction have not provided a tailored scalable algorithm to this end. A study focused on a related but different type of multiple core-periphery structure [30]. Other algorithms aim to detect multiple cores but do not assume that peripheral nodes are sparsely connected to each other [17,31,32]. A network can also have multiple disjoint cores in the form of k cores [33], k trusses [34], or dense subgraphs [35,36]. However, the corresponding algorithms do not tell how densely peripheral nodes are connected to each other or to which core a peripheral node belongs. An algorithm to find various mesoscale structure of networks including multiple core-periphery pairs [19] is computationally costly and only feasible for small networks (Appendix A).

---

[*]naoki.masuda@bristol.ac.uk

We present a scalable algorithm to detect multiple nonoverlapping core-periphery pairs in networks, each of which is as close as possible to an idealized core-periphery structure. Our algorithm automatically determines the number and the size of the core-periphery pairs. Various algorithms to detect core-periphery structure in networks are classified as density-based and transport-based algorithms [15,21,25]. Densely-based algorithms posit that a core is a densely connected group of nodes, whereas transport-based algorithms posit that a core is a group of nodes that can be reached from other nodes along short paths. In the present study, we focus on density-based algorithms.

II. METHODS

A. Algorithm

We extend the idealized core-periphery structure introduced by Borgatti and Everett [11] to the case of multiple pairs of a core and a periphery. In the Borgatti-Everett (BE) algorithm, one considers a graph (i.e., network) composed of N nodes and M edges. Let $A = (A_{ij})$ be the adjacency matrix, i.e., $A_{ij} = 1$ if node i and j are adjacent by an edge and $A_{ij} = 0$ otherwise. We assume an undirected and unweighted network without self-loops, i.e., $A_{ij} = A_{ji}$ and $A_{ii} = 0$ for all i and j. Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$ be a vector of length N, where $x_i = 0$ if node i is a peripheral node, and $x_i = 1$ if node i is a core node. We define the idealized core-periphery structure as the network where each core node is adjacent to all core and peripheral nodes, and each peripheral node is adjacent to all core nodes but no peripheral node. The corresponding adjacency matrix, $\boldsymbol{B}(\boldsymbol{x}) = [B_{ij}(\boldsymbol{x})]$, is given by

$$B_{ij}(\boldsymbol{x}) = \begin{cases} 1 & (x_i = 1 \text{ or } x_j = 1, \text{ and } i \neq j), \\ 0 & (\text{otherwise}). \end{cases} \quad (1)$$

The discrete version of the BE algorithm, which we consider in the present study, seeks $\boldsymbol{x}$ that maximizes similarity between A and $\boldsymbol{B}(\boldsymbol{x})$. We will explain the similarity measure in Sec. II C.

We extend the idealized core-periphery structure to the case of multiple core-periphery pairs. Let C be the number of core-periphery pairs and $\boldsymbol{c} = (c_1, c_2, \ldots, c_N)$ be a vector of length N, where $c_i \in \{1, 2, \ldots, C\}$ is the index of the core-periphery pair to which node i belongs. We exclude overlaps between core-periphery pairs, and between the core and periphery in each core-periphery pair. The corresponding adjacency matrix, $\boldsymbol{B}(\boldsymbol{c}, \boldsymbol{x})$, is given by

$$B_{ij}(\boldsymbol{c}, \boldsymbol{x}) = \begin{cases} \delta_{c_i, c_j} & (x_i = 1 \text{ or } x_j = 1, \text{ and } i \neq j), \\ 0 & (\text{otherwise}), \end{cases} \quad (2)$$

where δ is Kronecker delta.

We seek $(\boldsymbol{c}, \boldsymbol{x})$ that makes $\boldsymbol{B}(\boldsymbol{c}, \boldsymbol{x})$ the closest to A by maximizing

$$Q^{\text{cp}}(\boldsymbol{c}, \boldsymbol{x}) = \sum_{i=1}^{N} \sum_{j=1}^{i-1} A_{ij} B_{ij}(\boldsymbol{c}, \boldsymbol{x}) - \sum_{i=1}^{N} \sum_{j=1}^{i-1} p B_{ij}(\boldsymbol{c}, \boldsymbol{x})$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{i-1} (A_{ij} - p)(x_i + x_j - x_i x_j) \delta_{c_i, c_j}, \quad (3)$$
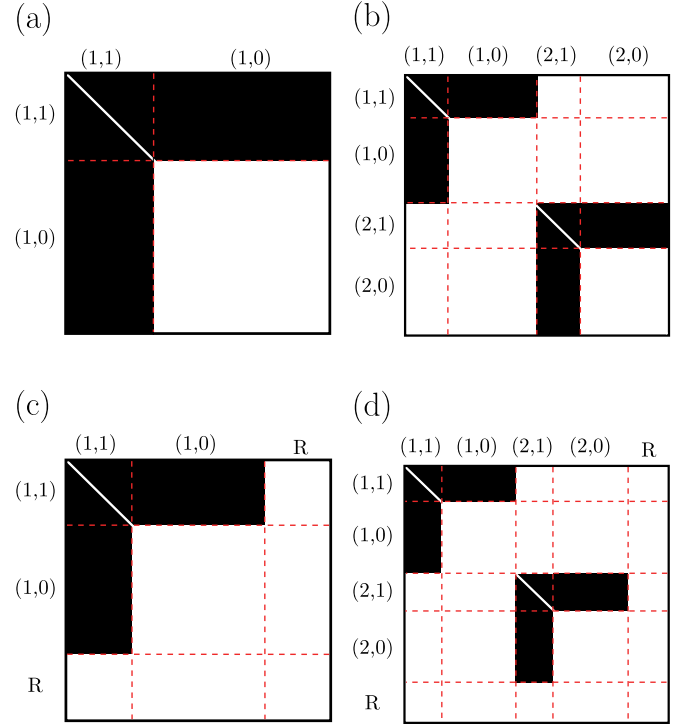

FIG. 1. Schematic illustrations of the adjacency matrices of the networks generated by stochastic block models. The filled blocks correspond to the entries that are equal to 1 with probability θ_1 and zero otherwise. The empty blocks correspond to the entries that are equal to 1 with probability θ_2 ($\theta_2 < \theta_1$) and zero otherwise. The diagonal entries are always set to zero and shown as empty entries in the figure for the sake of simplicity. The dashed lines indicate the borders separating different blocks. The labels $(\boldsymbol{c}, \boldsymbol{x})$ are also indicated at the top and left of the adjacency matrices. Label R represents a block of residual nodes. The networks are composed of (a) a single core-periphery pair, (b) two core-periphery pairs, (c) a single core-periphery pair and residual nodes, and (d) two core-periphery pairs and residual nodes. In all cases, we set $N = 400$.

where $p = M/[N(N - 1)/2]$ is the density of edges in the network. The term $\sum_{i=1}^{N} \sum_{j=1}^{i-1} A_{ij} B_{ij}(\boldsymbol{c}, \boldsymbol{x})$ represents the number of edges that are present in both the given network and the idealized core-periphery structure. The null-model term $\sum_{i=1}^{N} \sum_{j=1}^{i-1} p B_{ij}(\boldsymbol{c}, \boldsymbol{x})$ is the expected number of edges that are present in both the idealized core-periphery structure and an Erdős-Rényi random graph [37], in which each pair of nodes is adjacent with probability p. The Q^{cp} ranges between $-M$ and M. A large Q^{cp} value indicates that the given network and the idealized core-periphery structure share more edges than by chance. The Erdős-Rényi random graph model is widely used in the analysis of core-periphery structure [13,24,28,29,38,39]. Similar to modularity for community detection, our formulation permits the use of different null models such as the configuration model. See Sec. V for further discussion.

B. Maximization of Q^{cp}

We use a label switching heuristic [40,41] to maximize Q^{cp}. First, we assign each node to a different core by setting $(c_i, x_i) = (i, 1)$ $(1 \leqslant i \leqslant N)$. Then, we scan all nodes in a
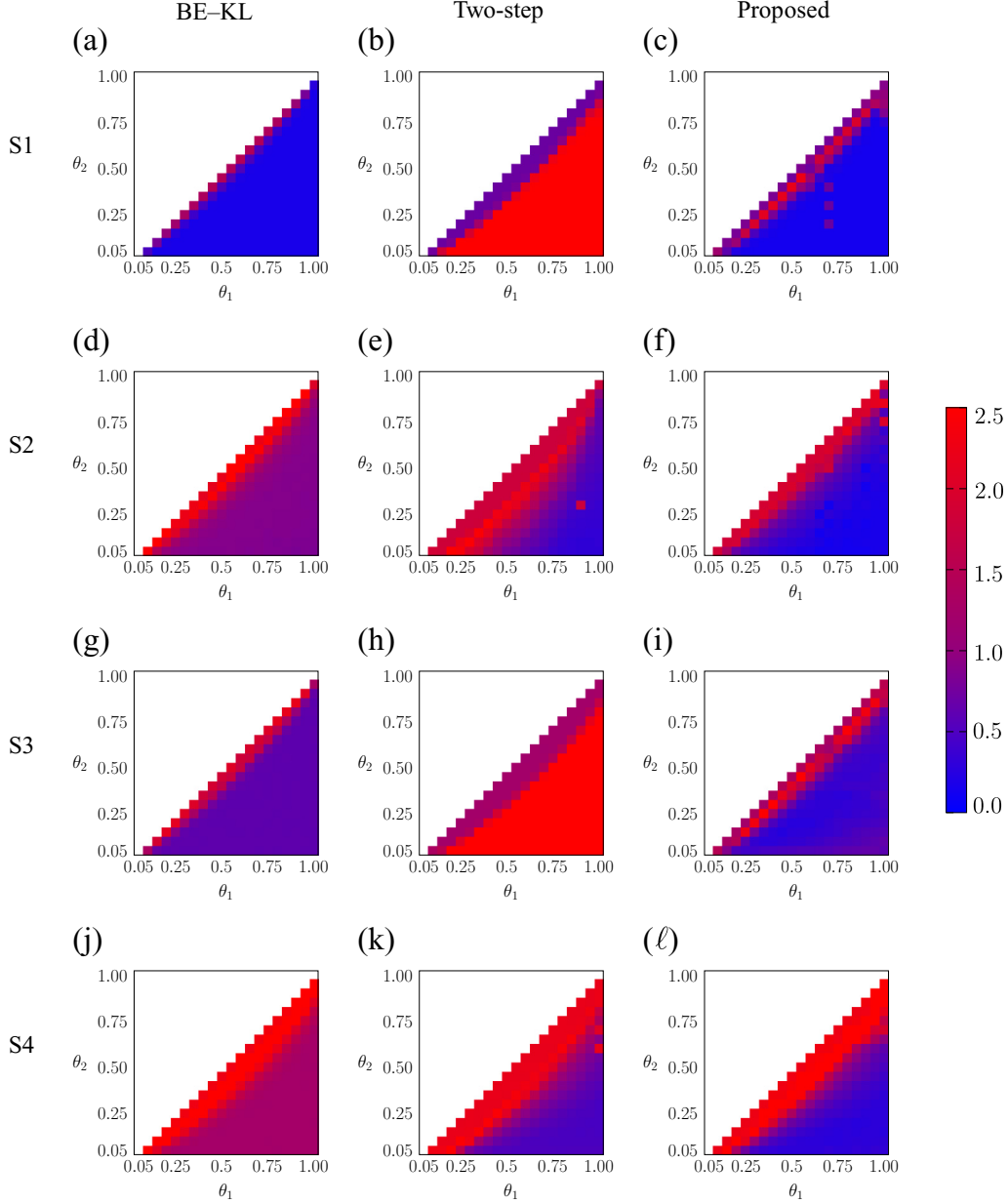
FIG. 2. The VI values between the true and inferred core-periphery structure for the three algorithms. Rows S1, S2, S3, and S4 correspond to the networks with planted core-periphery structure shown in Figs. 1(a), 1(b), 1(c), and 1(d), respectively. The color of each cell indicates the VI value. The white cells are those for which we did not calculate the VI values, i.e., we only computed them for $\theta_1 > \theta_2$.

random order. For each scanned node i, we calculate the increment in Q^{cp} when we tentatively update (c_i,x_i) to the core of the core-periphery pair that a neighbor of node i, denoted by j, belongs to, i.e., $(c_j,1)$. We also calculate the increment in Q^{cp} when we tentatively update (c_i,x_i) to $(c_j,0)$. Note that we experiment on these two cases regardless of whether $x_j = 0$ or $x_j = 1$. We carry out this procedure for all neighbors of i to measure the increment in Q^{cp} in each case. Finally, we update (c_i,x_i) to the label that has yielded the largest tentative increment in Q^{cp} [i.e., $(c_j,0)$ or $(c_j,1)$ for a neighbor j]. If any relabelling does not increase Q^{cp}, we do not update (c_i,x_i). When we have scanned all nodes, we stop the entire procedure if no node has changed its label in the present round. Otherwise, we draw a new random order of nodes and scan all nodes again according to the new random order.

The increment in Q^{cp} by changing node i's label from (c,x) to (c',x') is given by

$$\{\tilde{d}_{i,(c',1)} + x'\tilde{d}_{i,(c',0)} - p[\tilde{N}_{(c',1)} + x'\tilde{N}_{(c',0)} - \delta_{c,c'}]\}$$
$$- \{\tilde{d}_{i,(c,1)} + x\tilde{d}_{i,(c,0)} - p[\tilde{N}_{(c,1)} + x\tilde{N}_{(c,0)} - x]\}, \quad (4)$$

where $\tilde{d}_{i,(c,x)}$ is the number of neighbors of node i that have label (c,x), and $\tilde{N}_{(c,x)}$ is the number of nodes with label (c,x). For each scanned node i, we calculate Eq. (4) at most $2d_i$ times, where d_i is the degree of node i. Therefore, the time complexity for scanning all nodes in one round is $\mathcal{O}(\sum_{i=1}^N d_i) = \mathcal{O}(M)$, and that of the entire algorithm is $\mathcal{O}(rM)$, where r is the number of rounds. We run this algorithm 20 times starting from the same initial condition stated above and adopt the node labeling that produces the largest value of Q^{cp}.

C. Significance of the core-periphery structure

A detected core-periphery structure may be statistically insignificant [11,38]. Therefore, we adapt a statistical test in the case of a single core-periphery pair [38] to the case of multiple core-periphery pairs.

In the statistical test for a single core-periphery pair [38], we measure the significance of a core-periphery pair by a quality function based on the Pearson correlation coefficient [11], which is defined by

$$
Q_{\mathrm{BE}}^{\mathrm{cp}}
$$

$$
= \frac{\sum_{i=1}^{N} \sum_{j=1}^{i-1} (A_{ij} - p)(B_{ij}(\boldsymbol{x}) - p_B)}{\sqrt{\sum_{i=1}^{N} \sum_{j=1}^{i-1} (A_{ij} - p)^2} \sqrt{\sum_{i=1}^{N} \sum_{j=1}^{i-1} (B_{ij}(\boldsymbol{x}) - p_B)^2}},
$$

(5)

where $p_B = \sum_{i=1}^{N} \sum_{j=1}^{i-1} B_{ij}(\boldsymbol{x})/[N(N-1)/2]$. A core-periphery pair detected for the given network is deemed to be significant if $Q_{\mathrm{BE}}^{\mathrm{cp}}$ is statistically larger than $Q_{\mathrm{BE}}^{\mathrm{cp}}$ values calculated for the Erdős-Rényi random graph model, in which the number of edges is the same as that of the original network. One generates many networks using the Erdős-Rényi random graph and maximizes $Q_{\mathrm{BE}}^{\mathrm{cp}}$ for each network. The Kernighan-Lin (KL) algorithm [42] is used for maximizing $Q_{\mathrm{BE}}^{\mathrm{cp}}$. The core-periphery pair detected for the original network is significant at a significance level of $\alpha \in (0,1]$ if the $Q_{\mathrm{BE}}^{\mathrm{cp}}$ value for the original network is larger than a fraction $1 - \alpha$ of the $Q_{\mathrm{BE}}^{\mathrm{cp}}$ values for the randomized networks.

In the case of multiple core-periphery pairs, we apply essentially the same statistical test to each core-periphery pair detected in the original network. For each detected core-periphery pair, we first calculate $Q_{\mathrm{BE}}^{\mathrm{cp}}$. Second, we generate 3000 networks using the Erdős-Rényi random graph, which have the same number of nodes and edges as those of the core-periphery pair. In counting the number of edges, we only consider the edges connecting nodes within the core-periphery pair. Third, we detect a single core-periphery pair in each randomized network by maximizing $Q_{\mathrm{BE}}^{\mathrm{cp}}$ using the KL algorithm. Fourth, we compare the obtained $Q_{\mathrm{BE}}^{\mathrm{cp}}$ values between the original and randomized networks. If a core-periphery pair is judged to be insignificant, we call the corresponding nodes the residual nodes, i.e., those not belonging to any significant core-periphery pair.

If we test C core-periphery pairs at a significance level of α, then the probability of making at least one false positive (i.e., an insignificant core-periphery pair is judged to be significant) is $1 - (1 - \alpha)^C$, which increases as C increases. To remedy this multiple comparison problem, we adopt the Šidák correction, with which we test each core-periphery pair at a significance level of $\alpha_1 = 1 - (1 - \alpha)^{1/C}$, which is equivalent to $1 - (1 - \alpha_1)^C = \alpha$ [43]. We set $\alpha = 0.01$.

We have decided to use $Q_{\mathrm{BE}}^{\mathrm{cp}}$ maximized by the KL algorithm as the test statistic to compare the original and randomized networks. However, we can also use different algorithms to maximize $Q_{\mathrm{BE}}^{\mathrm{cp}}$. We can also use a different test statistic such as Q^{cp} restricted to the case of the one core-periphery pair (i.e., $c_i = 1, 1 \leqslant i \leqslant N$).

III. VARIATION OF INFORMATION

For the synthetic networks with planted core-periphery structure, we measure the difference between the true core-periphery structure $(\boldsymbol{c}, \boldsymbol{x})$ and the inferred core-periphery structure $(\hat{\boldsymbol{c}}, \hat{\boldsymbol{x}})$ by the variation of information (VI) [44]. The VI is given by

$$
\mathrm{VI} = -\sum_{(c,x)} \sum_{(\hat{c},\hat{x})} P(c,x;\hat{c},\hat{x}) \log
$$

$$
\times \frac{[P(c,x;\hat{c},\hat{x})]^2}{\left[\sum_{(\hat{c}',\hat{x}')} P(c,x;\hat{c}',\hat{x}')\right] \times \left[\sum_{(c',x')} P(c',x';\hat{c},\hat{x})\right]},
$$

(6)

where $P(c,x;\hat{c},\hat{x})$ is the fraction of nodes that have the true label (c,x) and inferred label $(\hat{c},\hat{x})$. The VI value is equal to zero if and only if the inferred core-periphery structure is the same as the true one. We measure the performance of an algorithm by averaging VI values over the 100 generated networks.

IV. RESULTS

We compare the proposed algorithm with the BE algorithm, which detects a single core-periphery pair by maximizing $Q_{\mathrm{BE}}^{\mathrm{cp}}$ using the KL algorithm [38,42]. We refer to the latter algorithm as the BE-KL algorithm. We also compare our algorithm with two other algorithms, two-step and divisive algorithms. The two-step algorithm partitions the nodes into core and peripheral nodes using the BE-KL algorithm and also partitions the same nodes into nonoverlapping communities by maximizing modularity using the Louvain algorithm [41]. Then we regard the core and peripheral nodes in each detected community as a core-periphery pair. The divisive algorithm [3–5,45] partitions the nodes into communities using the Louvain algorithm [41] and then partitions the nodes in each community into core and peripheral nodes using the BE-KL algorithm. The two-step and divisive algorithms provided similar results. Therefore, we report the results obtained from the two-step algorithm in this section and those obtained from the divisive algorithm in Appendix B. We apply the statistical test (Sec. II C) to the core-periphery pairs detected by the BE-KL, two-step, and our algorithms. We do not compare these algorithms with the algorithm introduced by Tunç and Verma [19] because of a low speed and insufficient performance of their algorithm on model networks with planted core-periphery structure (Appendix A).

A. Synthetic networks

We compare the performance of the three algorithms on four different types of synthetic networks with a planted core-periphery structure schematically shown in Fig. 1. We generate the synthetic networks using stochastic block models [9,16,20,21,46]. We draw label (c_i, x_i) for the ith node ($1 \leqslant i \leqslant N$) from a distribution $\pi_{(c,x)}$ ($1 \leqslant c \leqslant C$ and $x \in \{0,1\}$), where $\pi_{(c,x)} > 0$ is the probability that $(c_i, x_i) = (c,x)$ and satisfies $\sum_{c=1}^{C} \sum_{x=0,1} \pi_{(c,x)} = 1$. Then, we place edges between each pair of nodes with label (c,x) and (c',x') with probability $\Theta_{(c,x),(c',x')}$. For each type of the stochastic block
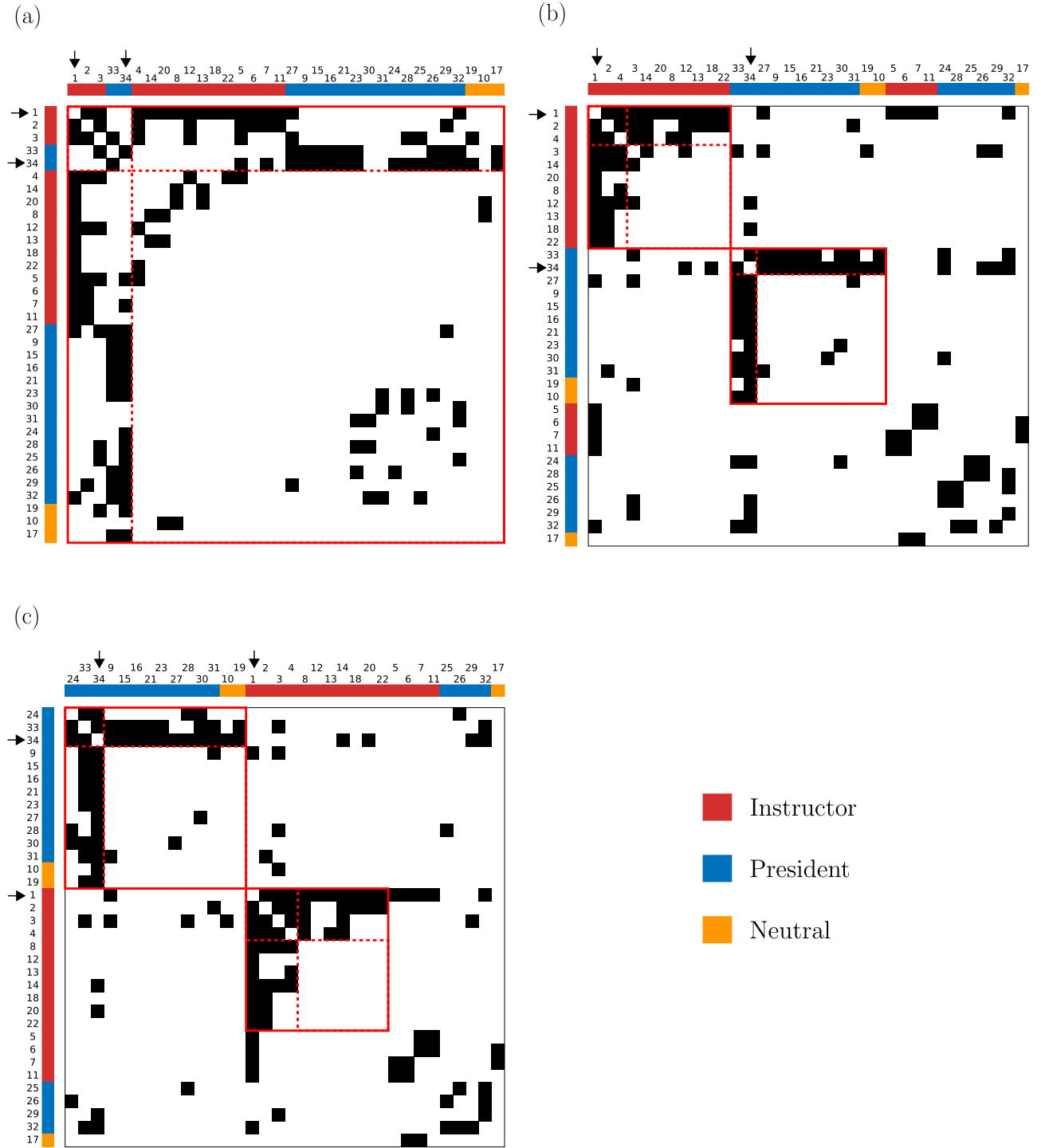
FIG. 3. Core-periphery structure of the karate club network detected by (a) the BE-KL algorithm, (b) the two-step algorithm, and (c) our algorithm. The filled and blank cells indicate $A_{ij} = 1$ and $A_{ij} = 0$, respectively. The solid lines indicate the partition into core-periphery pairs. The dashed lines indicate the partition into the core and periphery within a core-periphery pair. The color indicates the leaning of the members. The instructor (i.e., node 1) and president (i.e., node 34) are indicated by the arrows.

model, we generate 100 networks and detect core-periphery pairs by the three algorithms.

As a first example, we consider a network composed of a single core-periphery pair [Fig. 1(a)]. We set $\pi_{(1,1)} = 1/4$, $\pi_{(1,0)} = 3/4$, $\Theta_{(1,1),(1,1)} = \Theta_{(1,1),(1,0)} = \theta_1$, and $\Theta_{(1,0),(1,0)} = \theta_2$, where $\theta_1, \theta_2 \in \{0.05, 0.1, 0.15, \ldots, 1\}$ and $\theta_1 > \theta_2$. A generated network has strong core-periphery structure when

$\theta_1 \gg \theta_2$. If θ_1 is close to θ_2, then the generated network is close to the Erdős-Rényi random graph (i.e., noisy core-periphery structure). For this network model, the VI value, which quantifies the discrepancy between the true and inferred core-periphery structure, is compared between the three algorithms in Figs. 2(a)–2(c). The VI values for the BE-KL algorithm are approximately equal to zero in the entire parameter region
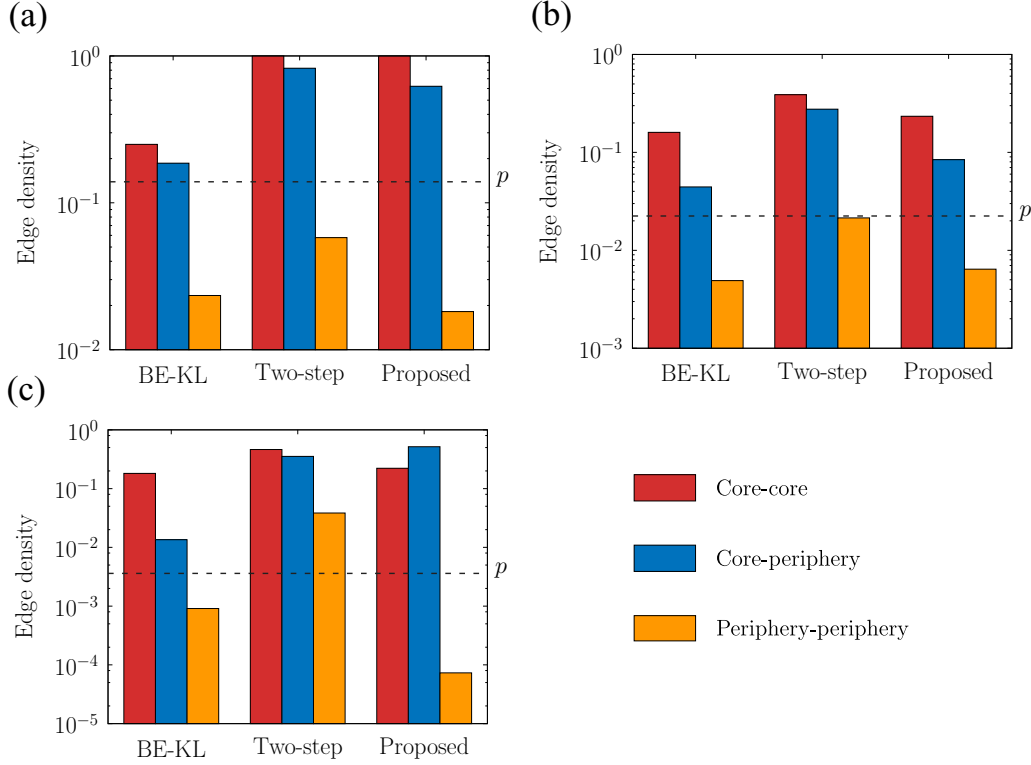
FIG. 4. Density of edges of different types within core-periphery pairs detected in (a) the karate club network, (b) blog network, and (c) airport network. The dashed line indicates the edge density for the entire network, p.

of θ_1 and θ_2. The VI values for the two-step algorithm are large even in the case of strong core-periphery structure (i.e., $\theta_1 \gg \theta_2$) because the two-step algorithm divides the single core-periphery pair into communities. In contrast, the VI values for the proposed algorithm are close to zero for most θ_1 and θ_2 values, as is the case for the BE-KL algorithm. Therefore, the performance of the proposed algorithm on this network model is comparable to that of the BE-KL algorithm.

As a second example, we examine networks composed of two core-periphery pairs [Fig. 1(b)]. We set $\pi_{(c,1)} = 1/8$, $\pi_{(c,0)} = 3/8$, $\Theta_{(c,1),(c,1)} = \Theta_{(c,1),(c,0)} = \theta_1$, and $\Theta_{(c,0),(c,0)} = \Theta_{(1,x),(2,x')} = \theta_2$ for $c \in \{1,2\}$ and $x,x' \in \{0,1\}$. The VI values for this network are shown in Figs. 2(d)–2(f). The VI values for the BE-KL algorithm are large in the entire parameter region of θ_1 and θ_2 because the BE-KL algorithm cannot find multiple core-periphery pairs. The VI values for the two-step algorithm are close to zero in the case of strong core-periphery structure (i.e., θ_1 is considerably larger than θ_2). The VI values for the proposed algorithm are smaller than those for the two-step algorithm for most θ_1 and θ_2 values, including the case of noisy core-periphery structure (i.e., when θ_1 is close to θ_2).

In empirical networks, there may be nodes that are better regarded not to belong to any core or periphery. Therefore, as a third example, we consider a network composed of a single core-periphery pair and residual nodes [Fig. 1(c)]. We regard the block of the residual nodes as a single group of nodes, like a core or periphery, when calculating the VI value. Let R be the index for the block of the residual nodes. We set

$\pi_{(1,1)} = \pi_R = 1/5$, $\pi_{(1,0)} = 3/5$, $\Theta_{(1,1),(1,1)} = \Theta_{(1,1),(1,0)} = \theta_1$ and $\Theta_{(1,0),(1,0)} = \Theta_{(1,x),R} = \theta_2$ for $x \in \{0,1\}$. The VI values for this model are shown in Figs. 2(g)–2(i). The VI values for the BE-KL algorithm are large even in the case of strong core-periphery structure. The VI values for the two-step algorithm are large in the entire parameter region of θ_1 and θ_2. The VI values for the proposed algorithm are smaller than those for the BE-KL and two-step algorithms for most θ_1 and θ_2 values, including the case of noisy core-periphery structure.

As a fourth example, we consider networks composed of two core-periphery pairs and residual nodes [Fig. 1(d)]. We set $\pi_{(c,1)} = \pi_R = 1/9$, $\pi_{(c,0)} = 1/3$, and $\Theta_{(c,1),(c,1)} = \Theta_{(c,1),(c,0)} = \theta_1$ for $c \in \{1,2\}$ and $\Theta_{(c,0),(c,0)} = \Theta_{(c,x),(c',x')} = \theta_2$, where $c = 1,2$, $c \neq c'$, and $x,x' \in \{0,1\}$. The VI values for this network model are shown in Figs. 2(j)–2(l). The VI values for the BE-KL algorithm are large in the entire θ_1-θ_2 parameter space. The VI values for the two-step algorithm are larger than those for the proposed algorithm for most θ_1 and θ_2 values.

B. Empirical networks

We apply the three algorithms to three empirical networks. For directed and weighted networks, we disregard the direction and the weight of edges.

1. Karate club network

Consider the karate club network [47], which has $N = 34$ nodes and $M = 78$ edges (edge density $p = 0.139$). A node
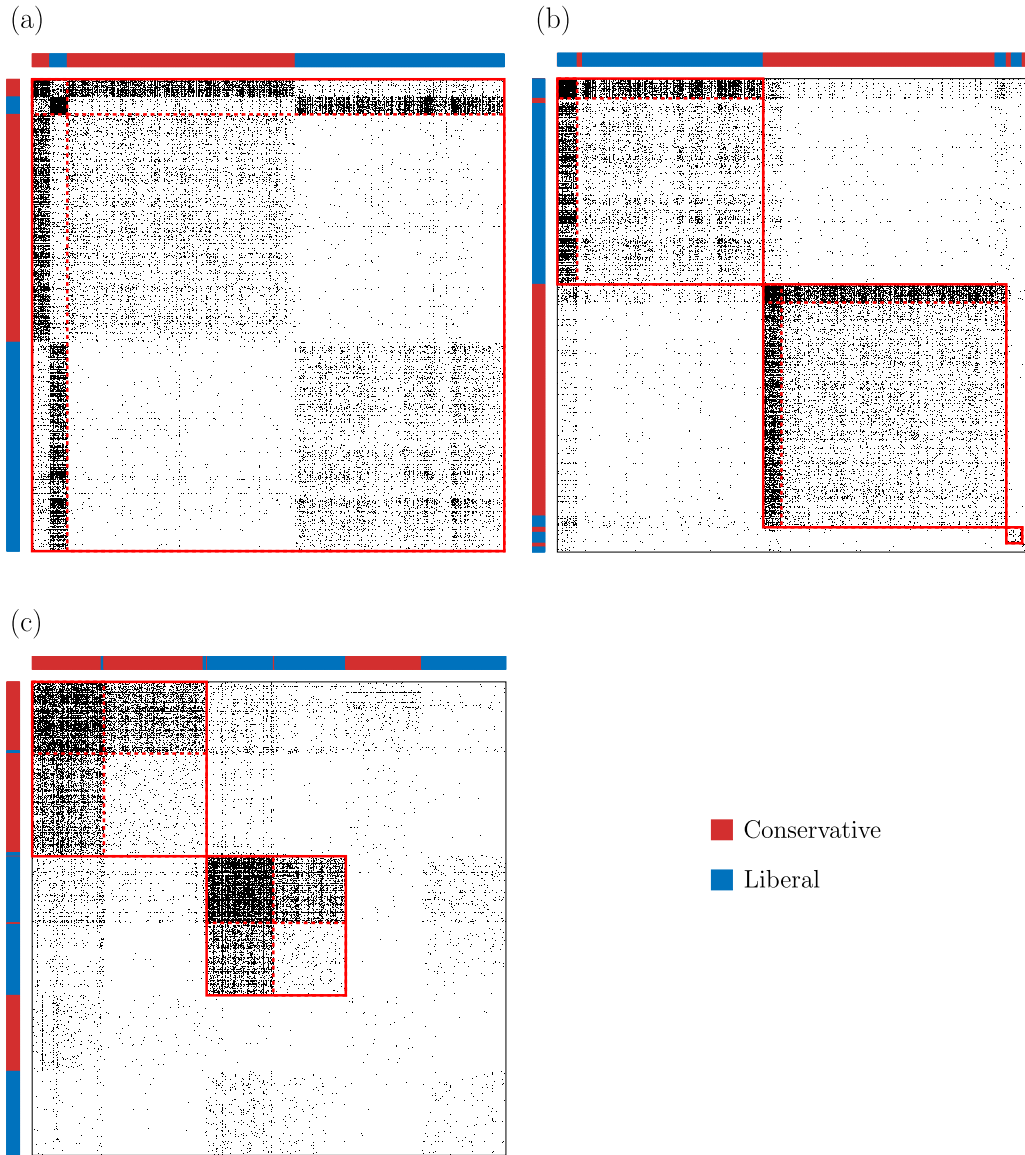
FIG. 5. Core-periphery structure of the blog network detected by (a) the BE-KL algorithm, (b) the two-step algorithm, and (c) our algorithm. The red and blue indicate the conservative and liberal blogs, respectively.

represents a member of a karate club at a university. Two members are adjacent if they have socially interacted outside club activities during the observation period. During the study, a conflict occurred between the instructor (node 1) and the president (node 34), which fissured the club. Based on self-reports, each member was labeled on the instructor's side (15 members), on the president's side (16 members), or neutral (3 members) [47].

The core-periphery structure detected by the three algorithms is shown in Fig. 3. The BE-KL algorithm detects a single core-periphery pair such that both the instructor and president are core nodes [Fig. 3(a)], neglecting the fissure of the club. The two-step algorithm detects two core-periphery pairs, each of which consists mostly of the members with the same leanings [Fig. 3(b)]. In particular, the instructor and the president belong to the core of the different core-periphery pairs. Two neutral members, nodes 10 and 19, are assigned to the president's core-periphery pair, which does not agree with

the self-reports by the members. The residual nodes consist of the members on the instructor's side, those on the president's side and a neutral member. Our algorithm detects almost the same two core-periphery pairs as that detected by the two-step algorithm [Fig. 3(c)].

Next, we compare the density of edges within core-periphery pairs. For each significant core-periphery pair, we compute the density of edges within the core, that of edges within the periphery and that of edges between the core and periphery. Then, we average each type of edge density over all significant core-periphery pairs (without weighing by the size of core-periphery pair when calculating the average).

We show the edge densities for the karate club network in Fig. 4(a). For all algorithms, the average density of intracore edges and that of core-periphery edges (i.e., edges connecting a core node and a peripheral node) are larger than the edge density for the entire network, $p = 0.139$. The average density of intraperipheral edges is smaller than $p = 0.139$. Therefore,
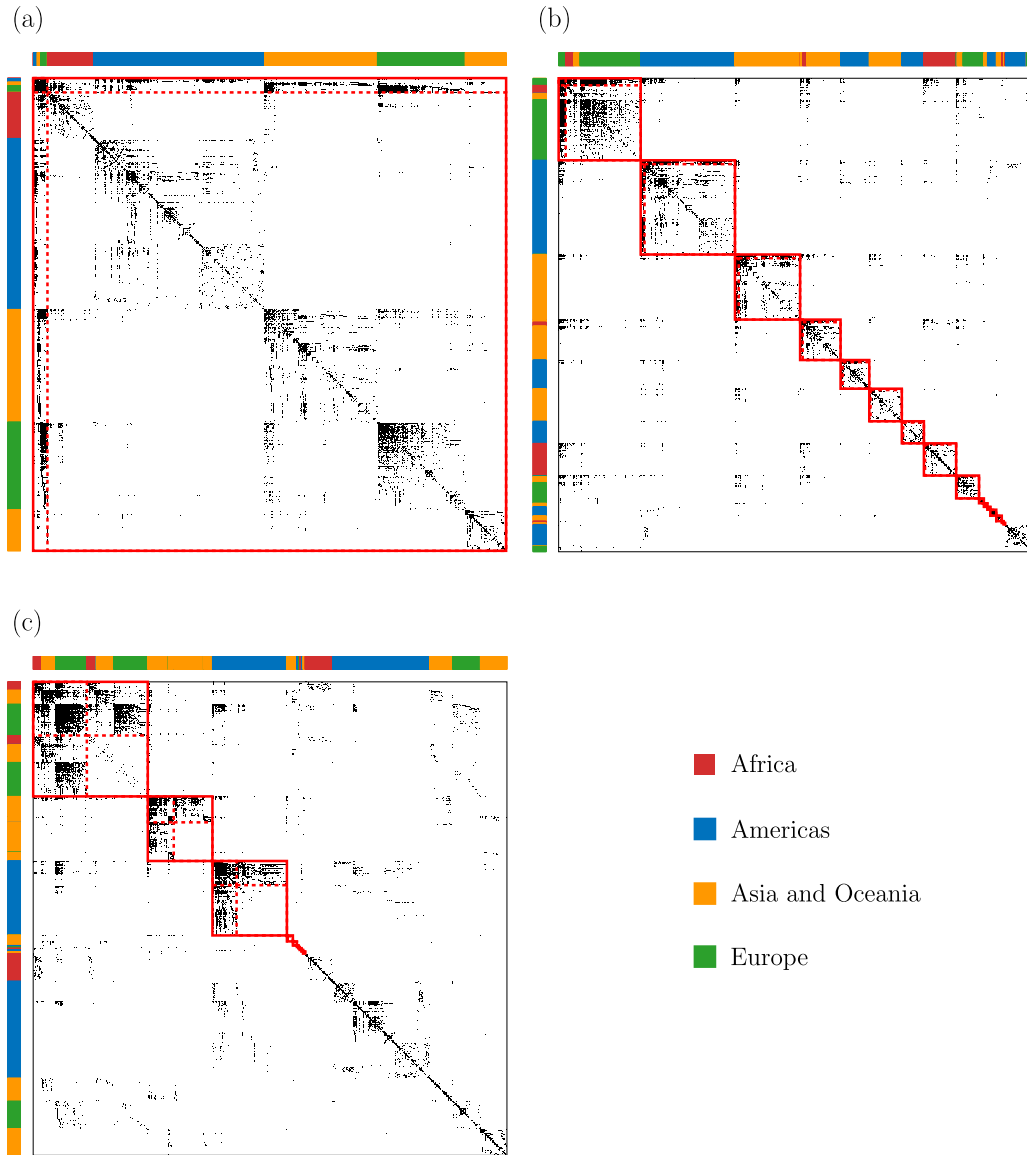
FIG. 6. Core-periphery structure of the airport network detected by (a) the BE-KL algorithm, (b) the two-step algorithm, and (c) our algorithm. The color indicates the geographical region.

we conclude that the structure detected by either of the three algorithms is consistent with the concept of core-periphery structure based on edge density.

2. Political blog network

The second example is a political blog network [6], which has $N = 1222$ nodes and $M = 16,714$ edges (edge density $p = 0.0224$). A node is a blog on the United States president election in 2004, and two blogs are adjacent if one blog cites the other blog on its front page. Each blog was labeled with one of the political leanings, liberal (586 blogs) or conservative (636 blogs), determined by automated categorisations by several weblog directories [6]. If a blog was uncategorized or classified to conflicting categories, then the authors of Ref. [6] manually judged the political leaning.

The core-periphery structure detected by the three algorithms is shown in Fig. 5. The unique core detected by the

BE–KL algorithm is a mixture of liberal and conservative blogs [Fig. 5(a)]. The peripheral blogs are mostly adjacent to blogs with the same political leaning. However, the structure detected by the BE-KL algorithm alone does not tell this unless we refer to the political leaning of the individual blogs. A different algorithm for a single core-periphery structure yielded similar results for the same network [20].

The two-step algorithm detects three core-periphery pairs, each of which mostly comprises the blogs with the same political leanings [Fig. 5(b)]. Two core-periphery pairs are much larger than the third one and have the opposite political leanings. The third small core-periphery pair is mainly composed of liberal blogs. In each core-periphery pair, a majority of the peripheral nodes is densely interconnected, which is against the idealized core-periphery structure. This is due to the community detection step that partitions a network into communities with dense intracommunity edges. In fact, the average density of intra-peripheral edges within
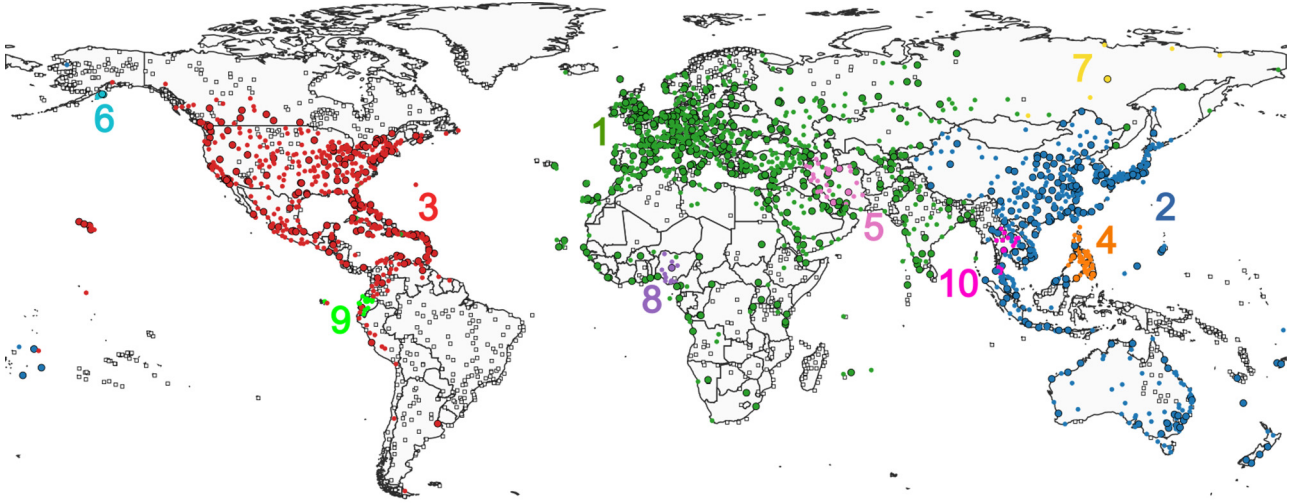
FIG. 7. Location of the airports. The large and small filled circles represent the core and peripheral airports, respectively. Each color represents a core-periphery pair. The open squares represent residual airports.

a core-periphery pair is 0.0214, which is approximately equal to the edge density for the entire network, $p = 0.0224$ [Fig. 4(b)]. This result indicates that the peripheral nodes are as densely adjacent to each other as expected for the Erdős-Rényi random graph, which contradicts the expectation from the core-periphery structure that peripheral nodes are rarely adjacent to each other.

Our algorithm detects two core-periphery pairs, each of which mostly comprises the blogs with the same political leaning [Fig. 5(c)]. The detected two core-periphery pairs are smaller than those detected by the two-step algorithm. More nodes are classified as residual nodes than by the two-step algorithm. The average density of intraperipheral edges within a core-periphery pair is 0.0064 [Fig. 4(b)], which is smaller than the edge density for the entire network, $p = 0.0224$, respecting the notion of the periphery.

3. Airport network

Our third example is a network of airports, which has $N = 2939$ nodes and $M = 15,677$ edges (edge density $p = 0.0036$) [48,49]. A node represents an airport. Two airports are adjacent if there is a direct commercial flight between them.

Figure 6 shows the core-periphery structure detected by the three algorithms. The BE-KL algorithm detects a dense core composed of 89 airports scattered in different geographical regions [Fig. 6(a)]. The peripheral airports are rarely adjacent to the core airports in other regions. Furthermore, the peripheral airports tend to be adjacent to other peripheral airports in the same region, which is inconsistent with the notion of the periphery.

The two-step algorithm detects 16 geographically concentrated core-periphery pairs [Fig. 6(b), in which some peripheral airports are densely interconnected within the core-periphery pairs. The average density of intraperipheral edges within a core-periphery pair is 0.0383, which is approximately 10 times larger than the edge density for the entire network,

$p = 0.0036$ [Fig. 4(c)]. Therefore, the structure detected by the two-step algorithm is not consistent with the concept of core-periphery structure with which peripheral nodes should be sparsely interconnected.

Our algorithm detects 10 geographically concentrated core-periphery pairs [Fig. 6(c)]. The partition of the worldwide airport network into geographically distinct groups of airports found here is consistent with the previous results derived with community detection algorithms [7,8]. Compared to the two-step algorithm, the peripheral airports detected by our algorithm are not densely interconnected; the average density of intraperipheral edges within a core-periphery pair is 0.000073, which is smaller than the edge density for the entire network, $p = 0.0036$ [Fig. 4(c)].

We further analyze the core-periphery structure obtained by our algorithm. Figure 7 maps the locations of the core and peripheral airports. The three largest core-periphery pairs labeled 1, 2, and 3 are mainly based in Europe, East Asia, and the United States, respectively. The core-periphery pairs 1, 2, and 3 consist of the airports in 125, 35, and 47 countries, respectively. Each of the other core-periphery pairs labeled 4–10 consists of the airports in one country.

The location of the airports and metropolises in Europe, East Asia, the United States and their surroundings are shown in Fig. 8. Here the metropolis is defined as the capital city of all countries, the provincial capitals of China and the state capitals of the United States because China and the United States have many airports. Core-periphery pair 1 contains 333 core airports and 378 peripheral airports, of which 405 (57%) airports are located in Europe [Fig. 8(a)]. However, this core-periphery pair excludes most airports in the Nordic countries (84 airports; 68%). There are 89 airports within 20 miles from a metropolis in Europe, among which there are 51 core airports (57%), 28 peripheral airports (31%), and 10 residual airports (11%). As a comparison, if we select the same number of the European airports with the largest degrees as that of the European core airports, then 64 airports (72%) are contained in the set of
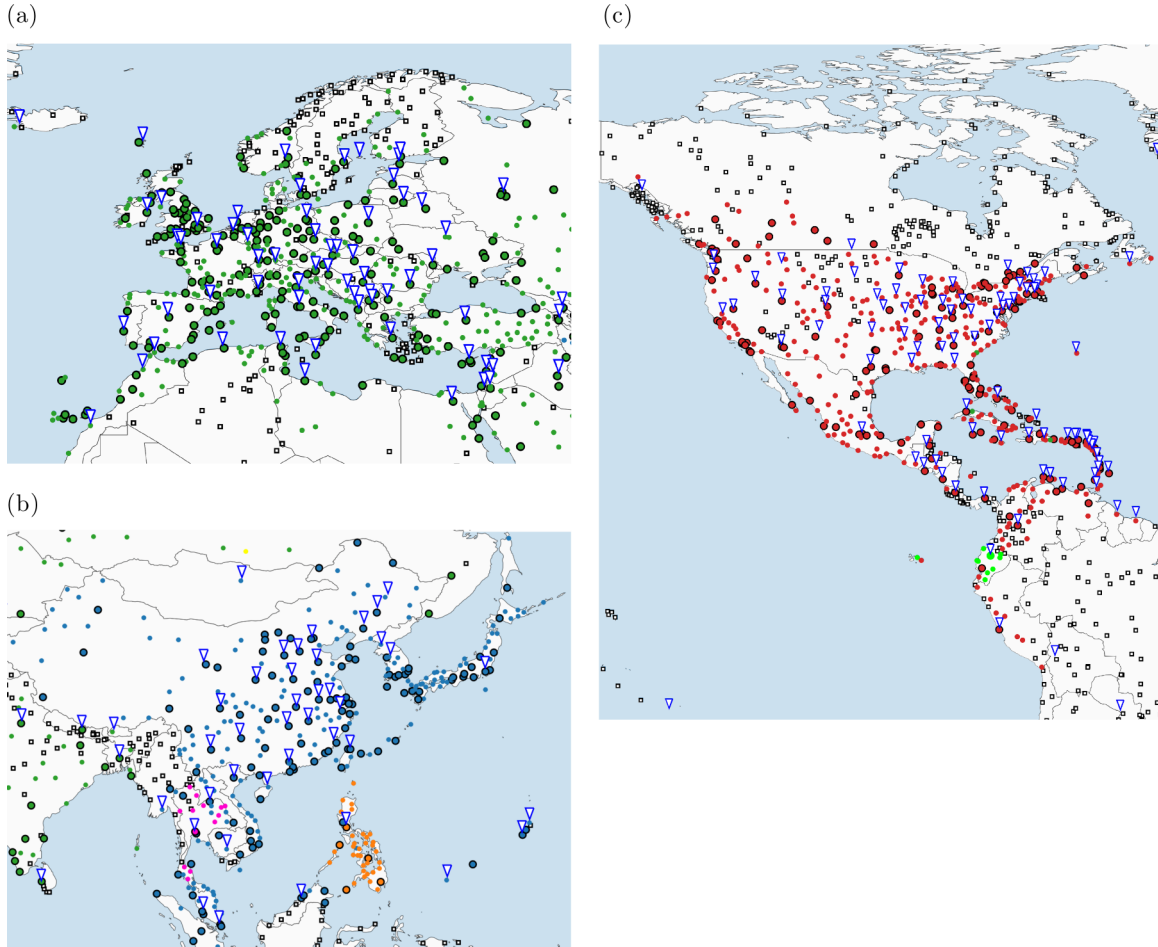
FIG. 8. Location of the airports in (a) Europe, (b) East Asia, and (c) the contiguous United States and their surrounding areas. The large and small filled circles represent the core and the peripheral airports, respectively. Each color represents a core-periphery pair. The open squares represent residual airports. The inverted triangles indicate the location of metropolises, i.e., the capital cities of all countries, the provincial capitals of China and the state capitals of the United States.

89 airports within 20 miles from a metropolis, which is more than the number of the core airports (51 airports; see above) contained in the same set of airports. This result indicates that hub metropolitan airports, which are common, are not necessarily core airports.

The second core-periphery pair contains 161 core airports and 240 peripheral airports, among which 217 (54%) airports are located in East Asia [Fig. 8(b)]. In this core-periphery pair, 31 airports are located within 20 miles from a metropolis in East Asia, among which there are 23 core airports (74%), eight peripheral airports, and no residual airport [Fig. 8(b)].

The third core-periphery pair contains 150 core airports and 312 peripheral airports, among which 210 (45%) airports are located in the United States [Fig. 8(c)]. In this core-periphery pair, 71 airports are located within 20 miles from a metropolis in the United States, among which there are 29 core airports (41%), 30 peripheral airports (42%), and 12 residual airports (17%) [Fig. 8(c)]. We have not found the partitioning of airports into core-periphery pairs corresponding to different major airline groups (e.g., American Airlines, Delta Airlines, Southwest Airlines, and United Airlines in the United States).

Table I lists the size of core-periphery pairs and the fractions of different types of edges. The airports in a large core are not densely interconnected compared to those in small core-periphery pairs, probably due to the limited capacity of the airports (e.g., a small number of runways). Core-periphery pairs 1, 2, and 3 contain hub airports in each region. The other small core-periphery pairs consist of a small number of core airports, i.e., at most 20% of the airports in each core-periphery pair. In these core-periphery pairs, most of the peripheral airports are adjacent to the core airports but not to other peripheral airports in the same core-periphery pair. This observation suggests that a small number of core airports relays most of the flights into these regions as gateway airports. For example, the representative core airport (i.e., the core airport that has the largest number of neighbors in the core-periphery pair) in pair 4, MNL (Philippines), and that in pair 8, LOS (Nigeria), serve most of the domestic airports in the respective countries. Such a structure is evident in small core-periphery pairs such as core-periphery pairs 4–10.

The subnetwork within the Philippines is shown in Fig. 9(a); see Table II for properties of all airports in the Philippines. Most of the airports (34 airports; 92%) in core-periphery pair

TABLE I. Properties of the core-periphery pairs in the airport network. The core-periphery pairs are ordered according to the number of the core nodes. The core nodes that have the largest number of neighbors within the same core-periphery pair are shown as representative core nodes. C–C, C–P, and P–P denote core–core, core–periphery, and periphery–periphery edges, respectively.

Pair	Number of nodes		Fraction of edges within a pair			Representative core nodes		
	Core	Periphery	C–C	C–P	P–P	IFTA	City	Country
1	333	378	0.0825	0.0194	0.0005	FRA	Frankfurt	Germany
						CDG	Paris	France
						AMS	Amsterdam	Netherlands
2	161	240	0.0964	0.0234	0.0000	PEK	Beijing	China
						CAN	Guangzhou	China
						PVG	Shanghai	China
3	150	312	0.1372	0.0265	0.0003	ATL	Atlanta	USA
						LAS	Las Vegas	USA
						MCO	Orlando	USA
4	5	32	0.4000	0.3312	0.0000	MNL	Manila	Philippines
5	4	21	0.5000	0.2976	0.0000	THR	Teheran	Iran
6	2	8	1.0000	0.5625	0.0000	ADQ	Kodiak	USA
7	1	6	0.0000	1.0000	0.0000	YKS	Yakutsk	Russia
8	1	9	0.0000	1.0000	0.0000	LOS	Lagos	Nigeria
9	1	9	0.0000	1.0000	0.0000	UIO	Quito	Ecuador
10	1	12	0.0000	1.0000	0.0000	DMK	Bangkok	Thailand

4 [shown in orange in Figs. 7, 8(b), and 9(a)] only serve domestic flights. Core airport 1 [labelled in Fig. 9(a)] has most of the edges (41 edges; 84%) between core-periphery pair 4 and the rest of the network. Therefore, core airport 1 functions as a gateway airport in the Philippines. Core airport 2 also functions as a gateway airport but to a lesser extent than core airport 1 does. Core-periphery pairs located in Alaska (core-periphery pair 6 in Table I), Russia (pair 7),

and Ecuador (pair 9) also contain a few core airports serving as gateway airports in the respective regions (Appendix C). Core airports 8 and 21 in the Philippines [Fig. 9(a)] have a small degree, which is counterintuitive. Core nodes having degree one or two are also found in core-periphery pair 6 [Fig. 14(c)]. The airports 8 and 21 in the Philippines are adjacent to one peripheral airport 7 and 4, respectively. If we assign airport 8 to the periphery, then two peripheral airports 7 and 8 would be adjacent. Similarly, if we assign airport 21 to the periphery, then two peripheral airports 4 and 21 would be adjacent. To avoid edges between peripheral nodes, our algorithm has identified airports 8 and 21 as core nodes. However, airports 8 and 21 may be better regarded as peripheral airports given that they are not densely interconnected to other core airports. Previous studies provided remedies for this issue [11,16] (see Sec. V for further discussion).

The subnetwork within Thailand is shown in Fig. 9(b); see Table III for properties of all airports in Thailand. Two major airports 1 and 14 are located in the capital city, Bangkok, and belong to different core-periphery pairs. All international airports in Thailand belong to core-periphery pair 2 [shown in blue in Figs. 7, 8(b), and 9(b)], including core airport 14. Most of the domestic airports (13 airports; 59%) belong to core-periphery pair 10 (shown in magenta), including core airport 1. The subnetwork composed of core-periphery pair 10 is largely separated from the other airports in Thailand, which belong to core-periphery pair 2, and the rest of the world. The separation of the domestic and international airports and their respective subnetworks is also observed in the Philippines [Fig. 9(a)], Iran, and Nigeria (Appendix C).

FIG. 9. Airport network within (a) the Philippines and (b) Thailand. The line color indicates the core-periphery pair to which the two airports belong. The edges connecting two airports in different core-periphery pairs are shown in gray. The numbers attached to some airports indicate the IDs of the airports listed in Tables II and III. We only show the IDs of all core airports, some peripheral airports and all residual airports.

C. Computation time

We implement the three algorithms in MATLAB and run simulations on a computer with Intel 2.6-GHz Sandy Bridge

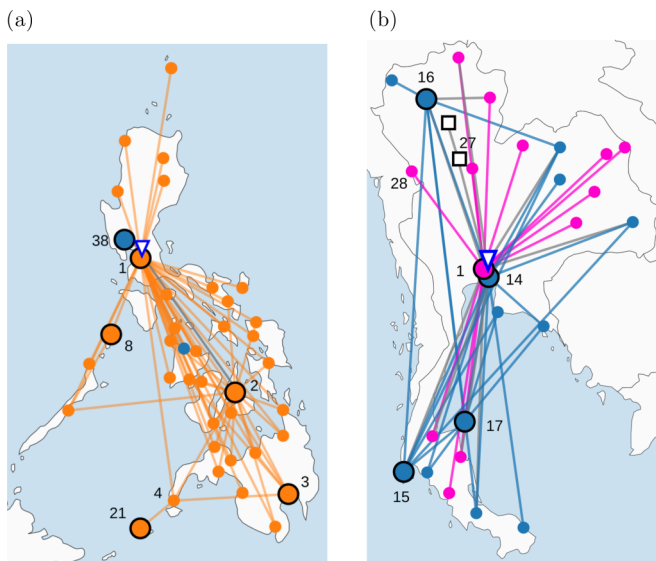TABLE II. Properties of the airports in the Phillipines. The airports are sorted in the descending order of the total number of edges. The internal edge is defined as that between two airports within the same core-periphery pair. The external edge is defined as that between an airport in the focal core-periphery pair and an airport in a different core-periphery pair or a residual airport.

| ID | IFTA | Pair | Type | Number of edges | | | |
				Domestic	International	Internal	External
1	MNL	4	Core	36	40	35	41
2	CEB	4	Core	19	6	18	7
3	DVO	4	Core	5	1	5	1
4	ZAM	4	Periphery	4	0	4	0
5	CGY	4	Periphery	3	0	3	0
6	ILO	4	Periphery	3	0	3	0
7	PPS	4	Periphery	3	0	3	0
8	USU	4	Core	2	0	2	0
9	PAG	4	Periphery	2	0	2	0
10	TAC	4	Periphery	2	0	2	0
11	BCD	4	Periphery	2	0	2	0
12	DGT	4	Periphery	2	0	2	0
13	MPH	4	Periphery	2	0	2	0
14	BXU	4	Periphery	2	0	2	0
15	DPL	4	Periphery	2	0	2	0
16	LGP	4	Periphery	2	0	2	0
17	OZC	4	Periphery	2	0	2	0
18	GES	4	Periphery	2	0	2	0
19	SUG	4	Periphery	2	0	2	0
20	CRM	4	Periphery	2	0	2	0
21	JOL	4	Core	1	0	1	0
22	CBO	4	Periphery	1	0	1	0
23	SJI	4	Periphery	1	0	1	0
24	TAG	4	Periphery	1	0	1	0
25	LAO	4	Periphery	1	0	1	0
26	ENI	4	Periphery	1	0	1	0
27	WNP	4	Periphery	1	0	1	0
28	BSO	4	Periphery	1	0	1	0
29	SFE	4	Periphery	1	0	1	0
30	TUG	4	Periphery	1	0	1	0
31	VRC	4	Periphery	1	0	1	0
32	CYP	4	Periphery	1	0	1	0
33	MBT	4	Periphery	1	0	1	0
34	RXS	4	Periphery	1	0	1	0
35	CYZ	4	Periphery	1	0	1	0
36	TBH	4	Periphery	1	0	1	0
37	MRQ	4	Periphery	1	0	1	0
38	CRK	2	Core	1	8	8	1
39	KLO	2	Periphery	1	3	3	1

TABLE III. Properties of the airports in Thailand.

| ID | IFTA | Pair | Type | Number of edges | | | |
				Domestic	International	Internal	External
1	DMK	10	Core	17	0	12	5
2	CEI	10	Periphery	2	0	1	1
3	NST	10	Periphery	2	0	1	1
4	URT	10	Periphery	2	0	1	1
5	NNT	10	Periphery	2	0	1	1
6	PHS	10	Periphery	1	0	1	0
7	TST	10	Periphery	1	0	1	0
8	SNO	10	Periphery	1	0	1	0
9	LOE	10	Periphery	1	0	1	0
10	KOP	10	Periphery	1	0	1	0
11	ROI	10	Periphery	1	0	1	0
12	BFV	10	Periphery	1	0	1	0
13	MAQ	10	Periphery	1	0	1	0
14	BKK	2	Core	14	122	69	67
15	HKT	2	Core	7	21	23	5
16	CNX	2	Core	8	6	12	2
17	USM	2	Core	6	3	9	0
18	HDY	2	Periphery	3	2	4	1
19	UTH	2	Periphery	4	0	3	1
20	KBV	2	Periphery	2	2	4	0
21	UBP	2	Periphery	3	0	2	1
22	UTP	2	Periphery	2	0	2	0
23	TDX	2	Periphery	2	0	2	0
24	NAW	2	Periphery	1	0	1	0
25	KKC	2	Periphery	1	0	1	0
26	HGN	2	Periphery	1	0	1	0
27	THS	–	Residual	2	0	0	2
28	LPT	–	Residual	1	0	0	1

on the blog and airport networks, it runs much faster than the BE-KL algorithm. Our algorithm runs in $\mathcal{O}(rM)$ time, where r is the number of rounds (Sec. II B). Therefore, our algorithm is expected to be fast on sparse networks.

V. DISCUSSION

We proposed a scalable algorithm to detect multiple core-periphery pairs in networks by maximizing a novel quality function Q^{cp}. The quality function Q^{cp} compares the number

TABLE IV. The average CPU time of the three algorithms on different networks. We generate synthetic networks 1–4 using the stochastic block models schematically shown in Figs. 1(a), 1(b), 1(c), and 1(d), respectively. For each of them, we set $\theta_1 = 0.9$ and $\theta_2 = 0.05$ and measure the CPU time for one generated network.

| Network | N | M | Average CPU time (s) | | |
			BE-KL	Two-step	Proposed
Synthetic 1	400	31 331	0.186	1.076	0.356
Synthetic 2	400	16 355	0.162	0.343	0.240
Synthetic 3	400	19 308	0.187	1.117	0.366
Synthetic 4	400	10 939	0.193	0.392	0.251
Karate	34	78	0.006	0.023	0.025
Blog	1222	0.022	5.076	9.080	1.321
Airport	2939	15 677	51.57	67.17	5.709

processors and 4 GB of memory. The speed of an algorithm is measured by averaging the CPU time over 100 runs. We do not run the statistical test because it is a common process for the three algorithms.

The average CPU time of the three algorithms is compared in Table IV. The BE-KL algorithm is the fastest on all synthetic networks and the karate club network. However, it is slower than our algorithm on the blog and airport networks. The two-step algorithm is the slowest on all but one network. Our algorithm is approximately two times slower than the BE-KL algorithm on the synthetic and karate club networks. However,
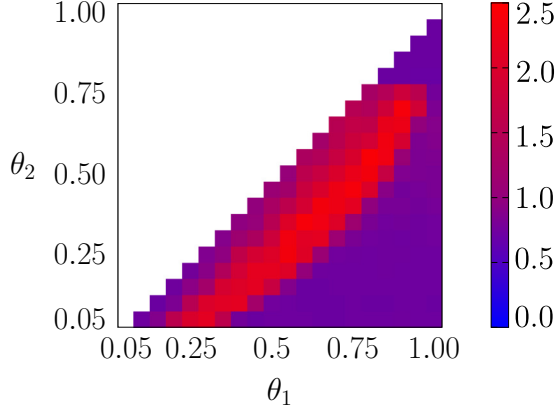
FIG. 10. The VI values between the true and inferred core-periphery structure obtained by the maximisation of $Q_{\text{config}}^{\text{cp}}$.

of edges of different types in the given network with the expected number for an Erdős-Rényi random graph. The present algorithm reveals the groups of nodes sharing a common property (e.g., a group of members sharing the same political leaning) and different roles of nodes within the groups (e.g., leaders and followers in each group). These properties are simultaneously revealed neither by partitioning of networks into a single core-periphery pair nor community detection.

In the airport network, we have found several core nodes having degree one or two [e.g., airports 8 and 21 in Fig. 9(a)], which contradicts the intuition that core nodes would have a large degree. Our algorithm assigned these nodes to a core to suppress the edges between peripheral nodes. However, these nodes may be better regarded as peripheral nodes because they are adjacent to at most one core node. One remedy is to weaken the suppression of the edges between peripheral nodes [11,16]. Adapting this idea to the case of multiple core-periphery structure warrants future research.

Previous studies provided algorithms to detect multiple core-periphery pairs based on community detection algorithms. Yang and Leskovec used an algorithm for detecting overlapping communities in networks [17,50,51]. They regarded the nodes belonging to many communities as core nodes and nodes belonging to few communities as peripheral nodes. The algorithm may detect densely interconnected peripheral nodes because the detected peripheral nodes in a single core-periphery pair belong to the same community. In

addition, a periphery may belong to multiple cores in these algorithms. These properties are shared by the algorithms proposed in Refs. [31,32]. In contrast, our algorithm detects disjoint core-periphery pairs such that peripheral nodes are interconnected sparsely within each core-periphery pair and across different core-periphery pairs. Yan and Luo focused on a different type of structure consisting of multiple cores and a single periphery [30]. In contrast, a core detected by our algorithm owns its exclusive periphery, including the case of an empty periphery.

We used the Erdős-Rényi random graph as the null model to define Q^{cp}. The Erdős-Rényi random graph model is also used for detecting communities in networks [9,35,52,53]. For example, the community detection algorithms based on the Potts model [52,53] and stochastic block models without degree correction (i.e., without assuming a heterogeneous degree distribution) [9] use the Erdős-Rényi random graph model as a null model. There are also null models that incorporate other properties of networks such as degree heterogeneity [5], weighted edges [54], signed edges [55], correlations [56], bipartiteness [57], and space embeddedness [58]. It is possible to incorporate these null models into the second term of the right-hand side of Eq. (3). For example, a popular null model is the configuration model, with which we randomly rewire edges while conserving the degree of each node [3]. With the configuration model, the quality function is given by $Q_{\text{config}}^{\text{cp}} = \sum_{i=1}^{N} \sum_{j=1}^{i} (A_{ij} - d_i d_j / 2M)(x_i + x_j - x_i x_j)\delta_{c_i,c_j}$. As is the case in community detection [9], the choice of the null model will affect the organization of the detected core-periphery pairs. To see this, we maximized $Q_{\text{config}}^{\text{cp}}$ using a label switching heuristic (Sec. II B) for the synthetic networks with a single core-periphery pair [Fig. 1(a)]. The VI values for $Q_{\text{config}}^{\text{cp}}$ were larger than 0.4 in the entire parameter region spanned by θ_1 and θ_2 (Fig. 10), indicating that the maximization of $Q_{\text{config}}^{\text{cp}}$ did not enable us to detect the planted core-periphery pair. This is due to the null-model term $d_i d_j / 2M$ in $Q_{\text{config}}^{\text{cp}}$. In the synthetic networks with a planted single core-periphery pair, the planted core nodes have a large degree. If we put the planted core nodes into the same core-periphery pair, then the null-model term $d_i d_j / 2M$ becomes large, giving a large decrement in $Q_{\text{config}}^{\text{cp}}$. Therefore, the maximization of $Q_{\text{config}}^{\text{cp}}$ assigned the planted core nodes to different core-periphery pairs (yielding $\delta_{c_i,c_j} = 0$) or a periphery (yielding $x_i + x_j - x_i x_j = 0$).
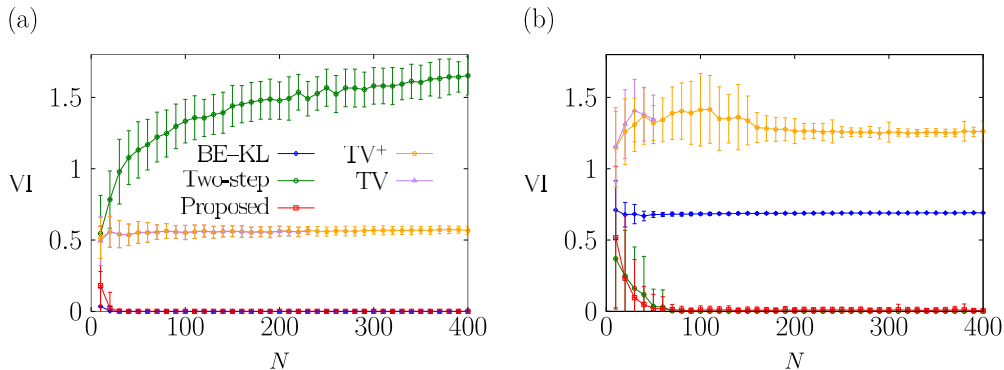

FIG. 11. The VI values between the true and inferred core-periphery structure for the five algorithms. Panels (a) and (b) correspond to the networks whose structure is shown in Figs. 1(a) and 1(b), respectively. The error bars indicate the ±1 standard deviation.
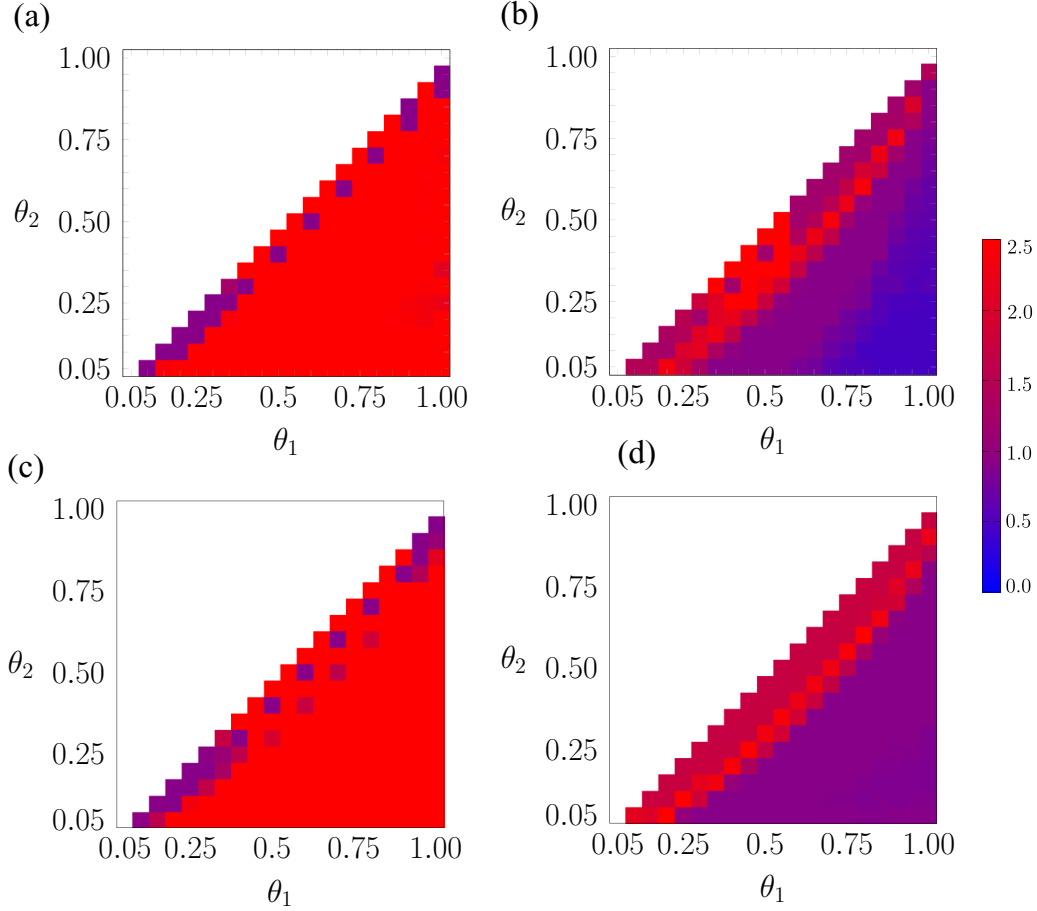
FIG. 12. The VI values between the true and inferred core-periphery structure for the divisive algorithm. Panels (a), (b), (c), and (d) show the VI values for the synthetic networks shown in Figs. 1(a), 1(b), 1(c), and 1(d), respectively.

There are several lines of possible extensions of the present work. First, we did not consider continuous versions of core-periphery structure, with which each node is assigned a strength (i.e., a coreness) value representing the belongingness of the node to a core [11,13,15,16,21,25,27,27]. Continuous versions of core-periphery structure can reveal nested structure of cores (i.e., cores within a core), which discrete versions of the algorithms would not. Borgatti and Everett generalized a discrete version of core-periphery structure defined by Eq. (1) to a continuous version by replacing binary variables $x_i \in \{0,1\}$ $(1 \leqslant i \leqslant N)$ by continuous variables (e.g., $x_i \in [0,1]$) [11]. This approach may allow us to generalize our algorithm to continuous versions.

Second, we have ignored the weight and direction of edges. It is straightforward to incorporate the weight of edges by redefining A_{ij} on the right-hand side of Eq. (3) as the weight of the edge. In contrast, it is not easy to extend our algorithm to the case of directed networks. In the case of community detection, a natural extension is to allow an adjacency matrix A to be asymmetric [59,60], which is, in fact, problematic [61]. Therefore, we expect that the same pitfall exists in the detection of core-periphery pairs if we extend our algorithm to the case of directed networks by simply allowing A to be asymmetric.

Third, our quality function, Q^{cp}, has a similar mathematical form to modularity [3,5]. If we constrain all nodes to be core

nodes, i.e., $x_i = 1$ $(1 \leqslant i \leqslant N)$, then Q^{cp} is equivalent to the modularity based on the Potts model [52,53]. Another class of approach is based on the likelihood maximization of stochastic block models. Stochastic block models are generative models of networks composed of blocks of nodes [9,46]. In fact, stochastic block models for detecting a single core-periphery pair [20] and multiple core-periphery pairs [19] have been proposed. Modularity maximization and likelihood maximization are equivalent in a particular situation but are different in general [62]. Modularity has some limitations such as the incapacity of finding small communities [63] and that of distinguishing nonrandom from random structure [64]. Therefore, we may benefit from using the stochastic block models.

How multiple core-periphery pairs emerge is unclear. An economic mechanism explains the emergence of a single core-periphery pair in networks [23]. The authors considered the trade-offs between the profit obtained by connecting nodes and the cost for maintaining edges. Core-periphery structure emerges if the cost is not extremely small or large relative to the profit [23]. Given their results, multiple core-periphery pairs may emerge when the cost of intergroup edges is significantly larger than the cost of intragroup edges. For example, in airport networks, interregional flights would be more costly than intraregional flights due to the different fuel expense and tax. Exploration of dynamical or economic mechanisms behind
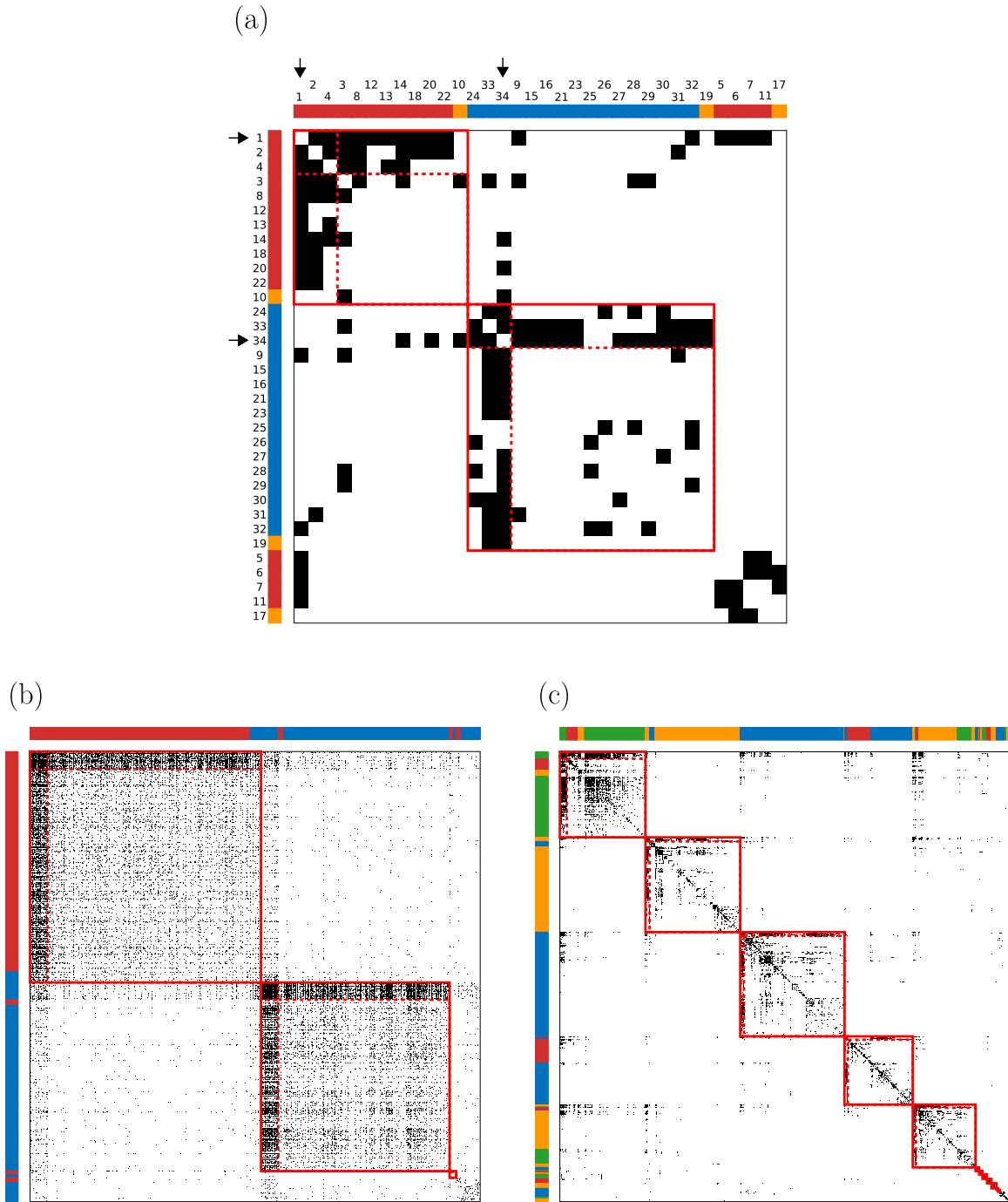
FIG. 13. Core-periphery structure identified by the divisive algorithm in (a) the karate club network, (b) blog network, and (c) airport network.

the formation of multiple core-periphery pairs in a network warrants future work.

APPENDIX A: TUNÇ-VERMA ALGORITHM

We evaluated the performance of the Tunç-Verma (TV) algorithm [19] using the synthetic networks used in Sec. IV. The Python code provided by the authors [19] is not fast enough. Therefore, we reimplement their algorithm based on the original code by changing the data structure and replacing

some functions by faster inbuilt functions in MATLAB. We did not change the algorithm itself including the parameters. We refer the original and reimplemented algorithms as the TV and TV^+, respectively. With the TV and TV^+ algorithms, each node belongs to multiple core-periphery pairs. Therefore, we assign each node to the core-periphery pair to which the degree of belonging [19] is the largest. The TV and TV^+ algorithms can infer the number of core-periphery pairs. However, to save their computational time, we provide the correct number of core-periphery pairs in the synthetic networks to these

TABLE V. Properties of airports in Iran.

| ID | IFTA | Pair | Type | Number of edges | | | |
				Domestic	International	Internal	External
1	THR	5	Core	36	4	24	16
2	ZAH	5	Periphery	5	2	3	4
3	KER	5	Core	6	0	3	3
4	KSH	5	Periphery	5	0	3	2
5	PGU	5	Core	4	0	2	2
6	ABD	5	Periphery	4	0	1	3
7	GBT	5	Core	3	0	2	1
8	MRX	5	Periphery	3	0	1	2
9	BUZ	5	Periphery	2	1	1	2
10	OMH	5	Periphery	1	1	1	1
11	XBJ	5	Periphery	2	0	1	1
12	GSM	5	Periphery	2	0	1	1
13	NSH	5	Periphery	2	0	1	1
14	ADU	5	Periphery	2	0	1	1
15	CQD	5	Periphery	1	1	1	1
16	SDG	5	Periphery	1	0	1	0
17	RZR	5	Periphery	1	0	1	0
18	KHD	5	Periphery	1	0	1	0
19	BXR	5	Periphery	1	0	1	0
20	BJB	5	Periphery	1	0	1	0
21	AFZ	5	Periphery	1	0	1	0
22	ACP	5	Periphery	1	0	1	0
23	IIL	5	Periphery	1	0	1	0
24	PFQ	5	Periphery	1	0	1	0
25	YES	5	Periphery	1	0	1	0
26	IKA	1	Core	1	43	40	4
27	MHD	1	Core	21	17	26	12
28	SYZ	1	Periphery	14	9	17	6
29	IFN	1	Core	11	5	11	5
30	BND	1	Core	12	2	12	2
31	AWZ	1	Core	8	5	12	1
32	TBZ	1	Core	6	4	9	1
33	KIH	1	Core	7	1	7	1
34	RAS	1	Periphery	7	1	6	2
35	ZBR	1	Periphery	4	2	4	2
36	LRR	1	Core	2	3	4	1
37	AZD	1	Periphery	3	1	3	1
38	SRY	1	Periphery	3	1	3	1
39	BDH	-	Residual	1	0	0	1

TABLE VI. Properties of the airports in Nigeria.

| ID | IFTA | Pair | Type | Number of edges | | | |
				Domestic	International	Internal	External
1	ABV	8	Core	11	8	9	10
2	AKR	8	Periphery	2	0	1	1
3	BNI	8	Periphery	2	0	1	1
4	CBQ	8	Periphery	2	0	1	1
5	PHC	8	Periphery	2	0	1	1
6	QOW	8	Periphery	2	0	1	1
7	QRW	8	Periphery	2	0	1	1
8	KAD	8	Periphery	2	0	1	1
9	ILR	8	Periphery	1	0	1	0
10	SKO	8	Periphery	1	0	1	0
11	LOS	1	Core	11	23	23	11
12	KAN	1	Periphery	3	5	6	2
13	ENU	–	Residual	2	0	0	2
14	JOS	–	Residual	1	0	0	1

all values of N because the BE-KL algorithm is not designed for multiple core-periphery pairs. The VI values for the two-step and our algorithms are comparable and close to zero for $N \geqslant 50$. The VI for the TV and TV^+ algorithms is larger than that for the other algorithms. The TV algorithm did not terminate for $N \geqslant 60$.
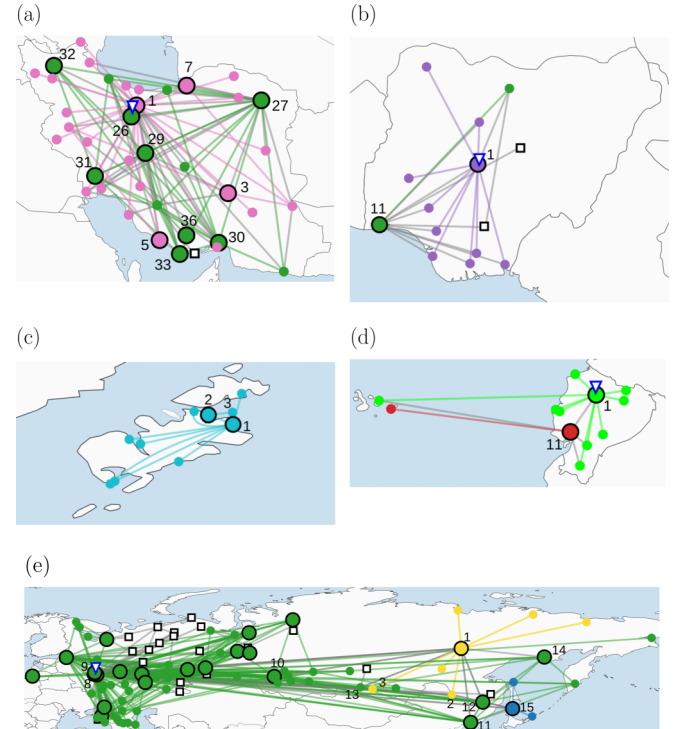

FIG. 14. Airport network within (a) Iran, (b) Nigeria, (c) core-periphery pair 6 based in Alaska, (d) Ecuador, and (e) Russia. The line color indicates the core-periphery pair to which the two airports belong. The edges connecting two airports in different core-periphery pairs are shown in gray. The numbers attached to some airports indicate the IDs of the airports listed in Tables V–IX. We only show the IDs of the core airports, some peripheral airports and some residual airports.

algorithms. We set $\theta_1 = 0.9$ and $\theta_2 = 0.05$, use the same Θ and π parameter values as those used in the main text and vary $N \in \{10, 20, \ldots, 400\}$.

Figure 11(a) shows the VI for the networks composed of a single core-periphery pair, whose structure is schematically shown in Fig. 1(a). The VI for the BE-KL and our algorithms is approximately equal to zero except for small N. The VI for the two-step algorithm increases as N increases. The VI for the TV algorithm is large for $N \leqslant 230$. The TV algorithm did not terminate for $N \geqslant 240$. As expected, the VI values for the TV^+ algorithm are similar to those for the TV algorithm and similarly large for $N \geqslant 240$.

Figure 11(b) shows the results for the networks composed of two core-periphery pairs, whose structure is schematically shown in Fig. 1(b). The VI for the BE-KL algorithm is large for

TABLE VII. Properties of the airports in core-periphery pair 6 based in Alaska.

ID	IFTA	Pair	Type	Number of edges			
				Domestic	International	Internal	External
1	ADQ	6	Core	10	0	9	1
2	ORI	6	Core	2	0	2	0
3	KOZ	6	Periphery	2	0	2	0
4	AKK	6	Periphery	1	0	1	0
5	KLN	6	Periphery	1	0	1	0
6	OLH	6	Periphery	1	0	1	0
7	ALZ	6	Periphery	1	0	1	0
8	AOS	6	Periphery	1	0	1	0
9	KKB	6	Periphery	1	0	1	0
10	KPY	6	Periphery	1	0	1	0

In both types of the synthetic networks, the TV and TV^+ algorithms do not infer the true planted core-periphery structure for the entire range of N. Tunç and Verma applied their algorithm to dense and weighted networks. In contrast, here we have analyzed sparse and unweighted networks, which might be a main reason for the poor performance of these algorithms.

Next, we measure the speed of the TV^+ algorithm (which is faster than the TV algorithm) for the synthetic networks used in the main text (Sec. IV C). For the synthetic networks, we set $N = 400$, $\theta_1 = 0.9$ and $\theta_2 = 0.05$. For the synthetic network with a single core-periphery pair [Fig. 1(a)], the TV^+ algorithm requires 29.6 s on average. This is 83 times slower than our algorithm (Table IV). For the synthetic network with two core-periphery pairs [Fig. 1(b)], the TV^+ algorithm requires 81.1 s on average, which is 338 times slower than our algorithm (Table IV). On the karate club network, for which we set the number of the core-periphery pair to two, the TV^+ algorithm requires 298.3 s on average, which is 11,932 times slower than our algorithm. For the blog and the airport networks, the TV^+ algorithm did not terminate in 12 h.

APPENDIX B: DIVISIVE ALGORITHM

The divisive algorithm [3–5,45] partitions the nodes into communities using the Louvain algorithm [41] and then par-

TABLE VIII. Properties of the airports in Ecuador.

ID	IFTA	Pair	Type	Number of edges			
				Domestic	International	Internal	External
1	UIO	9	Core	10	9	9	10
2	CUE	9	Periphery	2	0	1	1
3	GPS	9	Periphery	2	0	1	1
4	LOH	9	Periphery	2	0	1	1
5	OCC	9	Periphery	1	0	1	0
6	XMS	9	Periphery	1	0	1	0
7	MEC	9	Periphery	1	0	1	0
8	PVO	9	Periphery	1	0	1	0
9	ESM	9	Periphery	1	0	1	0
10	LGQ	9	Periphery	1	0	1	0
11	GYE	3	Core	5	11	11	5
12	SCY	3	Periphery	1	0	1	0

TABLE IX. Properties of the airports in Russia. Only the airports in core-periphery pair 7 and those in other core-periphery pairs that are adjacent to core-periphery pair 7 are shown.

ID	IFTA	Pair	Type	Number of edges			
				Domestic	International	Internal	External
1	YKS	7	Core	13	1	6	8
2	BQS	7	Periphery	4	0	1	3
3	UUD	7	Periphery	2	0	1	1
4	CKH	7	Periphery	1	0	1	0
5	CYX	7	Periphery	1	0	1	0
6	IKS	7	Periphery	1	0	1	0
7	NER	7	Periphery	1	0	1	0
8	DME	1	Core	62	97	134	25
9	VKO	1	Core	34	15	42	7
10	OVB	1	Core	14	18	27	5
11	VVO	1	Core	13	7	11	9
12	KHV	1	Core	14	5	11	8
13	IKT	1	Periphery	9	5	10	4
14	GDX	1	Core	8	0	7	1
15	UUS	2	Core	7	6	8	5

titions the nodes in each community into core and peripheral nodes using the BE-KL algorithm. We apply the statistical test (Sec. II C) to the core-periphery pairs detected by the divisive algorithm.

The VI values for the synthetic networks are shown in Fig. 12. For the synthetic network with a single core-periphery pair [Fig. 1(a)], the VI values are large in the entire θ_1-θ_2 parameter space [Fig. 12(a)] because the divisive algorithm divides the planted single core-periphery pair into multiple core-periphery pairs. For the synthetic network with two core-periphery pairs [Fig. 1(b)], the VI values are larger than those for the proposed algorithm for most θ_1 and θ_2 values [Figs. 2(f) and 12(b)]. For the synthetic network with a single core-periphery pair and residual nodes [Fig. 1(c)], the VI values are larger than those for the proposed algorithm for most θ_1 and θ_2 values [Figs. 2(i) and 12(c)]. For the synthetic network with two core-periphery pairs and residual nodes [Fig. 1(d)], the VI values are larger than those for the proposed algorithm in the entire parameter region of θ_1 and θ_2 [Figs. 2(l) and 12(d)].

The core-periphery structure in the karate club detected by the divisive algorithm is shown in Fig. 13(a). The divisive algorithm detects two core-periphery pairs, each of which mostly consists of the members supporting the same leader [Fig. 13(a)]. The average density of intracore edges and that of core-periphery edges within a core-periphery are 1.000 and 0.619, respectively, which are larger than the edge density for the entire network, $p = 0.139$. The average density of intraperipheral edges within a core-periphery pair is 0.080, which is smaller than the edge density for the entire network, $p = 0.139$. Therefore, the core-periphery structure in the karate network detected by the divisive algorithm is consistent with the concept of core-periphery structure based on edge density.

In the blog network, the divisive algorithm detects three core-periphery pairs, each of which mostly comprises the blogs with the same political leaning [Fig. 13(b)]. Two core-

periphery pairs are much larger than the third one and have the opposite political leanings. The divisive algorithm identifies more residual nodes than the two-step algorithm. The average density of intracore edges and that of core-periphery edges within a core-periphery pair are 0.3964 and 0.2906, respectively, which are larger than the edge density for the entire network, $p = 0.0224$. The average density of intraperipheral edges within a core-periphery pair is 0.0138, which is smaller than the edge density for the entire network, $p = 0.0224$. Therefore, the core-periphery structure in the blog network detected by the divisive algorithm is consistent with the concept of core-periphery structure based on edge density.

In the airport network, the divisive algorithm identifies 12 core-periphery pairs, each of which mostly consists of the airports located in the same geographical region [Fig. 13(c)]. The average density of intracore edges and that of core-periphery edges within a core-periphery pair are 0.6335 and 0.2978, respectively, which are larger than the edge density for the entire network, $p = 0.0036$. The average density of intraperipheral edges within a core-periphery pair is 0.0419, which is larger than the edge density for the entire network, $p = 0.0036$. Therefore, the core-periphery structure in the airport network detected by the divisive algorithm is inconsistent with the concept of core-periphery structure based on edge density.

APPENDIX C: PROPERTIES OF AIRPORTS IN CORE-PERIPHERY PAIRS 4–10

Our algorithm separates the international and domestic airports in the Philippines, Iran, and Nigeria into different core-periphery pairs. In Iran, the airports mainly serving the international and domestic flights belong to core-periphery pairs 1 and 5, respectively (Table V). In Nigeria, the airports mainly serving the international and domestic flights belong to core-periphery pairs 1 and 8, respectively (Table VI). There is no clear geographical division of the international and domestic airports in Iran and Nigeria [Figs. 14(a) and 14(b)].

Some core airports in Alaska, Ecuador, and Russia serve as gateway airports in the respective regions. In core-periphery pair 6 based in Alaska, core airport 1 is adjacent to all the other airports in this core-periphery pair [Fig. 14(c)]. In addition, only core airport 1 has an edge to the rest of the network (Table VII) and therefore is the unique gateway airport for this core-periphery pair. In Ecuador, most of the airports (10 airports; 83%) are adjacent to airport 1, which is the unique core airport in core-periphery pair 9 [Fig. 14(d)]. This core airport has most of the edges (10 edges; 77%) between core-periphery pair 9 and the rest of the network (Table VIII). Therefore, core airport 1 serves as a gateway airport in Ecuador. Airport 11 also functions as a gateway airport in Ecuador. The Russian airports belong to core-periphery pair 1, 2, or 7 [Fig. 14(e)]. Most of the airports in core-periphery pair 7 are located in Russian Far East. In core-periphery 7, all peripheral airports are adjacent to core airport 1. The core airport 1 has most of the edges (eight edges; 67%) between core-periphery pair 7 and the rest of the network (Table IX). Therefore, core airport 1 serves as a gateway airport for this core-periphery pair. There is no clear separation between the domestic and international airports into different core-periphery pairs in Russia.

[1] M. E. J. Newman, *Networks: An Introduction* (Oxford University Press, Oxford, 2010).

[2] A.-L. Barabási, *Network Science* (Cambridge University Press, Cambridge, 2016).

[3] S. Fortunato, Phys. Rep. **486**, 75 (2010).

[4] M. Girvan and M. E. J. Newman, Proc. Natl. Acad. Sci. USA **99**, 7821 (2002).

[5] M. E. J. Newman and M. Girvan, Phys. Rev. E **69**, 026113 (2004).

[6] L. A. Adamic and N. Glance, in *Proceedings of the 3rd International Workshop on Link Discovery* (ACM, New York, 2005), pp. 36–43.

[7] R. Guimerà, S. Mossa, A. Turtschi, and L. A. N. Amaral, Proc. Natl. Acad. Sci. USA **102**, 7794 (2005).

[8] M. Sales-Pardo, R. Guimerà, A. A. Moreira, and L. A. N. Amaral, Proc. Natl. Acad. Sci. USA **104**, 15224 (2007).

[9] B. Karrer and M. E. J. Newman, Phys. Rev. E **83**, 016107 (2011).

[10] R. Guimerà and L. A. N. Amaral, Nature **433**, 895 (2005).

[11] S. P. Borgatti and M. G. Everett, Soc. Netw. **21**, 375 (2000).

[12] P. Holme, Phys. Rev. E **72**, 046111 (2005).

[13] J. P. Boyd, W. J. Fitzgerald, M. C. Mahutga, and D. A. Smith, Soc. Netw. **32**, 125 (2010).

[14] P. Csermely, A. London, L.-Y. Wu, and B. Uzzi, J. Complex Netw. **1**, 93 (2013).

[15] S. H. Lee, M. Cucuringu, and M. A. Porter, Phys. Rev. E **89**, 032810 (2014).

[16] M. P. Rombach, M. A. Porter, J. H. Fowler, and P. J. Mucha, SIAM J. Appl. Math. **74**, 167 (2014).

[17] J. Yang and J. Leskovec, Proc. IEEE **102**, 1892 (2014).

[18] A. Ma and R. J. Mondragón, PLOS ONE **10**, e0119678 (2015).

[19] B. Tunç and R. Verma, PLOS ONE **10**, e0143133 (2015).

[20] X. Zhang, T. Martin, and M. E. J. Newman, Phys. Rev. E **91**, 032803 (2015).

[21] M. Cucuringu, P. Rombach, S. H. Lee, and M. A. Porter, Eur. J. Appl. Math. **27**, 846 (2016).

[22] J. Gamble, H. Chintakunta, A. Wilkerson, and H. Krim, IEEE Trans. Signal Inf. Process. Netw. **2**, 186 (2016).

[23] T. Verma, F. Russmann, N. A. M. Araújo, J. Nagler, and H. J. Herrmann, Nat. Commun. **7**, 10441 (2016).

[24] D. S. Bassett, N. F. Wymbs, M. P. Rombach, M. A. Porter, P. J. Mucha, and S. T. Grafton, PLOS Comput. Biol. **9**, e1003617 (2013).

[25] M. R. da Silva, H. Ma, and A.-P. Zeng, Proc. IEEE **96**, 1411 (2008).

[26] S. Bruckner, F. Hüffner, and C. Komusiewicz, Algo. Mol. Biol. **10**, 16 (2015).

[27] F. D. Rossa, F. Dercole, and C. Piccardi, Sci. Rep. **3**, 1467 (2013).

[28] B. Craig and G. von Peter, J. Financ. Intermed. **23**, 322 (2014).

[29] D. Fricke and T. Lux, Comput. Econ. **45**, 359 (2014).

[30] B. Yan and J. Luo, Preprint arXiv:1605.03286 (2016).

[31] D. Sardana and R. Bhatnagar, in *Proceedings of the 2016 IEEE/WIC/ACM International Conference on Web Intelligence* (IEEE, Nebraska, 2016), pp. 1–8.

[32] B.-B. Xiang, Z.-K. Bao, C. Ma, X.-Y. Zhang, H.-S. Chen, and H.-F. Zhang, (2016), preprint arXiv:1612.01704 (2016).

[33] J. I. Alvarez-Hamelin, L. Dall'Asta, A. Barrat, and A. Vespignani, in *Advances in Neural Information Processing Systems 18* (MIT Press, Cambridge, 2005), pp. 41–50.

[34] J. Cohen, Comput. Sci. Eng. **11**, 29 (2009).

[35] Y. Zhao, E. Levina, and J. Zhu, Proc. Natl. Acad. Sci. USA **108**, 7321 (2011).

[36] J. Chen and Y. Saad, IEEE Trans. Knowl. Data Eng. **24**, 1216 (2012).

[37] P. Erdős and A. Rényi, Publ. Math. Inst. Hung. Acad. Sci. **5**, 17 (1960).

[38] J. P. Boyd, W. J. Fitzgerald, and R. J. Beck, Soc. Netw. **28**, 165 (2006).

[39] D. in 't Veld and I. van Lelyveld, J. Bank. Finac. **49**, 27 (2014).

[40] U. N. Raghavan, R. Albert, and S. Kumara, Phys. Rev. E **76**, 036106 (2007).

[41] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, J. Stat. Mech. (2008) P10008.

[42] B. W. Kernighan and S. Lin, Bell Syst. Tech. J. **49**, 291 (1970).

[43] Z. Šidák, J. American Stat. Assoc. **62**, 626 (1967).

[44] M. Meilă, J. Multivariate Anal. **98**, 873 (2007).

[45] C. O. Flores, S. Valverde, and J. S. Weitz, ISME J. **7**, 520 (2013).

[46] T. P. Peixoto, Phys. Rev. E **95**, 012317 (2017).

[47] W. W. Zachary, J. Anthropol. Res. **33**, 452 (1977).

[48] J. Patokallio, "Openflights data" (2009) [http://openflights.org].

[49] T. Opsahl, "Why Anchorage is not (that) important: Binary ties and sample selection" (2011) [https://toreopsahl.com/2011/08/12/why-anchorage-is-not-that-important-binary-ties-and-sample-selection].

[50] J. Yang and J. Leskovec, in *Proceedings of the 2012 IEEE 12th International Conference on Data Mining* (IEEE, Washington, 2012), pp. 1170–1175.

[51] J. Yang and J. Leskovec, in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining* (ACM, New York, 2013), pp. 587–596.

[52] J. Reichardt and S. Bornholdt, Phys. Rev. Lett. **93**, 218701 (2004).

[53] J. Reichardt and S. Bornholdt, Phys. Rev. E **74**, 016110 (2006).

[54] R. Mastrandrea, T. Squartini, G. Fagiolo, and D. Garlaschelli, New J. Phys. **16**, 043022 (2014).

[55] V. A. Traag and J. Bruggeman, Phys. Rev. E **80**, 036115 (2009).

[56] M. MacMahon and D. Garlaschelli, Phys. Rev. X **5**, 021006 (2015).

[57] M. J. Barber, Phys. Rev. E **76**, 066102 (2007).

[58] P. Expert, T. S. Evans, V. D. Blondel, and R. Lambiotte, Proc. Natl. Acad. Sci. USA **108**, 7663 (2011).

[59] A. Arenas, J. Duch, A. Fernández, and S. Gómez, New J. Phys. **9**, 176 (2007).

[60] E. A. Leicht and M. E. J. Newman, Phys. Rev. Lett. **100**, 118703 (2008).

[61] Y. Kim, S.-W. Son, and H. Jeong, Phys. Rev. E **81**, 016103 (2010).

[62] M. E. J. Newman, Phys. Rev. E **94**, 052315 (2016).

[63] S. Fortunato and M. Barthélemy, Proc. Natl. Acad. Sci. USA **104**, 36 (2006).

[64] R. Guimerà, M. Sales-Pardo, and L. A. N. Amaral, Phys. Rev. E **70**, 025101 (2004).