Random graphs containing arbitrary distributions of subgraphs

Brian Karrer and M. E. J. Newman
Department of Physics, University of Michigan, Ann Arbor, MI 48109 and
Santa Fe Institute, 1399 Hyde Park Road, Santa Fe, NM 87501

Traditional random graph models of networks generate networks that are locally tree-like, meaning that all local neighborhoods take the form of trees. In this respect such models are highly unrealistic, most real networks having strongly non-tree-like neighborhoods that contain short loops, cliques, or other biconnected subgraphs. In this paper we propose and analyze a new class of random graph models that incorporates general subgraphs, allowing for non-tree-like neighborhoods while still remaining solvable for many fundamental network properties. Among other things we give solutions for the size of the giant component, the position of the phase transition at which the giant component appears, and percolation properties for both site and bond percolation on networks generated by the model.

I. INTRODUCTION

It was pointed out by Rapoport in the 1940s [1] and more recently by Watts and Strogatz [2] that many observed networks contain a statistically surprising number of triangles—sets of three vertices connected by three edges. More generally it has been noted that many small subgraphs occur in networks in numbers greater than one would expect purely on the basis of chance [3–5]. In some cases there are good reasons for the frequent occurrence of a particular subgraph. Social networks, for instance, display a lot of triangles because an individual's friends are likely themselves to be friends. In technological and biological networks certain subgraphs appear to perform modular tasks that contribute to the networks' overall operation [3] and hence may be evolutionarily favored. In other cases the reason for the appearance of the subgraphs is unclear, but the empirical evidence for their presence is nonetheless convincing, and hence if we wish to model these networks accurately the appropriate subgraphs must be included.

Unfortunately, few practical models of complex networks exist that include significant densities of arbitrary small subgraphs. Random graph models have been developed that incorporate numerous features of real-world networks, including arbitrary degree distributions [6, 7], correlations [8, 9], bipartite and multi-partite structure, hierarchy [10], vertex ordering [11], and even geometry [12], but until recently there were no equivalent random graph models incorporating nontrivial densities of general subgraphs. Exponential random graphs do allow general subgraphs in principle, but in practice they are difficult to analyze and moreover can display pathological behaviors that limit their usefulness.

In this paper we introduce a class of models that build upon conventional random graphs and incorporate nontrivial densities of arbitrary subgraphs, while remaining exactly solvable in the limit of large network size for a variety of properties both local and global, including degree distributions, clustering coefficients, component sizes, percolation properties, and others. A number of new models have appeared in recent years that achieve similar goals in other contexts [13–19], and many of these are, as we will see, special cases of the general formalism introduced in this paper. We begin our discussion with a description of a special case of the formalism, that of a network containing only random edges and triangles, which was described previously in Refs. [16, 19]. Then, using this model as a starting point, we further develop the general foundations that will allow us to model networks with any choice of subgraphs we please.

II. A SIMPLE EXAMPLE

Consider to begin with the standard Poisson random graph, famously studied by Erdős and Rényi in the 1950s and 60s [20, 21]. In this model each distinct pair of vertices, out of n vertices total, is (or is not) connected by an edge with independent probability $p = c/n$ (or $1 - p$), where c is a constant. A particular subgraph with n_s vertices and m_s edges can occur in $\binom{n}{n_s} \simeq n^{n_s}/n_s!$ positions in such a graph, each with probability

$$p^{m_s}(1-p)^{\binom{n_s}{2}-m_s} \simeq \left(\frac{c}{n}\right)^{m_s}. \qquad (1)$$

Thus the expected number of occurrences of such a subgraph is $\mathrm{O}(n^{n_s-m_s})$. If the subgraph is connected then $m_s \geq n_s - 1$ with the equality applying only in the case of a tree. Thus the number of connected subgraphs grows with system size only for the case of trees and for all other subgraphs is constant or decreasing, making the density of all such non-tree subgraphs vanish in the large-n limit. A similar argument applies to most of the extensions of the random graph. These graphs are thus "locally tree-like"—all small connected subsets of vertices within the network are trees. This means, for instance, that such graphs have a vanishing density of triangles in the limit of large size, in strong contrast to real networks, which frequently have a very high triangle density [22].

Although it is unrealistic, the locally tree-like nature of random graphs is also the crucial feature that makes the model tractable. The calculations of the size of the giant component, the size distribution of small components, the percolation features of the network, and many other properties are all crucially dependent on the tree-like property. This is unfortunate, since the tree-like nature of the network is destroyed by the introduction of a finite density of any subgraph that contains one or more loops, which suggests that generalizations of random graph models containing such subgraphs may be intrinsically unsolvable. As we show in this paper, however, this turns out not to be the case. By exploiting tree-like structure at a higher level, the level of the so-called "factor graph," we can introduce arbitrary distributions of subgraphs into the network, including those containing loops, and still solve exactly for global properties of the network, even though the network is now explicitly not locally tree-like. As a first demonstration of the process, consider the following simple model, which we will call the "edge–triangle" model.

The edge–triangle model was proposed previously in [16, 19] as a way to incorporate the phenomenon of clustering or transitivity into random graphs. It generalizes the configuration model, the standard model of a network with arbitrary degree distribution [6, 7]. In the configuration model one specifies the number of edges attached to each vertex i—the so-called "degree sequence"—as the fundamental parameters of the network. In the edge–triangle model one specifies instead the number t_i of triangles that each vertex participates in along with the number s_i of "single edges," meaning edges that are not part of a triangle. One can picture vertex i as having s_i "stubs" of edges emerging from it and t_i corners of triangles. Then an edge–triangle network is generated by choosing stubs randomly in pairs and joining them to form complete edges until no stubs remain, and also choosing triangle corners in threes and joining them to form complete triangles until no corners remain. The end result is a network drawn uniformly at random from the set of all possible matchings of the stubs and corners, and the edge–triangle model is defined to be the ensemble of such matchings in which each appears with equal probability. Note that to create a complete matching we require that the total number of stubs be a multiple of two and the total number of corners a multiple of three.

The undirected networks generated by the edge–triangle model are clearly not locally tree-like, since they contain triangles. Yet one can still proceed with analytic calculations. Consider for instance the following calculation of the size of the giant component (the extensive part of the network in which any vertex can reach any other via at least one path).

Let us define the joint degree distribution $p(s,t)$ for our network to be the fraction of vertices connected to s single edges and t triangles. As with other random graph models, it's helpful to define a generating function G_0 for this degree distribution:

$$G_0(z_1,z_2) = \sum_{st} p(s,t)\, z_1^s z_2^t. \tag{2}$$

Also important is the so-called "excess degree distribution." Excess degree is a property of the vertex one reaches by following an edge in a network and is normally defined to be the number of edges connected to such a vertex other than the edge one followed in the first place. In the edge–triangle model, where the "degree" of each vertex is denoted by the two numbers s and t, there are now two corresponding excess degrees. If one follows a single edge to reach a vertex then the excess degree is given by the number t of triangles attached to that vertex and the number s of single edges other than the edge via which we arrived. Similarly if one follows a triangle to reach a vertex then the excess degree is given by the number of single edges attached to that vertex and the number of triangles other than the one via which we arrived. It is straightforward to show that the distributions of these excess degrees are, respectively,

$$q(s,t) = \frac{(s+1)}{\langle s\rangle} p(s+1,t), \tag{3}$$

$$r(s,t) = \frac{(t+1)}{\langle t\rangle} p(s,t+1), \tag{4}$$

where $\langle s\rangle$ and $\langle t\rangle$ are the average numbers of stubs and corners at a vertex in the network as a whole. The generating functions for the excess degree distributions are

$$G_1(z_1,z_2) = \sum_{st} q(s,t)\, z_1^s z_2^t, \tag{5}$$

$$G_2(z_1,z_2) = \sum_{st} r(s,t)\, z_1^s z_2^t. \tag{6}$$

The calculation of the giant component size now proceeds as follows. Let u_1 be the probability that the vertex reached by following a single edge (an edge that is not part of a triangle) is not connected to the giant component by any of its other edges or triangles, and let u_2 be the probability that *neither* of the vertices reached by following a triangle is connected to the giant component by any of their other triangles or edges. If the vertex reached by following an edge is connected to s other edges and t triangles then the probability that none of them leads to the giant component is $u_1^s u_2^t$, where s and t are distributed according to the excess degree distribution $q(s,t)$. Averaging over this distribution, we find that

$$u_1 = \sum_{st} q(s,t) u_1^s u_2^t = G_1(u_1,u_2). \tag{7}$$

Similarly, if a vertex reached by following a triangle is is connected to t other triangles and s edges then the probability that none of them leads to the giant component is again $u_1^s u_2^t$, but with s and t now distributed according to $r(s,t)$. Averaging over $r(s,t)$ then gives an average probability of $G_2(u_1,u_2)$ and the total probability for both vertices reached via a triangle is the square of this quantity:

$$u_2 = \bigl[G_2(u_1,u_2)\bigr]^2. \tag{8}$$

Finally, a randomly selected vertex with s stubs and t triangles is not in the giant component if none of its neighbors are in the giant component, which happens with probability $u_1^s u_2^t$ where s and t are distributed according to $p(s,t)$. Averaging over $p(s,t)$, we find the probability of not being in the giant component to be $G_0(u_1,u_2)$, and the probability S of being in the giant component is one minus this:

$$S = 1 - G_0(u_1,u_2). \tag{9}$$

Between them Eqs. (7) to (9) give the size of the giant component in our edge–triangle network as a fraction of the size of the whole network. While they cannot always be solved analytically, they can be solved numerically by simple iteration: one makes an initial guess about the values of u_1 and u_2 and iterates (7) and (8) to convergence, then substitutes the resulting values into (9). (The size of the giant component is only one example of a quantity that can be calculated within the edge–triangle model. For other examples, including clustering coefficient and percolation properties see Ref. [16].)

The calculation of the giant component size in fact follows quite closely the method used for other, locally tree-like random graph models such as the configuration model [7] and does not appear significantly more complex despite the addition of triangles, which destroy the tree-like property. The reason for this is that, while the edge–triangle model is indeed not tree-like in a naive sense, it is still tree-like at a higher level, above the level of the triangles. Specifically, in the limit of large graph size, a finite-sized local neighborhood of a vertex in the edge–triangle model is a connected graph whose largest biconnected component (of which there can be many) is a triangle. This means that if we regard each triangle in the network as a single three-vertex unit (and each single edge as a two-vertex unit) then local neighborhoods are tree-like at the level of these units. We will develop this idea further shortly.

III. A GENERAL MODEL

The edge–triangle model provides a simple, solvable model of networks that contain triangles. It does, however, have some disadvantages. In particular, the probability that any two triangles connected to the same vertex will share an edge vanishes in the limit of large graph size, a direct consequence of the fact that the networks generated by the model are tree-like above the level of triangles. In real networks, by contrast, it is a common occurrence that triangles share an edge, and hence the model is unrealistic in this respect.

One way to solve this problem would be to modify the model in some way to encourage triangles to share edges, but this is unsatisfactory because, in so doing, we would destroy the locally tree-like property of the network (at the triangle level) that allows its solution. Instead, therefore, we propose an alternative approach: we consider two triangles that share an edge to form a new network element, analogous to the triangle, but now with four vertices instead of three—see Fig. 1. Our approach is to create a model that introduces a specified distribution of these larger elements in exactly the same way that we previously introduced triangles. More generally, it is possible to define a model in which we introduce arbitrary distributions of any subgraphs we please. (The possibility of such a model was mentioned briefly in Refs. [16] and [19].) As we will show, such models can always be viewed as tree-like at a suitable higher level and thereby solved exactly for properties both local and global in the limit of large size.

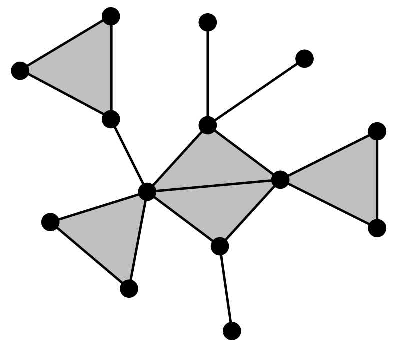


FIG. 1: A small network made of single edges, triangles, and "diamond" subgraphs composed of two overlapping triangles.

A. Subgraphs and roles

In the model we propose, we first specify a set of subgraphs that will be added to the network. Three examples of possible sets are shown in Fig. 2. The network will be created by specifying the number of each of the subgraphs attached to each vertex and then sampling randomly from the (usually large) set of compatible networks. The edge–triangle model of Section II is an example of such a model in which the set of subgraphs numbers just two—the single edge and the triangle as shown in Fig. 2b. The model of this section generalizes the edge–triangle model to arbitrary subgraph sets of arbitrary size.

This generalization, however, introduces an important new feature to the model that was not present in the edge–triangle model. It is not sufficient, in the general case, merely to specify how many copies of each subgraph are connected to each vertex because the vertices in the subgraphs can play more than one role. Consider the diamond-shaped subgraph of Fig. 2b. Two of the vertices in this subgraph have three incident edges while the others have two. Specifying only that a vertex belongs to such a subgraph is therefore ambiguous. We need to specify also which role the vertex plays (a point made previously by Miller [19]). A vertex could, for example,

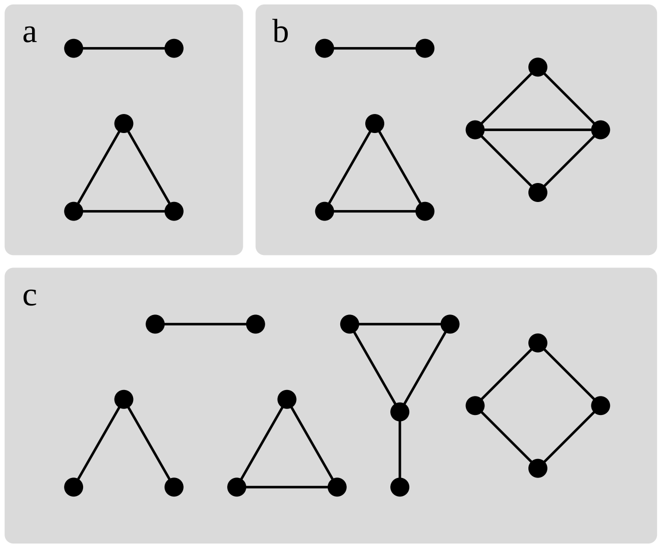


FIG. 2: Three examples of possible subgraph sets for the model described in the text. (a) The subgraph set for the edge–triangle model of Section II. (b) The subgraph set for the model with overlapping triangles shown in Fig. 1. (c) A more extensive set that includes both singly and doubly connected subgraphs.

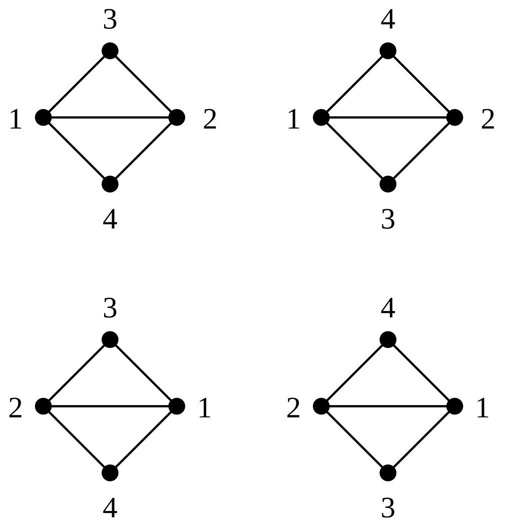


FIG. 3: Vertices 1 and 2 in this diamond-shaped subgraph constitute a role as defined in the text, as do vertices 3 and 4.

participate in five diamonds, playing the three-edge role in, say, four of them, and the two edge role in the fifth.

The concept of a role in a subgraph can be made precise by employing concepts from graph theory, specifically the concepts of automorphisms and orbits. Consider a subgraph in which each vertex has a unique identifying label. An *automorphism* of the subgraph is a permutation of the labels such that the set of label pairs joined by edges remains unchanged. Consider Fig. 3, for instance, which shows the "diamond" subgraph from Fig. 2b with integer labels on its vertices. The four panels within the figure show four different automorphisms of the graph so that there is, for instance always an edge between vertices 1 and 2 in each panel, regardless of the permutation of the labels. Formally, the set of all automorphisms of a subgraph G forms a group, which is called the automorphism group and denoted $\mathrm{Aut}(G)$.

Now consider a particular vertex within our subgraph. The *orbit* of the vertex is defined to be the set of other vertices with which it shares at least one label under the label permutations of the automorphism group. The left-most vertex in the subgraph of Fig. 3, for example, shares labels 1 and 2 with the right-most vertex, but shares no labels with either of the other vertices. Hence the left and right vertices form an orbit. Similarly the top and bottom vertices form an orbit. The orbits correspond to the subgraph "roles" discussed above: they enumerate each of the topologically distinct situations in which a vertex can find itself.

Unfortunately, the problem of calculating the orbits of the automorphism group of a graph is computationally hard and no polynomial-time algorithm is known. Our focus here, however, is on relatively small subgraphs for which roles can be found quite simply by hand.

Once subgraph roles are defined then we can specify completely the number of subgraphs in which each vertex participates and the manner in which it does so. We assign to each vertex a set of role indices $d_1, d_2, \ldots$, specifying the number of times each vertex plays each possible role. For instance, in the subgraph set shown in Fig. 2b, there are four different roles—one for each of the first two subgraphs and two for the third subgraph. Thus each vertex would in this case be assigned four role indices $d_1 \ldots d_4$ giving the number of times it plays each of these roles. The set of role indices for a given vertex form a vector $\mathbf{d}$ with c non-negative integer elements, where c is number of different roles in the subgraph set. This vector plays a part similar to that played by the degree in conventional network models and, by analogy with the degree sequence in such models, we will call the set of vectors for all vertices the *role sequence*.

Just as with degree sequences in other models, not all role sequences correspond to possible networks. Consider again the diamond subgraph of Fig. 3, with its two roles. Because both roles appear two times in this subgraph, the total number of times vertices in a network participate in each of the roles must be equal to the same multiple of two. More generally, the total number of times vertices play a particular role in a particular subgraph must always be an multiple of the number of times that role appears in the subgraph, and moreover that multiple must be the same for all roles of the given subgraph. Role sequences that fulfill these conditions are said to be *graphical*.

B. Network creation

Given a graphical role sequence, a random network drawn from the ensemble of our proposed model can be generated in a straightforward fashion. The role indices for each vertex dictate the numbers of stubs of each type attached to the vertex and creation of the network involves working through each of the subgraph types in turn and repeatedly choosing a set of stubs at random in the appropriate combination for that subgraph and connecting them together to make the subgraph in question. When all stubs for a given subgraph type have been exhausted we move on to the next subgraph until the list of subgraphs is also exhausted. The end result is a matching of stubs to form subgraphs, drawn uniformly from the set of all possible such matchings.

It is possible in the creation of a given subgraph that two or more stubs attached to the same vertex will be drawn from the set of available stubs. If this happens the resulting network will contain either an edge that connects a vertex to itself—a self-edge—or multiple edges between the same pair of vertices—a multiedge—or both. In the model we propose, such edges are allowed to exist, even though they are prohibited in many real-world networks. A similar situation arises in the standard configuration model of graph theory and, as in that model, the densities of self-edges and multiedges in our model both vanish in the limit of large network size as $1/n$ and hence can be neglected in this limit. It is possible to create models that generate "simple graphs," i.e., graphs without self-edges or multiedges, but such models are considerably more complicated to work with, both analytically and computationally.

In the calculations presented here, we will not assume that we are given a complete role sequence for all vertices, but only that we are given a role distribution $p(\mathbf{d})$, which specifies the probability that a randomly chosen vertex has role vector $\mathbf{d}$ (or equivalently specifies the fraction of vertices having role vector $\mathbf{d}$ in the large-n limit). The network itself will then be defined by drawing a role sequence from this distribution and forming a random matching of the set of stubs specified by the sequence.

The constraint that the role sequence must be graphical imposes a corresponding constraint on the role distribution. Let $\langle d_r\rangle = \sum_{\mathbf{d}} d_r p(\mathbf{d})$, where the sum is over all possible values of the role vector $\mathbf{d}$. Then the expected number of role-r stubs in the entire network is $n\langle d_r\rangle$ and the number of occurrences of the corresponding subgraph is $n\langle d_r\rangle/n_r$, where n_r is the number of times that role r appears in a single instance of the subgraph. We can calculate a corresponding figure for the number of occurrences of the subgraph using any of its other roles and all of these figures must be equal if the role sequence is to be graphical. Thus we must have

$$\frac{\langle d_r\rangle}{n_r} = \frac{\langle d_s\rangle}{n_s} \tag{10}$$

for all pairs r, s of roles within the same subgraph.

Assuming we have a role distribution $p(\mathbf{d})$ that satisfies this constraint, the procedure for generating a network is first to draw a complete role sequence $\mathbf{d}_i$, $i = 1 \ldots n$ for the network. It is possible that, despite the constraint on $p(\mathbf{d})$, this sequence will not be graphical, because of statistical fluctuations in the role vectors drawn. If this is the case, we choose a vertex uniformly at random, discard its role vector and draw another from $p(\mathbf{d})$. We repeat this process until the role sequence is graphical. Then the network itself is generated by random matching of stubs as described above.

The role distribution also fixes the conventional degree distribution for the network. Each role r has some number k_r of associated edges, so a vertex with role indices $d_1 \ldots d_c$ has $\sum_r k_r d_r$ edges. The fraction of vertices with total degree k is then

$$p(k) = \sum_{\mathbf{d}} p(\mathbf{d})\delta\Big(k, \sum_r k_r d_r\Big). \tag{11}$$

While nothing in the above strictly demands it, we will here consider only *sparse* graphs for which the average total degree of a vertex $\langle k\rangle$ is constant in the limit of large n. From Eq. (11) we have

$$\begin{aligned}\langle k\rangle &= \sum_k kp(k) = \sum_k k \sum_{\mathbf{d}} p(\mathbf{d})\delta\Big(k, \sum_r k_r d_r\Big)\\ &= \sum_r k_r \sum_{\mathbf{d}} d_r p(\mathbf{d}) = \sum_r k_r \langle d_r\rangle. \end{aligned}\tag{12}$$

IV. BIPARTITE GRAPH REPRESENTATION

Like the edge-triangle model (of which it is a superset), this subgraph model is not in general locally tree-like but can be thought of as tree-like at a higher level. If one considers the subgraphs as coherent graph units in their own right, then the network is still tree-like at the level of these units. This idea can be made more precise as follows.

Our subgraph model has an alternative representation as a bipartite graph: a network with two types of vertices and edges running only between unlike types. In this representation one of the types of vertices corresponds to the vertices of the original network while the other corresponds to the subgraphs, and each original vertex is connected by an edge to the subgraphs in which it participates. (In other circumstances this representation would be called a *factor graph.*) In order to distinguish the different roles that a vertex can play in a subgraph the edges can be labeled with the appropriate role numbers r.

A vertex in the original graph having role vector $\mathbf{d} = (d_1, d_2, \ldots, d_c)$ is represented in the bipartite graph by a vertex with d_1 incident edges marked with role label 1, d_2 edges marked with label 2, and so forth. On the other side of the bipartite graph, every vertex representing a given subgraph in the original network has the same number of stubs, again labeled by role, one for each vertex in the subgraph.

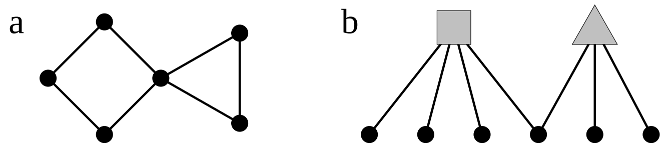


FIG. 4: (a) A small network composed of a square subgraph and a triangle and (b) the bipartite representation of the same network.

The process of creating a network in the original representation of the model is equivalent in the bipartite representation to a random matching of stubs subject to the constraint that every edge created joins a single vertex stub to a single subgraph stub having the same role label. Figure 4 illustrates the equivalence between the two representations.

The important point to notice about this construction is that in the limit of large graph size the bipartite graph is strictly locally tree-like for the same reason that standard random graphs are locally tree-like—the density of loops of any fixed size falls off as $1/n$ as n becomes large. Thus computations can be carried out on the bipartite graph using standard techniques for such networks [7] and the results can then be translated back into the language of the original network to calculate properties of the subgraph model. In the following sections we give a number of such calculations of increasing complexity. The first and simplest is the calculation of the expected number of occurrences of each subgraph in the subgraph model.

A. Subgraph counts

If we want to know how many times a particular subgraph occurs in our subgraph model we must allow for three distinct ways in which a subgraph can be created: it can be added explicitly as one of the set of allowed subgraphs; it can be added implicitly as part of a larger subgraph; or it can be created by putting two or more subgraphs together. For instance, a triangle can be formed if we explicitly add a triangle to the network, if we add a larger subgraph containing a triangle (such as the diamond subgraph in Fig. 2b), or we can add three independent single edges to the network that happen to coincide and create a triangle.

There is an important distinction to be drawn here between subgraphs that are biconnected and those that are only singly connected. Recall that a biconnected graph is one in which there exist at least two vertex-independent paths between any pair of vertices, or equivalently in which at least two vertices must be removed to disconnect any pair of vertices. It is trivial to see that a biconnected subgraph can only be created by joining two or more other subgraphs if the joined subgraphs meet at at least two places—if they meet at only one then the removal of that one vertex will disconnected them and hence they cannot have been biconnected. The locally tree-like property of the bipartite representation of the subgraph model, however, implies that the probability of two subgraphs of given size meeting at two places vanishes in the limit of large graph size. Such subgraphs would form a loop of length four in the bipartite graph composed of the two subgraphs and the two vertices at which they meet. Since the bipartite graph is locally tree-like such loops do not occur (or more properly occur with vanishing density in the limit of large n). This implies that the density of biconnected subgraphs formed by the joining of others vanishes in our subgraph model and hence that if we wish to know about biconnected subgraphs we need concern ourselves only with those added either explicitly to the network or implicitly as part of a larger subgraph. And the expected number of such subgraphs is trivial to calculate—we simply sum up the appropriate numbers.

One important consequence of this result is that we can calculate the clustering coefficient of our network in a straightforward manner. The clustering coefficient is defined to be [7]

$$C = \frac{3 \times (\text{number of triangles in network})}{(\text{number of connected triples})} = \frac{3N_\Delta}{N_3}. \tag{13}$$

Since the triangle is a biconnected subgraph (in fact the smallest such subgraph), the number of triangles N_Δ is given simply by counting how many triangles appear in each allowed subgraph, multiplying by the expected number of the respective subgraphs, and summing over subgraphs. The number of connected triples N_3 is given by

$$n \sum_k \binom{k}{2} p(k) = \tfrac{1}{2} n \biggl\langle \biggl[\sum_r k_r d_r\biggr]^2 - \sum_r k_r d_r \biggr\rangle, \tag{14}$$

and hence we can evaluate Eq. (13).

For singly connected subgraphs the situation is more complicated. Such subgraphs can be created by joining together two or more subgraphs at single vertices, which happens frequently even in locally tree-like networks. As a result it is not trivial to calculate the expected number of singly connected subgraphs in a network, except in a few special cases. (The connected triple or 2-star of Eq. (13) is an example of a singly connected subgraph whose number can be calculated in a straightforward fashion, but this is the exception rather than the rule.)

B. Giant component

One can also exploit the bipartite representation of the subgraph model to find the size of the giant component in the model. As with the treatment of the edge–triangle model in Section II, we introduce excess degree distributions, but defined now in terms of the bipartite network. Consider a subgraph node in the bipartite network and imagine following one of the edges connected to it, an edge labeled with role r, to the vertex at its other end. Then, if that vertex has overall role indices $d_1, \ldots, d_r+1, \ldots, d_c$ and we exclude the edge along which we arrived, it will have "excess" role indices $d_1, \ldots, d_r, \ldots, d_c$. Moreover, since the probability of our arriving at a vertex with role index d_r is proportional to d_r the excess role indices are distributed according to the probability distribution

$$q_r(\mathbf{d}) = \frac{d_r+1}{\langle d_r\rangle} p(d_1, \ldots, d_r+1, \ldots, d_c), \qquad (15)$$

and there is a corresponding distribution for every other role.

As in the edge–triangle model it is convenient to keep track of the various role distributions using their generating functions, defined by

$$G_0(\mathbf{z}) = \sum_{\mathbf{d}} p(\mathbf{d}) z_1^{d_1} \ldots z_c^{d_c}, \qquad (16)$$

$$G_r(\mathbf{z}) = \sum_{\mathbf{d}} q_r(\mathbf{d}) z_1^{d_1} \ldots z_c^{d_c}. \qquad (17)$$

Note that given the generating function $G_0(\mathbf{z})$ we can calculate the generating function $H(z)$ for the conventional degree distribution in a straightforward manner using Eq. (11) thus:

$$\begin{aligned} H(z) &= \sum_k p(k)\, z^k = \sum_k \sum_{\mathbf{d}} p(\mathbf{d})\, \delta\Big(k, \sum_r k_r d_r\Big) z^k \\ &= \sum_{\mathbf{d}} p(\mathbf{d}) z^{k_1 d_1} \ldots z^{k_c d_c} \\ &= G_0\big(z^{k_1}, \ldots, z^{k_c}\big), \end{aligned} \qquad (18)$$

where k_r is, as previously, the number of edges a vertex gains by virtue of playing role r in a subgraph.

We also need to define generating functions for the role distributions on the other side of the bipartite graph, the side that represents the subgraphs. These generating functions, however, take a relatively simple form, since every role belongs to only one subgraph and every instance of a given subgraph contains the same distribution of roles. Let us define n_r as before to be the number of times role r occurs in its own subgraph, and let us give the subgraphs unique labels such that g_r is the label of the graph in which role r appears. Then the generating function for the excess role distribution of a subgraph node reached by following an edge representing role r is

$$F_r(\mathbf{z}) = \frac{1}{z_r} \prod_{s=1}^{c} z_s^{n_s \delta_{g_r g_s}}, \qquad (19)$$

where δ_{ij} is the Kronecker delta. (We can in principle write down a generating function for the normal (non-excess) role distribution of the subgraph nodes, but we omit it because it's not needed for our calculations.)

Armed with these generating functions, we can compute the size of the giant component in the network as follows [23]. Define u_r to be the probability that a vertex is *not* connected to the giant component as a result of its playing role r in a given subgraph. This occurs if and only if none of the other vertices in that subgraph, regardless of role, are themselves members of the giant component, or, in the language of the bipartite representation, if none of the other edges connected to the appropriate subgraph node lead to vertices that are in the giant component.

Similarly, let v_r be the probability that a vertex that plays role r in a particular subgraph is not a member of the giant component by virtue of any of the *other* roles it plays. In the language of the bipartite representation, none of the other edges incident on such a vertex connect it to the giant component.

In terms of these variables we have

$$u_r = \frac{1}{v_r} \prod_{s=1}^{c} v_s^{n_s \delta_{g_r g_s}} = F_r(v_1, \ldots, v_c), \qquad (20)$$

$$v_r = \sum_{\mathbf{d}} q_r(\mathbf{d}) \prod_{s=1}^{c} u_s^{d_s} = G_r(u_1, \ldots, u_c), \qquad (21)$$

or, in vector notation, $\mathbf{u} = \mathbf{F}(\mathbf{v})$ and $\mathbf{v} = \mathbf{G}(\mathbf{u})$. Being primarily concerned with $\mathbf{u}$, we can also eliminate $\mathbf{v}$ and write $\mathbf{u}$ as the solution of the fixed point equation

$$\mathbf{u} = \mathbf{F}(\mathbf{G}(\mathbf{u})). \qquad (22)$$

Then the probability that a vertex with role indices $d_1 \ldots d_c$ is not in the giant component is $\prod_r u_r^{d_r}$ with d_r distributed according to $p(\mathbf{d})$, so that the average probability is

$$\sum_{\mathbf{d}} p(\mathbf{d}) \prod_r u_r^{d_r} = G_0(\mathbf{u}), \qquad (23)$$

and the average probability that a vertex *is* in the giant component is one minus this quantity:

$$S = 1 - G_0(\mathbf{u}). \qquad (24)$$

Between them, Eqs. (22) and (24) give us a complete solution for the size of the giant component. As with the edge–triangle model the equations are often not solvable in closed form, but can be solved numerically by simple iteration starting from a suitable initial value for $\mathbf{u}$.

As a first example, consider again the edge–triangle model, for which there are two subgraphs, the edge and the triangle, with one role each and generating functions

$$F_1(z_1, z_2) = z_1, \qquad F_2(z_1, z_2) = z_2^2. \qquad (25)$$

The excess degree generating functions for the vertices are

$$G_1(z_1, z_2) = \frac{1}{\langle d_1\rangle} \sum_{\mathbf{d}} (d_1+1) p(d_1+1, d_2) z_1^{d_1} z_2^{d_2}, \qquad (26)$$

$$G_2(z_1, z_2) = \frac{1}{\langle d_2\rangle} \sum_{\mathbf{d}} (d_2+1) p(d_1, d_2+1) z_1^{d_1} z_2^{d_2}, \qquad (27)$$

and Eqs. (22) and (24) reduce to the two equations

$$u_1 = G_1(u_1, u_2), \qquad u_2 = \bigl[G_2(u_1, u_2)\bigr]^2, \tag{28}$$

which are identical to Eqs. (7) and (8). And the size of the giant component as a fraction of the size of the whole network is given by $S = 1 - G_0(u_1, u_2)$ as before.

As a more complicated example consider a network built from the subgraphs shown in Fig. 2b: single edges, triangles, and diamonds. Of the four roles let us label the ends of single edges role 1, the corners of the triangles role 2, and the two roles in the diamond roles 3 and 4 (which is which will not matter). Now consider the role distribution $p(\mathbf{d}) = p_1(d_1)p_2(d_2)p_{34}(d_3, d_4)$ where

$$p_1(d_1) = \mathrm{e}^{-c_1}\frac{c_1^{d_1}}{d_1!}, \qquad p_2(d_2) = \mathrm{e}^{-c_2}\frac{c_2^{d_2}}{d_2!}, \tag{29}$$

$$p_{34}(d_3, d_4) = (1-2a)\delta_{r_3,0}\delta_{r_4,0} + a[\delta_{r_3,0}\delta_{r_4,1} + \delta_{r_3,1}\delta_{r_4,0}]. \tag{30}$$

In other words, participation is Poisson distributed with means c_1 and c_2 for roles 1 and 2, and vertices participate either in one diamond with equal probability a of taking role 3 or 4, or in no diamonds with probability $1 - 2a$.

This particular distribution is chosen because it has a nontrivial but still relatively simple solution: after some work it can be shown that the size S of the giant component obeys

$$S = 1 - (1-2a)\mathrm{e}^{-c_1S - c_2S(2-S)} - 2a\mathrm{e}^{-4c_1S - 4c_2S(2-S)}. \tag{31}$$

We show the form of this solution as a function for a for one particular choice of parameters in Fig. 5.

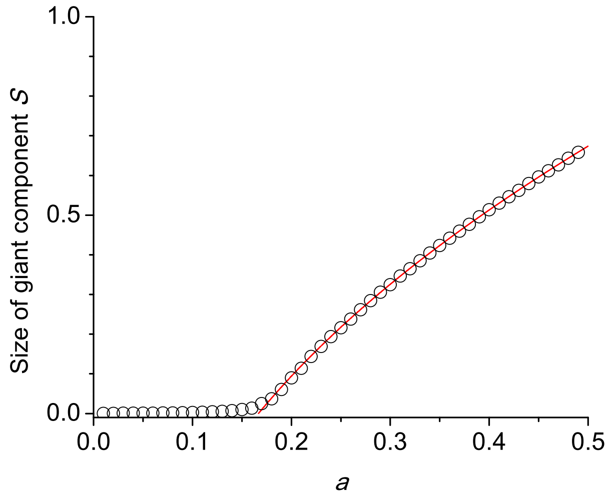


FIG. 5: The size S of the giant component in the network of edges, triangles, and diamonds described in the text, with the average role indices for edges and triangles fixed at $\langle d_1\rangle = \frac{1}{4}$ and $\langle d_2\rangle = \frac{1}{8}$, plotted as a function of the parameter a that controls the two role indices for the diamonds (see Eq. (30)). The solid line represents the analytic result and the circles are simulation results averaged over 100 networks with 10^5 vertices each.

C. Position of the phase transition

As with other random graph models, the fixed point equation, Eq. (22), has a trivial solution at $u_r = 1$ for all r corresponding to the state in which there is no giant component in the network and this fixed point undergoes a transcritical bifurcation at the point at which a giant component appears. We can locate the bifurcation, and hence the appearance of the giant component, by linear stability analysis of the fixed point. We write $u_r = 1 - \epsilon_r$ and expand to first order in the small quantities ϵ_r, which gives $\boldsymbol{\epsilon} = \mathbf{M}\boldsymbol{\epsilon}$ where $\mathbf{M}$ is the $c \times c$ matrix with elements

$$M_{rs} = \sum_t \bigl(n_t\delta_{g_rg_t} - \delta_{rt}\bigr)\frac{\partial G_t}{\partial z_s}\bigg|_{\mathbf{z}=\mathbf{1}}. \tag{32}$$

Using the definition of $G_r(\mathbf{z})$ given in Eq. (17) we can show that

$$\frac{\partial G_t}{\partial z_s}\bigg|_{\mathbf{z}=\mathbf{1}} = \frac{\langle d_s d_t\rangle}{\langle d_t\rangle} - \delta_{st}, \tag{33}$$

and hence that

$$M_{rs} = \delta_{rs} - \frac{\langle d_r d_s\rangle}{\langle d_r\rangle} - n_s\delta_{g_rg_s} + \sum_t \frac{\langle d_s d_t\rangle}{\langle d_t\rangle}n_t\delta_{g_rg_t}. \tag{34}$$

Physically the matrix element M_{rs} measures a branching ratio for the locally tree-like bipartite graph: a vertex that plays role r shares the relevant subgraph with some number of other vertices and M_{rs} measures the average number of times those other vertices collectively play role s in further subgraphs.

Now consider a set of randomly chosen vertices in a large network and suppose we grow that set repeatedly by adding to it all vertices with which its members share a subgraph. If we focus on the boundary of the set—meaning those vertices added on the most recent step—and represent the number of times the boundary vertices play roles 1 to c by a c-component vector, then the expected value of the vector is multiplied on each step by one factor of $\mathbf{M}$. If the sum of the vector elements grows we have a giant component and if it dwindles to zero, so that the set stops growing, then we do not, meaning that we have a giant component if and only if at least one eigenvalue of $\mathbf{M}$ is greater than one. If all eigenvalues are less than one then there is no giant component, and if one or more eigenvalues are exactly equal to one and none are greater then we are precisely at the phase transition point at which the giant component appears.

In general the eigenvalues of $\mathbf{M}$ are not trivial to find, but in some cases the problem simplifies. Consider, for instance, the case in which the role indices d_r are independent and Poisson distributed, in which case $\langle d_r d_s\rangle = \langle d_r\rangle\langle d_s\rangle + \langle d_r\rangle\delta_{rs}$ so that

$$M_{rs} = (N_r - 1)\langle d_s\rangle, \tag{35}$$

where $N_r = \sum_t n_t\delta_{g_rg_t}$ is the number of vertices in the subgraph in which role r appears. As an outer product of two vectors, this matrix is defective, having only a single eigenvector with corresponding eigenvalue $\lambda = \sum_r (N_r - 1)\langle d_r\rangle$. Thus in this network a giant component exists if and only if

$$\sum_r (N_r - 1)\langle d_r\rangle > 1. \tag{36}$$

In simple language, this equation says that a giant component exist if and only if the average number of vertices with which a random vertex shares a subgraph is greater than one.

Note that the degree k_r of role r within its subgraph trivially is never greater than $N_r - 1$ and hence when we are precisely at the transition point at which the giant component forms the average degree, Eq. (12), satisfies

$$\langle k\rangle = \sum_r k_r \langle d_r\rangle \le \sum_r (N_r - 1)\langle d_r\rangle = 1. \tag{37}$$

Recall that in the standard random graph of Erdős and Rényi the transition occurs precisely at $\langle k\rangle = 1$ and Eq. (37) thus tells us, in some sense, that the transition happens "earlier" in this subgraph model, or at least no later—in general a giant component will form when the average degree is less than one. The physical insight behind this observation is that belonging to a subgraph guarantees a vertex connections to all the other vertices in that subgraph, which vertices may be significantly greater in number than merely the immediate neighbors of the first vertex.

Another simple solvable case is that of a network composed of two types of subgraph, each with a single role. The edge–triangle model of Section II is a special case of such a model, but the general case is also solvable. The matrix $\mathbf{M}$ takes the form

$$\mathbf{M} = \begin{pmatrix} (n_1 - 1)\dfrac{\langle d_1^2\rangle - \langle d_1\rangle}{\langle d_1\rangle} & (n_1 - 1)\dfrac{\langle d_1 d_2\rangle}{\langle d_1\rangle} \\ \\ (n_2 - 1)\dfrac{\langle d_1 d_2\rangle}{\langle d_2\rangle} & (n_2 - 1)\dfrac{\langle d_2^2\rangle - \langle d_2\rangle}{\langle d_2\rangle} \end{pmatrix}, \tag{38}$$

and the largest eigenvalue is $\lambda = \frac{1}{2}(\tau + \sqrt{\tau^2 - 4\Delta})$ where τ and Δ are the trace and determinant of the matrix respectively. There is a giant component if and only if this eigenvalue is greater than one, or equivalently if $\sqrt{\tau^2 - 4\Delta} > 2 - \tau$. This condition is satisfied if either $\tau > 2$ or $\tau > \Delta + 1$. In terms of the matrix elements, these inequalities can be written

$$(n_1 - 1)\frac{\langle d_1^2\rangle - \langle d_1\rangle}{\langle d_1\rangle} + (n_2 - 1)\frac{\langle d_2^2\rangle - \langle d_2\rangle}{\langle d_2\rangle} > 2, \tag{39}$$

and

$$\frac{\langle d_1 d_2\rangle^2}{\langle d_1\rangle\langle d_2\rangle} - \left[\frac{\langle d_1^2\rangle}{\langle d_1\rangle} - \frac{n_1}{n_1 - 1}\right]\left[\frac{\langle d_2^2\rangle}{\langle d_2\rangle} - \frac{n_2}{n_2 - 1}\right] > 0. \tag{40}$$

Note that Eq. (39) does not depend on the correlation term $\langle d_1 d_2\rangle$ between the two role indices: in physical terms it says simply that there is a giant component if the densities of the two subgraphs independently are enough to create one. A giant component can, however, also be created by correlations between the placement of the subgraphs even when the overall density of subgraphs would otherwise be inadequate, and giant components of this kind are described by Eq. (40). Thus, for instance, if the vertices with many of one subgraph also have many of the other, then these high-degree vertices may join up to form a giant component even if there would be none were the same subgraphs allocated to different vertices.

In the examples we have examined, the size of the giant component and the position of the phase transition are determined solely by the role distribution $p(\mathbf{d})$ and the numbers n_r of roles in their respective subgraphs. The actual shape of the subgraphs themselves does not matter—once one vertex in a subgraph is in the giant component, the rest automatically are as well, regardless of patterns of connection within the subgraph. Thus, for instance, a network composed of a given distribution of cliques of given sizes would have a giant component of the same size as a network composed of the same distribution of loops of the same sizes. Not all network properties, however, show this behavior. In the next section we look at percolation on our networks, for which the shapes of the subgraphs matter greatly.

D. Percolation

Site and bond percolation have important implications for network resilience [24, 25] and the behavior of spreading processes on networks [26–28]. In site percolation, each vertex of a graph is present, functional, or "occupied" with independent probability ϕ_s. In bond percolation each edge is similarly occupied with independent probability ϕ_b.

It turns out that percolation on networks generated by our subgraph model can be treated using the same machinery developed above for calculating properties of the giant component. The only thing that needs to change is the generating function $F_r(z)$, Eq. (19), for the probability distribution of the number of vertices of each role that a vertex of role r can reach in its subgraph. In our previous treatment this distribution took a particularly simple form, since a vertex of role r could by definition reach all other vertices in its subgraph. Once we introduce percolation, however, some vertices in the subgraph may become unreachable, because there is no path of occupied vertices or edges along which to travel.

We will here consider the general case of simultaneous site and bond percolation—both vertices and edges may be occupied (or not) with probabilities ϕ_s and ϕ_b (or $1 - \phi_s$ and $1 - \phi_b$). Percolation on vertices or edges alone, i.e., conventional site or bond percolation, is the special case of this more general process when either ϕ_s

or ϕ_b equals one.

Let us denote by $p_r(\phi_s, \phi_b; \mathbf{d})$ the probability that an occupied vertex of role r in a subgraph is connected via occupied vertices and edges to $d_1 \dots d_c$ occupied vertices of roles $1 \dots c$ in the same subgraph, given the values ϕ_s and ϕ_b of the occupation probabilities. Then our generating function F_r is defined by

$$F_r(\phi_s, \phi_b; \mathbf{z}) = \sum_{\mathbf{d}} p_r(\phi_s, \phi_b; \mathbf{d}) \prod_{s=1}^{c} z_s^{d_s}. \qquad (41)$$

In terms of this function, the size of the giant percolation cluster is given by

$$\mathbf{u} = \mathbf{F}(\mathbf{G}(\mathbf{u})), \qquad S = \phi_s[1 - G_0(\mathbf{u})]. \qquad (42)$$

(See Eqs. (22) and (24) for the corresponding equations in our earlier calculation.) The additional factor of ϕ_s in the second equation accounts for the fact that a vertex can only be in the giant component if it is itself occupied. All we need now to complete the calculation is the form of F_r.

As an example, consider pure site percolation on a subgraph that takes the form of an m-clique—a set of m vertices with an edge between every pair. The vertices in a clique have only one role, and an occupied vertex i can reach another vertex j if and only if j is itself occupied. Thus the distribution p_r is binomial in this case:

$$p_r(\phi; d) = \binom{m-1}{d} \phi^d (1-\phi)^{m-1-d}, \qquad (43)$$

and

$$\begin{aligned} F_r(\phi; z) &= \sum_{d=0}^{m-1} \binom{m-1}{d} \phi^d (1-\phi)^{m-1-d} z^d \\ &= (1 - \phi + \phi z)^{m-1}, \end{aligned} \qquad (44)$$

where ϕ is the vertex occupation probability.

Suppose, for instance, that we have a network made entirely of cliques of various sizes m having one role each labeled with role labels $r = m$, and suppose the corresponding role indices d_m are independently Poisson distributed at each vertex. Then $G_m(\mathbf{d}) = G_0(\mathbf{d})$ for all m and

$$u_m = [1 - \phi + \phi G_0(\mathbf{u})]^{m-1}, \quad S = \phi[1 - G_0(\mathbf{u})]. \quad (45)$$

Thus in this case we have $u_m = (1-S)^{m-1}$ for all m and S satisfies the self-consistent equation

$$\begin{aligned} S &= \phi[1 - G_0(1-S, (1-S)^2, (1-S)^3, \dots)] \\ &= \phi[1 - H(1-S)], \end{aligned} \qquad (46)$$

where $H(z)$ is the generating function for the ordinary degree distribution, defined in Eq. (18). The percolation transition in this network corresponds to the point where the gradient of $\phi[1 - H(1-S)]$ at $S = 0$ is 1, i.e., the point

$$\phi_c = \frac{1}{H'(1)} = \frac{1}{\langle k \rangle}. \qquad (47)$$

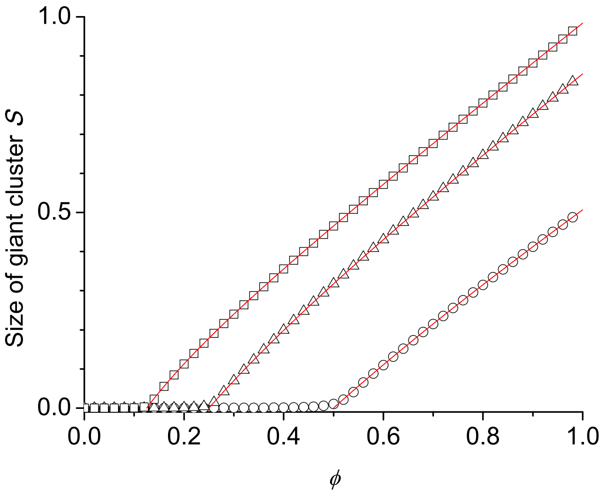


FIG. 6: The size S of the giant percolation cluster in the network of cliques described in the text, with cliques of size 2 to 5, as a function of the site percolation probability ϕ. The circles, triangles, and squares show simulation results averaged over 100 networks with 10^5 vertices each for $\langle k \rangle = 2$, 4, and 8 respectively, and the solid lines are the corresponding analytic results. If role $r = m$ is the role played by vertices in cliques of size m then the average role indices are $\langle d_m \rangle = \frac{1}{4}\langle k \rangle/(m-1)$ (which means that a randomly chosen edge is equally likely to belong to a clique of any size).

Note that this is the same condition as for the percolation point on an ordinary Poisson random graph, although the network is quite different. In Fig. 6 we show a plot of S from Eq. (46), along with the results of numerical simulations of finite sized networks in this class. As the figure shows, the agreement between simulation and theory is excellent.

The computation of F_r for general subgraphs is typically more involved than for the simple clique, since one must take into account possible paths between vertices and allow for different roles. Nonetheless, the computations are in principle merely a matter of counting the number of reachable vertices of each role for each possible configuration of occupied vertices or edges and multiplying by the probability of the configuration. We give in Table I the generating functions for all roles of all biconnected subgraphs of four vertices or fewer (plus the single edge), calculated by exhaustive enumeration in this fashion. Generating functions for singly connected subgraphs are simple composites of the biconnected ones.

If we only wish to determine the position of the percolation transition, the full subgraph generating functions are not necessary. Linearizing around the trivial fixed point in the equations for $\mathbf{u}$ we again derive an equation $\boldsymbol{\epsilon} = \mathbf{M}\boldsymbol{\epsilon}$, and the existence of a giant component depends on whether the matrix $\mathbf{M}$ has any eigenvalues

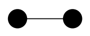
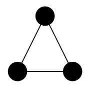
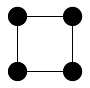
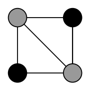
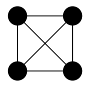
Subgraph	Role	Generating function F_r
	●	$1-\phi_s\phi_b+\phi_s\phi_b z$
	●	$(1-\phi_s\phi_b)^2+2\phi_s\phi_b(1-\phi_s\phi_b(2-\phi_b))z+\phi_s^2\phi_b^2(3-2\phi_b)z^2$
	●	$(1-\phi_s\phi_b)^2+2\phi_s\phi_b(1-\phi_s\phi_b)^2z+3\phi_s^2\phi_b^2(1-\phi_s\phi_b(2-\phi_b))z^2+\phi_s^3\phi_b^3(4-3\phi_b)z^3$
	1 ●	$(1-\phi_s\phi_b)^2+2\phi_s\phi_b(1-\phi_s\phi_b)(1-\phi_s\phi_b(2-\phi_b))z_2+\phi_s^2\phi_b^2(3-2\phi_b)(1-\phi_s\phi_b(2-\phi_b))z_2^2+2\phi_s^2\phi_b^2(1-3\phi_s\phi_b+3\phi_s\phi_b^2-\phi_s\phi_b^3)+\phi_s^3\phi_b^3(8-11\phi_b+4\phi_b^2)z_1z_2^2$
	2 ●	$(1-\phi_s\phi_b)^3+2\phi_s\phi_b(1-\phi_s\phi_b)(1-\phi_s\phi_b(2-\phi_b))z_1+\phi_s^2\phi_b^2(1-3\phi_s\phi_b+3\phi_s\phi_b^2-\phi_s\phi_b^3)z_1^2+\phi_s\phi_b(1-\phi_s\phi_b(2-\phi_b))^2z_2+2\phi_s^2\phi_b^2(3-2\phi_b)(1-\phi_s\phi_b(2-\phi_b))z_1z_2+\phi_s^3\phi_b^3(8-11\phi_b+4\phi_b^2)z_1^2z_2$
	●	$(1-\phi_s\phi_b)^3+3\phi_s\phi_b(1-\phi_s\phi_b(2-\phi_b))^2z+3\phi_s^2\phi_b^2(3-2\phi_b)(1-3\phi_s\phi_b+3\phi_s\phi_b^2-\phi_s\phi_b^3)z^2+\phi_s^3\phi_b^3(16-33\phi_b+24\phi_b^2-6\phi_b^3)z^3$

TABLE I: Subgraph generating functions F_r for all biconnected subgraphs of four vertices or fewer.

greater than one. In this case $\mathbf{M}$ is given by

$$M_{rs}=\sum_t\left[\frac{\partial F_r}{\partial z_t}\frac{\partial G_t}{\partial z_s}\right]_{\mathbf{z}=\mathbf{1}}. \tag{48}$$

The derivative $[\partial F_r/\partial z_t]_{\mathbf{z}=\mathbf{1}}$ is equal to the mean number of vertices of role t reachable from a vertex of role r within the appropriate subgraph. It is an open question whether there exists a method for calculating this mean number faster than the exhaustive enumeration of states used to calculate the generating function itself.

V. CONCLUSION

In this paper we have proposed and analyzed a random graph model that incorporates an arbitrary distribution of any chosen set of subgraphs or motifs, and hence mimics the properties of real-world networks, which are observed in many cases to contain certain subgraphs in significant numbers. The model is easily treated numerically and many of its properties can be calculated analytically by virtue of a mapping to a locally tree-like bipartite graph. In particular, we have given calculations of subgraph counts, the size of the giant component, the position of the transition at which the giant component appears, and percolation properties for site and bond percolation.

Useful though the model may be, however, it leaves open some important questions. We have not considered, for instance, how one should select the set of subgraphs to be used. If we wish to model a particular real-world network, then we would presumably want to look at the subgraphs that appear in that network and mimic their density and placement as accurately as possible in the model. But in that case, which subgraphs should be considered to occur sufficiently frequently as to require their inclusion in the model? And what should the distribution of the various subgraphs be? For biconnected subgraphs the density in the model network is, as we have shown, simply the density of the subgraphs we add explicitly, since those created by the combination of other subgraphs have density zero. For singly connected subgraphs, however, the density is more complicated, containing as it does not only the subgraphs we add ourselves but also those made from other subgraphs, and at present we do not have a good way to calculate it, making the matching of the real and model networks a challenge.

Some further generalizations of the model are possible. One could consider subgraphs in which the stubs are labeled, for instance with colors, so that even vertices in the same role could be distinguished. By specifying the colors of stubs connected to each vertex as well as the roles one could then induce additional kinds of structure, such as bipartite or k-partite structure or assortative mixing. One could also include role–role correlations, by analogy with the degree–degree correlations studied in ordinary random graphs.

The treatment given in this paper also leaves open some mathematical questions. In particular, it is unclear whether there is a quicker way to calculate the crucial subgraph generating functions of Section IV D than by exhaustive enumeration of states. For certain families of graphs, such as cliques, we have shown that it is possible to characterize the generating functions analytically, but it seems unlikely that this will be possible in more general cases. It is possible, however, that one could find a numerical algorithm for calculating the coefficients of the generating functions more quickly than the current method, which is exponentially slow.

Acknowledgments

The authors thank Lada Adamic, James Gleeson, and Lenka Zdeborova for useful comments. This work was funded in part by the National Science Foundation under grant DMS–0804778 and by the James S. McDonnell Foundation.

[1] A. Rapoport, Bulletin of Mathematical Biophysics **10**, 145 (1948).
[2] D. J. Watts and S. H. Strogatz, Nature **393**, 440 (1998).
[3] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, Science **298**, 824 (2002).
[4] R. Milo, S. Iztkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. AyzenShtat, M. Sheffer, and U. Alon, Science **303**, 1538 (2004).
[5] G. Bianconi, G. Caldarelli, and A. Capocci, Phys. Rev. E **71**, 066116 (2005).
[6] M. Molloy and B. Reed, Random Structures and Algorithms **6**, 161 (1995).
[7] M. E. J. Newman, S. H. Strogatz, and D. J. Watts, Phys. Rev. E **64**, 026118 (2001).
[8] R. Pastor-Satorras, A. Vázquez, and A. Vespignani, Phys. Rev. Lett. **87**, 258701 (2001).
[9] M. E. J. Newman, Phys. Rev. Lett. **89**, 208701 (2002).
[10] A. Clauset, C. Moore, and M. E. J. Newman, Nature **453**, 98 (2008).
[11] B. Karrer and M. E. J. Newman, Phys. Rev. Lett. **102** (2009).
[12] M. Penrose, *Random Geometric Graphs* (Oxford University Press, Oxford, 2003).
[13] M. E. J. Newman, Phys. Rev. E **68**, 026121 (2003).
[14] X. Shi, L. Adamic, and M. J. Strauss, Physica A **378**, 33 (2007).
[15] B. Bollobás, S. Janson, and O. Riordan, Preprint arXiv:0807.2040v1 (2008).
[16] M. E. J. Newman, Phys. Rev. Lett. **103**, 058701 (2009).
[17] J. P. Gleeson, Phys. Rev. E **80**, 036107 (2009).
[18] J. P. Gleeson and S. Melnik, Phys. Rev. E **80**, 046121 (2009).
[19] J. C. Miller, Phys. Rev. E **80**, 020901 (2009).
[20] P. Erdős and A. Rényi, Publicationes Mathematicae **6**, 290 (1959).
[21] P. Erdős and A. Rényi, Publications of the Mathematical Institute of the Hungarian Academy of Sciences **5**, 17 (1960).
[22] An exception is the geometric random graph [12], in which clustering occurs naturally on account of the graph's being embedded in a Euclidean space of low dimension (usually two). Geometric graphs are, however, something of a special case, most real networks having no such low-dimension embedding.
[23] One needs an additional constraint to avoid pathological graphs with two (or even more) giant components. Consider an ensemble with two subgraphs, a cycle and a clique, and assume that the probability distribution is such that a vertex participates either in cycles or in cliques, but not both. Then all graphs consist of a cycle part and a clique part that are disconnected from each other, so there could be two giant components, one composed of cycles and the other composed of cliques. To avoid this pathology, we impose the restriction that one should not be able to divide the roles into two sets such that every vertex and every subgraph only has roles from one of the sets.
[24] R. Cohen, K. Erez, D. ben-Avraham, and S. Havlin, Phys. Rev. Lett. **85**, 4626 (2000).
[25] D. S. Callaway, M. E. J. Newman, S. H. Strogatz, and D. J. Watts, Phys. Rev. Lett. **85**, 5468 (2000).
[26] D. Mollison, Journal of the Royal Statistical Society B **39**, 283 (1977).
[27] P. Grassberger, Math. Biosci. **63**, 157 (1982).
[28] M. E. J. Newman, Phys. Rev. E **66**, 016128 (2002).