The MIT Press

RESEARCH ARTICLE

Center–periphery structure in research communities

Eleanor Wedell[1] iD, Minhyuk Park[1] iD, Dmitriy Korobskiy[2] iD, Tandy Warnow[1] iD, and George Chacko[1,3] iD

[1]Department of Computer Science, University of Illinois Urbana-Champaign, Urbana, IL 61801
[2]NTT DATA, McLean, VA, 22102
[3]Office of Research, Grainger College of Engineering, University of Illinois Urbana-Champaign, Urbana, IL 61801

ABSTRACT

Clustering and community detection in networks are of broad interest and have been the subject of extensive research that spans several fields. We are interested in the relatively narrow question of detecting communities of scientific publications that are linked by citations. These publication communities can be used to identify scientists with shared interests who form communities of researchers. Building on the well-known k-core algorithm, we have developed a modular pipeline to find publication communities with center–periphery structure. Using a quantitative and qualitative approach, we evaluate community finding results on a citation network consisting of over 14 million publications relevant to the field of extracellular vesicles. We compare our approach to communities discovered by the widely used Leiden algorithm for community finding.

1. INTRODUCTION

We are interested in the structure of research communities that form and collaborate around research questions (Crane, 1972; Kuhn, 1970). Such research communities represent scientific specialization (Chubin, 1976; Morris & der Veer Martens, 2009). While prior studies have estimated the size of these research communities to be of the order of a few hundreds to a thousand (Crane, 1972; Kuhn, 1970; Morris, 2005; Mullins, 1985; Price, 1965), this question merits re-examination in an expanded, diversified, globalized, and electronically connected scientific enterprise. Accordingly, we would like to study research communities at scale.

From the perspective of scientometrics, detecting a research community can be framed as a community-finding problem in which communities of publications defining areas of research are discovered from the scientific literature. First, the scientific literature is modeled as a network with publications as nodes and citations as directed edges (Boyack & Klavans, 2019). In such a network, an area of research is defined by a community of publications—a sufficiently citation-dense area in the network. Because researchers can work on more than one problem and be members of more than one research community, we begin by finding communities of publications. Then, for each publication community, the authors of the publications in the community represent a researcher community.

Once constructed, citation networks can be analyzed using different community finding or clustering approaches (Ahlgren, Chen et al., 2020; Boyack & Klavans, 2019; Sciabolazza,

Vacca et al., 2017; Sjögarde & Ahlgren, 2018, 2020; Šubelj, van Eck, & Waltman, 2016; Traag, Waltman, & van Eck, 2019; Waltman & van Eck, 2012). Of these approaches, a recent development is the availability of the Leiden algorithm, which offers better partitioning and performance (Traag et al., 2019) compared to a preceding approach, the Louvain algorithm (Blondel, Guillaume et al., 2008).

The rich literature on community finding in complex networks is relevant (Fortunato, 2010; Fortunato & Castellano, 2009; Javed, Younis et al., 2018), particularly with modularity being proposed as a quality function (Newman, 2006). Here, community detection amounts to identifying groups within a complex network that share some common properties. However, as observed by Coscia, Giannotti, and Pedreschi (2011), imprecision is inherent in the definition of community-finding, given the diverse ways in which communities can be defined and a richness of perspectives. For example, a community detection approach may focus on vertex similarity or edge density; disjoint, overlapping or hierarchical community structure; and static versus dynamic networks. Thus, the context of the question being asked and the techniques being employed tends to determine the flavor of community detection in a study. In this study, we build upon this literature in focusing on edge density and disjoint communities.

Beyond detection, we are especially interested in substructure (structure within communities), as it reflects roles and dynamics among members. Price and Beaver (1966), in their study of the oxidative phosphorylation community, reported center–periphery substructure (Breiger, 2014): a small core of influential researchers and a much larger transient population. Core–periphery or center–periphery patterns have been reported in other networks using different techniques, such as block modeling and *k*-core decomposition (Borgatti & Everett, 2000; Gallagher, Young, & Welles, 2021; Malliaros, Giatsidis et al., 2019; Mullins, Hargens et al., 1977; Rombach, Porter et al., 2014, 2017), arguing for generality in their occurrence.

We have briefly explored substructure in an earlier study (Chandrasekharan, Zaka et al., 2021) where we developed an ensemble technique that combined the Leiden algorithm and the Markov Clustering (MCL) (Van Dongen, 2008) community-finding methods. The method was coupled to limited qualitative analysis and identified publication communities that were discovered by both Leiden and MCL. Subsequently, we identified the author communities of these publication communities in networks of biology literature. We were specifically interested in whether the communities we found exhibited substructure indicating the strong influence of a few researchers associated with the majority of publications. While we did detect center–periphery substructure in both publication and author communities, our findings were potentially limited by the clustering methods we used, Leiden and MCL, as neither is designed to detect substructure when identifying communities.

Here, we aim to investigate more carefully whether communities exhibiting center–periphery structure exist in citation networks of the scientific literature. The central idea is that each community contains core nodes representing the "center" of center–periphery organization and additional noncore nodes representing the "periphery"; our approach first finds the core nodes and then augments the cluster to include peripheral nodes.

To find the core nodes in a community, we combine two concepts represented in the clustering and community detection literature: First, that valid communities should have a positive modularity score (Fortunato & Barthelemy, 2007), and second that each community should be dense, which is expressed by the ratio between the average node degree and the number of nodes in the network. By distinguishing between core and noncore nodes, we can require that the positive modularity and sufficient citation density requirements hold for the subclusters of

core nodes but not necessarily for the clusters that combine both core and noncore nodes. This distinction potentially enables us to better model real world community structure.

Although optimizing the total modularity in a clustering has drawbacks (Fortunato & Barthelemy, 2007), we only require that the clusters individually have modularity scores above 0; this is a relatively mild criterion that seeks internal cohesion, and has been considered in the prior literature (e.g., Fortunato & Barthelemy, 2007; Newman & Girvan, 2004) to be evidence of a valid community.

Thus, our approach combines five ideas from the literature: the scientific enterprise organized into research communities, a center–periphery model, researcher community structure extended to a model of publication communities, positive modularity for individual communities, and communities defined by sufficient citation density.

Our modular pipeline is centered around finding dense clusters of core nodes, building on the ideas of Giatsidis, Thilikos, and Vazirgiannis (2011) who quantified the cohesiveness of a community using the *k*-core concept from graph theory (Matula & Beck, 1983). We tested this pipeline on a network of over 14 million publications that we constructed by harvesting citing and cited articles from a seed set defined by the keyword *exosome*. This keyword captures articles from the field of extracellular vesicles, which may be important for intercellular communication and development of some diseases, as well as having potential for therapy (Edgar, 2016; Kalluri & LeBleu, 2020; Raposo, van Niel, & Stahl, 2021). We chose extracellular vesicles (Raposo et al., 2021) as the focus of this study for two reasons: First, it is a large research area, and second, it is rapidly expanding—it has seen spectacular numbers of publications each year since 2010, and therefore represents an excellent test case for community-finding methods in the modern scientific enterprise.

Expecting that not all communities discovered in such a large network would be directly relevant to exosomes, we use 1,218 cited references from 12 recent review articles as expert-identified markers for specificity in the communities we discover. Any community containing at least one marker node is considered relevant and communities with significant numbers of markers are considered to be special interest. We report our findings in the following sections.

2. MATERIALS AND METHODS

2.1. Data

Citation networks of scientific literature can be constructed using different approaches. Direct citations were used to build clusters of articles from a data set of over 10 million publications (Waltman & van Eck, 2012); this methodology was also used in building citation maps from 19 and 43 million publications (Boyack & Klavans, 2014). Direct citation, bibliographic coupling, and cocitation have been compared for their relative value in identifying research fronts (Boyack & Klavans, 2010), with a hybrid approach involving bibliographic coupling and textual similarity performing the best. A subsequent study conducted at a larger scale and with improved evaluation criteria suggested that direct citation was the most promising (Klavans & Boyack, 2017). We use direct citations in this study.

A citation network consisting of 14,695,475 nodes and 99,663,372 edges was generated using the Dimensions bibliography (Hook, Porter, & Herzog, 2018). Briefly, a "seed" set, *S*, was obtained by performing a text search for the term "exosome" with years of publication restricted to 2010 or earlier. This constraint was applied to allow every element in the seed set to have accumulated at least 10 years of citations. The search retrieved 11,156 publications of type article from Dimensions. To capture publications proximal by citation to the seed set, a

network was constructed using a protocol we labeled *SABPQ*. First we start with the seed set *S*. Set *A* is the set of publications that cite at least one publication from set *S*. Similarly, set *B* is the set of publications that are cited by at least one publication from set *S*. Once the sets *S, A*, and *B* are identified, we define set *P* as the set of publications that cite at least one publication from the set *S* ∪ *A* ∪ *B*. Similarly, set *Q* is the set of publications that are cited by at least one publication from the set *S* ∪ *A* ∪ *B*. Thus, *S* and *A* are subsets of *P* and *B* is a subset of *Q*. The network contains directed edges defined by citations; if publication *x* cited *y* then we created an edge from *x* to *y*. The SABPQ protocol was implemented using Dimensions on BigQuery in Google Cloud Services. The data was then exported to Google Cloud Storage Bucket and subsequently exported to a PostgreSQL database for further analysis.

2.1.1. Marker nodes and specificity

As marker nodes for our analysis, we used 1,218 articles cited in 12 recent reviews on extracellular vesicles and exosome biology (Busatto, Morad et al., 2021; Clancy, Schmidtmann, & D'Souza-Schorey, 2021; Ghoroghi, Mary et al., 2021; He, Hamby, & Jin, 2021; Kalluri & LeBleu, 2020; Lananna & Imai, 2021; Le Lay, Rome et al., 2021; Leidal & Debnath, 2021; Raposo et al., 2021; Schnatz, Müller et al., 2021; van Niel, D'Angelo, & Raposo, 2018; Verdi, Bécot et al., 2021). All 1,218 markers are present in our 14,695,475 nodes network. Marker nodes were matched to clusters using identifiers in our network. For example, the marker node with title "Tumour exosome integrins determine organotropic metastasis" and DOI 10.1038/nature15756 is identified as node 4431204 in our network. The complete list of marker nodes is available on our Github site (Park et al., 2021).

2.2. Clustering Methods

2.2.1. Leiden

We used version 1.1.0 of the Java implementation for the Leiden algorithm (Traag et al., 2019) provided by the Centre for Science and Technology Studies and available in Github (Traag, 2021). We ran Leiden in default mode, which means that the quality function being optimized was the Constant Potts Model rather than modularity. Leiden includes a parameter for the resolution value, which we vary in our experiments from 0.0001 to 0.95.

2.2.2. New clustering methods

Here we describe at a high level the clustering and community-finding methods we developed in our study. The methods we developed and use in this study are freely available in Github, and the locations of the software and exact commands we used are provided in the Supplementary materials. We also provide software version numbers and commands for the existing code that we use in the Supplementary materials. We note here that the code we developed relies on NetworKit (Staudt, Sazonovs, & Meyerhenke, 2016), which is an open-source Python module designed for scalable network analysis.

In our approach, the objective was to produce a set of clusters, each of which has core nodes and peripheral nodes consistent with the "center–periphery" structure described earlier. These clusters are considered to be "publication communities," with two types of members: core members that are densely connected to each other and peripheral (noncore) members connected to the core members but with fewer edges within a cluster.

Our approach requires values for *k* and *p*, where *k* specifies a minimum connectivity between the core nodes, and *p* indicates a minimum connectivity between each noncore

("periphery") node and the core nodes. These parameter values for *k* and *p* are provided by the user, and different choices for these parameters will produce different clusterings.

The required minimum connectivity *k* between core nodes is related to the *k*-core concept in graph theory, which we now describe. A *k*-core of a network *N* is a largest connected subnetwork *A* of *N* such that every node in *A* is adjacent to at least *k* other nodes in *A* (Matula & Beck, 1983; Pittel, Spencer, & Wormald, 1996; Seidman, 1983). The *k*-cores can be calculated in polynomial time (Matula & Beck, 1983), and our new clustering methods build on these algorithms.

We are also interested in the modularity scores of the clusters that are produced by each method, as given in Definition 1:

Definition 1. The modularity of a single cluster *s* within a network *N*, denoted by mod(s), is given by

$$mod(s) = \frac{l_s}{L} - \left(\frac{d_s}{2L}\right)^2$$

where l_s is the number of internal edges in cluster *s*, d_s is the sum of the degrees of the nodes inside *s*, and *L* is the number of total edges in the network *N* (Fortunato & Barthelemy, 2007). The total modularity of a clustering is the sum of the modularity scores of its clusters.

Rather than aiming to maximize the total modularity of the clustering, we will only require that each cluster have positive modularity; as noted in Fortunato and Barthelemy (2007), this approach aims to detect valid communities. We now define some additional terms that we will use in designing new clustering methods:

Definition 2. Given a network *N*, a clustering *C*, and a cluster *C* drawn from *C* where *C* is partitioned into core nodes and noncore nodes, we will say

- *C* is *k*-valid if and only if each core node in *C* is adjacent to at least *k* other core nodes in *C*.
- *C* is *m*-valid if and only if the subcluster induced by the core nodes is connected and has a positive modularity score.
- *C* is *p*-valid if and only if each noncore node is adjacent to at least *p* core nodes in *C*.
- *C* is *kmp*-valid if and only if it is *k*-valid, *m*-valid, and *p*-valid.
- The clustering *C* is *kmp*-valid if and only if every cluster *C* ∈ *C* is *kmp*-valid.

Note that if a cluster does not contain noncore nodes then it is *vacuously p*-valid.

The clustering methods that we develop seek to produce *kmp*-valid clusters, so that we can interpret these clusters as communities with center–periphery structure. Moreover, we are interested in clusterings that produce a large number of *kmp*-valid clusters, as well as those that include as many nodes as possible in the *kmp*-valid clusters (which by definition must be nonsingleton when *k* > 1). Hence we explore different techniques that seek to optimize these two opposing criteria.

We also require positive modularity in the core node subclusters for each cluster. This is a relatively mild requirement that avoids cases where the core node subcluster may be *k*-valid and connected but might not reflect a preference for itself over the outside. Consider the case where a 10-clique, a complete graph on 10 nodes, is contained in a clique with 20 nodes. This 10-clique would satisfy *k*-validity for *k* ≤ 9 and would be connected, but would not have positive modularity. By enforcing positive modularity, we would avoid returning such clusters. This example illustrates the advantage of enforcing positive modularity even though the

probability of it occurring in a real-world network is likely to be small. We also note that enforcing positive modularity in the core node subcluster (or even in the final cluster that contains both core and noncore nodes) is not the same as trying to maximize the sum of the modularity scores of the individual clusters (total modularity score). In other words, enforcing positive modularity does not have the same vulnerability to the resolution limit that was established for the modularity criterion, which seeks to maximize the total modularity score (Fortunato & Barthelemy, 2007).

2.2.3. *Four-stage kmp-clustering*

We designed a four-stage pipeline that is designed to enable the user to explore different clustering options and guarantee that the output is a *kmp*-valid clustering. The input is a network N and values for the parameters k and p. At a high level, the first stages aim to construct the core member components. The second stage extracts valid subclusters from those generated by the first stage. The third stage augments these clusters with additional members, most likely noncore members, though some might qualify as core members, and the fourth stage assigns core or noncore status to the nodes, and retains only those clusters that are *kmp*-valid. We begin with a description of the overall multistage structure of our new clustering methods; note that stages 2 and 3 are optional.

- Stage 1: Cluster the network N into disjoint clusters (core members), so that each non-singleton cluster is k-valid and m-valid.
- Stage 2: Attempt to break each nonsingleton cluster produced in Stage 1 into a set of pairwise disjoint clusters, each of which is k-valid at minimum.
- Stage 3: For each nonsingleton cluster, add unclustered nodes (nodes that are not in any nonsingleton cluster) as noncore (peripheral) members, provided that they are adjacent to at least p core nodes in the selected cluster. This is the *augmentation* stage, which adds noncore nodes to the clusters produced in the earlier stage.
- Stage 4: Process the clustering that is received so that each final cluster is partitioned into core and noncore members, and so that the clustering is *kmp*-valid.

Thus, Stages 1 and 2 together are directed at finding core members of clusters, with Stage 1 directed at clusters with large numbers of core nodes and Stage 2 aimed at extracting smaller clusters within these larger clusters. At the end of Stage 1, all clusters will be k-valid and m-valid. If the optional Stage 2 is applied, the clusters it produces will be k-valid and connected, but may no longer have positive modularity. Neither of these stages introduces any noncore nodes, so the output of each stage is vacuously p-valid. Stage 3 augments the clusters to include noncore nodes; by design the clusters will be connected, k-valid, and p-valid, but depending on the outcome of Stage 2, they may not have positive modularity and so may not be m-valid. Stage 4 is designed to ensure that all final clusters are *kmp*-valid, and so may modify or discard clusters found in the earlier stages. However, after Stage 4 is run, the output clustering is guaranteed to be *kmp*-valid. Furthermore, the clusters produced are parsed into core and noncore nodes.

We now describe the techniques we have developed for each stage. For Stage 1, we present iterative k-core (IKC) clustering, a method that is inspired by the k-core concept in graph theory. For Stage 2, we also present two different techniques: recursive Graclus (RG) and iterative Graclus (IG), both of which are based on the Graclus (Dhillon, Guan, & Kulis, 2007) clustering method used in its default setting and applied to split a graph into two parts. Stage 3 is implemented using a straightforward algorithm that we describe below. In contrast to the earlier

stages, Stage 4 involves multiple steps, and is described below. This multistage design provides a flexible framework by allowing different techniques to be used at each stage.

2.2.4. *Stage 1: Iterative* **k**-*core clustering (IKC)*

To motivate the IKC algorithm, we first describe the technique that computes k-cores and then describe the iterative method.

2.2.4.1. ***k*-core**   For a network N and specified positive integer value for k, a k-core of a network N is a maximal connected subnetwork A of N such that every node in A is adjacent to at least k other nodes in A. Note that for every network that does not have any isolated vertices (i.e., nodes of degree 0), each connected component is a 1-core of the network. The distribution of k-core sizes also indicate how quickly a network shrinks as k increases (Leskovec & Horvitz, 2008).

The identification of the k-core for a maximum achieved value of k in a network has been proposed as a quality measure for community-finding that measures cohesiveness and suggests collaboration (Giatsidis et al., 2011). This idea is reiterated by Kong, Shi et al. (2019) and Malliaros et al. (2019), who discuss applications of the k-core in biology and real-world networks.

The k-cores can be calculated in polynomial time (Matula & Beck, 1983), as follows. First, we calculate the degree of every node in the network. Then, every node of degree less than k is deleted from the graph, and this process repeats until every node has degree at least k (i.e., every node is adjacent to at least k other nodes). Every connected component that remains is called a k-core of the network. As an example, given a network that contains two connected components, one is a clique of size 100 and the other has a node x that is adjacent to 2,000 other nodes, each of which is only adjacent to x (and so has degree 1). Note that for all k with $2 \leq k \leq 99$, the k-core of this network is the clique of size 100.

2.2.4.2. ***k*-core clustering**   The simple k-core clustering method takes as input a network N and a value for k, and computes the k-cores of the network. The set of the k-cores is returned as the clustering. By construction, the simple k-core clustering method produces clusters that are k-valid and connected. However, it does not constrain the clusters to have positive modularity.

2.2.4.3. **Iterative *k*-core clustering**   To improve on the simple k-core clustering technique for our purposes, we developed an iterative k-core (IKC) algorithm. The input to the IKC algorithm is a network N and a positive integer k. IKC then operates as follows:

- We will construct a bin B of clusters that will be returned by the IKC clustering. In this step, we initialize B to be the empty set.
- We run the k-core labeling algorithm, which labels every node in the network with a nonnegative integer. We let L be the largest label found in this labeling. If $L < k$, then IKC exits, and returns the clusters in the bin B. Otherwise, the L-cores (i.e., the connected components that are labeled by L) of the network are evaluated as potential clusters.
- An L-core A is added to the bin B if and only if A has positive modularity.
- The L-core is then deleted from the network, and the residual network is recursively analyzed by IKC. The stopping condition is when all the nodes have been deleted from the network.

This procedure produces a collection of clusters, each of which has the following properties: (i) each cluster is connected and has positive modularity, and hence each cluster is

m-valid; and (ii) each cluster is *k*-valid, which means every node in the cluster is adjacent to at least *k* other nodes in the cluster.

2.2.5. Stage 2: Finding clusters within clusters using Graclus

In Stage 2 we seek to discover *k*-valid clusters that exist within larger *k*-valid clusters. For this, we use the Graclus clustering method (Dhillon et al., 2007). Graclus has been previously used to cluster bibliometric data (Devarakonda, Korobskiy et al., 2020; Dhillon et al., 2007; Šubelj et al., 2016), but here we use it to split a given cluster into two subclusters. We use Graclus to optimize its default criterion, which is the normalized cut criterion. In this setting, we seek a partition of a given cluster C into two parts C_1 and C_2 so as to minimize

$$\frac{links(C_1, C_2)}{links(C_1, C)} + \frac{links(C_1, C_2)}{links(C_2, C)},$$

where *links*(*A*, *B*) denotes the number of edges with one endpoint in *A* and the other endpoint in *B*.

As is the case in other optimization methods, local search in Graclus helps the optimizer escape poor local minima; its default setting does no iterations but this can be modified by specifying the number of iterations. In this study we explore both the default mode, $l = 0$ (no iterations) and $l = 2,000$ iterations. We implement our use of Graclus in two different ways: recursively and iteratively. Thus, we use Graclus in four different ways.

2.2.5.1. Recursive Graclus   The input to this method is a clustering of the network N and the value for the parameter k. We create a bin B of clusters (setting it initially to the empty set). We take a cluster C from the clustering and apply Graclus recursively using either the default mode or the local search mode.

The result of this is a division of the cluster C into two nonempty sets A_1 and A_2. If A_1 has positive modularity and is *k*-valid, then we add A_1 to B (the bin we have created), and similarly for subset A_2. If neither A_1 nor A_2 is added to B, add cluster C to B and delete C from the network, effectively removing it from further consideration by RG.

The stopping condition for RG is that all nodes have been deleted from the network. When the stopping condition is reached, the final output of RG is the set of clusters in bin B. Though all clusters that are produced by RG are guaranteed to be *k*-valid and have positive modularity, they may not be *m*-valid, as this requires that the core node subsets be connected.

2.2.5.2. Iterative Graclus   To use Graclus iteratively, we follow a similar procedure as in RG but with two key differences. The first is that the user provides a parameter for the number of iterations, so that the procedure must stop after that number of iterations, if it hasn't already stopped. The second difference is that, in contrast to RG, the procedure is guaranteed to produce clusters that are *k*-valid and *m*-valid in each iteration, as we now describe.

When we apply Graclus, in either default or local search mode, to split a cluster into two subclusters, each of the created subclusters is parsed into core and noncore nodes (using a variant of the algorithm described for Stage 4; see Section 2.2.8). If the core node set is empty, the subcluster will be discarded. However, if the core is nonempty, then the core node set is by definition *k*-valid, and is then evaluated further. Each core node set is divided into its connected components, and each connected component that has positive modularity is passed

to the next iteration. Any cluster that does not end up producing a subcluster that is passed to the next iteration is added to the bin B as in RG and is then deleted from the network. IG stops when one of two conditions occurs: The number of allowed iterations has been completed, or all the nodes have been deleted from the network. By design, the output of IG is a set of clusters each of which is k-valid and m-valid (thus, every cluster is connected and has positive modularity, and every node is adjacent to at least k nodes in its cluster).

2.2.6. Stage 3: Augmentation

The purpose of the augmentation step is to assemble the periphery of center–periphery structures. The input to Stage 3 is a set of clusters, so that each nonsingleton cluster is k-valid and m-valid. Here we allow all nodes that are not in any nonsingleton cluster to be added to some cluster as long as it is adjacent to at least p core nodes in the (single) cluster to which it is added. In this study, we set $p = 2$ to ensure that we captured publications that are linked by cocitation or bibliographic coupling to core nodes in a community. If no such cluster exists such that a node can be added to it in a p-valid manner, the node remains unclustered.

We add x to the cluster C that maximizes

$$\frac{N_C(x)}{|C|}$$

where $N_C(x)$ is the number of core node neighbors of node x in cluster C and where $|C|$ denotes the number of nodes in cluster C. In other words, we add node x to the cluster where x has proportionally the most core node neighbors.

As an example, suppose C_1 and C_2 are clusters of core nodes and that x is not yet added as a noncore node to any cluster. Suppose x has five neighbors in cluster C_1 and 10 nodes in cluster C_2, where $|C_1| = 1000$ and $|C_2| = 20$. This procedure would add x as a noncore member to C_2 because 50% of the nodes in C_2 are neighbors of x while only $5/1000 = 0.5\%$ of the nodes in C_1 are neighbors of x.

2.2.7. Stage 4: Parsing clusterings to produce kmp-valid clusters

Although Stage 2 is guaranteed to produce k-valid clusters, it does not always produce m-valid clusters. Furthermore, the impact of Stage 3 (the augmentation step) is to add nodes to clusters that can participate as noncore nodes, and it is possible for a node added to a cluster in Stage 3 to have sufficient neighbors in its cluster to qualify for core membership. Hence, the result of these three stages is a set of clusters that needs to be "parsed" in order to know which nodes are core members, which nodes are noncore members, and whether the clusters are kmp-valid (as defined in Definition 2).

2.2.7.1. Parsing and modifying a single cluster
Here we describe how we perform this parsing on a given cluster C (taken from a clustering $\mathcal{C}$) and values for k and p.

- Step 1: We label every node in the cluster C using the k-core labeling algorithm, applied only to the subnetwork defined by C. We let C' denote the subset of nodes in C, each of whose labels is at least k, and we put the remaining nodes into a bin $B(C)$.
- Step 2: We compute the connected components of C', and delete the components that do not have positive modularity; the retained components thus have positive modularity, and are referred to as C-derived clusters (to indicate their derivation from the original

cluster C). The clusters produced in this step will be the core node members in the final clusters we produce at the end of Step 3.

- Step 3: We augment the C-derived clusters using the bin $B(C)$ (i.e., we find noncore members to add) as follows. We examine each node x in bin $B(C)$ to see if it has at least p neighbors in at least one C-derived cluster; if so, we select the best C-derived cluster A (using the algorithm from Stage 3) and add the node x to A_{nc} (where "nc" refers to "noncore"). After all nodes are examined and processed, we let $A' = A \cup A_{nc}$ (for each C-derived cluster A) and output the set of all such clusters A' as the "final clusters" derived from cluster C. Note that this process indicates the parsing of each final cluster A' into core (A) and noncore (A_{nc}) nodes.
- Step 4: We return all the final clusters derived from C.

Theorem 1. *For any clustering $\mathcal{C}$ of a network N, and any positive integers for k and p (with $p < k$), the output of kmp-processing is a clustering that is kmp-valid. Therefore, the output of the four-stage clustering method is kmp-valid for all networks N and values for k and p.*

Proof. Let $\mathcal{C}$ be an arbitrary clustering. Hence, some of its clusters may not be k-valid, may not be connected, and may not have positive modularity. We will prove that after the *kmp*-processing, all the clusters are *kmp*-valid. Specifically, we will prove that the parsing produced in Step 3 into core and noncore satisfies *kmp*-validity.

First, note that by construction the clusters (referred to as C') that are produced in Step 1 have the property that every node in these clusters is adjacent to at least k other nodes in their cluster. Hence, treating each cluster as only containing core nodes, these clusters are k-valid. In Step 2, these clusters are divided into components and the components are retained only if they have positive modularity; hence the C-derived clusters that are produced are k-valid and m-valid, under the interpretation that they contain only core nodes. In Step 3, the C-derived clusters are augmented. This augmentation step maintains connectivity, so the final clusters are connected. It remains to establish that after parsing any final cluster into core and noncore nodes, the final cluster would be k-valid (i.e., every core node would be adjacent to at least k other core nodes), p-valid (i.e., every noncore node would be adjacent to at least p core nodes), and the core node subcluster would have positive modularity, and hence also be m-valid.

Let A' be some final cluster. By construction, it is formed by taking a C-derived cluster A produced in Step 2, and then augmenting it. Thus, $A' = A \cup A_{nc}$, where A_{nc} is the set of nodes that are added during the augmentation step. We will establish that applying Stage 4 *kmp*-parsing to this cluster A' will not change its decomposition into core and noncore (i.e., A will still be the core nodes and A_{nc} will be the noncore nodes). Note that by construction, all the nodes in A are adjacent to at least k other nodes in A. Hence, when A' is *kmp*-parsed, the nodes that are identified as core nodes will contain all the nodes in A and then possibly some nodes from A_{nc}. Independent of whether there are new core nodes, A' will be p-valid. If there are no new core nodes, therefore, then A' will be *kmp*-valid. Here we show that in fact no node in A_{nc} will be labeled as core, so there are no new core nodes.

Let $x \in A_{nc}$ with A a C-derived cluster. Hence, x is drawn from bin $B(C)$. By Step 1, the label assigned to x during Step 1 (when the nodes in cluster C were labeled) was a value L_1 that is strictly less than k. Because A is a C-derived cluster, $A' \subseteq C$. Consider the label L_2 assigned to x by the k-core labeling of A'. Because $A' \subseteq C$, it follows that $L_2 \leq L_1$. Because $L_1 < k$, it follows that $L_2 < k$. Hence, x will not be placed in the core for A', and the final clusters output in Step 4 are therefore *kmp*-valid. □

2.2.8. Additional uses of *kmp*-parsing

The Stage 4 *kmp*-parsing routine is also used in modified forms in other aspects of this study:

- Strict *kmp*-parsing: This strict parsing routine evaluates each cluster in a clustering for being *kmp*-valid: Those clusters that are *kmp*-valid are retained and all others are discarded. We use this strict parsing routine to evaluate Leiden.
- Using *kmp*-parsing to extract core node clusters: We also have a variant where use the parsing to extract only the core nodes within each cluster, and then return the components of the core node subclusters that have positive modularity score; this variant is used within IG.

2.3. Multidimensional Scaling Analysis

We used Multidimensional Scaling analysis (MDS) to visualize clusters of marker nodes. For the distance between marker nodes *x* and *y*, we calculated the number of clusterings in which *x* and *y* were not in the same cluster. Using the matrix of pairwise distances, we produced a two dimensional visualization using metric MDS. See the Supplementary materials for additional details.

3. RESULTS AND DISCUSSION

3.1. Properties of the Citation Network

Exosomes are an area of investigation within the larger field of extracellular vesicles in biology (Kalluri & LeBleu, 2020; Raposo et al., 2021). This field has been exponentially expanding, as evidenced by a keyword search for *exosome* in the Dimensions bibliography yielding a count of less than 100 publications in 1990 and earlier, 11,100 publications from 1991 through 2010, and 115,300 publications from 2011–2021 (rounded). To find communities in this research area, we constructed a large citation network that captured articles concerning exosomes and extracellular vesicles and articles proximal to them through citation.

The citation network we built consisted of 14,695,475 publications in 13 components, of which the largest component accounts for 99.998% of the network. The network was generated through amplifying a seed set of 11,156 articles (Section 2). The degree distribution of the nodes in this network is typical of citation networks with a few nodes of high degree and many nodes of low degree (Figure 1). Roughly 68% of the nodes have degree at most five and the 90th, 95th, and 99th percentiles of degree counts are 6,113, 9,186, and 24,510. The highest degree is 256,836 and corresponds to an article describing an assay for protein measurement.

The nodes in this data set were roughly distributed by year of publication as follows: 1990 or earlier (1.17 million), 1991–2010 (6.05 million), and 2011–2021 (6.99 million), suggesting not only a rapid growth of exosome publications in the post-1990 period but also a substantial increase in publications linked by citation to the seed articles.

3.2. Results of Clustering Methods

We now present results using different clustering methods, including the *k*-core clustering method, iterative *k*-core clustering, our four-stage pipeline, and the Leiden algorithm. As noted earlier, we were explicitly interested in discovering citation-dense regions with center-periphery structure that reflect cohesiveness and collaboration (Breiger, 2014; Giatsidis et al., 2011). In this case, collaboration refers to the recognition of prior work by others in
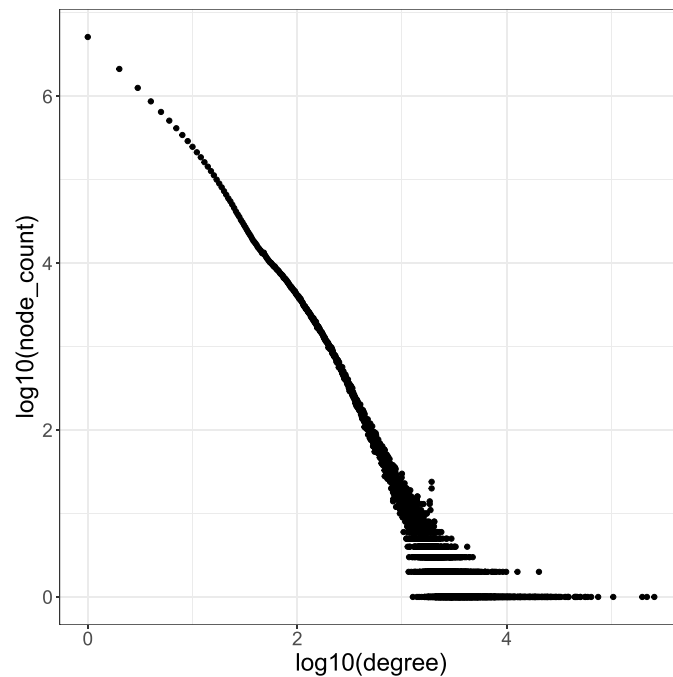
Figure 1. Intra-network degree distribution of the 14,695,475 nodes in the exosome citation network. *x*-axis: log degree, *y*-axis: log node count.

the community through citations. We were not interested in communities that consisted of a single heavily cited article or a single article that cited many references (Chandrasekharan et al., 2021, p. 193).

3.2.1. Results for the Leiden algorithm

We first used the Leiden algorithm, which guarantees connected communities (Traag et al., 2019), as a benchmark for community finding. We consider Leiden as a reference method in the scientometrics community, and have previously used it (Chandrasekharan et al., 2021) in combination with Markov Clustering (Van Dongen, 2008) to detect communities in the immunology and ecology literature. In the present study, we used the Leiden software (Traag, 2021) with the Constant Potts Model as quality function to cluster the exosome citation network. The resolution factor is designed to modify the clustering, as it determines the required minimum density within communities, and we varied it from 0.0001 to 0.95 (see Table 1). For each resolution factor value, we examined the number of clusters and their sizes, as well as the node coverage, which is defined to be the fraction of the nodes in the entire network that appear in a nonsingleton cluster. We also examined these statistics after restricting the clusters produced by Leiden to those that are *kmp*-valid for $k = 5$ or 10 and $p = 2$ (Section 2, Definition 2).

At the smallest resolution value we examined (0.0001), Leiden produced 12,266 nonsingleton clusters and 1,472,233 singletons, the median cluster size was 375, the largest cluster size was 70,001, and the node coverage was 90% (see also Supplementary materials, Table 2). Thus, at the smallest resolution value, there is excellent node coverage with a wide range in cluster sizes, including one large cluster. However, when restricting to the *kmp*-valid clusters and setting $k = 5$ and $p = 2$, there were only 69 clusters (median size 20,522) and the node coverage was only 11.5%, indicating a very large drop in number of clusters and node coverage (Supplementary materials, Table 3). There was an even bigger drop in number of clusters

Table 1. Cluster statistics for Leiden under different resolution values. Node coverage, expressed as percentage, refers to the the ratio of nodes in nonsingleton clusters to the total number of nodes in the network. # of clusters refers to the number of nonsingleton clusters. The last three columns refer to the sizes of nonsingleton clusters. Results shown here are before *kmp*-processing.

Resolution	Node coverage	# Clusters	# Singletons	Min.	Median	Max.
0.0001	90.0%	12,266	1,472,233	2	375	70,001
0.001	85.6%	67,304	2,367,965	2	98	20,802
0.01	74.9%	276,050	3,689,286	2	29	3,510
0.05	57.0%	488,285	6,323,695	2	17	960
0.25	15.6%	481,780	12,408,446	2	4	192
0.50	8.7%	434,973	13,412,183	2	2	97
0.75	8.2%	489,937	13,496,111	2	2	39
0.95	8.1%	497,757	13,499,132	2	2	16

and node coverage when setting $k = 10$ and $p = 2$, where the number of clusters was 18 and node coverage was 4.5% (Supplementary materials, Table 4). Thus at this resolution value, while Leiden does find clusters that are *kmp*-valid and so exhibit substructure, most of the clusters it finds are not *kmp*-valid.

Results for other resolution values showed similar trends, but increases in resolution value very quickly decrease the node coverage and median cluster size and increase the number of singleton clusters; furthermore, when restricted to *kmp*-valid clusters, the node coverage and median cluster size drop very quickly. As an example, using resolution value 0.10 and setting $k = 5$ and $p = 2$, Leiden produces 5,393 clusters with median size 58, more than 14 million nodes are in singleton clusters, and the overall node coverage is 2.6% (setting $k = 10$ produces 2,961 clusters and node coverage of 2%). Further increases in the resolution value drop the node coverage and the median and maximum cluster sizes, and increase the number of singleton clusters. Thus overall, reasonable results in terms of node coverage when restricted to *kmp*-valid clusters are obtained only for the small resolution values (between 0.0001 and 0.01), and range from 7.3% to 13.8% for $k = 5$ and from 4.5% to 8.8% when $k = 10$.

In summary, while Leiden was able to efficiently cluster our network into communities that represented between 8% and 90% of the network, depending on the resolution value employed, only a small fraction of these communities exhibited *kmp*-validity. This is not surprising, because the Leiden algorithm and optimization criterion were not designed to produce *kmp*-valid communities, but these trends indicate that Leiden has some limitations for our purposes.

3.2.2. Results for k-core clustering

To identify citation-dense regions of the network, we ran the *k*-core clustering method on our network for different values of $k \geq 1$ (Figure 2). The largest value of k for which a cluster is returned is 56, indicating that the degeneracy of the network is 56. The network has no isolated vertices, so $k = 0$ and $k = 1$ produce the same output, which is 13 clusters, each corresponding to a connected component in the network. When $k = 2$, two clusters were returned. For each $3 < k \leq 56$, only a single cluster was returned; hence only $k = 0, 1, 2$ produced more than one cluster, and in each of these cases, a single cluster dominated in size.
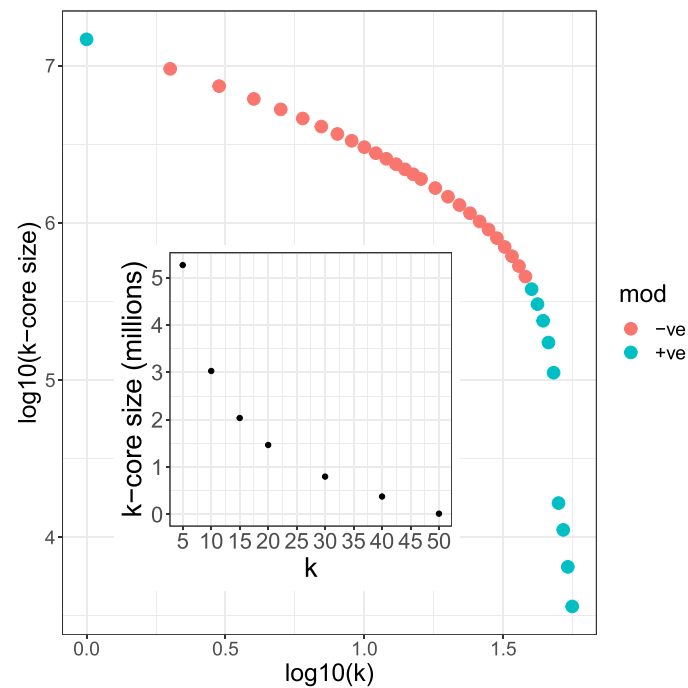
Figure 2. Cluster sizes using the simple *k*-core clustering algorithm. The input exosome citation network (Section 2) consists of 13 components summing to 14,695,475 nodes. Of these, a single component accounts for 14,695,226 articles (99.998% of the network). The *k*-core clustering algorithm was applied to the exosome citation network for multiple values of *k* (*x*-axis). The size of the *k*-core is shown on the ordinate (*y*-axis). A single component is returned in each case, with the exception of $k = 1$ and $k = 2$. Node coverage decreases as *k* increases, and is approximately 36% and 21% when $k = 5$ and $k = 10$ respectively. The single cluster at $k = 56$ has 3,630 nodes, amounting to node coverage of 0.02%. At $k = 57$ or greater, the entire network dissolves, thus the degeneracy of the network is 56. Clusters shown have positive modularity (*mod + ve*) when $k = 1$ or $k \geq 40$ and are colored teal. By definition, the connected components of the network are the 1-cores so 13 clusters are returned for $k = 1$. For $k = 2$, only two clusters are returned. Clusters of size 100 or less (0.001% of the network) are not included in this plot and pertain only to $k = 1$ or $k = 2$. *Inset*: *k*-core sizes sampled at intervals of five are displayed using a linear scale.

An interesting trend in this analysis is how *k* impacts the modularity scores of the clusters. By definition, the components in the graph all have positive modularity, so when $k = 1$ the clusters all have positive modularity. For larger values of *k*, the modularity scores do not become positive until $k = 40$, and then all subsequent values of *k* produce clusters with positive modularity scores.

Cluster size decreased monotonically as *k* increased to 56. However we observe a pattern of relatively stable core sizes at lower values of *k* followed by a more rapid decrease as *k* increases above 40. Leskovec and Horvitz (2008) reported similar findings about changes in core sizes on a much larger network of instant messaging data that consisted of 180,000,000 nodes. These authors suggest that the rapid decrease in core size occurs once nodes on the fringe of the network are removed. On our network, this more rapid decrease of core sizes also coincides with the appearance of positive modularity of clusters; modularity was not reported in Leskovec and Horvitz (2008). Thus, we are mainly interested in the *k*-cores for large values of *k*: they have positive modularity, small changes in *k* result in large changes to their sizes, and they have been identified in the prior literature as what is left after the fringe of the network is removed.

While the simple *k*-core clustering method identifies citation-dense areas in the network, suggesting cohesiveness and collaboration, it has two limitations: It does not ensure positive modularity for every cluster it produces, and, on our data, for every $k \geq 3$ it produced only a single large cluster. We note that for $k = 0, 1, 2$, it produced two or more clusters, one of which was very large. These limitations impact the ability to find multiple communities of interest in the exosome literature, especially considering the possibility that some of the communities of interest could be contained within larger clusters with nonpositive modularity.

3.2.3. Results for iterative k-core (IKC)

We designed iterative *k*-core (IKC) to improve on the *k*-core clustering algorithm. The input to IKC is the network *N* and the parameter *k*. The first cluster that is found in the network is the *L*-core where *L* is the largest label assigned to any node. If $L \geq k$, then the *L*-core is produced as a cluster and removed from the network, and otherwise the algorithm stops. If the algorithm has not stopped, it is run recursively on the reduced network. Therefore, IKC(*k*) will contain all the clusters in IKC(*k'*) for $k \leq k'$.

We explored IKC varying *k* between 5 and 50, and examined the distribution of cluster sizes generated. We also recorded the minimum degree in each cluster. Because IKC clusters satisfy *k*-validity, the nodes in the clusters are all "core" members, so this minimum degree is also the Minimum Core Degree (MCD) of the cluster. As expected, increasing *k* results in decreases in cluster size and increases in the MCD value (Figure 3). To include as much of the network as is reasonable and still have sufficient density to define community structure, we selected $k = 5$ and $k = 10$ for IKC.
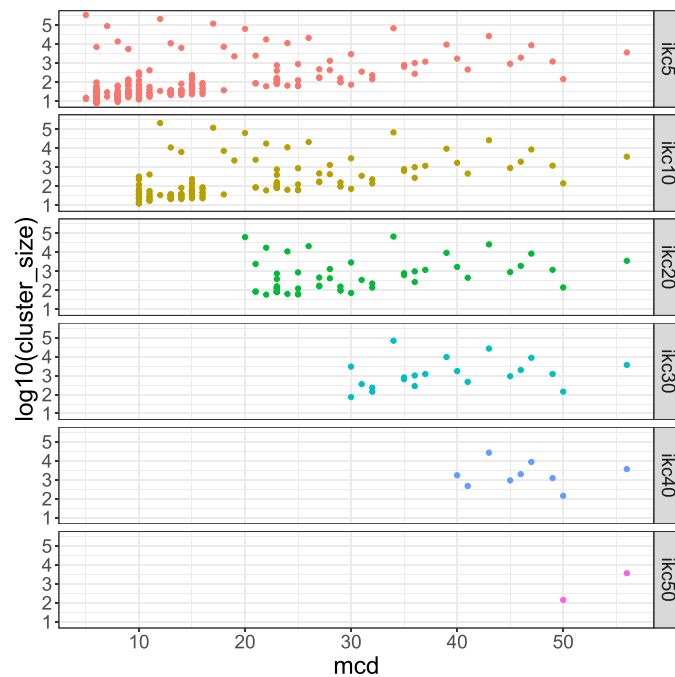

Figure 3. Empirical statistics of the iterative *k*-core (IKC) clustering methods (varying *k*) on the exosome network. The *y*-axis shows the cluster sizes (logarithmic scale) and the *x*-axis shows the Minimum Core Degree (MCD) values, where the MCD of a cluster is the minimum degree of any node in the cluster. By design, IKC(*k*) contains all the clusters of IKC(*k'*) if $k \leq k'$; thus, the panels showing IKC at lower values of *k* contain greater numbers of clusters.

While IKC discovered *km*-valid communities that were trivially *p*-valid, it too has limitations. It identifies only core nodes with modest node coverage of 7.38% and 4.22% at $k = 5$ and $k = 10$, respectively. It also generates some large clusters with lower MCD values that leave open the question of whether denser *kmp*-valid communities exist within them.

3.2.4. Results for the four-stage clustering pipeline

To address the limitations of IKC, we designed a four-stage pipeline (Figure 4). This four-stage pipeline is guaranteed to produce a *kmp*-valid clustering of the network for user-provided values of k and p.

In Stage 1 we use IKC with $k = 5$ and $k = 10$. Stage 2 breaks the clusters found in Stage 1 into smaller clusters; for this stage we apply either RG or IG, and for each of these we use Graclus either in default mode or in local search mode. Thus, for each setting of k we have four versions of the four-stage pipeline: Iterative and RG run in either default or local search mode. In Stage 3, the clusters produced by Stage 2 are augmented by the addition of nodes that satisfy *p*-validity for $p = 2$. In Stage 4, we parse the output of Stage 3 and retain only those clusters that are *kmp*-valid.

We show results for the different versions of this four-stage pipeline in Table 2, where the rows correspond to different ways of setting k and running Stages 2 and 3. Using IKC alone without Stages 2 and 3 resulted in 276 and 119 nonsingleton clusters and had total node coverage of 7.38% and 4.22%, with maximum cluster sizes of 345,139 and 213,670, for $k = 5$ and $k = 10$, respectively. Skipping Stage 2 (breaking down clusters) but adding Stage 3 (augmentation) greatly increases the node coverage to at least 33% for both settings of k but
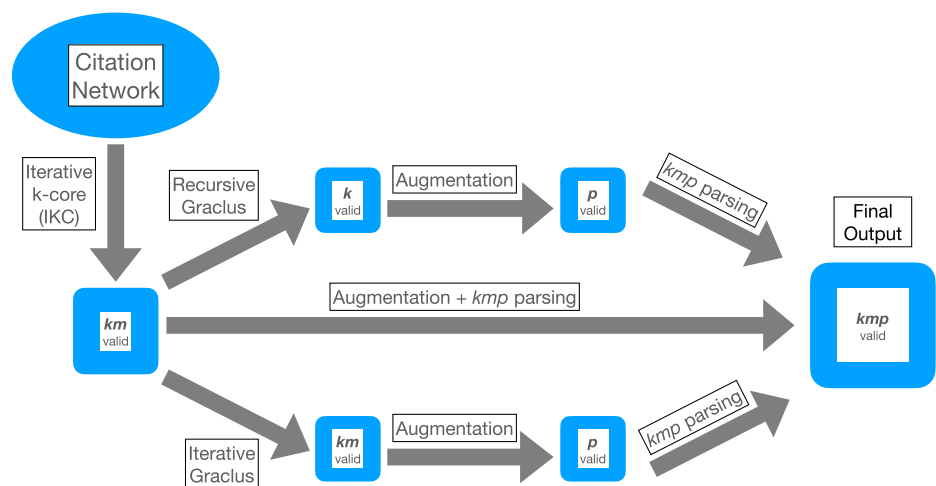
Figure 4. Our four-stage clustering pipeline takes as input a citation network and produces clusters based on values selected for parameters *k* and *p*. Boxes adjacent to edges indicate stages in the pipeline; boxes with blue borders indicate tests that are performed to determine which clusters are passed to the next stage. In Stage 1, it runs the iterative *k*-core (IKC) algorithm for the selected value of *k*; clusters that are *km*-valid are then passed to the next stage. Stage 2 (optional) breaks the clusters from the first stage into smaller clusters, using either Recursive Graclus or Iterative Graclus (and in each case, using either the default version or a heuristic search version). Clusters that pass the required validity check (*k*-validity for Recursive Graclus and *km*-validity for Iterative Graclus) are then passed to the next stage. These clusters are then enlarged with additional nodes in Stage 3, the "Augmentation" step. All clusters produced are *p*-valid at this point, and are passed to Stage 4, *kmp*-parsing, which produces a set of *kmp*-valid clusters, each of which is parsed into their core subcluster and noncore subcluster.

Table 2. Cluster statistics for 12 variants of four-stage clustering. All results include Stages 1 and 4, but some pipelines do not use Stages 2 or 3; all clusterings are *kmp*-valid. Top: results for $k = 5$, Bottom: results for $k = 10$. Stage 2 is performed using either Iterative Graclus (IG) or Recursive Graclus (RG), which are each run with either 0 or 2,000 local search iterations. Node coverage refers to the percentage of network nodes contained in nonsingleton clusters and singletons refers to the number of nodes in singleton clusters. All other statistics refer to nonsingleton clusters, with the last three columns refering to the sizes of nonsingleton clusters. Specific noteworthy trends include: (a) All clusterings that use Stage 3 have node coverage above 18%; (b) all clusterings have at least one very large cluster; (c) Stage 2 choice impacts maximum cluster size; and (d) setting $k = 5$ produces more clusters with a smaller median cluster size than setting $k = 10$.

Stage 2	Stage 3	Node coverage	Clusters (number)	Singletons (number)	Min.	Median	Max.
$k = 5$							
No	No	7.38%	276	13,611,485	8	28.0	345,139
No	Yes	36.63%	276	9,312,583	15	265.5	856,623
IG(0)	Yes	20.01%	13,709	11,752,582	6	118.0	19,934
IG(2000)	Yes	20.84%	9,698	11,632,959	6	106.0	32,487
RG(0)	Yes	26.27%	2,261	10,835,596	10	145.0	579,720
RG(2000)	Yes	26.09%	3,417	10,861,670	11	105.0	578,265
$k = 10$							
No	No	4.22%	119	14,075,787	12	85.0	213,670
No	Yes	33.69%	119	9,744,368	62	1638.0	964,503
IG(0)	Yes	18.51%	4,185	11,975,796	33	427.0	27,007
IG(2000)	Yes	20.00%	3,044	11,757,196	31	488.5	60,189
RG(0)	Yes	27.26%	359	10,689,730	67	1014.0	679,922
RG(2000)	Yes	27.61%	473	10,637,388	55	761.0	620,491

also increases the maximum cluster size to 856,623 and 964,503 for $k = 5$ and $k = 10$, respectively. This approach also has large median cluster sizes, especially for $k = 10$ where the median is 1638. Thus, using Stage 3 (augmentation) but not also Stage 2 produces high node coverage and large cluster sizes.

Using all four stages, and hence using Stage 2, reduces the node coverage to values that range from 18.51% to 27.61% and also reduces the median and maximum cluster sizes, but the choice of how Stage 2 is run has a significant impact. Node coverage is higher when using RG rather than IG. Using RG in default mode rather than local search mode tends to produce a smaller number of nonsingleton clusters that are also somewhat larger; for example, when $k = 10$, default usage of Graclus that doesn't employ the local search strategy produces a median cluster size of 1,014 as opposed to 761 when using 2,000 local search iterations.

The choice between $k = 5$ and $k = 10$ also impacts results, with $k = 10$ producing a much smaller number of nonsingleton clusters that are substantially larger than the results for $k = 5$. For example, using default RG at $k = 10$ produces 359 nonsingleton clusters with median size 1,014, while the same setting for $k = 5$ produces 2,261 nonsingleton clusters with median size 145.

3.2.5. Comparison between different clustering methods

We now compare the different clustering outputs with respect to node coverage, number of nonsingleton clusters, and the distribution of nonsingleton cluster sizes. As our discussion of the different variants of the four-stage pipeline revealed, how Stages 2 and 3 are run and the value for k impact these statistics. If node coverage is the most important criterion, then skipping Stage 2 but using Stage 3 is recommended. However, as these approaches produce very large clusters, including Stage 2 is more likely to be desirable. Among the techniques that use Stage 2, IG tends to produce smaller clusters, but RG produces higher node coverage; the choice between these should be made based on the specific question that is being addressed. Similarly, how k is set should depend on the features of the citation network, and picking larger values of k may be suitable under some conditions.

A comparison to Leiden is also helpful: Before restricting to *kmp*-valid clusters, Leiden has very high node coverage (57%) for resolution value 0.05 and 90% at 0.001. After restricting to *kmp*-valid clusters, however, the best node coverage seen is 13.8% at resolution value 0.0005 when $k = 5$ and 8.8% at resolution value 0.001 when $k = 10$. In contrast, the four-stage pipeline has node coverage that varies from 18.51% to 27.61%, depending on k and how Stage 2 is performed (Table 2). Interestingly, the Leiden algorithm also discovers *kmp*-valid communities even though it is not designed to specifically identify such communities. These findings support the idea of such center–periphery communities existing in networks and being more or less efficiently discovered depending on the algorithm being used.

Thus, a potential user of this approach is presented with options that can be used to address contextual needs. For example, after considering features of the source data and the purpose of community finding, a user may choose Leiden, IKC, IKC with augmentation, or the complete pipeline with its *kmp*-parsing requirements.

3.3. Marker Node Analysis

The network we constructed for this study has more than 14 million nodes. We assumed that some of the communities discovered would represent areas of investigation peripheral to exosomes and extracellular vesicles. Consequently, we used a set of 1,218 independently selected articles from the extracellular vesicle literature, all of which are present in the network, as marker nodes (Section 2) and used them to identify clusters of interest. Any community containing at least one marker node was considered relevant; however, we were particularly interested in communities with high numbers of marker nodes.

For IKC clustering at either $k = 5$ or $k = 10$ (Figure 2), two clusters contained 256 and 227 marker nodes respectively, and together accounted for approximately 40% of the 1,218 markers. The first of these clusters, with 256 marker nodes, was the 56-core of the network, and comprised 3,630 nodes. The second, with 227 marker nodes, exhibited an MCD (minimum core degree) value of 12 and consisted of 213,670 nodes. To visualize the distribution of the entire set of marker nodes we used a multidimensional scaling approach using the frequency of co-occurrence in clusters across 12 different clustering methods as the measure of similarity between publications. Each of these two sets of marker nodes is found in dense and clearly defined clusters after multidimensional scaling (Figure 5).

These two clusters were similar in having a large number of marker nodes but were otherwise different from each other with respect to MCD and size; therefore, we used the two sets of 256 and 227 markers as examples for further study.
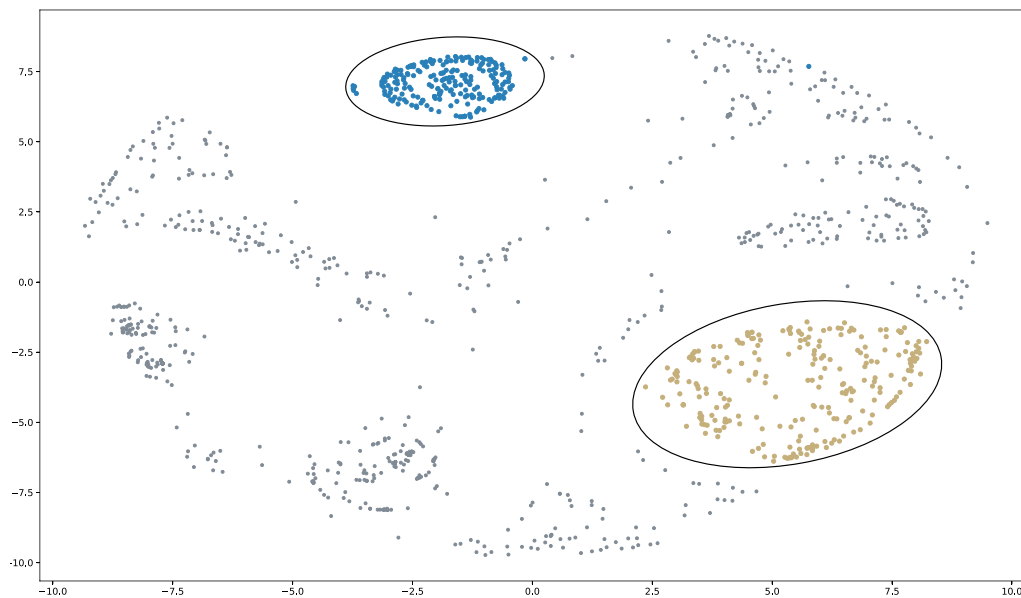
Figure 5. Multidimensional scaling (MDS) of 1,218 marker nodes based on the frequency with which they are placed in the same cluster in 12 clustering outputs. The circled beige cluster (bottom right) corresponds to the 256 markers found using either IKC(5) or IKC(10) in a single cluster of minimum cluster degree (MCD) of 56, while the circled blue cluster (top) corresponds to the 227 markers found in a different cluster with an MCD value of 12.

A second criterion considered was robustness to clustering method, which we measure by the frequency with which marker nodes were found co-located in the same cluster across the 12 clusterings we studied. We began this evaluation by first determining which of these nodes are always placed in nonsingleton clusters for all 12 clustering methods. We found that 27 of 256 and 35 of 227 marker nodes respectively were always placed in nonsingleton clusters, and studied these two sets further.

The first set (A) consisting of 27 marker nodes contained 17 articles and eight reviews that were published between 2006 and 2019, with 23 of these published in 2010 or later. Based upon inspection of titles and journal, the contents of articles in set A spanned basic cell biology, the role of exosomes in cancer, and exosome isolation methods with some variation in terms of being descriptive or mechanistic. The second set (B) of 35 markers contained 26 articles and nine reviews published between 2013 and 2021, largely focused on basic and translational studies of exosomes in nervous tissue but also including a few articles on exosomes in pregnancy and exosome isolation methods.

We then examined the publications in sets A and B for co-occurrence in the same cluster across all 12 clusterings. We found that eight articles from set A were always found in the same cluster and four articles from set B were always found in the same cluster. While the numbers of markers in these sets are small, they serve to identify a larger community, which can then be characterized further by other techniques, such as detailed scholarly examination or textual content analysis.

We identified the smallest cluster across the 12 clusterings that contained the eight articles from A; the selected cluster (Cluster 1) consisted of 73 articles and was focused on extracellular vesicles in cancer. We performed a similar analysis for the four articles from B, and found a cluster (Cluster 2) of 145 articles that was focused on extracellular vesicles in the nervous system. Thus, both these clusters were relatively homogeneous with respect to article content, one focused on

Table 3. Statistics on the two selected small clusters. Cluster 1 and Cluster 2 were selected based on marker nodes: The greatest number of marker nodes that were found co-located in the same cluster across all 12 clustering methods were used to define their common clusters, and the smallest of the common clusters for each set was returned. Both clusters are *kmp*-valid for $k = 5$ and $p = 2$. Articles refers to the total number of nodes in the cluster, core nodes refers to the number of these nodes that are core, and markers refers to the number of nodes that are markers. Authors denotes the total number of authors of articles in the cluster, Auth_5 refers to the number of authors that have at least five publications in the cluster, and Max_Pubs refers to the largest number of publications in the cluster by any single author.

	Articles	Core nodes	Markers	Authors	Auth_5	Max_Pubs
Cluster 1	73	47	16	356	9	17
Cluster 2	145	129	31	742	4	7

extracellular vesicles in cancer and the other on extracellular vesicles in the nervous system. Our use of a single label for each cluster should be considered a subjective approximation; an alternate view is that Cluster 2 also includes articles on cancer and has some focus on astrocytes. Both clusters were derived from the IKC(5)-IG branch of the pipeline.

We then extracted the authors of articles in these two clusters (Table 3). Cluster 1 involved 356 authors of which nine were authors of at least five articles in the cluster and one person was an author of 17 articles in the cluster. However, 301 authors (84.6%) had contributed to only one article in the cluster. Cluster 2 involved 742 authors, of which four were authors of at least five articles in the cluster. One author had contributed seven articles and 650 authors (87.6%) had contributed only one article each. Interestingly, the two publication clusters share 14 authors. Thus, the two author communities defined by two disjoint publication clusters overlap.
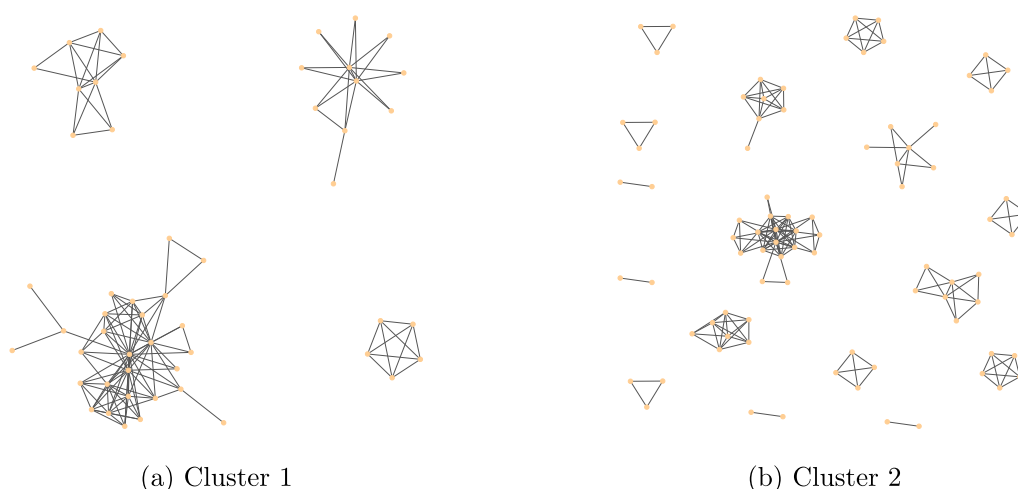

(a) Cluster 1   (b) Cluster 2

Figure 6. Coauthor clusters in two communities identified using marker nodes. Two clusters were selected for analysis because they contained the greatest number of marker nodes that were co-located across all 12 clustering methods, and were the smallest of such clusters (see text). Discrete coauthor groups are found in both clusters when inclusion in a coauthor groups requires at least two instances of intracluster coauthorship between two authors. (a) Cluster 1 consists of 73 articles contributed by 356 authors. This cluster contains four nonoverlapping coauthor groups, with 5, 8, 11, and 28 authors. (b) Cluster 2 consists of 145 articles authored by 356 authors. This cluster contains 17 nonoverlapping groups, with four groups of two authors, three groups of three authors, three groups of four authors, two groups of five authors, four groups of seven authors, and one group of 18 authors.

These trends are strikingly similar to the observations of Price and Beaver (1966) in that the authors segregate into a small number with large numbers of papers in the cluster and many with only one paper in the cluster. Although we only examined two clusters, both exhibited the center–periphery structure described in Price and Beaver (1966); our sample is too small to draw conclusions beyond suggesting that the trends observed may be true for other clusters, and this should be evaluated.

We also found discrete coauthor groups within these clusters (Figure 6). Cluster 1 featured four nonoverlapping coauthor groups where authors were linked to each other if they had coauthored at least two articles in the cluster. Cluster 2 featured 17 such discrete coauthor groups suggesting, despite its larger size, that influence within the group was more distributed considering the larger number of coauthor groups and the smaller number of articles written by individual authors. These examples are provided to illustrate the potential utility of the pipeline and the use of marker nodes.

4. CONCLUSIONS

Based on historical studies of research communities, we posed corresponding properties for the graphical structure of communities in networks. We developed an analytic pipeline to ask whether communities of publications with center–periphery substructure exist in citation networks. In designing a four-stage pipeline to find communities of this form, we were implicitly asking whether the information encoded in the graphical structure of communities can be used to make inferences on the social structure of these communities, for example, discrete coauthor groups. We examined these questions using a citation network representative of exosomes and extracellular vesicles, a field that has rapidly expanded in recent years.

In this citation network, our pipeline found many publication communities that exhibit center–periphery structure. This finding supports our hypothesis that communities of this type exist within the extracellular vesicles research community, and shows that the pipeline we used can find such communities. Whether such communities exist in other citation networks and whether our pipeline is successful at finding such communities are important questions that future work should address. We note that our work explored one very large network with respect to a particular center–periphery model. However, other center–periphery models have been proposed that can be explored and evaluated (Borgatti & Everett, 2000; Gallagher et al., 2021; Havemann, Gläser, & Heinz, 2018; Rombach et al., 2014, 2017), and future work should examine whether these provide better insight into the structure of scientific research communities.

Our pipeline is designed to enable investigators to interrogate their data with different options for each stage and different settings for k and p. As we observed in our study, changes to the settings for the parameters k and p as well as how each stage was performed produced clusterings that differ from each other in terms of the the node coverage as well as the number and sizes of nonsingleton clusters, effectively providing different views of the network. Thus, the specific question of interest and the properties of the citation network are important in choosing how to set these parameters. Alternatively, the pipeline can be used to generate many different clusterings, and the investigator may assess community structure through an integrative analysis that does not depend on a single clustering method.

We also saw significant differences between clusterings produced by our pipeline and those produced by the Leiden software. While there is some overlap in the range of cluster sizes generated, our pipeline tends to generate much larger clusters than the Leiden software, and all of our pipeline clusterings produced at least one very large cluster with more than

19,000 nodes and, in the case of RG, the largest cluster contained more than 500,000 nodes. As we note earlier, the Leiden algorithm was not designed to identify substructure detected as *kmp*-valid clusters but it also detects citation-dense communities.

Given our focus on the small clusters produced in our various pipelines, we did not explore large publication clusters. It is not clear to us what insights can be gained from clusters that have tens of thousands of nodes. We speculate that they may reflect the many connections in a rapidly expanding field. Alternatively, they are targets for future versions of Stage 2, in which we break large clusters into smaller valid ones, perhaps with different criteria for validity. Further work is clearly needed.

This study suggests several other directions for future work. For method development, our four-stage pipeline is designed to enable substitutions to how each stage is performed; as noted above, breaking up large clusters might be more successfully executed using new approaches rather than either recursive or iterative Graclus. We also note that all the clustering methods we developed produce disjoint clusters, yet publications may be expected to be members of more than one research community. Some clustering methods exist that can produce overlapping clusters (Rossetti, 2020); those that cluster edges rather than nodes (Ahn, Bagrow, & Lehmann, 2010; Evans & Lambiotte, 2009; Havemann, 2021) are a next step towards nondisjoint approaches for us. Exploring overlapping clusters in the context of large citation networks is likely to provide additional insight. Combining information from multiple clustering methods could also lead to greater insight, so principled development of ensemble methods (a standard approach in machine learning) is another direction for future research.

One of the most intriguing directions for future research is the life cycle of these research communities, both in terms of how ideas and questions being focused on by the research community as a whole change over time and how authors move between different communities over time. This is challenging to study because the network evolves through growth each year, and community-finding in dynamic graphs is a promising direction (Rossetti, 2020).

Additional clusters could be examined for in-depth examination based, for example, on specific authors, articles of particular importance, funding sources, or clusters identified by marker nodes co-located in many but not all 12 clusterings. Other insights could be obtained by examining the relationship between publication communities (e.g., citations between communities) in a single clustering or comparisons of communities obtained using different clustering methods; such investigations would help elucidate how the research ideas and communities relate to each other, and the extent to which these communities are hierarchically organized.

For extracellular vesicles, insight into community life cycles can be obtained by studying, over time, communities containing key studies such as those on transferrin recycling (Harding, Heuser, & Stahl, 1983; Pan & Johnstone, 1983), the observation that B-lymphocytes secrete antigen-presenting vesicles (Raposo, Nijman et al., 1996), a report of exosome-mediated transfer of mRNA and microRNA (Ratajczak, Miekus et al., 2006; Valadi, Ekström et al., 2007), and the biological effects of transferring exosomes between lean and obese mice (Ying, Riopel et al., 2017). Our analysis was based on a single set of marker nodes; extending the set of marker nodes and annotating each marker node with respect to content would allow finer-grained evaluation.

Exploration of author communities associated to these publication communities would offer additional insights into the social structure of the research community, potentially identifying authors with high influence within a particular emerging research area, and others that are highly influential across several areas.

We close with comments about the high-level approach we took to understanding community structure. Our approach relied entirely on the graphical structure of the citation network. This restriction was used in order to provide a scalable approach that did not rely on any other information or expert knowledge; however, this limits which communities can be detected (McCain, 1986). Textual analysis, relationships between authors based on institutions, and other social interactions such as conference presentations and sources of funding could be used to used to supplement citation data and would likely lead to a different set of publication or author communities with potentially different properties.

Understanding author role is important, and again our reliance on citations to identify influential researchers is biased towards well-cited and well-funded authors. Perhaps one of the benefits, therefore, of our approach to community detection is that we can use it to find small and thematically focused publication communities and hence identify those authors who are influential within these small communities. Nevertheless, we propose that while scalable methods for community detection may generally tend to rely on purely graph-theoretic properties, mixed method approaches support more a more nuanced understanding of social structures and dynamics within the scientific enterprise.

DATA AVAILABILITY

Access to the bibliographic data analyzed in this study requires access from Digital Science. Code generated for this study is freely available from our Github site (Park et al., 2021).

REFERENCES

Ahlgren, P., Chen, Y., Colliander, C., & van Eck, N. J. (2020). Enhancing direct citations: A comparison of relatedness measures for community detection in a large set of PubMed publications. *Quantitative Science Studies*, *1*(2), 714–729. https://doi.org/10.1162/qss_a_00027

Ahn, Y.-Y., Bagrow, J. P., & Lehmann, S. (2010). Link communities reveal multiscale complexity in networks. *Nature*, *466*(7307), 761–764. https://doi.org/10.1038/nature09182, PubMed: 20562860

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(10), P10008. https://doi.org/10.1088/1742-5468/2008/10/P10008

Borgatti, S. P., & Everett, M. G. (2000). Models of core/periphery structures. *Social Networks*, *21*(4), 375–395. https://doi.org/10.1016/S0378-8733(99)00019-2

Boyack, K. W., & Klavans, R. (2010). Co-citation analysis, bibliographic coupling, and direct citation: Which citation approach represents the research front most accurately? *Journal of the American Society for Information Science and Technology*, *61*(12), 2389–2404. https://doi.org/10.1002/asi.21419

Boyack, K. W., & Klavans, R. (2014). Including cited non-source items in a large-scale map of science: What difference does it make? *Journal of Informetrics*, *8*(3), 569–580. https://doi.org/10.1016/j.joi.2014.04.001

Boyack, K. W., & Klavans, R. (2019). Creation and analysis of large-scale bibliometric networks. In W. Glänzel, H. F. Moed, U. Schmoch, & M. Thelwall (Eds.), *Springer handbook of science and technology indicators* (pp. 187–212). Cham: Springer International. https://doi.org/10.1007/978-3-030-02511-3_8

Breiger, R. (2014). *Explorations in structural analysis (RLE social theory): Dual and multiple networks of social interaction.* New York: Routledge. https://doi.org/10.4324/9781315765488

Busatto, S., Morad, G., Guo, P., & Moses, M. A. (2021). The role of extracellular vesicles in the physiological and pathological regulation of the blood–brain barrier. *FASEB Bio Advances*, *3*(9), 665–675. https://doi.org/10.1096/fba.2021-00045, PubMed: 34485835

Chandrasekharan, S., Zaka, M., Gallo, S., Zhao, W., Korobskiy, D., ... Chacko, G. (2021). Finding scientific communities in citation graphs: Articles and authors. *Quantitative Science Studies*, *2*(1), 184–203. https://doi.org/10.1162/qss_a_00095

Chubin, D. E. (1976). State of the field the conceptualization of scientific specialties. *The Sociological Quarterly*, *17*(4), 448–476. https://doi.org/10.1111/j.1533-8525.1976.tb01715.x

Clancy, J. W., Schmidtmann, M., & D'Souza-Schorey, C. (2021). The ins and outs of microvesicles. *FASEB Bio Advances*, *3*(6), 399–406. https://doi.org/10.1096/fba.2020-00127, PubMed: 34124595

Coscia, M., Giannotti, F., & Pedreschi, D. (2011). A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining*, *4*(5), 512–546. https://doi.org/10.1002/sam.10133

Crane, D. (1972). *Invisible colleges; diffusion of knowledge in scientific communities.* Chicago: University of Chicago Press.

Devarakonda, S., Korobskiy, D., Warnow, T., & Chacko, G. (2020). Viewing computer science through citation analysis: Salton and Bergmark redux. *Scientometrics*, *125*(1), 271–287. https://doi.org/10.1007/s11192-020-03624-0, PubMed: 33746310

Dhillon, I., Guan, Y., & Kulis, B. (2007). Weighted graph cuts without eigenvectors: A multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *29*(11), 1944–1957. https://doi.org/10.1109/TPAMI.2007.1115, PubMed: 17848776

Edgar, J. R. (2016). Q & A: What are exosomes, exactly? *BMC Biology*, *14*(46). https://doi.org/10.1186/s12915-016-0268-z, PubMed: 27296830

Evans, T. S., & Lambiotte, R. (2009). Line graphs, link partitions, and overlapping communities. *Physical Review E*, *80*(1), 016105. https://doi.org/10.1103/physreve.80.016105, PubMed: 19658772

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, *486*(3–5), 75–174. https://doi.org/10.1016/j.physrep.2009.11.002

Fortunato, S., & Barthelemy, M. (2007). Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, *104*(1), 36–41. https://doi.org/10.1073/pnas.0605965104, PubMed: 17190818

Fortunato, S., & Castellano, C. (2009). Community structure in graphs. In *Computational complexity* (pp. 490–512). New York: Springer. https://doi.org/10.1007/978-1-4614-1800-9_33

Gallagher, R. J., Young, J.-G., & Welles, B. F. (2021). A clarified typology of core-periphery structure in networks. *Science Advances*, *7*(12), eabc9800. https://doi.org/10.1126/sciadv.abc9800, PubMed: 33731343

Ghoroghi, S., Mary, B., Asokan, N., Goetz, J. G., & Hyenne, V. (2021). Tumor extracellular vesicles drive metastasis (it's a long way from home). *FASEB Bio Advances*, *3*(11), 930–943. https://doi.org/10.1096/fba.2021-00079, PubMed: 34761175

Giatsidis, C., Thilikos, D. M., & Vazirgiannis, M. (2011). Evaluating cooperation in communities with the *k*-core structure. In *2011 International Conference on Advances in Social Networks Analysis and Mining*. IEEE. https://doi.org/10.1109/asonam.2011.65

Harding, C., Heuser, J., & Stahl, P. (1983). Receptor-mediated endocytosis of transferrin and recycling of the transferrin receptor in rat reticulocytes. *Journal of Cell Biology*, *97*(2), 329–339. https://doi.org/10.1083/jcb.97.2.329, PubMed: 6309857

Havemann, F. (2021). Topics as clusters of citation links to highly cited sources: The case of research on international relations. *Quantitative Science Studies*, *2*(1), 204–223. https://doi.org/10.1162/qss_a_00108

Havemann, F., Gläser, J., & Heinz, M. (2018). Communities as well separated subgraphs with cohesive cores: Identification of core-periphery structures in link communities. In *Studies in computational intelligence* (pp. 219–230). Springer International. https://doi.org/10.1007/978-3-030-05411-3_18

He, B., Hamby, R., & Jin, H. (2021). Plant extracellular vesicles: Trojan horses of cross-kingdom warfare. *FASEB Bio Advances*, *3*(9), 657–664. https://doi.org/10.1096/fba.2021-00040, PubMed: 34485834

Hook, D. W., Porter, S. J., & Herzog, C. (2018). Dimensions: Building context for search and evaluation. *Frontiers in Research Metrics and Analytics*, *3*, 23. https://doi.org/10.3389/frma.2018.00023

Javed, M. A., Younis, M. S., Latif, S., Qadir, J., & Baig, A. (2018). Community detection in networks: A multidisciplinary review. *Journal of Network and Computer Applications*, *108*, 87–111. https://doi.org/10.1016/j.jnca.2018.02.011

Kalluri, R., & LeBleu, V. S. (2020). The biology, function, and biomedical applications of exosomes. *Science*, *367*(6478). https://doi.org/10.1126/science.aau6977, PubMed: 32029601

Klavans, R., & Boyack, K. W. (2017). Which type of citation analysis generates the most accurate taxonomy of scientific and technical knowledge? *Journal of the Association for Information Science and Technology*, *68*(4), 984–998. https://doi.org/10.1002/asi.23734

Kong, Y.-X., Shi, G.-Y., Wu, R.-J., & Zhang, Y.-C. (2019). *k*-core: Theories and applications. *Physics Reports*, *832*, 1–32. https://doi.org/10.1016/j.physrep.2019.10.004

Kuhn, T. (1970). *The structure of scientific revolutions*. Chicago: University of Chicago Press.

Lananna, B. V., & Imai, S. (2021). Friends and foes: Extracellular vesicles in aging and rejuvenation. *FASEB Bio Advances*, *3*(10), 787–801. https://doi.org/10.1096/fba.2021-00077, PubMed: 34632314

Le Lay, S., Rome, S., Loyer, X., & Nieto, L. (2021). Adipocyte-derived extracellular vesicles in health and diseases: Nano-packages with vast biological properties. *FASEB Bio Advances*, *3*(6), 407–419. https://doi.org/10.1096/fba.2020-00147, PubMed: 34124596

Leidal, A. M., & Debnath, J. (2021). Emerging roles for the autophagy machinery in extracellular vesicle biogenesis and secretion. *FASEB Bio Advances*, *3*(5), 377–386. https://doi.org/10.1096/fba.2020-00138, PubMed: 33977236

Leskovec, J., & Horvitz, E. (2008). Planetary-scale views on a large instant-messaging network. In *Proceeding of the 17th International Conference on World Wide Web—WWW '08*. ACM Press. https://doi.org/10.1145/1367497.1367620

Malliaros, F. D., Giatsidis, C., Papadopoulos, A. N., & Vazirgiannis, M. (2019). The core decomposition of networks: Theory, algorithms and applications. *VLDB Journal*, *29*(1), 61–92. https://doi.org/10.1007/s00778-019-00587-4

Matula, D. W., & Beck, L. L. (1983). Smallest-last ordering and clustering and graph coloring algorithms. *Journal of the Association for Computing Machinery*, *30*(3), 417–427. https://doi.org/10.1145/2402.322385

McCain, K. W. (1986). The paper trails of scholarship: Mapping the literature of genetics. *Library Quarterly*, *56*, 258–271. https://doi.org/10.1086/601770

Morris, S. A. (2005). Manifestation of emerging specialties in journal literature: A growth model of papers, references, exemplars, bibliographic coupling, cocitation, and clustering coefficient distribution. *Journal of the American Society for Information Science and Technology*, *56*(12), 1250–1273. https://doi.org/10.1002/asi.20208

Morris, S. A., & der Veer Martens, B. V. (2009). Mapping research specialties. *Annual Review of Information Science and Technology*, *42*(1), 213–295. https://doi.org/10.1002/aris.2008.1440420113

Mullins, N. C. (1985). Invisible colleges as science elites. *Scientometrics*, *7*, 357–368. https://doi.org/10.1007/BF02017154

Mullins, N. C., Hargens, L. L., Hecht, P. K., & Kick, E. L. (1977). The group structure of cocitation clusters: A comparative study. *American Sociological Review*, *42*(4), 552–562. https://doi.org/10.2307/2094554

Newman, M. E. J. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, *103*(23), 8577–8582. https://doi.org/10.1073/pnas.0601602103, PubMed: 16723398

Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, *69*(2), 026113. https://doi.org/10.1103/physreve.69.026113, PubMed: 14995526

Pan, B.-T., & Johnstone, R. M. (1983). Fate of the transferrin receptor during maturation of sheep reticulocytes in vitro: Selective externalization of the receptor. *Cell*, *33*(3), 967–978. https://doi.org/10.1016/0092-8674(83)90040-5, PubMed: 6307529

Park, M., Wedell, E., Korobskiy, D., Warnow, T., & Chacko, G. (2021). Community Finding and Clustering Project. *Github repository*, University of Illinois Urbana-Champaign. https://github.com/chackoge/ERNIE_Plus/tree/master/Illinois/clustering.

Pittel, B., Spencer, J., & Wormald, N. (1996). Sudden emergence of a giant *k*-core in a random graph. *Journal of Combinatorial Theory, Series B*, *67*, 111–151. https://doi.org/10.1006/jctb.1996.0036

Price, D. d. S. (1965). Networks of scientific papers. *Science*, *149*, 510–515. https://doi.org/10.1126/science.149.3683.510, PubMed: 14325149

Price, D. d. S., & Beaver, D. D. (1966). Collaboration in an invisible college. *American Psychologist*, *21*(11), 1011–1018. https://doi.org/10.1037/h0024051, PubMed: 5921694

Raposo, G., Nijman, H. W., Stoorvogel, W., Liejendekker, R., Harding, C. V., ... Geuze, H. J. (1996). B lymphocytes secrete antigen-presenting vesicles. *Journal of Experimental Medicine*, *183*(3), 1161–1172. https://doi.org/10.1084/jem.183.3.1161, PubMed: 8642258

Raposo, G., van Niel, G., & Stahl, P. D. (2021). Extracellular vesicles and homeostasis—An emerging field in bioscience research. *FASEB BioAdvances*, *3*(6), 456–458. https://doi.org/10.1096/fba.2021-00009, PubMed: 34124600

Ratajczak, J., Miekus, K., Kucia, M., Zhang, J., Reca, R., ... Ratajczak, M. Z. (2006). Embryonic stem cell-derived microvesicles reprogram hematopoietic progenitors: Evidence for horizontal transfer of mRNA and protein delivery. *Leukemia*, *20*(5), 847–856. https://doi.org/10.1038/sj.leu.2404132, PubMed: 16453000

Rombach, M. P., Porter, M. A., Fowler, J. H., & Mucha, P. J. (2014). Core-periphery structure in networks. *SIAM Journal on Applied Mathematics*, *74*(1), 167–190. https://doi.org/10.1137/120881683

Rombach, P., Porter, M. A., Fowler, J. H., & Mucha, P. J. (2017). Core–periphery structure in networks (revisited). *SIAM Review*, *59*(3), 619–646. https://doi.org/10.1137/17m1130046

Rossetti, G. (2020). ANGEL: Efficient, and effective, node-centric community discovery in static and dynamic networks. *Applied Network Science*, *5*, 26. https://doi.org/10.1007/s41109-020-00270-6

Schnatz, A., Müller, C., Brahmer, A., & Krämer-Albers, E.-M. (2021). Extracellular vesicles in neural cell interaction and CNS homeostasis. *FASEB Bio Advances*, *3*(8), 577–592. https://doi.org/10.1096/fba.2021-00035, PubMed: 34377954

Sciabolazza, V. L., Vacca, R., Okraku, T. K., & McCarty, C. (2017). Detecting and analyzing research communities in longitudinal scientific networks. *PLOS ONE*, *12*(8), e0182516. https://doi.org/10.1371/journal.pone.0182516, PubMed: 28797047

Seidman, S. B. (1983). Network structure and minimum degree. *Social Networks*, *5*(3), 269–287. https://doi.org/10.1016/0378-8733(83)90028-x

Sjögarde, P., & Ahlgren, P. (2018). Granularity of algorithmically constructed publication-level classifications of research publications: Identification of topics. *Journal of Informetrics*, *12*(1), 133–152. https://doi.org/10.1016/j.joi.2017.12.006

Sjögarde, P. & Ahlgren, P. (2020). Granularity of algorithmically constructed publication-level classifications of research publications: Identification of specialties. *Quantitative Science Studies*, *1*(1), 207–238. https://doi.org/10.1162/qss_a_00004

Staudt, C. L., Sazonovs, A., & Meyerhenke, H. (2016). NetworKit: A tool suite for large-scale complex network analysis. *Network Science*, *4*(4), 508–530. https://doi.org/10.1017/nws.2016.20

Šubelj, L., van Eck, N. J., & Waltman, L. (2016). Clustering scientific publications based on citation relations: A systematic comparison of different methods. *PLOS ONE*, *11*(4), e0154404. https://doi.org/10.1371/journal.pone.0154404, PubMed: 27124610

Traag, V. (2021). Network analysis, GitHub page for the Leiden algorithm. https://github.com/vtraag/networkanalysis (accessed October 18, 2021).

Traag, V. A., Waltman, L., & van Eck, N. J. (2019). From Louvain to Leiden: Guaranteeing well-connected communities. *Scientific Reports*, *9*, 5233. https://doi.org/10.1038/s41598-019-41695-z, PubMed: 30914743

Valadi, H., Ekström, K., Bossios, A., Sjöstrand, M., Lee, J. J., & Lötvall, J. O. (2007). Exosome-mediated transfer of mRNAs and microRNAs is a novel mechanism of genetic exchange between cells. *Nature Cell Biology*, *9*(6), 654–659. https://doi.org/10.1038/ncb1596, PubMed: 17486113

Van Dongen, S. (2008). Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications*, *30*(1), 121–141. https://doi.org/10.1137/040608635

van Niel, G., D'Angelo, G., & Raposo, G. (2018). Shedding light on the cell biology of extracellular vesicles. *Nature Reviews Molecular Cell Biology*, *19*(4), 213–228. https://doi.org/10.1038/nrm.2017.125, PubMed: 29339798

Verdi, V., Bécot, A., van Niel, G., & Verweij, F. J. (2021). In vivo imaging of EVs in zebrafish: New perspectives from "the waterside." *FASEB Bio Advances*, *3*(11), 918–929. https://doi.org/10.1096/fba.2021-00081, PubMed: 34761174

Waltman, L., & van Eck, N. J. (2012). A new methodology for constructing a publication-level classification system of science. *Journal of the American Society for Information Science and Technology*, *63*(12), 2378–2392. https://doi.org/10.1002/asi.22748

Ying, W., Riopel, M., Bandyopadhyay, G., Dong, Y., Birmingham, A., … Olefsky, J. M. (2017). Adipose tissue macrophage-derived exosomal miRNAs can modulate in vivo and in vitro insulin sensitivity. *Cell*, *171*(2), 372–384. https://doi.org/10.1016/j.cell.2017.08.035, PubMed: 28942920