Generalization of core percolation on complex networks

N. Azimi-Tafreshi
Physics Department, Institute for Advanced Studies in Basic Sciences, 45195-1159 Zanjan, Iran

S. Osat
Quantum Complexity Science Initiative, Skolkovo Institute of Science and Technology, Skoltech Building 3, Moscow, 143026, Russia

S. N. Dorogovtsev
Departamento de Física da Universidade de Aveiro & I3N, Campus Universitário de Santiago, 3810-193 Aveiro, Portugal and A.F. Ioffe Physico-Technical Institute, 194021 St. Petersburg, Russia

We introduce a $k$-leaf removal algorithm as a generalization of the so-called leaf removal algorithm. In this pruning algorithm, vertices of degree smaller than $k$, together with their first nearest neighbors and all incident edges are progressively removed from a random network. As the result of this pruning the network is reduced to a subgraph which we call the Generalized $k$-core ($G_k$-core). Performing this pruning for the sequence of natural numbers $k$ we decompose the network into a hierarchy of progressively nested $G_k$-cores. We present an analytical framework for description of $G_k$-core percolation for undirected uncorrelated networks with arbitrary degree distributions (configuration model). To confirm our results, we also derive rate equations for the $k$-leaf removal algorithm which enable us to obtain the structural characteristics of the $G_k$-cores in another way. We suggest that the $k$-leaf pruning algorithm constructs a hierarchy of configurations approaching optimal ones in combinatorial optimization problems. In particular, implying rate equation approach we estimate from above the size of a minimum vertex cover of random networks. Also we apply our algorithm to a number of real-world networks and perform the $G_k$-core decomposition for them.

PACS numbers: 05.70.Fh, 64.60.-p, 02.10.Ox

I. INTRODUCTION

Structural decomposition of complex networks providing classification the vertices into different subsets is one of effective approaches for studying the structural properties of networks. As a primary and well-known example, one can indicate the $k$-core decomposition, which is an efficient technique for uncovering structural properties of large networks [1]. The $k$-core of a network is defined as the largest subgraph whose vertices have degree at least $k$ [2]. There is a pruning algorithm enabling one to obtain $k$-core subgraphs for a given network: at each step, a vertex of degree less than $k$ is randomly chosen and removed together with its neighbor and all incident edges. The pruning is continued until no further removal is possible. As the result of this pruning the network is decomposed to a set of enclosed $k$-cores. The vertices belonging to higher (more central) cores are more strongly connected. It was also shown that the vertices of inner core are more influential spreaders in epidemic processes [2]. A giant $k$-core emerges above a percolation threshold [4]. The most remarkable result is that for $k \geq 3$ the giant $k$-core shows a discontinues hybrid phase transition combining discontinuity and a critical singularity [2,3,4].

Another key subgraph of a random network is simply called its core. These subgraphs significantly differ from the $k$-cores. A core of an undirected network is obtained only by a pruning algorithm in contrast to the $k$-core which is, in addition, defined by a specific constraint on the connectivity of its vertices. The pruning algorithm producing a core is called the leaf removal algorithm. It was introduced by Karp and Sipser [6]. In this pruning algorithm, a vertex of degree one (a leaf) is randomly chosen and removed together with its neighbor and all incident edges. The algorithm is continued until no leaves remain. The resulting subgraph is formed by some isolated subgraphs and the giant one which is called the core. For the Erdős–Rényi (ER) random graphs, Bauer and Golinelli showed that the core percolation threshold is located at the mean degree $\langle q \rangle = e \approx 2.718...$, so that above this point the network contains the giant core while below the threshold the size of the giant core is zero [2,7]. The core structure and the phase transition at $\langle q \rangle = e$ is related to a number of phenomena in physics such as conductor-insulator transitions [8] and replica symmetry breaking in minimal vertex cover [9]. Moreover it was shown that the formation of the core is related to controllability robustness [10] and some combinatorial optimization problems such as maximum matching and minimum vertex cover [10,11,12].

The leaf removal algorithm first was applied to finding maximum matching (and equivalently minimum vertex cover) in random graphs [10,11]. However this algorithm is able to construct a maximum matching (minimum vertex cover) for a vertex mean degree smaller than $e$, i.e. before the emergence of the core. For more dense networks, the algorithm stops since no leaf is left, while some edges or vertices remain uncovered. On the other
hand, the Gazmuri algorithm provides an upper bound to the optimal solution for the networks with the large mean vertex degree \[13\]. In this algorithm, at each step a vertex is randomly chosen, and this vertex together with all first neighbors and incident edges are removed from the graph. However this algorithm is less successful in approaching an optimal solution for small and intermediate connectivities \[14\].

In this paper, focusing on the problem of vertex cover on random graphs, we generalize the leaf removal algorithm to draw a near optimal configuration for vertex cover on random graphs with a vertex mean degree higher than \(e\). The algorithm is based on the fact that the vertices of small degree are more likely to be uncovered. We generalize definition of the leaf to the “\(k\)-leaf”, defined as a vertex of degree less than \(k\). Removing recursively a \(k\)-leaf together with all its first neighbors and their incident edges, we can improve Gazmuri algorithm and obtain a near-optimal vertex cover for the networks with small mean degree. Furthermore, following this pruning approach, the network is decomposed to a hierarchy of nested cores, similarly to the ordinary \(k\)-core decomposition. We call this structure the Generalized \(k\)-core (\(G_k\)-core). In this notation, the ordinary core is represented by the \(G_2\)-core. The vertices belonging to inner \(G_k\)-cores and also their first neighbors are of high degree and well connected. Analytical calculation is possible only for the networks with locally tree-like structure. For these kind of networks, and using the generating function technique, we study the structural transitions and emergence points of the \(G_k\)-core subgraphs.

The leaf removal algorithm is a Markovian process. We describe evolution of the network structure during the pruning process by applying rate equations. The rate equations have been derived for the ordinary leaf removal algorithm on undirected and directed graphs \[12, 14, 16\]. This approach provided the size and the emergence point of the ordinary core. Furthermore, using the rate equations one can find the size of minimal vertex covers, constructed by the leaf removal algorithm in the networks that have no core \[14\]. In this paper, we also derive rate equations for the degree distribution of a network during the execution of the \(k\)-leaf algorithm, which enables us to obtain the structure of the \(G_k\)-cores in an alternative way. Using the rate equations and for each value of \(k\), we can find the number of vertices covered by the \(k\)-leaf algorithm in the vertex cover problem.

The paper is organized as follows. In Sec. \(\text{II}\) we present an analytical framework to study \(G_k\)-core percolation for random networks with arbitrary degree distributions. We apply our general results to the Erdős–Rényi and scale-free networks. We compare our results with numerical simulations. In Sec. \(\text{III}\) we derive the rate equations for the \(k\)-leaf removal algorithm and using these equations we find in another way how \(G_k\)-cores are organized. We obtain the size of minimum vertex covers constructed by the \(k\)-leaf algorithm in Sec. \(\text{IV}\). In Sec. \(\text{V}\) a set of real-world networks are analyzed in the framework of our approach.

II. ANALYTICAL FRAMEWORK

Let us consider an uncorrelated network with an arbitrary degree distribution \(P(k)\). To produce a generalization of the core subgraph, we use the following pruning algorithm: at each step we randomly choose a \(k\)-leaf (i.e. a vertex of degree less than \(k\)), remove it together with its neighbors and all incident edges to the neighbors. As a result of the pruning, the degrees of some vertices change. The procedure is iterated until no vertices of degree less than \(k\) remain in the network. The residual network, if it exists, is called \(G_k\)-core.

To find the size of the \(G_k\)-core, we classify the vertices into three groups: 1) \(\alpha\)-removable: the vertices that can become a leaf. 2) \(\beta\)-removable: the vertices that can become a neighbor of a leaf. 3) the vertices that are neither \(\alpha\)-removable nor \(\beta\)-removable and hence belong to \(G_k\)-core. Using the assumption that the network has a locally tree-like structure, we can write self-consistency equations for probabilities that a randomly chosen edge leads to an \(\alpha\)-removable node, to a \(\beta\)-removable or to a non-removable node. We call these probabilities \(\alpha\), \(\beta\) and \(1-\alpha-\beta\), respectively. These probabilities are represented graphically in Fig. \(\text{I}\).

At least one of the neighbors of a \(\beta\)-removable vertex must be \(\alpha\)-removable. Furthermore, an end vertex of a randomly chosen edge belongs to the \(G_k\)-core, if it has at least \(k-1\) neighbors which belong to the \(G_k\)-core and none of its neighbors are of type \(\alpha\). Taking into account these facts, we write the following two self-consistent equations:
one can obtain

$$1 - \alpha - \beta = \sum_{q} \frac{qP(q)}{\langle q \rangle} \times \sum_{s=k-1}^{q-1} \left( \begin{array}{c} q - 1 \\ s \end{array} \right) (1 - \alpha - \beta)^s \beta^{q-1-s},$$

$$\beta = 1 - \sum_{q} \frac{qP(q)}{\langle q \rangle} (1 - \alpha)^q - 1. \quad (1)$$

The first equation represents the probability that an end vertex of a randomly chosen edge belongs to the $G_k$-core. $qP(q)/\langle q \rangle$ is the probability that the end vertex of a uniformly randomly chosen edge has degree $q$ and the combinatorial multiplier $\left( \frac{m}{n} \right)$ gives the number of ways one can choose $n$ edges from a sample of $m$ edges. At least $k-1$ edges of $q-1$ edges (other edges than the starting one) must lead to the $G_k$-core. Eq. (1) also shows the probability that an end vertex of a randomly chosen edge is $\beta$-removable. At least one of the neighbors of a $\beta$-removable vertex must be a leaf, i.e., an $\alpha$-removable vertex. These two equations are schematically represented in Fig. 2.

From Eq. (1), one can derive the following self-consistency equation for $\alpha$:

$$\alpha = \sum_{q} \frac{qP(q)}{\langle q \rangle} \sum_{s=0}^{k-2} \left( \begin{array}{c} q - 1 \\ s \end{array} \right) (1 - \alpha - \beta)^s \beta^{q-1-s}. \quad (2)$$

The probabilities $\alpha$ and $\beta$ enable us to obtain the probability $n_{kc}$ that a randomly chosen vertex belongs to the $G_k$-core, which is also the relative size of the $G_k$-core. Fig. 3 shows a schematic representation of this probability. A vertex is in the $G_k$-core if the vertex has at least $k$ neighbors which belong to the $G_k$-core. Hence we can write the following equation for the relative size of the general $G_k$-core:

$$n_{kc} = \sum_{q>k} P(q) \sum_{s=k}^{q} \left( \begin{array}{c} q \\ s \end{array} \right) (1 - \alpha - \beta)^s \beta^{q-s}. \quad (3)$$

To be able to solve Eqs. (1)–(3) analytically, we rewrite these equations using generating functions [13]. For a network with a given degree distribution $P(q)$, the generating function $G(x)$ is defined as

$$G(x) \equiv \sum_{q} P(q)x^q. \quad (4)$$

Hence, we obtain the following equations for $\alpha$, $\beta$ and $n_{kc}$ in terms of the generating function:

$$\alpha = \frac{1}{\langle q \rangle} \sum_{s=0}^{k-2} \frac{(1 - \alpha - \beta)^s}{s!} G^{s+1}(\beta),$$

$$\beta = 1 - \frac{G^{(1)}(1 - \alpha)}{\langle q \rangle},$$

$$n_{kc} = G(1 - \alpha) - \sum_{s=0}^{k-1} \frac{(1 - \alpha - \beta)^s}{s!} G^{s}(\beta), \quad (5)$$

where we used the notation $G^{(s)}(x)$ for the $s$-th derivatives of $G(x)$.

Furthermore, the probability that both end vertices of an edge in the network belong to the $G_k$-core is $(1 - \alpha - \beta)^2$. Hence, the fraction of edges in the $G_k$-core, denoted by $l_{kc}$, is obtained as

$$l_{kc} = \frac{c}{2} (1 - \alpha - \beta)^2. \quad (6)$$

Let us first consider Erdős–Rényi networks with Poisson degree distributions, $P(q) = c^qe^{-c}/q!$, where $c$ is the vertex mean degree for the network. For the Poisson distribution, the generating function and its $s$-th derivative are $G(x) = e^{-c(1-x)}$ and $G^{s}(x) = c^se^{-c(1-x)}$, respectively. One can easily find the relation between $\alpha$ and $\beta$ probabilities as $\beta = 1 - e^{-c\alpha}$, which is independent of the value of $k$. The largest core is labeled with $k = 2$. This is the $G_2$-core, i.e., in other words, the ordinary core [17]. The relative size of the $G_2$-core can be simplified as

$$n_{2c} = (1 - \beta - \alpha)(1 - \alpha). \quad (7)$$

The next core is the $G_3$-core. From Eqs. (5) one can obtain the relative probabilities for $G_3$-core:

$$\alpha = 1 + (1 - \beta)e^{-c(1-\beta)} + e^{-c(1-\beta)} + c,$$

$$n_{3c} = e^{-c\alpha} - e^{-c(1-\beta)} - c\alpha(1 - \alpha - \beta) - \frac{1}{2} e^{-2(1 - \alpha - \beta)} e^{-c(1-\beta)}. \quad (8)$$

The size of $G_4$-core is also obtained by the following equations:

$$\alpha = e^{-c\alpha} + ce^{-c\alpha}(1 - \alpha - \beta) + \frac{1}{2} e^{-2(1 - \alpha - \beta)} e^{-c(1-\beta)},$$

$$n_{4c} = e^{-c\alpha} - e^{-c(1-\beta)} - c(1 - \alpha - \beta)e^{-c(1-\beta)} - \frac{1}{2} e^{-2(1 - \alpha - \beta)} e^{-c(1-\beta)} - \frac{1}{6} e^{-3(1 - \alpha - \beta)} e^{-c(1-\beta)}. \quad (9)$$

Similarly, we can obtain the size of more internal cores, if they exist, from Eqs. (5) Figure 4 shows the relative size and the normalized number of edges of the $G_k$-core for
\( k = 2, 3 \) and 4. The analytic results (curves) are compared with numerical simulations (symbols). As we can see in the figure, in contrast to the ordinary core, for \( k \geq 3 \) a \( G_k \)-core emerges discontinuously at the percolation threshold.

Next we consider scale-free networks. It was shown that for the purely power-law scale-free networks the ordinary core does not exist [17]. Hence we consider the asymptotically scale-free, uncorrelated networks generated by the static model with the degree distribution

\[
P(q) = \frac{\Gamma(q-1)}{2\Gamma(\gamma-1)}\Gamma(q-\gamma+1, \frac{(\gamma-2)}{2(\gamma-1)}) / \Gamma(q+1). \]

where \( \Gamma(s, x) \) is the upper incomplete gamma function [18, 19]. For this degree distribution the generating function is

\[
G_k(y) = \sum_{q=0}^{\infty} \frac{P(q)}{q!} y^q = \int_0^y \frac{dx}{e^{x(\gamma-1)}}. \]

In Fig. 4 we compare the emergence of cores for asymptotically scale-free and Erdős–Rényi networks. As one can see, the dependence of the cores on \( c \) for these networks is similar and, as expected, the curves with larger \( \gamma \) approach the result for Erdős–Rényi networks.

We define pruning times steps in a way that enables us to classify the vertices of the network into a set of layers for a given \( k \). At time step \( t' = 1 \), we select the vertices of degree less than \( k \) (k-leaves) and remove these vertices and their neighbors by applying the \( k \)-leaf algorithm. Removing the vertices in the first step may produce new \( k \)-leaves, which will be removed at \( t' = 2 \) and so on. The vertices removed at each step \( t' \) form a layer of the network. In other words, the network is pruned layer by layer until there is no \( k \)-leaf left. We denote the total number of pruning steps as \( \tau \) so that \( t' = 1, 2, \ldots, \tau \). After \( \tau \) steps, the network consists of finite components or a giant \( G_k \)-core. For different networks we obtain \( \tau(c) \) using numerical simulation, see in Fig. 5. As we can see, the dependencies \( \tau(c) \) diverge at the birth points of the cores.

### III. RATE EQUATIONS

The structural evolution of the network during pruning processes is described by the so-called rate equations for the degree distribution of the remaining network [12, 14]. Here we derive rate equations for the \( k \)-leaf removal algorithm. Let us consider a network of \( N \) vertices and \( L \) edges. For simplicity we remove only the edges during the pruning process. In other words, at each time step \( t \) we choose randomly a \( k \)-leaf, remove all \( k \) edges incident to it, together with all edges incident to its \( k \) neighbors. In this way, the number of vertices of the network remain constant. Note that the time steps \( t \) differ from \( t' \). The algorithm is iterated until \( P(q) = 0 \) for all \( q < k \). The important point of this approach is that the dynamics is self-averaging in the thermodynamic limit, i.e. \( N \to \infty \). After a certain number of time steps, almost all random networks have the same degree distribution, which is independent of the (random) order of the removal of leaves.

We introduce the rescaled time \( t = \frac{T}{N} \), where \( T \) is the total number of steps of the pruning algorithm. So, \( \Delta t = 1/N \) is the rescaled time of one iteration. Let \( N(q, t) \) be the average number of vertices with degree \( q \) at time \( t \). Since the total number of vertices is constant, i.e. \( N(t) = N \), we have \( N(q, t) = NP(q, t) \). We can write the change of \( N(q, t + \Delta t) - N(q, t) \) after one iteration. In the large network limit, we can pass from the discrete difference to the time derivative of the degree distribution and obtain the following evolution equation for the degree distribution:

\[
N(q, t + \Delta t) - N(q, t) = \frac{d}{dt} N(q, t) = \frac{\partial}{\partial t} \left[ \frac{N(q, t)}{N} \right] = \frac{\partial}{\partial t} \left[ \sum_{q} N(q, t) \right] = \frac{\partial}{\partial t} \left[ N(q, t) \right] \frac{1}{N} \frac{dN}{dt}
\]

Let us explain different terms on the right hand side of Eq. (10). First we choose a random vertex of degree less than \( k \) and remove all edges incident to it. The probability that a vertex has degree less than \( k \) is \( \frac{\theta(k-q)P(q, t)}{\sum_{q} \theta(k-q)P(q, t)} \), where \( \theta(i) \) is defined for integers: \( \theta(i) = 1 \) and \( \theta(i < 0) = 0 \). Thus with this probability, the number of vertices with \( q < k \) decreases by 1. This gives the first term. After removing the edges incident to the leaf and all edges incident to its neighbors, the leaf and all its neighbors become vertices of degree zero. The average number of neighbors of a vertex of degree less than \( k \) is \( \frac{\sum_{q} q \theta(k-q)P(q, t)}{\sum_{q} \theta(k-q)P(q, t)} \). Hence the second term shows the number of vertices whose degrees become zero. On the other hand, the degree distribution of the end vertices of
a randomly chosen edge is \( qP(q) \). When we remove the edges incident to the nearest neighbors of the leaf, the number of vertices of degree \( q \) is decreased by the mean degree of the leaf with probability \( 2P(q) \). Finally the last contribution is resulted from modification of degrees of the second neighbors of the leaf. After removal of all edges incident to the leaf and its nearest neighbors, the number of connections of the second nearest neighbors of the leaf decreases by one. The average number of the second neighbors is equal to the mean degree of the nearest neighbors except one (connection to the leaf), multiplied by the average number of the nearest neighbors of the leaf. Equation (10) is a set of differential equations, describing the evolution of a network during the pruning.

For \( k = 2 \), these equations coincide with the known ones [14]. Solving Eq. (10) iteratively, we can obtain the degree distribution of the network at each time step \( t \).

As we already mentioned, we do not remove the vertices during the leaf removal algorithm and so the total number of the vertices remains constant. However, at each time step all edges incident to the leaf and also the edges incident to all its nearest neighbors are removed. Hence, at each time step the average number of removed edges is equal to the mean number of nearest neighbors multiplied by their mean degree. This results to the following evolution equation for the average number of re-

FIG. 4. (color online). The relative sizes and the normalized number of edges of the \( G_k \)-core for \( k = 2, 3, 4 \). The points are the results of numerical simulation for the Erdős–Rényi and asymptotically scale-free networks of size \( N = 10^6 \), averaged over 10 realizations. The lines are analytical results obtained from Eqs. (5–6).

FIG. 5. (color online). Total number of pruning steps \( \tau \) vs. mean degree \( c \). The curves shows diverging of \( \tau(c) \) at the emergence point of (a) \( G_2 \)-core, (b) \( G_3 \)-core and (c) \( G_4 \)-core for the Erdős–Rényi and asymptotically scale-free networks of size \( N = 10^6 \), averaged over 10 realizations.
remained edges in the network:

$$\frac{\dot{L}(t)}{N} = -\frac{(q^2)_t}{(q)_t} \times \frac{\sum_q q \theta(q - q) P(q, t)}{\sum_q \theta(q - q) P(q, t)}$$  \hspace{1cm} (11)

We apply the leaf removal algorithm to an uncorrelated network with degree distribution $P(q, t=0)$ and a vertex mean degree equal to $c_0$ as the initial conditions. For each value of $k$, the algorithms are iterated until no vertices of degree less than $k$ remain. To find the $G_k$-core, the algorithm must continue until time $t_k^*$ at which $P(1, t_k^*) = P(2, t_k^*) = \ldots = P(k-1, t_k^*) = 0$. Our numerical results for different networks show that $P(1, t)$ is the last probability to become zero. That is, the vertices of degree 1 disappear after all other leaves. This is why during iteration we look at the behavior of $P(1, t)$ and the algorithm stops at time $t_k^*$ for a given $k$. The remaining subgraph is the $G_k$-core. For $k = 2$ the algorithm coincides with the ordinary leaf-removal algorithm and the remaining subgraph is the $G_2$-core or simply the core. After we find $t_k^*$, the size and the number of edges of the $G_k$-core can be obtained from the following relations:

$$N_{kc} = N[1 - P(0, t_k^*)], \hspace{1cm} (12)$$

$$L_{kc} = L(t_k^*). \hspace{1cm} (13)$$

We apply this approach to the Erdős–Rényi random graphs. The Poisson degree distribution rapidly decays and it is sufficient to consider $q_{\text{max}} = 20$, i.e. we solve the set of the first 21 equations. Figure 6 shows the size and the number of edges calculated from Eqs. (9)–(13), for the Erdős–Rényi networks. In this figure we compare the results obtained by solving the rate equations (points) with the analytic results of the previous section (lines) for the general 2, 3 and 4-cores.

**IV. MINIMUM VERTEX COVER**

The leaf-removal algorithm can be directly applied to such combinatorial optimization problems as minimum vertex cover and maximum matching. A vertex cover $V_c$ of a graph is a set of vertices such that for every edge at least one of its end vertices is in $V_c$. The vertices in $V_c$ are called covered and the remaining vertices are called uncovered. Vertex cover is the complement of a matching (independent edge set) problem. A matching is defined as a set of edges of a graph which have no common vertices. Finding a minimum vertex cover, or equivalently maximum matching, on a graph is a combinatorial optimization problem.

There is a class of linear time heuristic algorithms allowing to construct a minimum vertex cover \[14, 20\]. In these heuristic algorithms, at each step a central vertex with weight $w_q$ is randomly chosen and uncovered (i.e., they are in the set $G^\prime V_c$). Then all its neighbors are covered (i.e., they are in the set $V_c$). The central vertex, its neighbors together with all incident edges are removed from the graph. The algorithm is repeated until no edges remain. The case $w_q = 1$ has been analysed by Gazmuri \[13\]. However the Gazmuri algorithm can give an estimation for the minimum vertex cover only for networks with large mean degree. In \[6, 7\], the selected central vertices are those with $w_q = \delta_q, 1$, i.e. the ordinary leaf-removal algorithm, where a vertex of degree one is selected. The selected vertex is uncovered, while its neighbor is covered. The algorithm results in two possible situations: (1) all edges are removed and the constructed vertex cover has minimum size. (2) No leaf is left, while there still exist un-removed edges. In this case algorithm stops without producing a minimum vertex cover.

Bauer and Golinski showed that for the random Erdős–Rényi networks with mean degree $c < c_2 = e$, the case (1) occurs, while for networks with $c > c_2$ the leaf removal algorithm stops before all vertices are pruned, and so it cannot construct a minimum vertex cover \[7\]. The point $c = c_2$ is the emergence point of the $G_2$-core (or the ordinary core).

To find an estimation for minimum vertex cover for $c > e$, we use the $k$-leaf removal algorithm. The 2-leaf, i.e., the ordinary leaf removal algorithm has the best
from above of a minimum vertex cover for \( e \leq c < c_3 \). Similarly, 4-leaf removal algorithm constructs a vertex cover before the emergence of the G4-core, which occurs at \( c = c_4 = 8.917 \ldots \). For the ER random graphs with mean degree \( c_3 < c < c_4 \), the 4-leaf removal produces a vertex cover markedly more close to a minimum one than the Gazmuri algorithm. In other words, the \( k \)-leaf removal algorithm improves the Gazmuri algorithm for random networks with small and intermediate mean degree.

Using the rate equation approach presented in Sec. 3, we can find the size of a vertex cover constructed with the \( k \)-leaf algorithm. In this algorithm, at each time \( t \), a vertex of degree less than \( k \) is randomly chosen and uncovered. The average number of its neighbors, covered at time \( t \), is \( \frac{\sum_q q \theta(k - q) P(q, t)}{\sum_q \theta(k - q) P(q, t)} \). Denoting the average number of covered vertices at time \( t \) by \( X(t) = x(t)N \), we obtain the following differential equation for \( x(t) \) in the limit of large networks, \( N \to \infty \):

\[
\dot{x}(t) = \frac{\sum_q q \theta(k - q) P(q, t)}{\sum_q \theta(k - q) P(q, t)}.
\]

Figure 7 shows the size of a vertex cover constructed by the \( k \)-leaf removal algorithm for ER random networks. The curves are for \( k = 2, 3, \) and 4, and for the results obtained by the Gazmuri algorithm. In the Gazmuri algorithm, the average number of covered vertices at each time \( t \) is \( \langle q \rangle_t \), since at each time step a vertex with \( w_q = 1 \) is selected. As we can see in Fig. 7, the \( k \)-leaf removal algorithm constructs vertex covers of smaller size than the Gazmuri algorithm for \( c < c_k \).

---

**TABLE I.** \( G_k \)-core decomposition of real networks with the number of vertices \( N \) and the number of edges \( L \). \( k_{\text{max}} \) is the label of the innermost core. \( n_{k_{\text{max}} \text{-core}} \) and \( n_{2 \text{-core}} \) show the relative size of the innermost and outermost cores respectively. Similarly, \( l_{k_{\text{max}} \text{-core}} \) and \( l_{2 \text{-core}} \) show the normalized number of edges of the innermost and outermost cores respectively. \( x \) is the relative size of a vertex cover constructed by the \( k_{\text{max}} \)-leaf removal algorithm.

| Name                  | N   | L             | Ref | \( k_{\text{max}} \) | \( n_{k_{\text{max}} \text{-core}} \) | \( l_{k_{\text{max}} \text{-core}} \) | \( n_{2 \text{-core}} \) | \( l_{2 \text{-core}} \) | \( x \) |
|-----------------------|-----|---------------|-----|----------------------|--------------------------------|--------------------------------|-------------------|-------------------|-------|
| E. Coli, transcriptio | 97  | 212           | [21]| 3                    | 0.319                          | 0.793                          | 0.917             | 2.051             | 0.701 |
| AS Oregon             | 6474| 12572         | [22]| 2                    | 0.001                          | 0.001                          | 0.001             | 0.001             | 0.165 |
| Astrophysics          | 16046| 121251        | [23]| 31                   | 0.002                          | 0.045                          | 0.769             | 5.980             | 0.900 |
| C. Elegans, neural    | 297 | 2148          | [24]| 3                    | 0.885                          | 6.037                          | 0.915             | 6.447             | 0.656 |
| Cond-Mat              | 16264| 47594        | [25]| 10                   | 0.006                          | 0.003                          | 0.618             | 1.884             | 0.768 |
| Dolphins              | 62  | 159           | [26]| 3                    | 0.161                          | 0.290                          | 0.618             | 1.884             | 0.768 |
| Email-Enron           | 36692| 183831        | [27]| 3                    | 0.0004                         | 0.001                          | 0.389             | 1.052             | 0.435 |
| Linux                 | 30834| 213217        | [28]| 5                    | 0.0003                         | 0.0008                         | 0.147             | 0.375             | 0.442 |
| petster-friendship-hamster | 1858 | 12534       | [29]| 8                    | 0.010                          | 0.047                          | 0.584             | 2.664             | 0.500 |
| Sociopatterns-Infectious | 410 | 2765         | [30]| 9                    | 0.056                          | 0.443                          | 0.912             | 6.090             | 0.814 |
| PGPgiantcompo         | 10680| 24316         | [31]| 17                   | 0.001                          | 0.014                          | 0.158             | 0.483             | 0.776 |
| US Air Trasportation  | 500  | 2980          | [32]| 3                    | 0.008                          | 0.012                          | 0.260             | 0.494             | 0.432 |
| Yeast- protein        | 2284| 6646          | [33]| 3                    | 0.003                          | 0.011                          | 0.025             | 0.052             | 0.367 |
V. REAL-WORLD NETWORKS

We apply the $k$-leaf removal to a number of real-world networks and find cores of these networks. The characteristics of real-world networks, analyzed in the paper, are listed in Table I. The most outer core is the $G_2$-core or ordinary core and the innermost core is labeled by $k_{\text{max}}$. We present the relative size and number of edges of the innermost and the outermost ($k = 2$) $G_k$-cores in Table I. The vertex covers are hierarchically constructed by applying the $k$-leaf removal. In Table I, $x$ shows the relative size of a vertex cover before appearance of the $G_{k_{\text{max}}}$-core. Clearly, the smaller $k$ is, the more near-optimal vertex cover is constructed by the $k$-leaf algorithm. We find that many real social networks are decomposed to a large hierarchy of the $G_k$-cores. For instance the layers of arxiv networks, i.e. cond-mat, astro-ph, hep-th, . . . , have the highest numbers of the $G_k$-cores nested into each other among networks analyzed in this paper. In contrast, the food webs and biological networks have a small number of cores. Using the visualization algorithm proposed in [1], visualization of astrophysics network in 2005 [23] and transcriptional regulation network [21] are presented as two examples in Fig. 8. The regulation network has a few cores while the astrophysics network has around 30 cores in our proposed network decomposition scheme.

VI. CONCLUSION

In this work we have generalized the ordinary core subgraph to the $G_k$-cores. We proposed the $k$-leaf removal algorithm as a generalization of the ordinary leaf removal which is useful in constructing of near-optimal configurations in combinatorial optimization problems, e.g. a minimum vertex cover. The $k$-leaf pruning algorithm enables us to decompose large random networks into a hierarchical set of progressively nested subgraphs which we called the $G_k$-cores. Our approach can also be considered as a generalization of the ordinary $k$-core decomposition. In our pruning at each time step, not only the vertices of degree less than $k$ but also their nearest neighbors are removed. Following this pruning, the network is decomposed into a hierarchy of progressively nested $G_k$-cores such that the vertices, belonging to the inner cores, and also their first neighbors are of higher degree and well connected. Using generating function technique, we found the structural characteristics and the emergence point of the $G_k$-cores for the Erdős–Rényi and scale-free random networks. Similarly to the ordinary $k$-core percolation, $G_k$-cores show a discontinuous phase transition for $k \geq 3$. We compared our results with numerical simulations and observed a complete agreement. In addition, we used the rate equation approach to describe the evolution of degree distribution of random networks during the $k$-leaf pruning algorithm. We checked that the result of the application of this approach to the Erdős–Rényi graph completely coincides with the exact result obtained by the analytical calculations. Also the rate equation approach allows us to find the size of minimum vertex covers. We showed that with applying the $k$-leaf pruning algorithm, we can obtain hierarchically near optimal vertex covers for random networks. We have applied the $k$-leaf removal algorithm to a number of real-world networks. Among the real networks explored, the social networks have a large $k_{\text{max}}$.

We emphasize that in contrast to the $k$-core decomposition, the $G_k$-cores are not about the classification of vertices in a network according to their properties but rather about the characterization of a specific robustness of this network. Suppose that a network be attacked by a virus infecting/removing weak vertices (of degree less than $k$) and their nearest neighbors. The $G_k$-cores show what will remain of the network after this epidemics. The resilience/robustness of a network against this kind of epidemics is characterized by the size of its $G_k$-core. This may explain why the social networks that we explored have a large $k_{\text{max}}$. 

FIG. 8. (color online). Graphical visualization of the $G_k$-core decomposition of (a) astrophysics and (b) transcriptional regulation networks.
[1] I. Alvarez-Hamelin, L. Dell-Asta, A. Barrat, and A. Vespignani, K-core Decomposition of Internet Graphs: Hierarchies, Self-Similarity and Measurement Biases, arXiv:cs.NI/0511035 (2005).
[2] S.B. Seidman, Network Structure and Minimum Degree, Social Networks 5, 269 (1983).
[3] M. Kitsak, L. K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H. E. Stanley, and H. A. Makse, Identification of Influential Spreaders in Complex Networks, Nature Phys. 6, 888 (2010).
[4] S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes, K-core Organization of Complex Networks, Phys. Rev. Lett. 96, 046001 (2006).
[5] S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes, Critical Phenomena in Complex Networks, Rev. Mod. Phys. 80, 1275 (2008).
[6] R. M. Karp and M. Sipser, Maximum Matching in Sparse Random Graphs, Proc. 22nd Annual IEEE Symp. on Foundations of Computer Science pp. 364-375 (1981).
[7] M. Bauer and O. Golinelli, Core Percolation in Random Graphs: A Critical Phenomena Analysis, Eur. Phys. J. B 24, 339 (2001).
[8] M. Bauer and O. Golinelli, Exactly Solvable Model with Two Conductor-Insulator Transitions Driven by Impurities, Phys. Rev. Lett. 86, 2621 (2001).
[9] M. Weigt and A.K. Hartmann, Number of Guards Needed by a Museum: A Phase Transition in Vertex Covering of Random Graphs, Phys. Rev. Lett. 84 6118 (2000).
[10] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, Controllability of complex networks, Nature 473, 167 (2011).
[11] T. Jia, and M. Pósfai, Connecting Core Percolation and Controllability of Complex Networks, Sci. Rep. 4, 5379 (2014).
[12] M. Weigt and A. K. Hartmann, Phase Transitions in Combinatorial Optimization Problems, (Wiley-VCH, Weinheim, 2005).
[13] P.G. Gazmuri, Independent Sets in Random Sparse Graphs, Networks 14, 367 (1984)
[14] M. Weigt, Dynamics of Heuristic Optimization Algorithms on Random Graphs, Eur. Phys. J. B 28, 369 (2002).
[15] M. E. J. Newman, S. H. Strogatz, and D. J. Watts, Random Graphs with Arbitrary Degree Distributions and Their Applications, Phys. Rev. E 64, 026118 (2001).
[16] N. Azimi-Tafreshi, S. N. Dorogovtsev, and J. F. F. Mendes, em Core Organization of Directed Complex Networks, Phys. Rev. E. 87, 032815 (2013).
[17] Y.-Y. Liu, E. Csóka, H. Zhou, and M. Pósfai, Core Percolation on Complex Networks, Phys. Rev. Lett. 109, 205703 (2012).
[18] M. Catanzaro and R. Pastor-Satorras, Analytic Solution of a Static Scale-Free Network Model, Eur. Phys. J. B 44, 241 (2005).
[19] K. -I. Goh, B. Kahng, and D. Kim, Universal Behavior of Load Distribution in Scale-Free Networks, Phys. Rev. Lett. 87, 278701 (2001).
[20] M. Weigt and A. K. Hartmann, Phase Transitions in Combinatorial Optimization Problems, (Wiley-VCH, Weinheim, 2005).
[21] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon, Network motifs in the transcriptional regulation network of Escherichia coli , Nat. Genet. 31, 64 pmid:11967538 (2002).
[22] J. Leskovec, J. Kleinberg, and C. Faloutsos, KDD ’05, Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations, Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, 177 (2005).
[23] M. E. J. Newman, The Structure of Scientific Collaboration Networks, Proceedings of the National Academy of Sciences 98(2), 404 (2001).
[24] J. D. Watts, and S. H. Strogatz, Collective Dynamics of Small-World Networks, Nature 393 440 (1998).
[25] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, The Bottlenose Dolphin Community of Doubtful Sound Features a Large Proportion of Long-Lasting Associations, Behavioral Ecology and Sociobiology 54 (4), 396 (2003).
[26] M. W. Mahoney, A. Dasgupta, K. J. Lang, and J. Leskovec, Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters, Internet Mathematics 6 29 (2009).
[27] J. Kunegis, KONECT: The Koblenz Network Collection, WWW ’13 Companion, Proceedings of the 22Nd International Conference on World Wide Web, 1343 (2013).
[28] L. Isella, J. Stehlé, A. Barrat, C. Cattuto, J.-F. Pinton, and W. Van den Broeck, What’s in a Crowd? Analysis of Face-to-Face Behavioral Networks, J. Theor. Biol. 271, 166 (2011).
[29] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera and A. Arenas, Models of Social Networks Based on Social Distance Attachment, Phys. Rev. E 70, 056122 (2004).
[30] V. Colizza, R. Pastor-Satorras and A. Vespignani, Reaction-Diffusion Processes and Metapopulation Models in Heterogeneous Networks, Nature Physics 3, 276 (2007).
[31] D. Bu, Y. Zhao, L. Cai, H. Xue, X. Zhu, H. Lu, J. Zhang, S. Sun, L. Ling, N. Zhang, G. Li, and R. Chen1, Topological Structure Analysis of the Protein-Protein Interaction Network in Budding Yeast, Nucleic Acids Res. 31(9) 2443 (2003).