nature physics

ARTICLES

https://doi.org/10.1038/s41567-021-01417-7

Degree-preserving network growth

Shubha R. Kharel[1,2], Tamás R. Mezei[3], Sukhwan Chung[1,2], Péter L. Erdős[3] and Zoltan Toroczkai[1,2]✉

Real-world networks evolve over time through the addition or removal of nodes and edges. In current network-evolution models, the degree of each node varies or grows arbitrarily, yet there are many networks for which a different description is required. In some networks, node degree saturates, such as the number of active contacts of a person, and in some it is fixed, such as the valence of an atom in a molecule. Here we introduce a family of network growth processes that preserve node degree, resulting in structures substantially different from those reported previously. We demonstrate that, despite it being an NP (non-deterministic polynomial time)-hard problem in general, the exact structure of most real-world networks can be generated from degree-preserving growth. We show that this process can create scale-free networks with arbitrary exponents, however, without preferential attachment. We present applications to epidemics control via network immunization, to viral marketing, to knowledge dissemination and to the design of molecular isomers with desired properties.

Many network growth models have been introduced in the literature, with possibly the preferential attachment model by Barabási and Albert[1] being the most popular. In most of these models, when the incoming node joins the existing network, it also increases (including in the Barabási and Albert model) the degrees of the nodes to which it connects. However, there are networks in which the nodes cannot accept additional connections because their connectivity is saturated or even fixed. For example, in a chemical complex, a node represents a chemical species and an edge represents a covalent bond, and thus node degree is the number of bonds a species can form, which is given by its electron configuration and is fixed. Through reactions, the complexes grow in size, but the degrees of their atoms stay constant in the process. In physical networks, creating and maintaining connections bears a cost, leading to degree saturation, and the only way a new node can join such nodes is if some of them lose their existing connections and form a new connection with the incoming node. In social networks, for example, this is known as tie dissolution and formation[2,3]. One may, therefore, consider growth processes that do not force degree increases on nodes (such as in the Barabási and Albert model) but instead respect the existing degrees over time. Although there are many variations, here we consider the simplest of such processes, namely those that preserve the existing degrees. We call this 'degree-preserving growth' (DPG) and the created graph structures as DPG graphs. When nodes represent physical entities, they will have limited connections and possibly even a preferred degree, with a distribution that decays on both sides of this degree. The DPG formulation assumes precisely constant degrees; however, this helps with mathematical tractability and understanding. Despite its simplicity, DPG-generated networks have rich mathematical properties and a wide range of potential applications.

Definition of the DPG process

Here we work with graphs (called simple) that have no multiple edges between any pair of nodes and no self-loops. The basic DPG mechanism is that, at every step, a new node w joins a graph G by first cutting and removing k of its edges, then connecting w to the $2k$ nodes of the removed edges, thus preserving their degrees; G gains a new node of degree $d_w = 2k$. One could also say that the incoming node w, shown as a node with $2k$ 'stubs' in Fig. 1, is joined to the stubs resulting from the cut edges. The only constraint is that the cut edges must be independent (no common node), otherwise the resulting graph would not be simple. Figure 1b shows two more DPG scenarios. In the upper scenario, $d_w = 4$ and we show two possible outcomes corresponding to cutting either the blue edges or the green ones. In the lower scenario, w ($d_w = 6$) joins by cutting the red edges, which is the only possibility. The above implies even degrees for all incoming nodes; however, it can be generalized to arbitrary degrees by allowing a stub in the graph (an 'unsatisfied bond'), to be picked up in later steps (see Supplementary Information section III.B for the general algorithm). Note that non-edges stay non-edges at all times during the process, and thus DPG cannot build graphs that are too dense, including complete graphs.

A set of independent edges (selected at every step in DPG) is called a matching. Matchings[4] have applications in many areas including computer science, computational chemistry[5] and network control[6]. A matching is maximal if no additional edges from the graph can be added to it. A maximal matching of the largest size is a maximum matching, and its size $\nu(G)$ is the graph's matching number. Clearly, $1 \leq \nu \leq \frac{1}{2}n$ for any G with at least one edge, where n is the number of nodes in G. There can be several maximal or maximum matchings in a graph (Fig. 1b). If $\nu = \lfloor \frac{n}{2} \rfloor$, then G has a perfect/near-perfect matching for even/odd n (the red set in Fig. 1b is perfect). Note that not all graphs have perfect/near-perfect matchings. In DPG, ν limits the largest degree that can be added in a step: $d \leq 2\nu + 1$ (Supplementary Information section I.A).

In trade or political networks, the DPG can be interpreted as 'middleman' dynamics, when a new agent inserts itself between (trading) partners and overtakes their flow. Middleman positions are advantageous, allowing flow control by either facilitating cooperation or exploiting differences among the connected nodes. If two nodes u and v have been trading a commodity (an edge), a new node w may offer incentives to both for trading with it, thus overtaking this trade (or edge). Related to Burt's structural holes theory[7,8], this was studied from a game-theoretic point of view by Kleinberg et al.[9]. The 'middleman' structure is also present in biology and chemical biology, especially in vivo (see, for example, ref. [10]).

[1]Department of Physics, University of Notre Dame, Notre Dame, IN, USA. [2]Center for Network and Data Science, University of Notre Dame, Notre Dame, IN, USA. [3]Alfréd Rényi Institute of Mathematics, Eötvös Loránd Research Network, Budapest, Hungary. ✉e-mail: toro@nd.edu

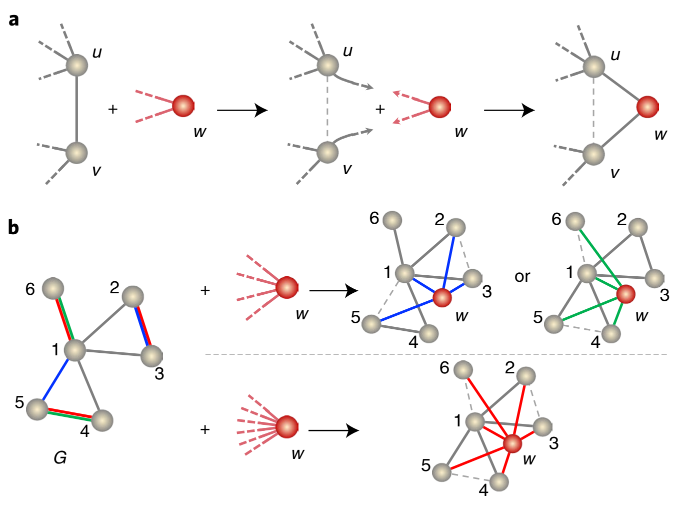

Fig. 1 | Degree-preserving growth. a, In the simplest case, a new vertex w joins the graph by removing some edge (u, v) and then connecting to its nodes. The thin dashed line indicates the non-edge between u and v. **b**, Two more scenarios, one with an incoming degree of $d=4$ with two outcomes (above the horizontal dashed line), and another with an incoming degree of $d=6$ (below). The pair of blue edges, the pair of green edges and the three red ones in G each form a matching. The blue set is maximal but the green is not; adding edge $(2, 3)$ yields the red matching, which is also a maximum (and a perfect) matching.

Due to the freedom in choosing the incoming degrees and the matchings, DPG defines a large family of network models, which makes it well suited for network modelling; for example, it can produce scale-free networks (without preferential attachment) or be used as a degree-constrained method for maximizing graph statistics with applications to epidemics, marketing, knowledge dissemination and chemistry.

Real-world networks as realizations of DPG processes

Here we ask the question: given a real-world network G on n nodes, is there a DPG sequence that can form G, starting from a small graph G_f on $n_f = \mathcal{O}(1)$ nodes, and if so, how easy is it to find such a sequence? To answer this question, we employ the inverse of the DPG process (the exact reverse of the steps in Fig. 1). A single degree-preserving (DP) reduction step consists in cutting all the links to some node w, removing w and then joining the stubs of the cut links remaining in the graph between themselves, avoiding multiple edges (see Supplementary Information section III.C for the general case). Clearly, for a node w to be DP-removable, there must be a sufficient number of non-links between the neighbours of w. One can show (Supplementary Information section II.A) that G is DP-reducible by one step if and only if the graph of non-edges of the neighbours of w has a perfect/near-perfect matching. It is known that deciding whether a graph has a perfect matching takes polynomially many steps in n (it is said to be in class P), due to Edmonds' blossom algorithm[11]. Thus, deciding whether G is DP-reducible by one node is in class P and so is reducibility by k nodes, where k is a finite constant. We can prove, however[12], that if $k > n^{\varepsilon}$ with $\varepsilon > 0$, then DP reduction is an NP (non-deterministic polynomial time)-complete problem (only exponential-time algorithms are known) and finding the largest possible k for a given graph is NP-hard.

A simple algorithm allows us to test how well a network can be reduced by inverse DPG: pick a node uniformly at random and remove if allowed, then repeat. Once no more nodes are DP-removable, the algorithm stops, giving a final graph G_f. As there is randomness in both the choice of the nodes and in the way the stubs are linked after removals, the algorithm generates one stochastic backward path out of many possible ones. Note that due to reversibility, this implies a forward DPG path from G_f to G (with $n_s = n$ nodes). We record the reduction fraction $\phi = n_f/n_s$, corresponding to a stochastic backward path. Repeating the process from G we may arrive at another G_f value and ϕ value. If $n_f = \mathcal{O}(1)$ and thus $\phi = \mathcal{O}(n^{-1})$, we say that G is a DPG-feasible network. Figure 2 shows ϕ values obtained from 50 repeated runs on 36 randomly picked real-world networks from a wide range of domains (Supplementary Table 2). We see that despite the deconstruction process being NP-hard, most networks are actually DPG-feasible, that is, there is a forward DPG process that can generate them, even though it is not necessarily how they actually formed. Note the small variability in ϕ for most networks, which shows that finding a deep DP deconstruction path is easy and that, when tested, many paths result in identical or similar G_f graphs (Supplementary Figs. 5–8). Of the least decomposable networks, three (1, 3 and 4) have the highest densities, which agrees with the earlier observation that DPG cannot construct high-density networks. However, in this case, it is easy to decide that these networks are not DPG-feasible. Note that other work[12] shows NP-hardness of DP deconstruction using sparse networks, and thus sparsity is not a good criterion for DPG feasibility. A point in case here is the sparse disease–gene association network DiseaseGene ($\rho = 4.8 \times 10^{-3}$, $\langle \phi \rangle = 0.53$). However, most of the networks tested here seem DPG-feasible, implying that they share specific properties that make them so.

DPG models

To develop an understanding of DPG networks, we present several models based on different rules for the incoming degree sequence and matching selection. As it does not change the conclusions but is simpler to deal with analytically, here we focus on even incoming degrees. We first establish the simple bound (valid for any graph G)

$$\nu(G) \geq \frac{m(G)}{\Delta(G) + 1}, \tag{1}$$

for the matching number of G, where $m(G)$ is its number of edges and $\Delta(G)$ is its maximum degree. See Supplementary Information section II.B for proof and for similar bounds[13,14].

It is known that large homogeneous random graphs have a large ν and, under mild conditions, even perfect matching[15]. We call ν 'large' if $\nu(G) \propto N$, where N is the number of nodes of G. If G_n denotes the DPG graph after n steps, its number of nodes is $N_n = N_{n-1} + 1 = N_0 + n$, and thus it has large matching if $\nu(G_n) \equiv \nu_n \propto n$ for $n \gg N_0$. The time step i when a node joins the graph indexes that node (i) and its degree (d_i). The edge count of G_n is $m_n = m_{n-1} + \frac{1}{2} d_n = m_0 + \frac{1}{2} \sum_{i=1}^{n} d_i$. We write $\Delta_n \equiv \Delta(G_n) = \max(\Delta_{n-1}, d_n) = \max\limits_{i \leq n}(\Delta_0, d_i)$. Note that ν_n can grow at most by unity in a step, that is, $\nu_{n+1} \leq \nu_n + 1$, but it can drop by large amounts (Supplementary Fig. 1), which also shows that the evolution of ν_n depends on both the graph structure and the incoming degrees. All DPG algorithms and pseudocodes are presented in Supplementary Information section III.

The simplest case is bounded-degree DPG dynamics, that is, $d_n \leq \Delta = \text{constant}$ for all $n \geq 0$. As $d_n \geq 2$, $m_n \geq m_0 + n \propto n$ for $n \gg 1$. From equation (1), $\nu_n \geq (\Delta + 1)^{-1}(m_0 + n) \propto n$, showing that these graphs have large matching numbers, asymptotically. Choosing $d_n = \Delta$ for all $n \geq 0$, we obtain a dynamic model for random regular graphs. Supplementary Fig. 9 shows that random regular DPG graphs have asymptotically perfect/near-perfect matching. Four-regular random graphs are used as client-server architectures in peer-to-peer networking called SWAN technology[16], using a special case of the DPG process, which results in reliable and efficient TCP/IP (transmission control protocol/internet protocol) fabrics of connections[17]. DPG have also been used in two-dimensional vortex

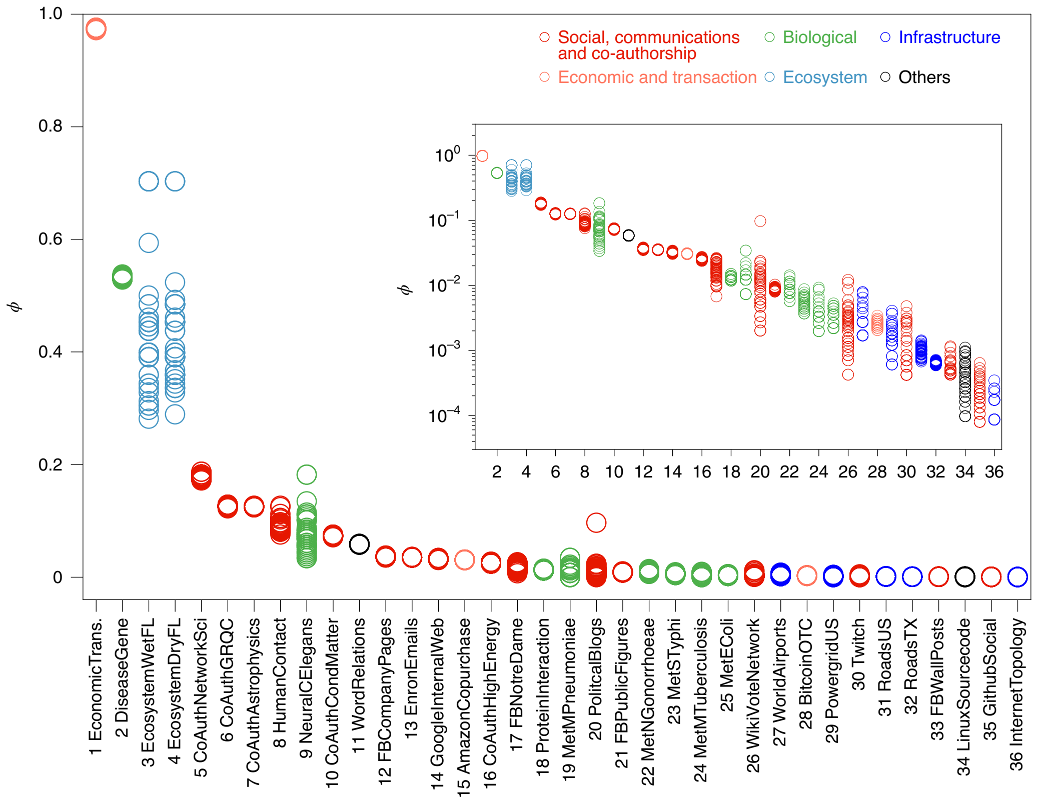

Fig. 2 | Real-world networks from DPG process. Fraction $\phi = n_f/n_s$ of the number of nodes n_f in the final network to those of the starting network ($n_s = n$), after DPG deconstruction. For every network, 50 runs were performed and for each, the final fraction ϕ is plotted. Real-world networks were chosen from social, communications and co-authorship networks and five other categories as indicated in the figure legend. Networks are ordered by decreasing value of $\langle\phi\rangle$ (over the 50 runs). The inset shows the same on a log–linear scale. The largest network (32 RoadsTX) has $n = 1{,}379{,}917$ nodes and $m = 1{,}921{,}660$ edges, for a density of $\rho = 2m/n(n-1) = 2.02\times10^{-6}$. Networks 32-36, namely InternetTopology ($\rho = 1.8\times10^{-4}$, $\langle\phi\rangle = 1.2\times10^{-4}$), GithubSocial ($\rho = 4.1\times10^{-4}$, $\langle\phi\rangle = 2.2\times10^{-4}$), LinuxSourcecode ($\rho = 4.5\times10^{-4}$, $\langle\phi\rangle = 3.7\times10^{-4}$), FBWallPosts ($\rho = 1.7\times10^{-4}$, $\langle\phi\rangle = 5.1\times10^{-4}$) and RoadsTX ($\rho = 2.02\times10^{-6}$, $\langle\phi\rangle = 6.4\times10^{-4}$) are the most decomposable, whereas networks 1-5 with the highest $\langle\phi\rangle$ ($\langle\phi\rangle > 0.1$) are the least DPG-decomposable. Of the latter, three have the highest densities: EconomicTrans ($\rho = 0.34$, $\langle\phi\rangle = 0.97$), EcosystemWetFl ($\rho = 0.44$, $\langle\phi\rangle = 0.44$) and EcosystemDryFl ($\rho = 0.43$, $\langle\phi\rangle = 0.43$).

liquids analysis[18] and as proof techniques[19,20]. Three-regular graphs formed by complex but degree-preserving rules are used in the design of self-reproducing graph automata[21].

Linear DPG. Here we consider the case in which the incoming vertex's degree is proportional to the matching number:

$$d_n = 2\lceil c\nu_{n-1}\rceil\,,\quad 0 < c \le 1\,, \tag{2}$$

where c is a constant. We call this model Linear DPG (or LinDPG) because, as we show, the degrees grow linearly with n and thus the degree distribution is uniform (all degrees are equally likely). Figure 3a–c shows three sample graphs. Figure 3d shows that, asymptotically, LinDPG graphs have almost perfect matching for any c, which implies from equation (2) that the degree sequence is linear, with slope c.

To characterize these networks, we study the $c = 1$ limit, called the Maximum DPG (MaxDPG) model. Here, at every step, we join the new node with as many edges as the graph can take, while keeping the graphs simple. Thus, for MaxDPG, $d_n = 2\nu(G_{n-1})$, $n \ge 1$. However, in this case we need a stronger bound than that of equation (1). In Supplementary Information section II.C we prove that for large enough n,

$$\nu_n \ge \frac{1}{2}n - \log_2 n + \mathcal{O}(1)\,, \tag{3}$$

that is, the matching number is asymptotically equal to that of perfect matching. From equation (2) (with $c = 1$) and equation (3),

$$k - 2\log_2 k + \mathcal{O}(1) \le d_{k+1} \le k\,,\quad 1 \le k \le n-1\,, \tag{4}$$

with $d_1 = 2\nu_0$. Thus, $\frac{1}{4}n(n-1) - n\log_2 n + \mathcal{O}(n) \le m_n \le \frac{1}{4}n(n-1) + \mathcal{O}(1)$, showing that MaxDPG graphs are dense, with density $\rho_n = \frac{1}{2} - \mathcal{O}\left(\frac{\ln n}{n}\right)$; due to their maximal nature, no densities higher than 1/2 can be reached by any DPG process as $n \to \infty$.

To better understand the structure of MaxDPG graphs, we call on the notion of split graphs. A graph G with node set V is called a split graph[22,23] if its nodes can be partitioned into a set S and a set $V - S$ such that the S nodes form a complete graph $K_{|S|}$ in G but those in $V - S$ have no connections between them (an independent set), except to S. Thus, split graphs have a maximal core (S) and periphery ($V - S$) structure. By sorting a degree sequence $\mathbf{d}$ non-increasingly as $d_1 \ge \ldots \ge d_n$, one defines the pivot index[24] as $s(\mathbf{d}) = \max\{i : d_i \ge i\}$ (so $d_{s+1} < s + 1$). (In ref. [23] the slightly different 'pivot' index $m(\mathbf{d}) = \max\{i : d_i \ge i - 1\}$ is defined; however, s is a better choice as shown in ref. [24], slightly strengthening the results of ref. [23].) By analogy, this is the scientometric Hirsch, or h-index, if d_i is the number of citations of the ith-most cited paper. In Supplementary Information section I.C we show (with s as the pivot index of $\mathbf{d}$) that

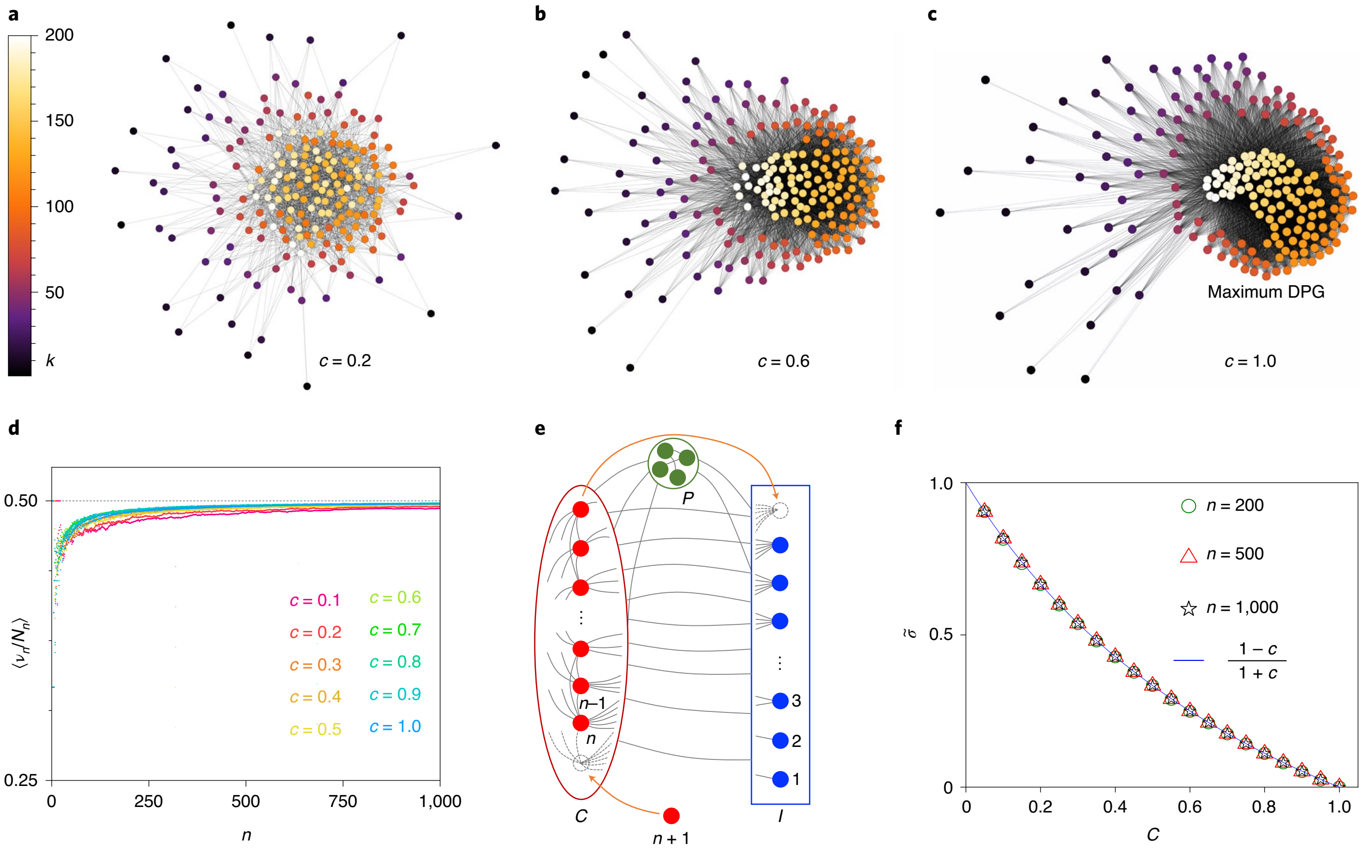

Fig. 3 | DPG models. a–**c**, Networks grown from a triangle in 200 steps of the LinDPG process with $c = 0.2$ (**a**), 0.6 (**b**) and 1.0 (**c**). Nodes are coloured by their index, that is, by their join time k, with darker colours indicating old nodes (small k). Networks are drawn using spring-electrical embedding force-directed layout. A MaxDPG network is shown in **c** with recently added nodes (larger k) forming a complete subgraph (core), whereas the older nodes (smaller k) form an independent set (periphery); splitness $\tilde\sigma = 2\times 10^{-4}$. **d**, Linear DPG networks have perfect/near-perfect matching, with $\langle \nu_n/N_n \rangle \to 0.5$ asymptotically and independently of c. Averages are taken over 50 runs. **e**, Steady-state dynamics of MaxDPG graphs. A newly incoming node $n+1$ joins the complete subgraph K_C and also connects to most nodes in the independent set I (see Methods for explanation). **f**, Decay of splitness $\tilde\sigma$ with c for LinDPG and its theoretic expression (continuous curve).

$$\tilde\sigma(G) = \frac{s(s-1) - \sum_{i=1}^{s} d_i + \sum_{i=s+1}^{n} d_i}{\sum_{i=1}^{n} d_i} \in [0,1)\,, \qquad (5)$$

is the fraction of edges to be added or removed to make a graph into a split graph[23], thus serving as a 'splitness' measure. $\tilde\sigma$ can also be regarded as a core-periphery measure with $\tilde\sigma = 0$ corresponding to maximal core periphery. In Fig. 3e and Methods, we show that MaxDPG graphs are near split-graphs and describe their steady-state dynamics.

Returning to the general case of $c \le 1$ and $d_k \simeq c(n-k)$, the pivot is simply $s = cn/(c+1)$ and thus $\lim_{n\to\infty} \tilde\sigma(G_n) = (1-c)/(1+c)$ (see Fig. 3f for agreement with simulations). Thus, to obtain graphs with a given core-periphery strength $\tilde\sigma$ we may set $c = (1-\tilde\sigma)/(1+\tilde\sigma)$.

Another analytically tractable model is 'Random Fraction dpg', with $d_n = 2\lceil r_n \nu_{n-1} \rceil$, where $r_n \in (0,1]$ is a uniformly distributed random variable (Supplementary Information section II.D). This model also shows linear scaling, $\nu_n \simeq an$, where a is a constant, but has a logarithmically decaying degree distribution.

Scale-free DPG. The following simple algorithm produces scale-free networks:

(1) Find a maximum matching M of G_{n-1}, $\nu_{n-1} = |M|$ and let $\mathcal{A} \equiv \{1, \ldots, \nu_{n-1}\}$.
(2) Generate the probabilities $p_i = i^{-\gamma}/c$, $i \in \mathcal{A}$, where $c = \sum_{k\in\mathcal{A}} k^{-\gamma}$.
(3) Sample an integer $m \in \mathcal{A}$ with probability p_m.
(4) Select uniformly at random a subset of m edges from M.
(5) Perform a DPG step on these edges, joining a new vertex of degree $d_n = 2m$. To set the minimum degree $d_{\min}$, all integers less than $d_{\min}/2$ must be excluded from $\mathcal{A}$.

Figure 4a shows the degree distributions for various γ values, and Supplementary Fig. 10 shows their matching fractions $\langle \nu_n/N_n \rangle$ versus n. Supplementary Fig. 11 shows a scale-free DPG graph (even degrees) for $n = 10^4$, $d_{\min} = 4$ and $\gamma = 2.5$ with its magnified centre presented in Fig. 4b, showing the hubs. The scale-free network in Fig. 4c was generated with the generalized algorithm (arbitrary parity degrees).

As every node in G_{n-1} contributes with at most one edge to a matching, independently of its degree, edge formation is not based on degree preference. This contrasts with the Barabási and Albert[1], configuration[25,26] and Chung–Lu[27] models, where the probability of an edge is proportional to the product of the degrees of their nodes. Figure 4d shows that the node participation ratio does not scale with node degree except at small, finite degrees. Although there is no degree preference in a single step, the networks are degree-correlated. Note that ours, like the Barabási and Albert model, is a growth algorithm (unlike the configuration and Chung–Lu models), but is not an endogenous scale-free network generator

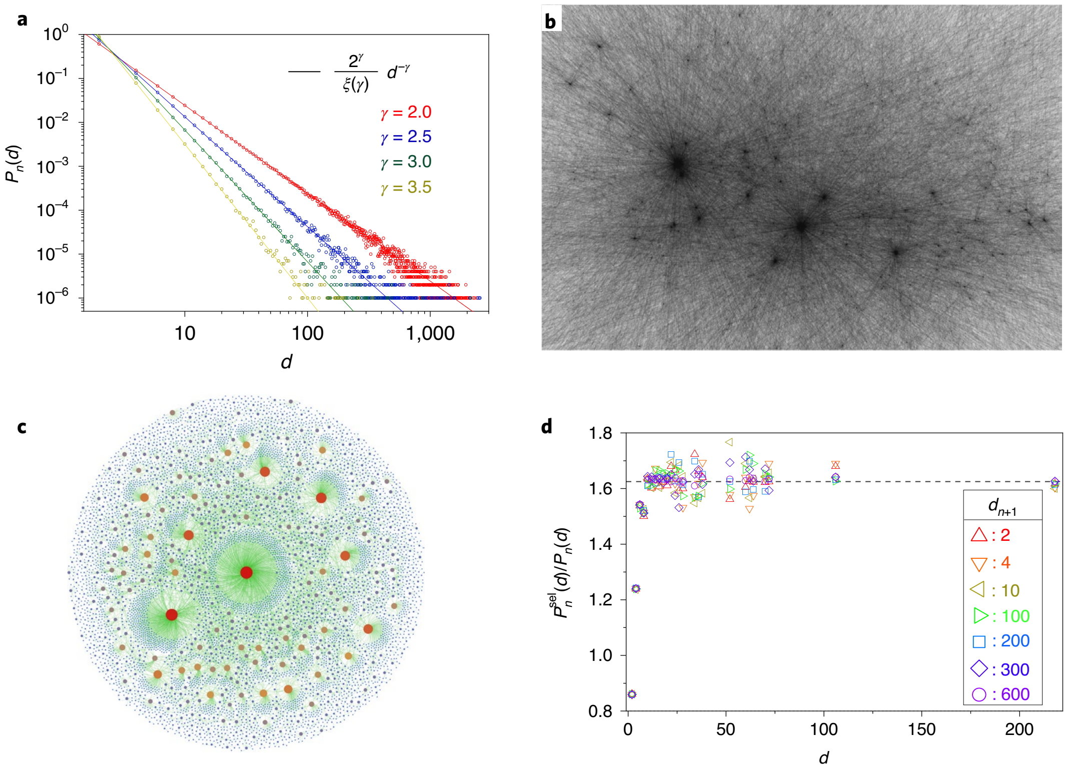

Fig. 4 | Scale-free DPG networks. a, Degree distributions for $\gamma = 2.0$, 2.5, 3.0 and 3.5, after $n = 10^4$ DPG steps. **b**, A part of a scale-free DPG network (even degrees) for $n = 10^4$, $\gamma = 2.5$ and $d_{\min} = 4$ (full graph is in Supplementary Fig. 11). **c**, A connected, scale-free network obtained with the general algorithm (arbitrary parity degrees), $\gamma = 2.5$ and $n = 7{,}035$ nodes, started from a single-edge ForceAtlas2 layout[43]. **d**, Node participation ratio $P_n^{\mathrm{sel}}(d)/P_n(d)$ versus its degree d, in a matching of size $d_{n+1}/2$, in a scale-free DPG graph with $n = 10^3$ nodes, for several incoming node degrees d_{n+1}. $P_n^{\mathrm{sel}}(d)$ is the probability of a node of degree d being selected into the matching and $P_n(d)$ is the probability of a node of degree d. For every d_{n+1}, the matchings were drawn $6 \times 10^5/d_{n+1}$ times.

(unlike the Barabási and Albert model); it is more similar to the configuration and Chung–Lu models in that respect.

Network design through DPG and applications

In network modelling, we often create graphs with a given degree sequence[28]. This can also be done via DPG, for example by either joining degrees increasingly (iconf-DPG) or greedily (gconf-DPG) (Methods and Fig. 5a).

The freedom in the choice of matchings allows the tuning of graph properties beyond degrees, for example degree correlations[29]. Given a graph, $R_\alpha = \sum_{i\sim j}(d_i d_j)^\alpha$ (sum over edges) is called its general Randić index[30,31]. R_1 is the second Zagreb index, used in chemistry as a topological descriptor for molecular graphs[32] and is related to the graph's assortativity coefficient r (ref. [33]) via $r = 1 - C^{-1}\ (S_3 - 2R_1)$ where $C = S_3 - (S_2)^2/S_1$ and $S_k = \sum_{i=1}^n (d_i)^k$ (ref. [34]). It was shown that r (or R_1) is strongly related to the graph's spectral radius λ_1 (the largest eigenvalue of its adjacency matrix)[34–36] and that minimizing λ_1 increases the epidemic threshold, thus curbing disease spread[37,38], whereas maximizing λ_1 aids in viral marketing or in knowledge spread in research and development collaboration networks[39]. $R_{-1/2}$ is the original Randić index applied in chemistry and pharmacology to study structure–property or structure–activity relations[30,31]. In particular, $R_{-1/2}$ closely tracks the boiling points of alkanes (Fig. 1 of ref. [31]), formation enthalpies and chromatographic retention times. $R_{-1/2}$ is also related to the normalized Laplacian matrix[40].

In terms of algorithms, edge rewiring is currently used to tune R_1 (ref. [29]), which can be costly. Here we propose configurational DPG with R_α tuning as an alternative. In tuning, we assign weights to the edges, then select a matching such that the total weight in the matching is within a desired range, or the largest or smallest possible (Supplementary Information section III.D); Fig. 5b,c shows two DPG graphs (same degree sequence) with extremal degree assortativity coefficients. The relationship between r and λ_1 (ref. [34]) allows the tuning of λ_1 (Fig. 5d). As a chemistry application, this can be used, for example, to efficiently generate specific molecular structures such as alkane isomers with desired boiling points. Moreover, the DPG algorithm also samples pathways towards these end-structures, informing the chemical synthesis of these substances.

Finally, we discuss an alternative route to epidemics control using the DPG process. Because the DPG reroutes connections between old nodes, we can use it to disrupt the transmission routes of a pathogen (a biological or computer virus) by adding immune DPG nodes, which still allow transmission pathways for other quantities. Let G_0 be the virus-transmitting network. We ask how the smallest number of immune DPG nodes should be added such that viral spread is maximally disrupted. To quantify disruption, we monitor the fraction of old nodes q in the largest cluster, still connected through virus-transmitting edges, after the addition of $N - N_0 \equiv xN_0$ DPG nodes. The DPG node degrees follow the degree distribution of G_0. For matchings (Supplementary Table 1) we choose from the set of old edges with the highest betweenness centrality within that set. Figure 5e shows in different colours the components disconnected by this algorithm for a human contact network: all paths between components pass through the immune nodes (grey). We next consider two-dimensional random geometric graphs[41], which are spatial networks without shortcuts and Erdős–Rényi random graphs (non-spatial with shortcuts). Figure 5f shows a 'shattering' phase transition in both cases as fraction x is increased: above the critical x_c, the disease-transmitting network breaks into many small pieces. Interestingly, Erdős–Rényi networks have a fairly large x_c of 0.205, when compared to 0.011 for random geometric graphs. Thus, DPG immunization of spatial networks is much more efficient than

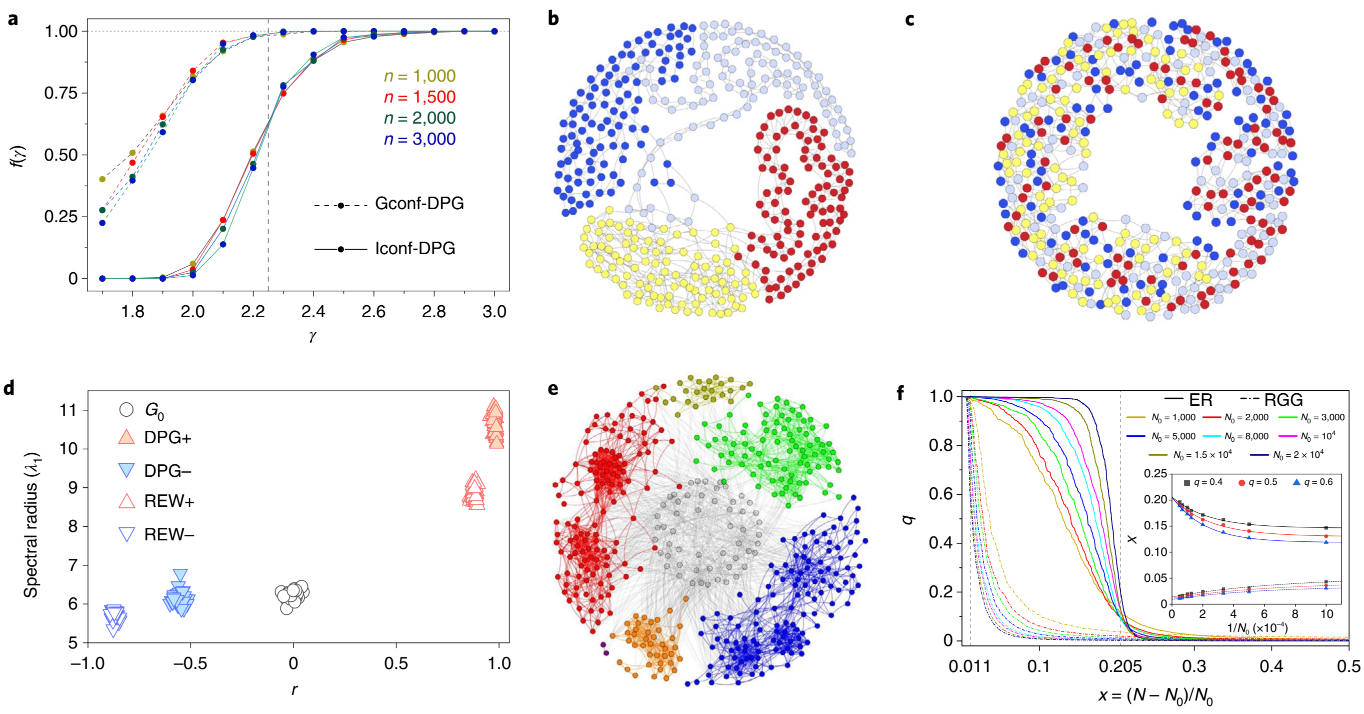

Fig. 5 | DPG in network design. a, Fraction of successful configurational DPG runs for scale-free degree distributions using gconf-DPG and iconf-DPG (Methods). **b**,**c**, Configurational DPG graphs of the same degree sequence with extremal assortativity coefficients 0.99 (**b**) and −0.71 (**c**), respectively. The degree sequence is $d_1 = 3$ (×100, blue), $d_2 = 4$ (×100, light blue), $d_3 = 5$ (×100, yellow) and $d_4 = 6$ (×100, red). **d**, Spectral radius λ_1 as function of assortativity r when tuning r with the rewiring method (REW) versus the DPG method, starting from 50 Erdős–Rényi graphs (G_0) on 10^3 nodes and average degree of $\langle d \rangle = 5$. For every G_0 its degree sequence was used to construct graphical realizations of it with conf-DPG, while tuning r to extreme positive (DPG+) and negative (DPG−) values. Degree-sequence-preserving rewiring of G_0 was done to increase (REW+) or decrease (REW−), greedily. The DPG approach achieves good improvement in the largest λ_1 over REW values (viral spread), whereas REW is a bit better for low λ_1 values (epidemics control). Unlike the REW, the DPG algorithm is rejection-free and thus one expects better DPG performance when REW has many rejections. **e**, DPG immunization (partitioning) with 53 (grey) nodes of a human contact network with 410 individuals, using matchings based on the highest betweenness centrality edges. Paths between partitions (different colours) run through DPG nodes. **f**, Same as in **e**, but on Erdős–Rényi graphs (ER, $\langle k \rangle = 2.4$) and random geometric graphs (RGG, $\langle k \rangle = 10$), monitoring the fraction of nodes q (averaged over 50 runs) in the largest component still connected through the edges of the original network, as function of fraction x of DPG nodes. N_0 is the number of nodes in the original network. For both types, a shattering transition happens at x_c ($x_c = 0.011$ for RGG and $x_c = 0.205$ for ER). The inset shows the convergence to the transition point with increasing N_0, for both cases.

that of non-spatial ones. As human physical contacts and computer networks have a strong spatial component, we expect DPG immunizations to work well in such cases.

In summary, we introduced a family of network growth models based on degree saturation, a realistic constraint in physical networks. The DPG mechanism can generate the exact structure of most real-world networks, despite the fact that this is an NP-hard problem, highlighting its modelling potential in real-world applications. An interesting open question then is what properties make real-world networks DPG-feasible. The flexibility of the DPG mechanism allows the tuning of various graph statistics. In addition to those shown above, we can tune for clustering; for example, the iconf-DPG constructs tree-like networks (like the configuration model) but gconf-DPG generates clustered ones. Proper selection of matchings can also tune network communities. Regarding scale-free networks, DPG builds them without preferential attachment, unlike other models. If preferential attachment represents the 'rich-gets-richer' mechanism, then DPG is a 'tinkering' mechanism that preserves function (in this case degrees) during growth while expanding its functional repertoire, a property observed in real systems[42].

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41567-021-01417-7.

Received: 31 August 2019; Accepted: 13 October 2021;
Published online: 23 December 2021

References

1. Barabasi, A.-L. & Albert, R. Emergence of scaling in random networks. *Science* **286**, 509–512 (1999).
2. Burt, R. S. Decay functions. *Soc. Netw.* **22**, 1–28 (2000).
3. Bahulkar, A., Szymanski, B. K., Lizardo, O., Dong, Y., Yang, Y. & Chawla, N. V. Analysis of link formation, persistence and dissolution in NetSense data. In *Proc. 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)* 1197–1204 (IEEE, 2016).
4. Lovász, L. & Plummer, M. D. *Matching Theory* (AMS Chelsea, 2009).
5. Trinajstić, N., Klein, D. J. & Randić, M. On some solved and unsolved problems of chemical graph theory. *Intl J. Quant. Chem.* **30**, 699–742 (1986).
6. Liu, Y. Y., Slotine, J.-J. & Barabási, A.-L. Controllability of complex networks. *Nature* **473**, 167–173 (2011).
7. Burt, R. S. *Structural Holes: The Social Structure of Competition* (Harvard Univ. Press, 1992).
8. Burt, R. S. Reinforced structural holes. *Soc. Netw.* **48**, 149–161 (2015).
9. Kleinberg, J., Suri, S., Tardos, É. & Wexler, T. Strategic network formation with structural holes. In *Proc. 9th ACM Conference on Electronic Commerce (EC '08)* 284–293 (Association for Computing Machinery, 2008).
10. Roy, S. & Jones, A. K. Cutting out the middleman. *Nat. Chem. Biol.* **9**, 603–605 (2013).

11. Edmonds, J. Paths, trees, and flowers. *Can. J. Math.* **17**, 449–467 (1965).
12. Erdös, P. L., Kharel, S. R., Mezei, T. R. & Toroczkai, Z. Degree-preserving graph dynamics—a versatile process to construct random networks. Preprint at https://arxiv.org/abs/2111.11994 (2021).
13. Biedl, T., Demaine, E. D., Duncan, C. A., Fleischer, R. & Kobourov, S. G. Tight bounds on maximal and maximum matchings. *Discrete Math.* **285**, 7–15 (2004).
14. Henning, M. A. & Yeo, A. Tight lower bounds on the matching number in a graph with given maximum degree. *J. Graph Theory* **89**, 115–149 (2018).
15. Frieze, A. & Pittel, B. in *Mathematics and Computer Science III: Algorithms, Trees, Combinatorics and Probabilities* (eds Drmota, M. et al.) 95–132 (Birkhäuser, 2004).
16. Bourassa, V. & Holt, F. SWAN: Small-world wide area networks. In *Proc. International Conference on Advances in Infrastructure (SSGRR 2003w)*, paper 64 (2003).
17. Holt, F. B., Bourassa, V., Bosnjakovic, A. M. & Popovic, J. in *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks* (ed. Wu, J.) 799–824 (CRC, 2005).
18. Hu, J., MacDonald, A. H. & McKay, B. D. Correlations in two-dimensional vortex liquids. *Phys. Rev. B* **49**, 15263–15270 (1994).
19. Robinson, R. W. & Wormald, N. C. Almost all regular graphs are hamiltonian. *Random Struct. Algorithms* **5**, 363–374 (1994).
20. Cooper, C., Dyer, M. & Greenhill, C. Sampling regular graphs and a peer-to-peer network. *Comb. Probab. Comput.* **16**, 557–593 (2007).
21. Tomita, K., Kurokawa, H. & Murata, S. Graph automata: natural expression of self-reproduction. *Physica D* **171**, 197–210 (2002).
22. Földes, S., Hammer, P.L. Split graphs. In *Proc. 8th Southeastern Conference on Combinatorics, Graph Theory and Computing* (eds Hoffman, F. et al.) 311–315 (Utilitas Mathematica, 1977).
23. Hammer, P. L. & Simeone, B. The splittance of a graph. *Combinatorica* **1**, 275–284 (1981).
24. Barrus, M. D., Hartke, S. G., Jao, K. F. & West, D. B. Length thresholds for graphic lists given fixed largest and smallest entries and bounded gaps. *Discrete Math.* **312**, 1494–1501 (2012).
25. Bollobás, B. A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *Eur. J. Comb.* **1**, 311–316 (1980).
26. Molloy, M. & Reed, B. A critical point for random graphs with a given degree sequence. *Random Struct. Algorithms* **6**, 161–180 (1995).
27. Chung, F. & Lu, L. The average distances in random graphs with given expected degrees. *Proc. Natl Acad. Sci. USA* **99**, 15879–15882 (2002).
28. Kim, H., Toroczkai, Z., Erdős, P. L., Miklós, I. & Székely, L. A. Degree-based graph construction. *J. Phys. A* **42**, 392001 (2009).
29. Xulvi-Brunet, R. & Sokolov, I. M. Changing correlations in networks: assortativity and dissortativity. *Acta Phys. Pol. B* **306**, 1431–1455 (2005).
30. Li, X. & Shi, Y. A survey on the Randić index. *MATCH Commun. Math. Comput. Chem.* **59**, 127–156 (2008).
31. Randić, M. On characterization of molecular branching. *J. Am. Chem. Soc.* **97**, 6609–6615 (1975).
32. Devillers, J. & Balaban, A. T. (eds) *Topological Indices and Related Descriptors in QSAR and QSPR* (Wiley-VCH, 1999).
33. Newman, M. E. J. Assortative mixing in networks. *Phys. Rev. Lett.* **89**, 208701 (2002).
34. Van Mieghem, P., Wang, H., Ge, X., Tang, S. & Kuipers, F. A. Influence of assortativity and degree-preserving rewiring on the spectra of networks. *Eur. Phys. J. B* **76**, 643–652 (2010).
35. Winterbach, W., de Ridder, D., Wang, H. J., Reinders, M. & Van Mieghem, P. Do greedy assortativity optimization algorithms produce good results? *Eur. Phys. J. B* **85**, 151 (2012).
36. Abdo, H., Dimitrov, D., Réti, T. & Stevanović, D. Estimating the spectral radius of a graph by the second Zagreb index. *MATCH Commun. Math. Comput. Chem.* **72**, 741–751 (2014).
37. Wang, Y., Chakrabarti, D., Wang, C. & Faloutsos, C. Epidemic spreading in real networks: an eigenvalue viewpoint. In *Proc. 22nd International Symposium on Reliable Distributed Systems (SRDS)* 25–34 (IEEE, 2003).
38. Saha, S., Adiga, A., Aditya Prakash, B., & Vullikanti, A. K. S. Approximation algorithms for reducing the spectral radius to control epidemic spread. In *Proc. 15th SIAM International Conference on Data Mining (SDM)* 568–576 (2015).
39. Cvetković, D. & Simić, S. Graph spectra in computer science. *Linear Algebra Its Appl.* **434**, 1545–1562 (2011).
40. Chung, F. R. K. *Spectral Graph Theory* (American Mathematical Society, 1997).
41. Dall, J. & Christensen, M. Random geometric graphs. *Phys. Rev. E* **66**, 016121 (2002).
42. Jacob, F. Evolution and tinkering. *Science* **196**, 1161–1166 (1977).
43. Jacomy, M., Venturini, T., Heymann, S. & Bastian, M. ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software. *PLoS ONE* **9**, 98679 (2014).

Methods

Steady state growth dynamics of MaxDPG networks. Because $\tilde{\sigma}$ depends only on $\mathbf{d}$, similar degree sequences should have similar splitnesses. Let C be the set of nodes in the complete subgraph $K_{|C|}$ and I the periphery. Because for a linear degree sequence, equation (4) with unit slope $s \simeq \lfloor n/2 \rfloor$, we have $|C| \simeq |I| \simeq n/2$. As MaxDPG graphs are not quite split graphs, one has a small set of nodes P with connections to both C and I, so that $|I| \simeq n/2$, $|C| \simeq n/2 - \mathcal{O}(\ln n)$ and $|P| = \mathcal{O}(\ln n)$. Due to equation (3), nearly all the nodes are matched and because half the nodes are in I, almost all the edges of a maximum matching are between C and I. This allows us to describe the steady-state MaxDPG dynamics as in Fig. 3e. Accordingly, the new node $n+1$ must connect to all nodes in C (thus joining C) and to almost all the nodes in I. The new C becomes a K_{n+1} graph. During cutting, however, all nodes in C lose a link to I, and eventually a node in C (Fig. 3e, top) will lose its last connection to I, thus joining it.

Configurational DPG. The task is the following: given an initial graph G_0 and a graphic set of degrees D, find an ordering $\mathbf{d} = (d_1, \ldots, d_n)$ of D in which to join these degrees via DPG into a graph. We call this method 'configuration DPG' or conf-DPG, and it is also a sampling method. Because DPG graphs grow by a node per step, one expects the process to succeed with a higher probability when D is increasingly ordered, which we call the iconf-DPG. Figure 5a shows the fraction of successful iconf-DPG runs $f(\gamma)$ on scale-free sequences chosen at random, as a function of γ. The curves cross around $\gamma_{\text{iconf}} \simeq 2.25$, which by a finite-size scaling argument shows a transition at γ_{iconf} between almost never to almost always successful iconf-DPG runs. Note that the graphicality transition is at $\gamma_c = 2$ (ref. [44]). However, there is a better ordering of D: choose in every step the largest available degree that the current graph can accept; we call this model the greedy conf-DPG, or gconf-DPG. Figure 5a shows a higher success rate for gconf-DPG than for iconf-DPG, with only a weak dependence on sequence length.

Data availability

Source data are provided with this paper. The data that support the findings of this study are available from the corresponding author upon reasonable request.

Code availability

The codes used for simulation and analysis are available from the corresponding author upon reasonable request.

References

44. Del Genio, C. I., Gross, T. & Bassler, K. E. All scale-free networks are sparse. *Phys. Rev. Lett.* **107**, 178701 (2011).

Acknowledgements

We thank D. Soltész, I. Miklós, N. Rupprecht and J. Baker for useful discussions. The project was supported in part by the United States National Science Foundation through grant IIS-1724297 (Z.T.) and by the National Research, Development and Innovation Office of Hungary through NKFIH grants K 116769, KH 126853, K 132696 and SNN 135643 (P.L.E. and T.R.M.).

Author contributions

S.R.K. performed mathematical modelling, contributed proofs, provided analysis tools, designed and ran experiments, analysed data and generated the figures. S.C. ran experiments. T.R.M. and P.L.E. performed mathematical modelling and provided proofs. Z.T. proposed the main idea, performed mathematical modelling, contributed proofs and was the lead writer of the manuscript. All authors contributed to proofreading the paper.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41567-021-01417-7.

Correspondence and requests for materials should be addressed to Zoltan Toroczkai.

Peer review information *Nature Physics* thanks Thilo Gross and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

nature portfolio

https://doi.org/10.1038/s41567-021-01417-7

Supplementary information

Degree-preserving network growth

Supplementary Information

Degree-preserving network growth

Shubha R. Kharel,[1] Tamás R. Mezei,[2,3] Sukhwan Chung,[1] Péter L. Erdős,[2,3] and Zoltán Toroczkai[1]

[1]*Department of Physics, 225 Nieuwland Science Hall,*
University of Notre Dame, Notre Dame, IN, 46556 USA
[2]*Alfréd Rényi Institute, Reáltanoda u 13-15 Budapest, 1053 Hungary*
[3]*Eötvös Loránd Research Network, Budapest, 1052 Hungary*

CONTENTS

I. MATHEMATICAL BACKGROUND

In the following, we work with simple, undirected graphs, i.e., graphs in which there is at most one edge between any pair of distinct nodes and no edge starts and ends in the same node (no self-loops). Graph or network is used interchangeably and it is typically denoted by $G(V,E)$ (or, simply by G), where V (or $V(G)$) is the set of nodes and E (or $E(G)$) is the set of edges of G. An edge is labeled by the pair (u,v) of nodes that it connects. Given some set A, $|A|$ denotes the number of elements (cardinality) of A. We will also use the standard asymptotic notation $\mathcal{O}$ and Θ meaning that $f(x) = \mathcal{O}(g(x))$ if $f(x)$ does not grow faster than $g(x)$ for $x \to \infty$ and $f(x) = \Theta(g(x))$ when $f(x)$ grows like $g(x)$ as $x \to \infty$.

A. Matching

A matching in a graph is a set of independent edges, which are pairwise non-adjacent, i.e., no two of them share a common vertex. A matching in a graph is *maximal* if no additional edges from the graph can be included in the matching. *Maximum matching* is a maximal matching of the largest size and this size is called the graph's *matching number*, $\nu(G)$. We have $1 \le \nu(G) \le \frac{1}{2}n$ (for graphs with at least one edge), where $n = |V(G)|$ is the order of G. When $\nu(G) = \lfloor \frac{n}{2} \rfloor$ the matching is called *perfect matching* for n even and *near perfect matching* for n odd. We say a node is *matched* if it is incident on an edge from the matching. Nodes that are not matched by any edge from the matching are called *exposed*. There can be several maximal or maximum matchings in a graph as discussed in the main text and Fig. 1 of the main text. Simple examples of graphs with perfect matching are the $2n$-cycle C_{2n} (sparse) and the complete graph K_{2n} (dense): C_{2n} has two, whereas K_{2n} has $(2n-1)!!$ perfect matchings. Clearly, when $\nu(G)$ is low then a small number of vertices collect a large number of edges, with the star graph S_n having the lowest value, $\nu(S_n) = 1$.

Since only (t by
unity: $\nu_{n+1} \le$ can
drop by almos

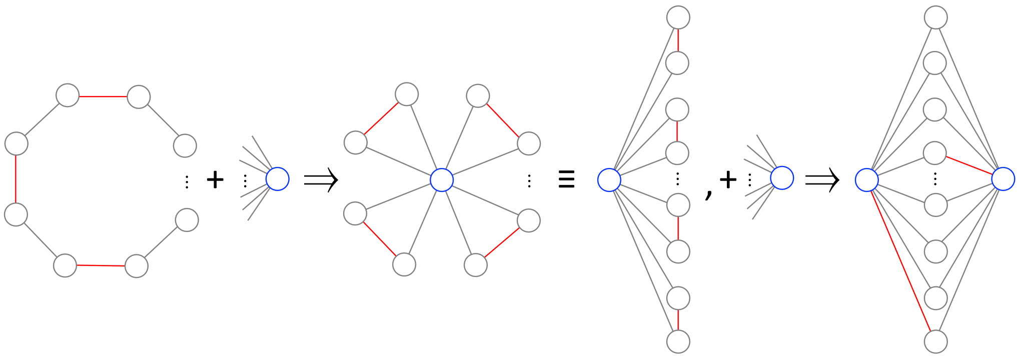

FIG. S1. **Matching number drop during DPG:** An example for a large drop in the matching number. Starting with a cycle C_{2n}, $n \ge 3$, and thus $\nu(C_n) = n$, we join a new node of degree $2n$ in a DPG step, resulting in a so-called friendship graph [1]. The edges of the triangles opposite to the common node form an independent set of size n and thus we can join another DPG node of degree $2n$, resulting in the end graph shown in the figure. However, that graph has a matching number of 2, independently of n. Thus, the matching number dropped from $\Theta(n)$ to $\Theta(1)$ in two DPG steps. The red edges form a maximum independent set.

B. Graphical sequences

A sequence of integers $\mathbf{d} = (d_1, \ldots, d_n)$, $d_i \ge 1$, $i = 1, \ldots, n$ is *graphical* if there is a simple graph G with $\mathbf{d}$ as its degree sequence, denoted by $\mathbf{d}(G)$. Sufficient and necessary conditions for a given sequence of integers to be graphical is provided by the Erdős-Gallai theorem [3]:

Theorem S1 (Erdős-Gallai, EG). *Let* $\mathbf{d} : d_1 \ge d_2 \ge ...... \ge d_n \ge 0$ *be a sequence of non-negative integers, arranged non-increasingly. Then* $\mathbf{d}$ *forms the degree sequence of a simple graph iff* (i) $d_1 + ... + d_n$ *is even, and* (ii) *for all*

$k = 1, ..., n$ *we have*

$$\sum_{i=1}^{k} d_i \leq k(k-1) + \sum_{i=k+1}^{n} \min\{k, d_i\} \ . \tag{S1}$$

Theorem S1 is not minimal in terms of the number of inequality checks that need to be done; it was shown by several authors for example in [4], that it is sufficient and necessary to check these inequalities only for special k values, in particular for only those $1 \leq k \leq s$ values for which $d_k > d_{k+1}$ (*step-indices*) and at s, where s is the pivot index introduced in the main text (also introduced in [4] but not called as such).

C. Graph splitness

Hammer and Simeone [6] define the "splittance" $\sigma(G)$, as the minimum number of edges that need to be added/removed from G to make it into a split graph, thus serving as a measure of how far G is from a split graph. They show that

$$\sigma(G) = \frac{1}{2}\left(s(s-1) - \sum_{i=1}^{s} d_i + \sum_{i=s+1}^{n} d_i \right) \ , \tag{S2}$$

where s is $\mathbf{d}$'s pivot index. In the main paper we define the normalized version of splittance via $\tilde{\sigma}(G) = \sigma(G)/m(G)$, where $m(G) = \frac{1}{2}\sum_{i=1}^{n} d_i$ is the total number of edges, and we call $\tilde{\sigma}(G)$ the "splitness " of the graph.

Some particular values for the splittance $\sigma(G)$ of a graph G: $\sigma(P_n) = n-4$, $\sigma(K_{m,n}) = q(q-1)/2$, $q = \min(m,n)$ and for a regular graph of degree d: $\sigma(R_{d,n}) = d(n-d-1)/2$. For any graph, we also have the tight upper bound $\sigma(G) \leq \left\lfloor \frac{1}{2}\left(\binom{n}{2} - \left\lfloor \frac{n^2}{4} \right\rfloor \right) \right\rfloor$, or roughly $n^2/8$.

II. THEOREMS AND PROOFS

A. Degree-preserving reduction

In the main text we discussed the conditions under which a node w can be DP reduced from a network, i.e., performing the "inverse" of a DPG step for the simpler case when w has an even degree. Below we first provide its precise formulation along with its proof, then we turn to the more general case, corresponding to the general forward DPG algorithm which allows also odd degree nodes to be added to the graph, and do the same. We first introduce some notations. Given a set of nodes S let $\Gamma(S)$ denote the subgraph induced in G by the nodes of S. Let $N(S)$ denote the set of neighbors of S in G and $\overline{\Gamma}(S)$ denote the complementary graph of $\Gamma(S)$ in G.

Lemma S2. *A simple graph G is DP-reducible under the even degree DPG process iff G has at least one vertex whose complement neighborhood graph has a perfect matching.*

Proof. Let $w \in V(G)$ be a node with even degree d such that $\overline{\Gamma}(N(w))$ has a matching $\overline{M}$ with $d/2$ edges (perfect matching). This partitions the neighbors of w in G into $d/2$ disjoint pairs, each pair connected by an edge in $\overline{M}$. Remove w with its edges in G, then transfer all edges of $\overline{M}$ into edges of G (edges of $\overline{\Gamma}(N(w))$ are non-edges in G), obtaining another simple graph G'. The other direction is also simple. ■

We now turn to discuss the general DPG case and its inverse operation. The general forward DPG step and also its inverse (or the DP reduction step) are described in the algorithms sections III B 2, III C. As presented there, in the general forward DPG step we allow one stub to be present at times in the graph (since now we allow also odd degree nodes to join the graph), considered connected to an "auxiliary" node (see III B 1 for description). When reversing this procedure, therefore, we also need to consider the existence of such an auxiliary node, denoted by s. The following Lemma is a generalization of Lemma S2 for the general DPG process:

Lemma S3. *A simple graph G (which may contain an auxiliary node s) is DP-reducible iff G has at least one node w such that one of the following is true: 1) the degree d_w of w is even and $\overline{\Gamma}(N(w))$ (which may include s) has perfect matching; 2) d_w is odd and $\overline{\Gamma}(N(w))$ has near-perfect matching when s is absent or $\overline{\Gamma}(N(\{w,s\}) \setminus \{w,s\})$ has perfect matching when s is present.*

Proof. The proof proceeds along similar lines as in Lemma S2, however, considering now all the possible cases corresponding to the generalized DPG process, presented in section III B 2. ■

This provides a polynomial algorithm for checking if a graph is DP-reducible. The reduction step is then repeated on G', obtaining G'' and so on, until a smallest graph G_0 obtained that is non-reducible. Using Lemma S2 one can show that one family of such G_0 graphs is obtained from taking m copies of K_ℓ and adding a 1-factor between every pair of them, for $\ell \geq 4$, $\ell + m$ even and $\ell > m$. The graph obtained is a non-DP-reducible $(\ell + m - 2)$-regular graph. However, there are other, non-DP reducible graph families, including sparse graphs, with some examples being shown in Figs. S5-S8.

B. The Vizing bound on the matching number

Denoting by $\Delta(G)$ the maximum degree of graph G, Vizing's theorem [5] says that for all finite simple graphs $\Delta(G) \leq \chi'(G) \leq \Delta(G) + 1$, where $\chi'(G)$ is the chromatic index of G. The chromatic index is the minimum number of colors needed for a proper edge-coloring of G. A proper edge coloring of a graph is any coloring in which no two incident edges (incident edges share a node) are colored with the same color. Since every color class forms a matching in G we can write $\chi'(G) \geq \frac{m(G)}{\nu(G)}$ where $m(G) = |E(G)|$ is the number of edges in G, thus obtaining

$$\nu(G) \geq \frac{m(G)}{\Delta(G) + 1} \; . \tag{S3}$$

C. Maximum DPG

The Maximum DPG (or MaxDPG) model is a limiting case of the DPG family of models, and it allows rigorous analytical treatment, although not easily, by any means. In MaxDPG the incoming degree (working with even degrees, only) in the n-th step is completely determined by the graph's matching number after the $n-1$-th step via

$$d_n = 2\nu(G_{n-1}) = 2\nu_{n-1} \; , \quad n \geq 1 \tag{S4}$$

which is also the maximum degree of a DPG node that the graph can take, at that step. Here we provide the proof of the bound (3) (in the main text) on the matching number, presented here as Theorem S8. While the bound itself looks simple, the proof below is fairly involved, requiring several other results, that we first need to establish.

We start by invoking an observation according to which a Hamiltonian graph (of even order) always has a perfect matching. A graph is Hamiltonian if there is a closed path (a cycle) that visits each node exactly once. Given such a graph, we can easily construct a perfect matching in it, by taking every other edge along the Hamiltonian cycle. We now recall a theorem by Pósa:

Theorem S4 (Pósa [2])**.** *Let G be a graph on n vertices and let $t(k)$ denote the number of vertices of degree not exceeding k. If $t(k) < k$, $\forall\, 1 \leq k < \frac{n}{2}$, then G has a Hamiltonian cycle.*

Pósa's theorem expresses the intuitive observation that if a graph has sufficiently many edges, it will also have a Hamiltonian cycle. As we will see, G in general does not satisfy the conditions in Pósa's theorem for MaxDPG, however, it is not far from it. The idea then is to extend G with the smallest number of $r(G)$ vertices, each connected to all vertices of G, such that the extended graph G^+ satisfies Pósa's theorem. Note that an equivalent formulation of the conditions in Theorem S4 is to demand that $d_k \geq k + 1$. Thus, this procedure amounts to "lifting" all the degrees of G such that $d_k + r(G) \geq k + 1$ for all $1 \leq k < (n + r(G))/2$. Having a Hamiltonian cycle in the extended graph G^+, we now delete the added vertices and their links and thus the remaining edges of the matching along the cycle in G^+ will form a matching in G. One can therefore announce:

Theorem S5. *Let G be a simple graph on n vertices and let $t_G(k)$ be the number of vertices of degree not exceeding k. Let*

$$r(G) := \min\left\{ \ell \in \mathbb{Z}^+ \colon t_G(k - \ell) < k \, , \; \forall\, \ell \leq k < \frac{n + \ell}{2} \right\} \tag{S5}$$

then G has a matching of size: $\lceil \frac{n - r(G)}{2} \rceil \leq \nu(G)$.

Proof. Based on (S5), $r(G)$ is the smallest positive integer ℓ for which the inequality holds *for all* degrees k in the shown range. One can think of ℓ as the extra degree coming from the ℓ added vertices. The value of $r(G)$ guarantees that the larger G^+ satisfies the conditions in Pósa's theorem. Proceeding as described just before Theorem S4, when $n + r(G)$ is even, we obtain a matching of size $\frac{n-r(G)}{2}$ in G and when odd, the matching is of size $\frac{n+r(G)-1}{2}$. Thus, at least $\frac{n+r(G)-1}{2} - (r(G) - 1) = \frac{n-r(G)+1}{2}$ of the chosen edges are inside G. ■

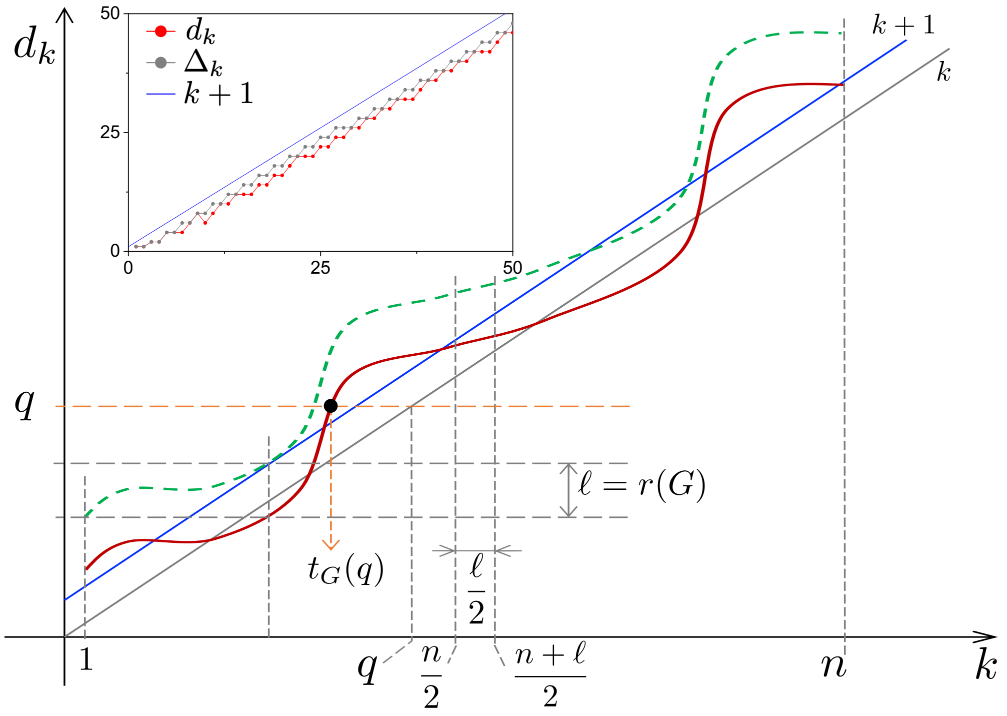

FIG. S2. **Schematic for the proof of Theorem S5.** The dashed green curve is the red curve shifted up by ℓ. Inset: the degree sequence $\mathbf{d}$ for a MaxDPG run starting from K_2 shown in red and the sequence $\boldsymbol{\Delta}$ (defined in the text) in gray.

For $r(G)$ small, we get a high lower bound on maximum matching. This bound is useful only for sufficiently dense graphs but it is noninformative for sparse ones. E.g., if $G = C_{2n}$ then $r(G) = 2n - 4$, so $\nu(C_{2n}) \geq 2$, far from $\nu(C_{2n}) = n$.

Next, we estimate $r(G)$ for the MaxDPG process. Let $(G_n)_{n=0}^{\infty}$ be the sequence of MaxDPG graphs and denote $r_n = r(G_n)$ and $t_n = t_{G_n}$. Let us introduce

$$s_n := \min\left\{\ell \in \mathbb{Z}^+ \colon t_n(k-\ell) < k \,,\ \forall\, \ell \leq k < n\right\} \,, \tag{S6}$$

which simply extends the range of k to n, compared to the definition of r_n. Since existing degrees are preserved, $(r_n)_{n=0}^{\infty}$, $(s_n)_{n=0}^{\infty}$ are both monotonically increasing sequences and $r_n \leq s_n$ for all $n \geq 0$ (since for s_n more constraints need to be satisfied). We next prove the following:

Lemma S6. *For any m: $n \leq m \leq 2n - s_n$, we have $r_m \leq s_n$.*

Proof. To simplify notations, let $p_n := 2n - s_n$. We proceed by induction on m. The statement is clearly true for $m = n$. Let us assume that $r_{m-1} \leq s_n$, $m \geq n+1$. Want to show that $r_m \leq s_n$. Since Eq. (4) of the main text defines r_m as the smallest ℓ for which $t_m(k-\ell) < k$, for all $\ell \leq k < (m+\ell)/2$, it is sufficient to show that $t_m(k-s_n) < k$ holds for all $s_n \leq k < (m+s_n)/2$. Based on Theorem S5 we have $\nu_{m-1} \geq (m - 1 - r_{m-1})/2$ and thus for all $m \in [n+1, p_n]$ we have

$$d_m = 2\nu_{m-1} \geq m - 1 - r_{m-1} \geq m - 1 - s_n \geq n - s_n \,. \tag{S7}$$

Clearly, $m \leq p_n$ implies that $s_n \leq k < (m + s_n)/2 \leq n$ and thus the new vertex's degree $d_m \geq n - s_n$ falls outside the needed range for k, so

$$t_m(k - s_n) = t_n(k - s_n) \,,\ \ s_n \leq k < \frac{m + s_n}{2} \,, \tag{S8}$$

for all $m \in [n+1, p_n]$. By Eq. (5) of the main text, however, $t_n(k - s_n) < k$ and thus $t_m(k - s_n) < k$, which we wanted to prove. ■

We can now use this result to prove:

Lemma S7. *For any $n \geq 0$ we have: $s_{2n-s_n} \leq s_n + 2$.*

Proof. It is sufficient to prove that $t_{p_n}(k - (s_n + 2)) < k$ for all $s_n + 2 \leq k < p_n$. Choosing $m = p_n$ in (S8) we obtain that $t_{p_n}(k - s_n) = t_n(k - s_n)$ for all $s_n \leq k < n$. But, by the definition of s_n, $t_n(k - s_n) < k$ for all $s_n \leq k < n$, so $t_{p_n}(k - s_n) < k$, $\forall k \in [s_n, n)$. Changing the variable k to $k-2$ gives $t_{p_n}(k-(s_n+2)) < k-2 < k$, $\forall k \in [s_n+2, n+2)$ (noting that the lhs is zero for $k = 1, 2$). In the following, we will make use of the above inequality for $k = n+1$

$$t_{p_n}(n - 1 - s_n) < n + 1 \; . \tag{S9}$$

We are left to prove that $t_{p_n}(k - (s_n + 2)) < k$ for all $n + 2 \leq k < p_n$. G_{p_n} contains the degree sequence of G_n and, in addition, the degrees $d_{n+1} \leq \ldots \leq d_{p_n}$. Let $u_n(k)$ be the number of vertices of degree k in G_n. Thus

$$t_{p_n}(k - 2 - s_n) = t_{p_n}(n - 1 - s_n) + \sum_{j=0}^{k-2-n} u_{p_n}(n + j - s_n) \; . \tag{S10}$$

Since $d_m \geq m - 1 - s_n$ (see (S7)), $d_{n+1} \geq n - s_n$ but $d_{n+2} \geq n + 1 - s_n$, so $u_{p_n}(n - s_n) \leq 1$; $d_{n+3} \geq n + 2 - s_n$, so $u_{p_n}(n-s_n)+u_{p_n}(n+1-s_n) \leq 2$, and so on. This means from (S10) that $t_{p_n}(k-2-s_n) \leq t_{p_n}(n-1-s_n)+k-1-n$ or after using (S9): $t_{p_n}(k - 2 - s_n) < n + 1 + k - 1 - n = k$, for all $k \in [n + 2, p_n)$. ■

Armed with the results above we now focus on our main theorem and provide two proofs for it. The first is a short technical proof, the second is more involved but provides more insight.

Theorem S8. *If $s_{n_0} \leq n_0 - 3$ then $s_n \leq 2\log_2 n + \mathcal{O}(1)$ and thus*

$$\nu_n \geq \frac{1}{2}n - \log_2 n + \mathcal{O}(1) \; . \tag{S11}$$

Proof. This is the first, short proof. Let us introduce $n_i = n_0 + 2^i + 2i - 1$, $i \in \mathbb{N}_0$. We prove by induction that $s_{n_i} \leq n_0 + 2i - 3$. For $i = 0$ the statement is just the condition assumed. Let us assume that $s_{n_j} \leq n_0 + 2j - 3$, for all $1 \leq j \leq i$. Then, $2n_i - s_{n_i} \geq 2(n_0 + 2^i + 2i - 1) - (n_0 + 2i - 3) = n_0 + 2^{i+1} + 2i + 1 = n_{i+1}$. Since s_n is non-decreasing, $s_{n_{i+1}} \leq s_{2n_i - s_{n_i}} \leq s_{n_i} + 2$, using Lemma S7 for the last inequality. But $s_{n_i} + 2 \leq (n_0 + 2i - 3) + 2$, so $s_{n_{i+1}} \leq n_0 + 2(i+1) - 3$, which we wanted to show. Now let $f(i) = n_0 + 2i - 3$. We have $n_i = 2^{[f(i)+3-n_0]/2} + f(i) + 2$. For $i \geq 1$, $f(i) + 2 \geq 0$, so $2\log_2(n_i) + n_0 - 3 \geq f(i) \geq s_{n_i}$. As s_n is monotone increasing and $f(i+1) - f(i) = \mathcal{O}(1)$, the theorem holds. ■

1. *Alternative proof of the main Theorem S8*

Before proving this theorem, let us consider the sequence of positive integers z_n obeying:

$$z_{2n-z_n} = z_n + 2 \tag{S12}$$

$$z_{n+1} \geq z_n \tag{S13}$$

$$z_{n_0} \geq s_{n_0} + 2 \; , \tag{S14}$$

for all $n \geq n_0$.

Lemma S9. *We have $s_n \leq z_n$, for all $n \geq n_0$.*

Proof. First we show: if $s_m \leq z_m$, then $s_{2m-s_m} \leq z_{2m-s_m}$. Clearly, $s_m \leq z_m$ implies $2m - z_m \leq 2m - s_m$ and since z_m is non-decreasing (from (S13)), $z_{2m-z_m} \leq z_{2m-s_m}$. Thus, from Lemma 6 of the main text: $s_{2m-s_m} \leq s_m + 2 \leq z_m + 2 = z_{2m-z_m} \leq z_{2m-s_m}$, which is what we wanted to show. Next, let $n_i = 2n_{i-1} - s_{i-1}$, $i \geq 1$. Starting from $s_{n_0} < z_{n_0}$, using induction and the above, we get $s_{n_i} \leq z_{n_i}$, $\forall i \geq 0$. Moreover, $z_{n_{i+1}} - z_{n_i} = z_{2n_i - s_{n_i}} - z_{n_i} \geq z_{2n_i - z_{n_i}} - z_{n_i} = 2 \geq s_{2n_i - s_{n_i}} - s_{n_i} = s_{n_{i+1}} - s_{n_i}$ (the last inequality is based on Lemma 6 of the main text). Hence: $z_{n_{i+1}} - s_{n_{i+1}} \geq z_{n_i} - s_{n_i}$, for all $i \geq 0$ and therefore $z_{n_i} - s_{n_i} \geq \ldots \geq z_{n_0} - s_{n_0} \geq 2$ (from (S14)). Thus $z_{n_i} - s_{n_i} \geq 2$ or $z_{n_i} \geq s_{n_i} + 2 \geq s_{n_{i+1}}$. Since z_{n_i} is the minimum of z_n and $s_{n_{i+1}}$ is the maximum of s_n in $n \in [n_i, n_{i+1}]$, we must have $s_n \leq z_n$, $n \in [n_i, n_{i+1}]$, $\forall i \geq 0$ and thus for all $n \geq n_0$. ■

Consider the embedding of the sequence z_n into $\mathbb{R}^+$, through the functional equation

$$g(2x - g(x)) = g(x) + 2 \tag{S15}$$

with $g'(x) > 0$ (monotonic) and $g(x) < x - 2$, for all $x \geq n_0$ and with fixed boundary condition $g(n_0) = z_{n_0}$. If such a $g(x)$ exists, $z_n = g(n)$ will satisfy (S12)-(S14). Note, the functional equation (S15) is an Abel equation. To solve it we use the fact that g is monotonic and thus it has an inverse $x = g^{-1}(y) \equiv h(y)$. Taking the inverse of (S15) we obtain

$$2h(y) = h(y+2) + y\ , \tag{S16}$$

which, differentiated twice gives $2h''(y) = h''(y+2)$, solved by $h''(y) = a2^{y/2}$, where a is an arbitrary constant. Integrating twice and inserting the result into (S16) we find: $h(y) = \frac{2a}{\ln 2}2^{y/2} + y + 2 = g^{-1}(y) = x$. Note that $a = 0$ gives the linear solution of (S16), in which case $g(x) = x - 2$, but that is excluded by $g(x) < x - 2$ and thus $a > 0$. The inverse is: $g(x) = x - 2 - \frac{2}{\ln 2}W\left(a2^{(x-2)/2}\right)$, where W is the Lambert function. Asymptotically, for large z, $W(z) \simeq \ln z - \ln\ln z + \frac{\ln\ln z}{\ln z} + \mathcal{O}((\ln\ln z/\ln z)^2)$ and thus

$$g(x) = 2\log_2 x + \frac{2}{\ln 2}\ln\left(\frac{\ln 2}{2a}\right) + \mathcal{O}\left(\frac{\ln x}{x}\right)\ ,\quad x \gg 1\ , \tag{S17}$$

with $a = (n_0 - z_{n_0} - 2)2^{-1-z_{n_0}/2}$. Note that the leading term does not depend on a. Thus, $g(n) = 2\log_2 n + \mathcal{O}(1) = z_n \geq s_n$ (from Lemma S9). Using Theorem S5: $\nu_n \geq \lceil \frac{n-r_n}{2} \rceil \geq \frac{n-r_n}{2} \geq \frac{n-s_n}{2} \geq \frac{n}{2} - \log_2 n + \mathcal{O}(1)$, which finishes the proof. Note that the logarithmic deviation from $n/2$ is only an estimate, and in fact, ν_n stays closer to $n/2$. ■

D. Random fraction DPG and the derivation of its degree distribution

This DPG model is defined via:

$$d_n = 2\lceil r_n \nu_{n-1} \rceil\ , \tag{S18}$$

where $r_n \in (0,1)$ is a uniformly distributed random variable, $\rho(r) = 1$. Fig. S3a shows a sample graph and Fig. S3b shows that here too, ν_n has linear growth: $\nu_n \simeq an$, with $a = 0.38$. The degree distribution is no longer linear but

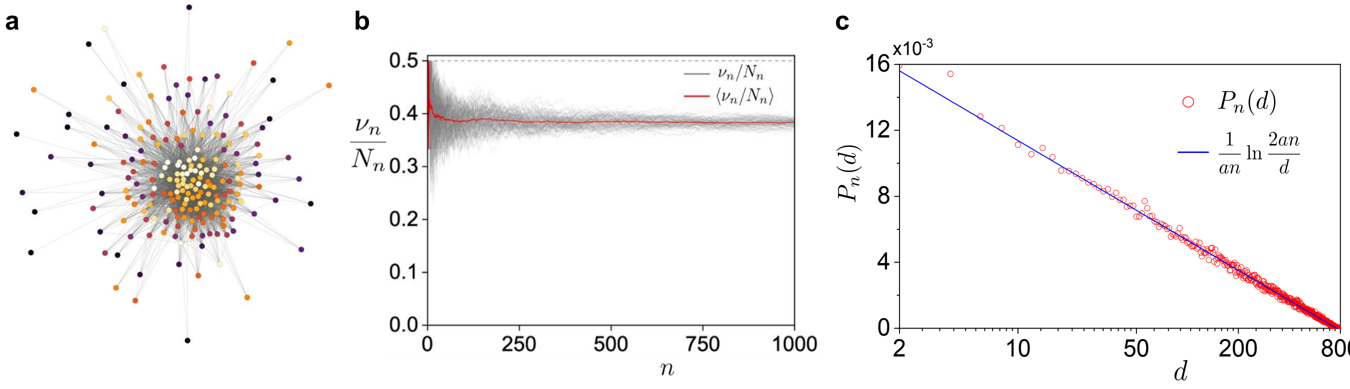

FIG. S3. **Random fraction DPG:** (a) A Random fraction DPG built sample graph (200 nodes), same node colors as in main text Figs. 3a-c; note, here newer nodes can join also the periphery. (b) The fraction $\langle \nu_n/N_n \rangle$ (average over 100 runs) saturates to a value under 0.5. (c) The degree distribution of random fraction DPG networks (obtained from 100 networks with 10^3 nodes) is logarithmic; the continuous line is the theoretical curve.

The calculations below are generalizable to other DPG processes. Let $N_n(d)$ denote the number of vertices of degree d after n steps (in the G_n DPG graph). The degree distribution is then

$$P_n(d) = \left\langle \frac{N_n(d)}{n} \right\rangle \tag{S19}$$

where $\langle\cdot\rangle$ denotes averaging over all G_n DPG graphs started from the same initial condition. Normally, we should divide by $n+|V(G_0)|$, but here we are interested in the $n \gg 1$ limit. One can write: $N_n(d) = N_{n-1}(d) + I_n(d)$, where $I_n(d) = \delta_{d_n,d}$ is an indicator function. Clearly, $\langle N_n(d)\rangle = \langle N_{n-1}(d)\rangle + \langle I_n(d)\rangle$. However, $\langle I_n(d)\rangle = \text{Prob}\{d_n = d\} \equiv p_n(d)$. Although the matching number ν_n itself is a stochastic variable, here we will replace it with its average value, which in all of the cases studied here is linear in n, with good approximation, i.e., $\nu_n \simeq \langle \nu_n\rangle = an$, where $0 < a \leq 1/2$. Neglecting the ceiling function in (S18) and considering variables as continuous ones, $d_n \simeq 2arn$, $2 \leq d_n \leq 2an$. Thus, we can express

$$p_n(d) = \langle \delta(d - d_n)\rangle = \int_0^1 dr\, \rho(r)\, \delta(d - 2anr) = \frac{1}{2an}\theta(2an - d)\ , \tag{S20}$$

where $\theta(\cdot)$ is the Heaviside step function. Let us denote $S_n(d) = \langle N_n(d)\rangle$. We have $S_n(d) = S_{n-1}(d) + \frac{1}{2an}\theta(2an-d)$, with $S_0(d) = N_0(d)$ being the number of nodes of degree d in G_0 (for K_2, $N_0(1) = 2$, $N_0(d) = 0$, $\forall d \geq 2$). Let $D = \frac{d}{2a}$. Then $S_n(d) = S_{n-1}(d)$, $\forall\, n < D$. Assuming $G_0 = K_2$ and $d \geq 2$, we thus have $S_n(d) = S_{n-1}(d) = \ldots = S_0(d) = 0$ for all $n < D$. For $n \geq D$ the recursion is easily solved to give

$$S_n(d) = \frac{1}{2a}\sum_{k=0}^{n-D} \frac{1}{D+k}\ , \quad n \geq D \tag{S21}$$

and thus

$$P_n(d) = \frac{2}{n}\sum_{k=0}^{n-\frac{d}{2a}} \frac{1}{d+2ak} \tag{S22}$$

expressing the sought degree distribution. The extra factor of 2 comes from the fact that only every second d value is allowed due to the evenness restriction. Using a Poisson resummation formula we can extract the leading terms as

$$P_n(d) \simeq \frac{1}{an}\left[\ln\left(\frac{2an}{d}\right) + \frac{1}{2}\left(\frac{2a}{d} + \frac{1}{n}\right) + \ldots\right] \tag{S23}$$

plotted in Fig. S3c.

III. ALGORITHMS

A. Finding a matching

In any DPG step, we need to find a matching of size $k \leq \nu$, where ν is the graphs's matching number. The simplest approach to find a matching of size k, which works well if k is not too large (compared to ν), is the greedy method. In the greedy method we first pick a random edge, add it to our matching set M (M initially is the empty set) then we remove this edge along with all the edges adjacent to it. We repeat this process by picking another random edge from the set of leftover edges (if non-empty), etc. until we find k edges or there are no more edges left. Note, that when the latter occurs, we have just obtained a *maximal* matching, M. It is well known, that the size of any maximal matching is at least $\nu/2$ [8] so one expects for the greedy method to succeed as long as $k \leq \nu/2$.

When k is larger than the size of the obtained maximal matching, we need something more sophisticated, in order to find k independent edges. This is based on Edmond's Blossom algorithm, which allows us to *continue* growing M (found by the greedy part) and increase its size one edge at a time. A path is called an alternating path if its every other edge is from the matching, M. An *augmenting path* P is an alternating path of odd length with exposed start and end nodes (see subsection I A). Edmond's algorithm starts with a set M of independent edges (for example, the maximal matching found greedily), then proceeds by constructing an augmenting path in the graph. If an augmenting path P exists, it removes the edges at the even positions from M and adds the edges at the odd positions to M. The resulting matching will have one more edge because P had an odd length. This process is repeated until there are no more augmenting paths, which, according to Berge's Lemma, implies that a *maximum matching* has been found [9, 10].

Note that the greedy algorithm is of $\mathcal{O}(|E|)$ complexity, whereas Edmond's Blossom algorithm finds a maximum matching in time $O(|V|^2|E|)$, which can be improved to $O(|V|^{1/2}\,|E|)$ [35].

Implementation of Edmond's algorithm in C++ can be found in Boost Graph Library [12] and LEMON Graph Library [18]. Wolfram *Mathematica*® also provides a specific function (FindIndependentEdgeSet[G]) to find a maximum matching in graph G.

B. The DPG algorithm

Degree-preserving growth is a not a single model but a family of models and this is reflected in the algorithm as well. Accordingly, it has some components that are internal and some that are external to the code. The external components (those which specify the actual model) are the incoming new node w's degree d_w, and the specificity of the set M of independent edges that are cut to form connections with node w. Given an incoming degree d_w, the size of M, is determined by that ($|M| \in \{\lfloor d_w/2 \rfloor, \lceil d_w/2 \rceil\}$ see subsection III B 2), however, the identity of the edges forming a set of that size is an external variable. It is this variable that we can use to tune other properties (such as degree correlations, or connectivity) of the DPG graph, as explained in a later section. The internal component is how the new node is connected to the existing graph, given in section III B 2, below.

1. Auxiliary node

To formulate our general DPG algorithm that allows nodes with odd incoming degree, we introduce the notion of the *auxiliary* node. Let w be the new node of degree d_w to be added through DPG to a simple graph G. If d_w is odd, no matter how we connect it to G without creating multiple edges, there will be a leftover stub or "half-edge" connected to a node (w or otherwise). Since graph with odd degree-sum cannot exists, we introduce a *auxiliary node* s with degree at most one to which we connect this dangling stub, so the whole structure actually forms a simple graph. When there is no stub, the auxiliary node is isolated (zero degree) and we consider it not present, removed from the graph, but created again, as necessary. There can be at most one auxiliary node in a DPG graph.

2. DPG Step

In a DPG step we always remove the edges in the selected M. The neighborhood of w may include the auxiliary node s depending on conditions that are outlined below.

—

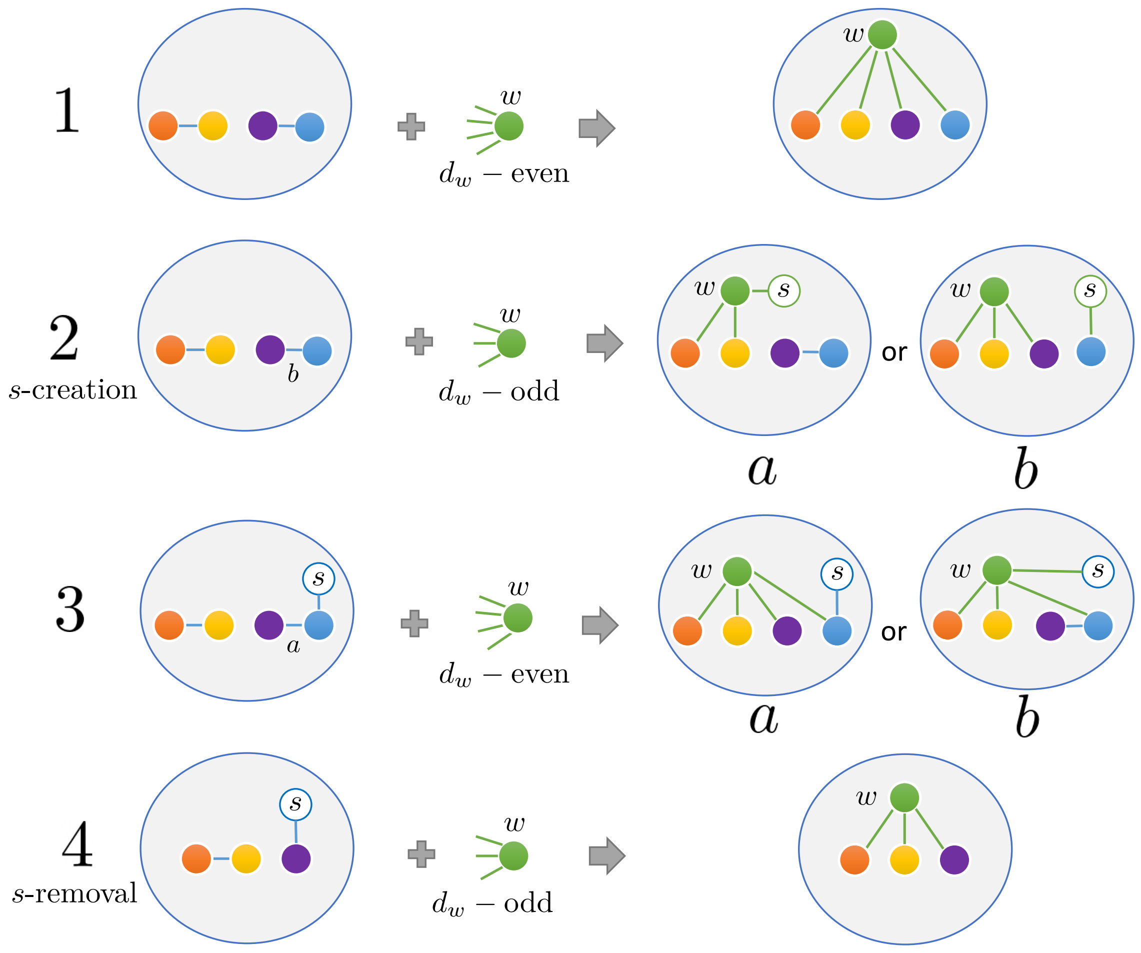

FIG. S4. **Schematic for the general DPG step:** Insertion of a new node w through DPG process. The left hand side (lhs) shows the subgraphs necessary to perform the DPG step, leading to the outcomes on the right hand side. The labeled edges (by a or b) on the lhs are only necessary for the corresponding outcome on the rhs, with the same label. The *auxiliary node* s is created/removed when d_w is odd. All new nodes and edges that are created during the DPG step are colored green.

d_w is even, s is either present or not
: In this case the neighbours of w are picked from a matching M of size $d_w/2$; all edges in M are removed in the process (Figs. S4.1,S4.3a). The chosen matching M may include the edge connected with s (if s present), in which case, w connects to s as well (Fig S4.3b).

d_w is odd, s absent (s-creation)
: We first create the auxiliary node s. There are $d_w + 2$ possible ways of joining s to the graph. With probability $(d_w+2)^{-1}$, we pick a matching M of size $|M| = \lfloor d_w/2 \rfloor$ and the rest of times (with probability $(d_w + 1)/(d_w + 2)$) of size $|M| = \lceil d_w/2 \rceil$.

  – If $|M| = \lfloor d_w/2 \rfloor$, we join s to w, cut the edges in M and join w to all its $d_w - 1$ nodes (the case Fig S4.2a).

  – If $|M| = \lceil d_w/2 \rceil$, we select a node uniformly at random from M and join it with s, cut the edges in M then connect its remaining d_w nodes to w (the case in Fig. S4.2b).

d_w is odd, s present (s-removal)
: Select a matching M of size $|M| = \lceil d_w/2 \rceil$ that includes s in it. Cut all edges of M, remove s, then connect w to the remaining d_w nodes of M. This joins the neighbour of s with w (Fig. S4.4)

The pseudo-code for the DPG step is on the next page.

Algorithm 1 DPG

```
 1: procedure DPG($G, d_w, s$)   ▷ Adds new vertex $w$ with degree $d_w$ to $G$ where $s$ is the auxiliary vertex
 2:     if $d_w$ is even then
Require: A Matching $M$ of size $d_w/2$ that can include the edge with $s$ as well   ▷ Fig S4.1/S4.3
 3:         Add vertex $w$ to $G$
 4:         Remove edges in $M$
 5:         Add edges between $w$ and vertices in $M$   ▷ $s$ transfers to $w$ if $\exists(s,u) \in M$ (Fig S4.3b)
 6:     else [$d$ is odd]
 7:         if $s$ doesn't exists then
 8:             Pick random number $I = \{0,1\}$ with probabilty $P(I=0) = \frac{1}{d+2}, P(I=1) = \frac{d_w+1}{d_w+2}$
 9:             if $I = 0$ then
Require: A Matching $M$ of size $\lfloor d_w/2 \rfloor$
10:                 Add vertex $w$ to $G$
11:                 Add edges between $w$ and vertices in $M$
12:                 Create auxiliary vertex $s$
13:                 Add edge $(w,s)$   ▷ $s$-creation at $w$ (Fig S4.2a)
14:                 Remove edges in $M$
15:             else
Require: A Matching $M$ of size $\lceil d_w/2 \rceil$
16:                 Add vertex $w$ to $G$
17:                 Pick an random edge $(u,v) \in M$
18:                 Add edges between $w$ and vertices in $M - \{(u,w)\}$
19:                 Create a auxiliary vertex $s$
20:                 Add edges $(w,u)$ and $(s,v)$   ▷ $s$-creation at $v$ (Fig S4.2b)
21:                 Remove edges in $M$
22:         else [$s$ exists]
Require: A Matching $M$ of size $\lceil d_w/2 \rceil$ that includes the edge $(s,u)$ with $s$
23:             Add vertex $w$ to $G$
24:             Add edges between $w$ and vertices in $M - \{(s,u)\}$
25:             Add edge $(w,u)$
26:             Remove $s$ from $G$   ▷ $s$-removal (Fig S4.4)
27:             Remove edges in $M$
28:     if $s$ is present in $G$ then
29:         return $(G,s)$
30:     else [$s$ was not present or removed]
31:         return $(G,\varnothing)$
```

In the above pseudo code, failure to find a matching M of required size in the '**Require**' steps above implies that vertex w cannot be joined with G through the DPG process.

C. The Degree-preserving reduction step

For the reduction (inverse) DPG step, the goal is to remove a node w from the graph, while keeping the degrees of remaining nodes in the graph unchanged. We start by selecting a node w at random. According to Lemma S2, this requires that there are sufficient non-edges between the neighbors of w, and if that is not the case then w cannot be DP removed from the graph. Otherwise, the specific operations depend on the degree d_w and the existence of the auxiliary node s that is outlined below.

The following description uses the notations $S, N(S), \Gamma(S), \overline{\Gamma}(S)$ introduced in section II A as: the set of nodes, its neighbors, the subgraph induced by S in G and the complementary graph of $\Gamma(S)$ in G, respectively.

d_w is even	Find a perfect matching M in $\overline{\Gamma}(N(\{w\}))$ ($N(\{w\})$ may include s). If such M exists, remove w along with its edges and then add the edges of M to the graph.
d_w is odd	The DP reduction step depends on the absence/presence of the auxiliary node s. – If s is absent, we must add s to G. Then find a near perfect matching M in $\overline{\Gamma}(N(\{w\}))$. If such M exists, remove w along with its edges and add edges of M into G. Finally, join s with the node in $\overline{\Gamma}(N(\{w\}))$ that is left unmatched by M. – If s is present, then find a perfect matching M in $\overline{\Gamma}(N(\{w,s\}) \setminus \{w,s\})$. If such M exists, remove w along with its edges and add the edges of M into G. Finally, remove s from G.

Algorithm 2 DPR

```
1: procedure DPR($G, w, s$)          ▷ Removes vertex $w$ with degree $d_w$ from the graph $G$ with auxiliary vertex $s$
2:     if $d_w$ is even then          ▷ Inverse of Fig S4.1, S4.3
Require: A perfect matching $M$ in $\overline{\Gamma}(N(\{w\}))$, which can include $s$
3:         Add edges of $M$ into $G$
4:         Remove $w$ from $G$
5:     else [$d_w$ is odd]
6:         if $s$ is absent in $G$ then          ▷ Inverse of Fig S4.4
Require: A near perfect matching $M$ in $\overline{\Gamma}(N(\{w\}))$
7:             Add $s$ into $G$
8:             Join $s$ with the node left unmatched by $M$ in $\overline{\Gamma}(N(\{w\}))$
9:             Remove $w$ from $G$
10:            Add edges of $M$ into $G$
11:        else [$s$ is present in $G$]          ▷ Inverse of Fig S4.2
Require: A perfect matching $M$ in $\overline{\Gamma}(N(\{w,s\}) \setminus \{w,s\})$
12:            Remove $w$ and $s$ from $G$
13:            Add edges of $M$ into $G$
14:    if $s$ is present in $G$ then
15:        return $(G, s)$
16:    else [$s$ was not present or was removed]
17:        return $(G, \varnothing)$
```

In the above pseudo code, failure to find a matching M of required size in the '**Require**' steps above, implies that vertex w is not DP reducible.

D. Configurational DPG and tuning

In the main text we discuss the configurational version of the DPG, in which case we want to create graphical realizations of a set of degrees D using the DPG process. Here, the choice of the initial graph G_0 for DPG is important, as it has to be a simple graph and its set of degrees D_0 must be a proper subset of D. We want G_0 with the smallest N_0 number of nodes with degrees $D_0 \subseteq D$. There can be many possibilities for G_0, which may include the empty graph (in the case of D containing degrees of 1). The choice depends on the degree values in D; for example, if $(2, 2, 2, 4, \ldots)$ are the sorted degrees in D then G_0 is K_3.

After defining the initial graph G_0 with degrees $D_0 \subseteq D$, we need to add all the nodes with degrees $D \setminus D_0$, through the DPG process. As discussed in the main text, we can do that either via the iconf-DPG process or the gconf-DPG process. In all such DPG steps, the degree d_w of the new node should obey the constraint $d_w \leq 2\nu(G)$ (or $d_w \leq 2\nu(G) + 1$ if auxiliary node is present). Here $\nu(G)$ is the matching number of G at that particular stage, which may include an edge to the auxiliary node. If the latter condition cannot be satisfied, that run cannot finish and must be started from the beginning. Note, there can be sets of degrees D that are graphical (see subsection I B) but non-DPG graphical, that is, not realizable by a DPG process.

If the DPG process doesn't need to tune network properties (other than degree), then we can just use the non-weighted matching finding algorithm described in subsection III A to find us a matching of the needed size. However, if we need to tune higher-order network properties, the matching is chosen based on edge weight information. We are still choosing independent edges, but we also enforce a criterion that the weights of those edges need to obey, formulated as an optimization function (that is maximum or minimum total weight), in order to be selected into the matching. There are several exact and approximate algorithms to find such weighted matching (outlined in Ref. [19]).

Here we use a simple greedy algorithm, similar to that for maximal matching described in subsection III A: pick the edge with max/min weight, add it to M then remove it along with its adjacent edges before choosing another edge with max/min weight, etc. Although this greedy method does not select the most optimal set of weighted independent edges, it produces sufficiently good approximations for network properties that we needed.

E. Algorithms for DPG models

In the previous section we presented the inner components of the DPG algorithm, which referred to the steps taken once an incoming degree d_w and a corresponding matching M were chosen. Below, in Algorithm 3 we present the outer components, corresponding to the different models discussed in the main text. Other choices can easily be added, by modeller's preference. Algorithm 3 builds a DPG graph with N nodes, from an initial graph G_0, adding new nodes with degrees determined by the function d_w, while finding matching for DPG considering edge weights given by the function W. Note, here d_w and the weight distribution W are represented as functions. Different functions d_w and W are used for different DPG models. Table S1 outlines how functions d_w and W are chosen corresponding to the DPG models discussed in main text. Refer to section on IV for details on seed graph and weighted matching.

Algorithm 3 DPG_Model($G_0, N, d_w, W, < params >$)

```
1: $(G, s) \leftarrow (G_0, \varnothing)$
2: while vertex count of $G < N$ do
3:     $d \leftarrow d_w(G, < params >)$
4:     $(G, s) \leftarrow DPG(G, s, d)$ with matching weighted by function $W$
5: return $G$
```

DPG Model	d_w	W
Maximum	$d_w(G) = 2\nu(G)$	1
Linear	$d_w(G, c) = \lceil 2c\nu(G) \rceil, c \in (0, 1]$	1
Rand-fraction	$d_w(G) = R, P(R = r) = 1/(2\nu(G)), R \in [1, 2\nu(G)]$	1
Scale-Free	$d_w(G, \gamma) = R, P(R = r) = r^{-\gamma}/\sum r^{-\gamma}, R \in [1, 2\nu(G)]$	1
i-config	$d_w(G, D) = n^{th}$ element in sorted degrees D n = vertex count in G	model dependent
g-config	$d_w(G, D) =$ max DPG-feasible $d \in D$ that hasn't been added to G	model dependent
Randic	same as i-config	$W(e(u, v), \alpha) = (d_u d_v)^\alpha$
Paritioning	same as i-config	$W(G, V_0, e_0) =$ Edge betweeness centrality of edge e_0 in induced subgraph of G with seed nodes V_0

TABLE S1. **Functions in DPG models:** Function d_w determines the degree of the next node being added through DPG whereas the function W assigns weights to the edges for matching. $W = 1$ corresponds to the unweighted matching. Symbols $e(u, v), d_u, V_0$ represent an edge connecting node u with v, degree of node u and nodes in initial graph G_0, respectively.

IV. REAL-WORLD NETWORKS

Id	Graph Name	Node Count	Edge Count	Density	$min(n_s)$	$< \phi >$	Details
1	EconomicTrans.	496	41668	0.33943	482	0.9740	US economic transactions in 1972 – industries x industries [14]
2	DiseaseGene	1777	7491	0.00475	939	0.5322	Network of disorders and disease genes linked by known disordersand gene associations [21]
3	EcosystemWetFL	128	2075	0.25529	36	0.4417	Food web in south Florida during wet season [16][43]
4	EcosystemDryFL	128	2106	0.25910	37	0.4283	Food web in south Florida during dry season [16][43]
5	CoAuthNetworkSci	379	914	0.01276	65	0.1770	Co-authorship network in network science (2006) [37]
6	CoAuthGRQC	4158	13422	0.00155	507	0.1259	Co-authorship network in general relativity and quantum cosmology (1993-2003) [32][33]
7	CoAuthAstrophysics	14845	119652	0.00109	1839	0.1249	Co-authorship network in Astrophysics (1995-1999) [36]
8	HumanContact	410	2765	0.03298	31	0.0903	Face-to-face contact network active for at least 20 seconds during a conference [29][22]
9	NeuralCElegans	297	2148	0.04887	10	0.0781	Network representing the neural network of C. Elegans [45][15]
10	CoAuthCondMatter	13861	44619	0.00046	978	0.0732	Co-authorship network in Condensed Matter (1995-1999) [46]
11	WordRelations	146005	656999	0.00006	8383	0.0581	Lexical network of words from the WordNet dataset [20]
12	FBCompanyPages	14113	52126	0.00052	490	0.0363	Facebook page (2017) networks of companies. Edges are mutual likes among them [33][41]
13	EnronEmails	33696	180811	0.00032	1160	0.0350	Enron email communication network covers all the email communication in Enron [33] [26] [34]
14	GoogleInternalWeb	15763	148585	0.00120	479	0.0321	Hyperlink network from pages within Google's own sites, i.e., on google.com [29] [39]
15	AmazonCopurchase	334863	925872	0.00002	10030	0.0305	Co-purchase network in Amazon between products that have been bought together [29] [47]
16	CoAuthHighEnergy	5835	13815	0.00081	139	0.0251	Co-authorship network in High-Energy Theory (1995-1999) [36]
17	FBNotreDame	12149	541336	0.00734	82	0.0174	Facebook friends network of students in University of Notre Dame [42]
18	ProteinInteraction	1870	2203	0.00126	22	0.0125	Protein interaction network of yeast [24]
19	MetMPneumoniae	411	926	0.01099	0	0.0123	Metabolic cellular network data for Mycoplasma Pneumoniae [25]
20	PolitcalBlogs	1490	16715	0.01507	0	0.0110	Political blogs network in during 2004 US election [17]
21	FBPublicFigures	11565	67038	0.00100	93	0.0087	Facebook page (2017) networks of Public Figures.Edges are mutual likes among them [33][41]
22	MetNGonorrhoeae	1055	2531	0.00455	6	0.0087	Metabolic cellular network data for Neisseria Gonorrhoeae [25]
23	MetSTyphi	2982	7117	0.00160	11	0.0053	Whole cellular network data for Salmonella Typhi [25]
24	MetMTuberculosis	1520	3655	0.00317	0	0.0034	Metabolic cellular network data for Mycobacterium Tuberculosis [25]
25	MetEColi	2275	5627	0.00218	5	0.0032	Metabolic cellular network data for Escherichia Coli [25]
26	WikiVoteNetwork	7115	100762	0.00398	3	0.0029	Network of Wikipedia admins and users active in the voting for new admins [33][31][30]
27	WorldAirports	2939	15677	0.00363	5	0.0027	The network contains flights between airports of the world [29][38]
28	BitcoinOTC	5875	21489	0.00125	12	0.0025	Who-trusts-whom network of people trading on Bitcoin OTC [33][27][28]
29	PowergridUS	4941	6594	0.00054	0	0.0015	The power grid network of the western states of the United States [45][29]
30	Twitch	7126	35324	0.00139	0	0.0011	Networks of gamers who stream in english language in Twitch [33][40]
31	HighwaysUS	126146	161950	0.00002	85	0.0010	Continental US road network [13]
32	RoadsTX	1379917	1921660	2.02E-6	813	0.0006	Road network of Texas. [33][34]
33	FBWallPosts	46952	183412	0.00017	20	0.0005	Network of posts to other user's wall on Facebook [29][44]
34	LinuxSourcecode	30837	213217	0.00045	0	0.0004	Dependency network of Linux 3.16 source code[29]
35	GithubSocial	37700	289003	0.00041	3	0.0002	Social network of GitHub (2019) developers with at least 10 repositories [40]
36	InternetTopology	34761	107720	0.00018	0	0.0001	Network of connections between autonomous systems of the Internet. [29][48]

TABLE S2. **Real-world networks reduced using the DP reduction process, shown in Fig. 2 of the main text:** Entries are sorted by average fraction $\langle\phi\rangle = \langle n_f\rangle / n_s$ of final nodes n_f remaining after inverse-DPG on network with starting n_s nodes. The table also outlines the minimum (found in the 50 runs) starting node count n_f from where we can grow the exact structure of these networks, though a forward DPG process. In particular, seven of these real-world networks can be grown using DPG starting from an empty graph.

V. SUPPLEMENTARY FIGURES

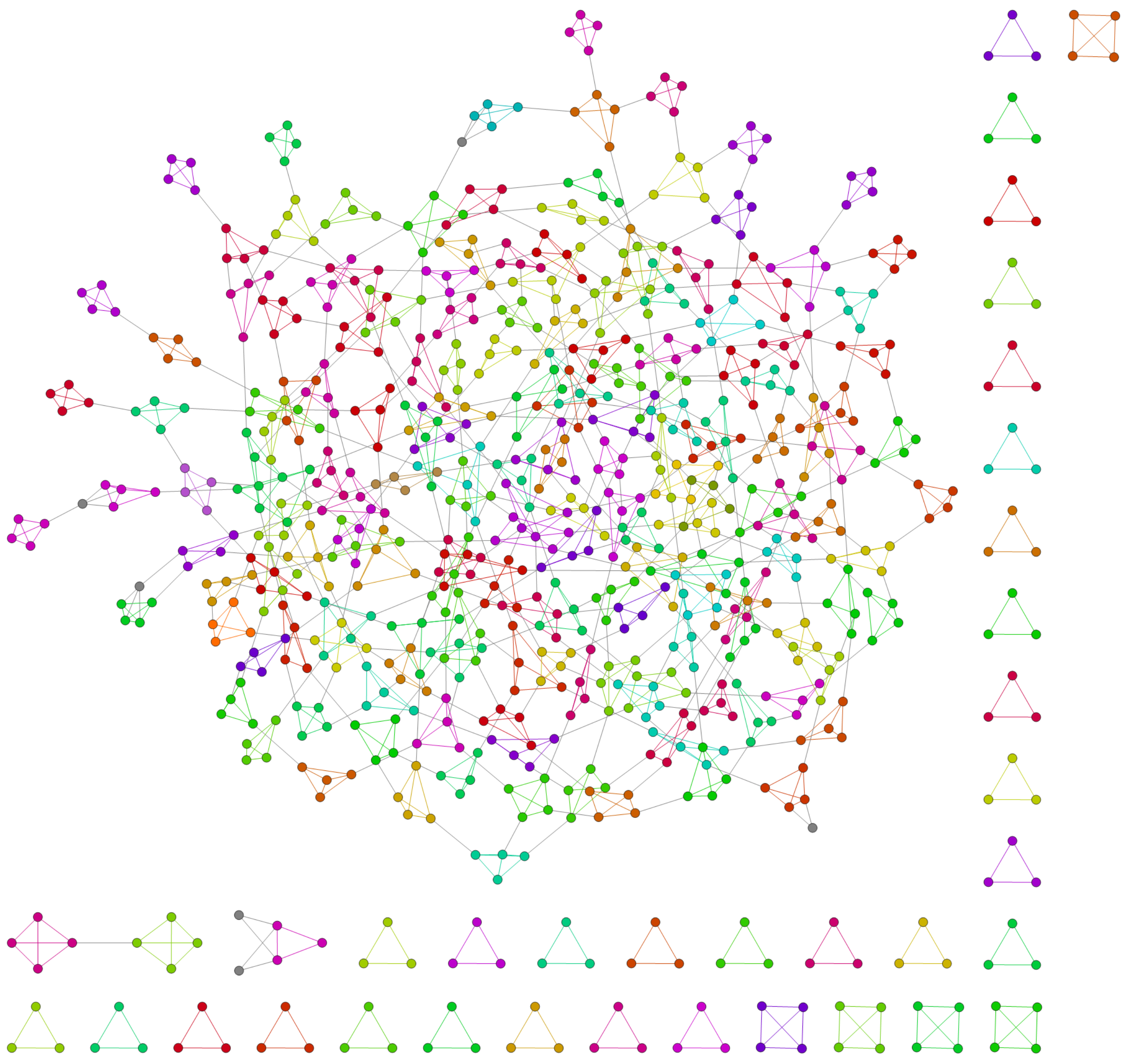

FIG. S5. **Sample** G_f of the Road network of Texas with 1.38 million nodes (intersections/endpoints) can be grown from this much smaller network with 813 nodes using DPG. Network was obtained using a run of the inverse-DPG process. Nodes and edges in cliques larger than 2 are drawn using the same (non-gray) color.

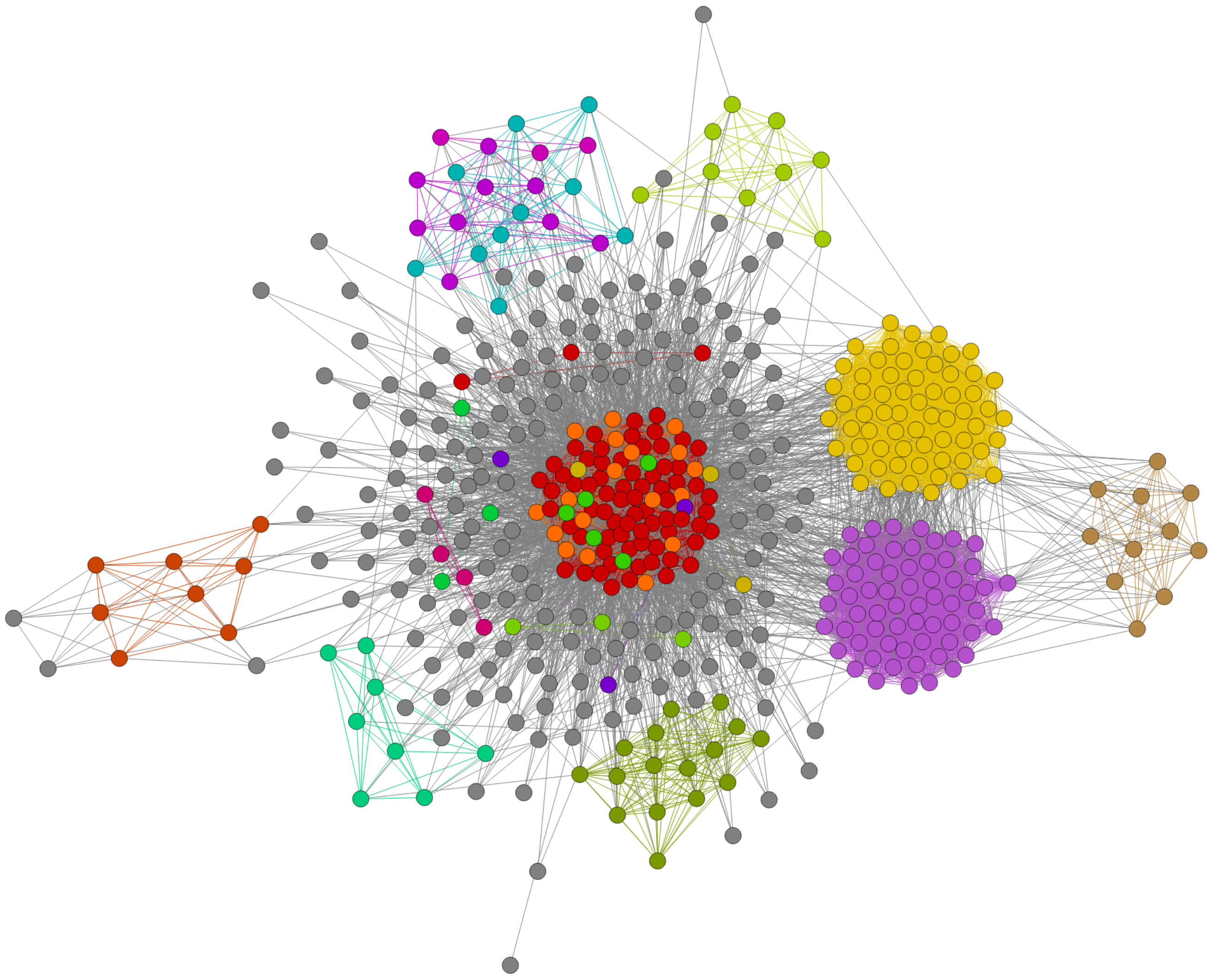

FIG. S6. **Sample** G_f using the inverse-DPG process of the hyperlink network of 15,763 pages within domain google.com. One can grow the original network from this much smaller one with 479 nodes using forward DPG. Nodes and edges in cliques larger than 2 are drawn using the same (non-gray) color.

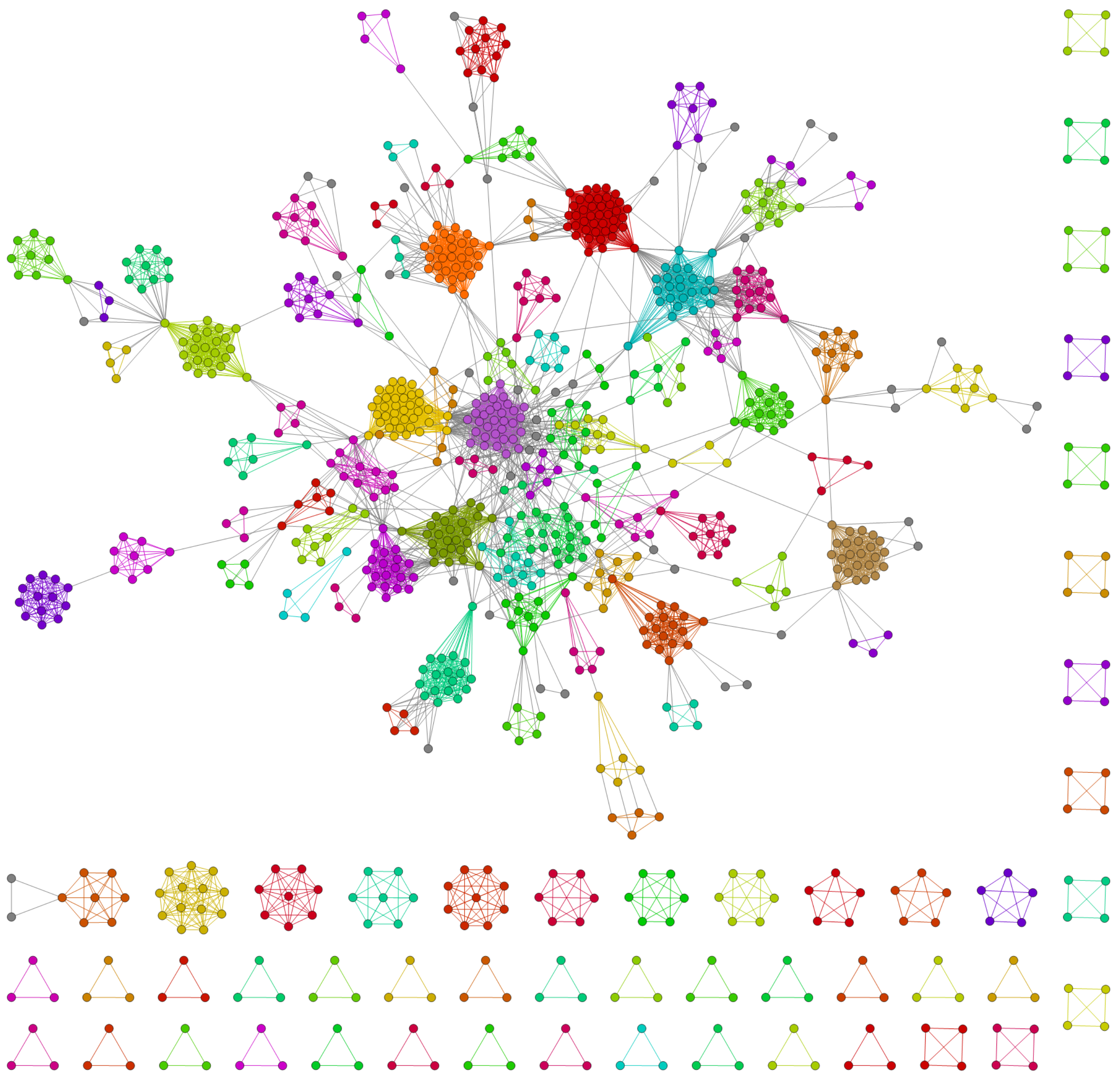

FIG. S7. **Sample** G_f of the co-authorship network with 4,158 authors in general relativity and quantum cosmology. The original network can be grown from this G_f with 507 nodes using DPG. Nodes and edges in cliques larger than 2 are drawn using the same (non-gray) color.

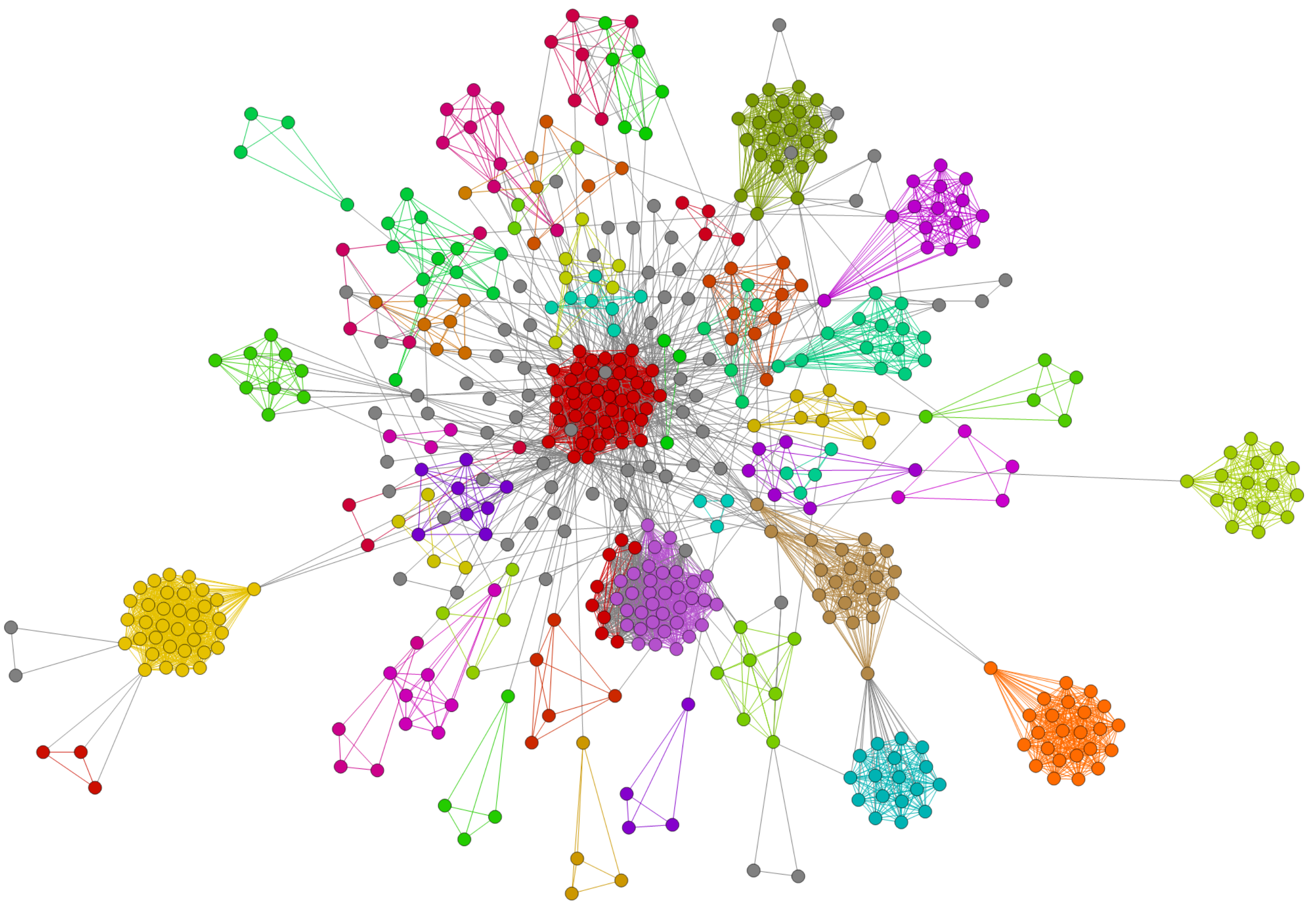

FIG. S8. **Sample** G_f of the disease-gene network (of 1,777 disorders and disease genes linked by known disorders and gene associations) obtained with a typical inverse-DPG run. The original network can be grown from this one with 939 nodes using DPG. Nodes and edges in cliques larger than 2 are drawn using the same (non-gray) color.

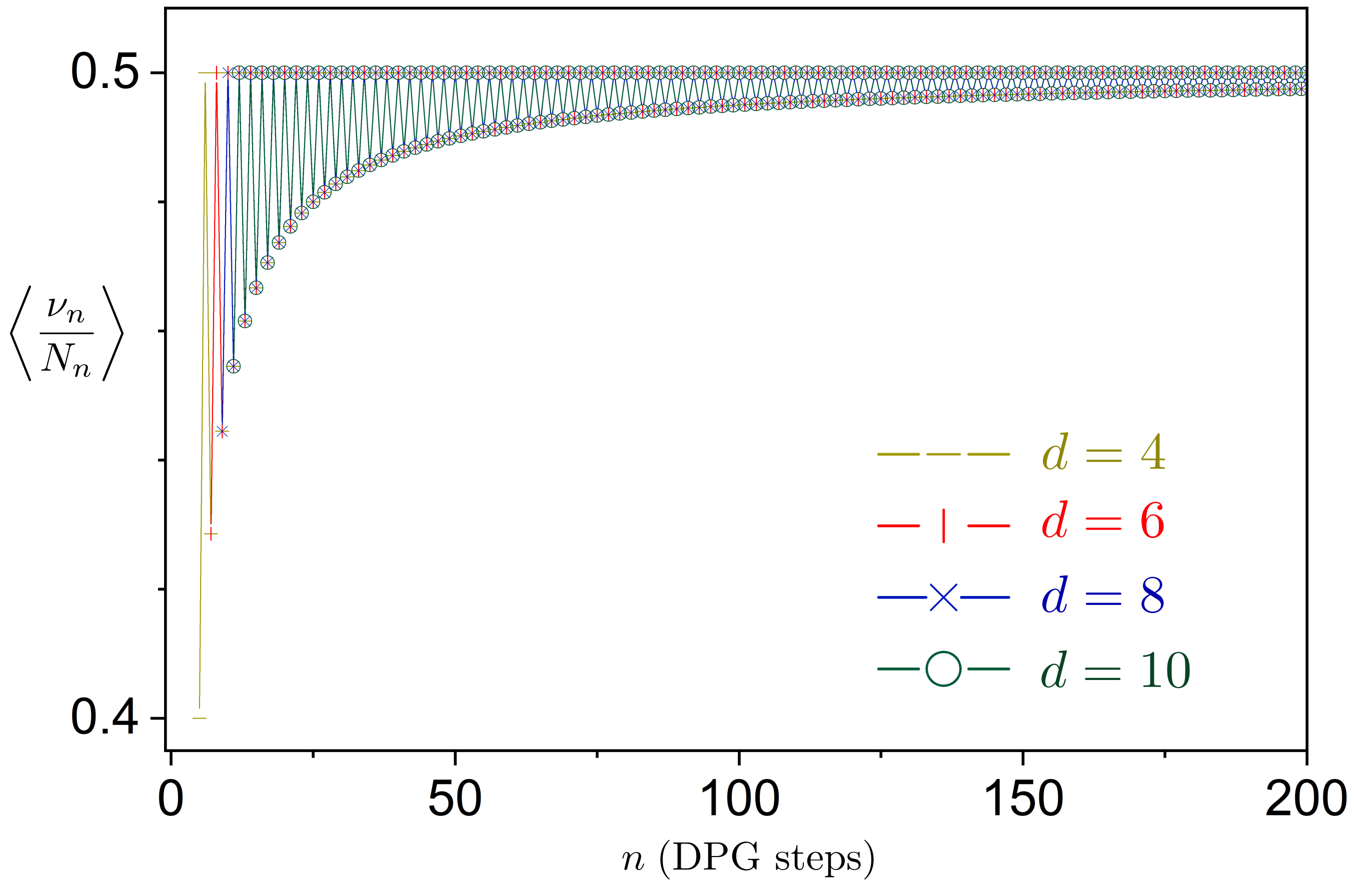

FIG. S9. **Evolution of the matching numbers for random regular DPG graphs.** Ratio of maximum matching size and the number of nodes in the graph ($N_n = n + N_0$) during a random regular DPG process (with fixed degree d), averages over 100 runs. Initial graph for all DPG runs is $G_0 = K_{d+1}$, hence $N_0 = d + 1$. The maximum matching in all cases here is either perfect matching or near perfect matching.

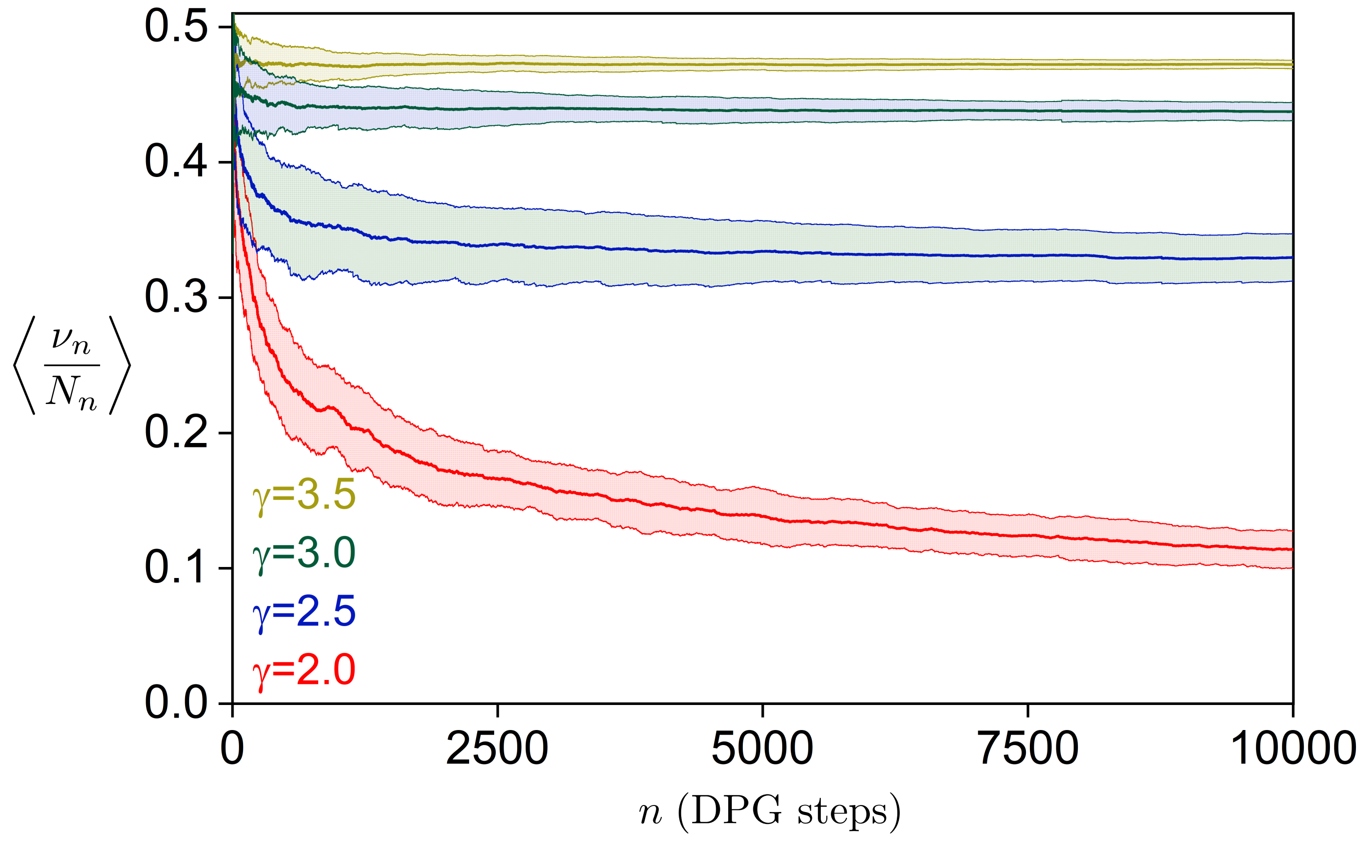

FIG. S10. **Evolution of the matching numbers for scale-free DPG graphs.** Ratio of maximum matching size and the number of nodes in the graph ($N_n = n + N_0$) during scale-free DPG process for different γ-s, each averaged over 100 runs. Shaded region show one standard deviation and the center line is the mean of the data. Initial graph is $G_0 = K_2$, so $N_0 = 2$.

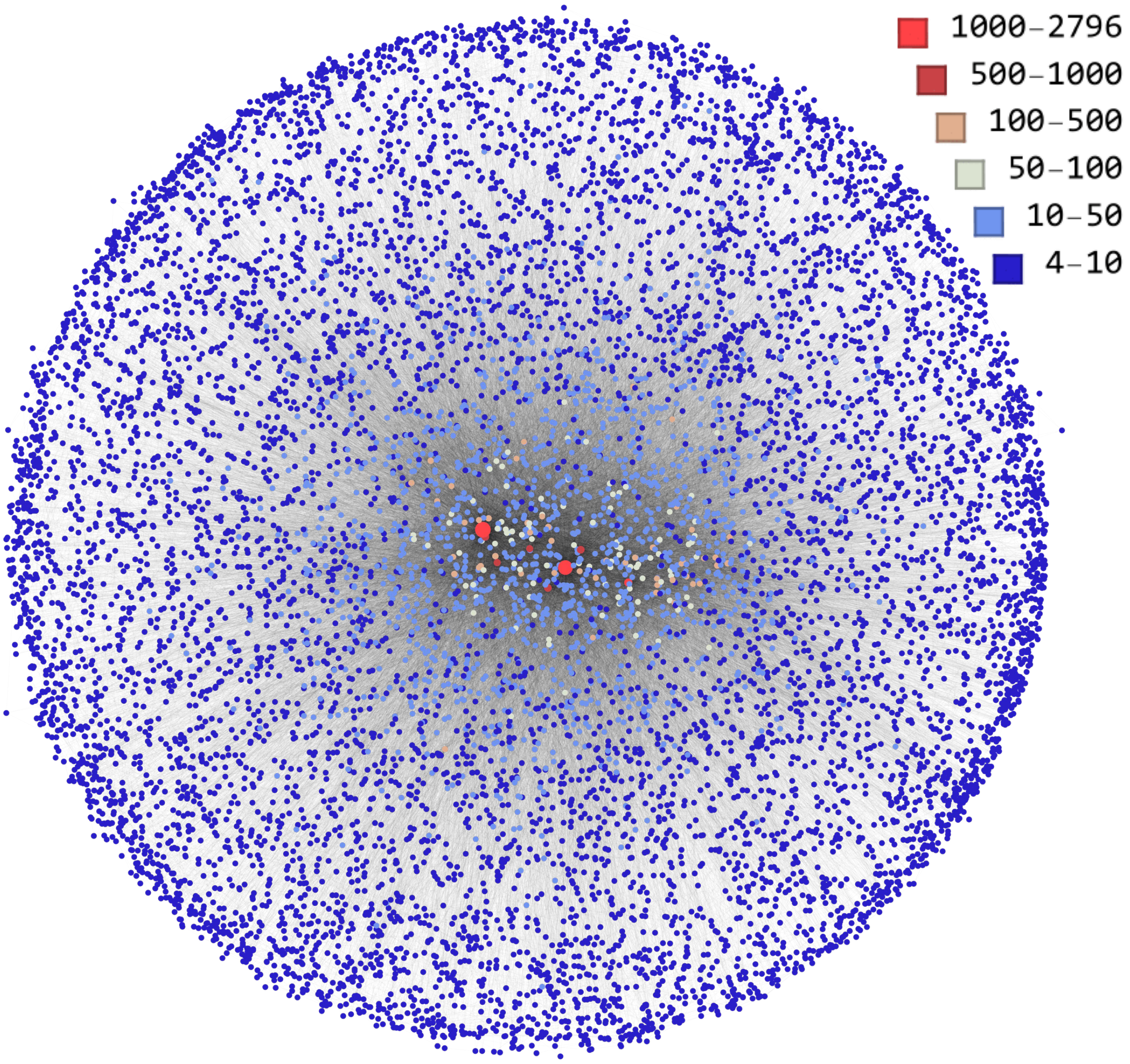

FIG. S11. **A scale-free DPG network** with $\gamma = 2.5, d_{min} = 4$ and with 10^4 added vertices, started from $G_0 = K_5$. The color legend indicates vertex degree ranges.

[1] Mertzios, George B., and Walter Unger. 'The friendship problem on graphs.' *Journal of multiple-valued logic and soft computing* 27.2-3 (2016): 275-285.

[2] Posa, Lajos. 'A theorem concerning Hamilton lines.' *Magyar Tud. Akad. Mat. Kutató Int. Közl* 7 (1962): 225-226.

[3] Erdős P, Gallai T (1960) Gráfok előírt fokszámú pontokkal. *Matematikai Lapok* 11(4):264–274.

[4] Tripathi A., Vijay S. 'A note on a theorem of Erdős & Gallai.' *Discrete Mathematics* 265.1-3 (2003): 417-420.

[5] Vizing V.G. On an estimate of the chromatic class of a p-graph. *Diskret. Analiz.* **3**, 25–30 (1964)

[6] Hammer P.L., Simeone B. The splittance of a graph. *Combinatorica* **1**(3), 275–284 (1981)

[7] Barrus MD, Hartke SG, Jao KF, West DB (2012) Length thresholds for graphic lists given fixed largest and smallest entries and bounded gaps. *Discr Math* 312:1494–1501.

[8] Biedl T, Demaine ED, Duncan CA, Fleischer R and Kobourov SG (2004) Tight bounds on maximal and maximum matchings. *Discr. Math.* 285:7–15.

[9] Berge C (1957) Two theorems in graph theory. *Proc. Natl. Acad. Sci.* 43(9):842–844.

[10] Lovász L and Plummer MD (2009) *Matching Theory* (American Mathematical Society)

[11] Edmonds J (1965) Paths, trees, and flowers. *Canad. J. Math.* 17:449–467.

[12] Siek JG, Lee L-Q, and Lumsdaine A (2001) *The Boost Graph Library: User Guide and Reference Manual* (Addison-Wesley Professional, PAP/CDR edition).

[13] Gleich/usroads-48 sparse matrix. `https://www.cise.ufl.edu/research/sparse/matrices/Gleich/usroads-48`. (Accessed on 08/05/2020).

[14] Matrix mbeaflw. `https://math.nist.gov/MatrixMarket/data/Harwell-Boeing/econiea/mbeaflw.html`. (Accessed on 08/05/2020).

[15] Network data. `http://www-personal.umich.edu/~mejn/netdata/`. (Accessed on 08/05/2020).

[16] Pajek datasets. `http://vlado.fmf.uni-lj.si/pub/networks/data/`. (Accessed on 08/05/2020).

[17] Adamic, L. A., and Glance, N. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery* (2005), pp. 36–43.

[18] Dezső, B., Jüttner, A., and Kovács, P. Lemon–an open source c++ graph template library. *Electronic Notes in Theoretical Computer Science 264*, 5 (2011), 23–45.

[19] Duan, R., and Pettie, S. Linear-time approximation for maximum weight matching. *Journal of the ACM (JACM) 61*, 1 (2014), 1–23.

[20] Fellbaum, C., et al. Wordnet: An electronic lexical database mit press. *Cambridge, Massachusetts* (1998).

[21] Goh, K.-I., Cusick, M. E., Valle, D., Childs, B., Vidal, M., and Barabási, A.-L. The human disease network. *Proceedings of the National Academy of Sciences 104*, 21 (2007), 8685–8690.

[22] Isella, L., Stehlé, J., Barrat, A., Cattuto, C., Pinton, J.-F., and Van den Broeck, W. What's in a crowd? analysis of face-to-face behavioral networks. *Journal of theoretical biology 271*, 1 (2011), 166–180.

[23] Israeli, A., and Itai, A. A fast and simple randomized parallel algorithm for maximal matching. *Information Processing Letters 22*, 2 (1986), 77–80.

[24] Jeong, H., Mason, S. P., Barabási, A.-L., and Oltvai, Z. N. Lethality and centrality in protein networks. *Nature 411*, 6833 (2001), 41–42.

[25] Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., and Barabási, A.-L. The large-scale organization of metabolic networks. *Nature 407*, 6804 (2000), 651–654.

[26] Klimt, B., and Yang, Y. Introducing the enron corpus. In *CEAS* (2004).

[27] Kumar, S., Hooi, B., Makhija, D., Kumar, M., and Faloutsos, C. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (2018), ACM, pp. 333–341.

[28] Kumar, S., Spezzano, F., Subrahmanian, V., and Faloutsos, C. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on* (2016), IEEE, pp. 221–230.

[29] Kunegis, J. Konect: the koblenz network collection. In *Proceedings of the 22nd International Conference on World Wide Web* (2013), pp. 1343–1350.

[30] Leskovec, J., Huttenlocher, D., and Kleinberg, J. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web* (2010), pp. 641–650.

[31] Leskovec, J., Huttenlocher, D., and Kleinberg, J. Signed networks in social media. In *Proceedings of the SIGCHI conference on human factors in computing systems* (2010), pp. 1361–1370.

[32] Leskovec, J., Kleinberg, J., and Faloutsos, C. Graph evolution: Densification and shrinking diameters. *ACM transactions on Knowledge Discovery from Data (TKDD) 1*, 1 (2007), 2–es.

[33] Leskovec, J., and Krevl, A. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[34] Leskovec, J., Lang, K. J., Dasgupta, A., and Mahoney, M. W. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics 6*, 1 (2009), 29–123.

[35] Micali, S., and Vazirani, V. V. An o (v— v— c— e—) algoithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science (sfcs 1980)* (1980), IEEE, pp. 17–27.

[36] Newman, M. E. The structure of scientific collaboration networks. *Proceedings of the national academy of sciences 98*, 2 (2001), 404–409.

[37] Newman, M. E. Finding community structure in networks using the eigenvectors of matrices. *Physical review E 74*, 3 (2006), 036104.

[38] Opsahl, T., Agneessens, F., and Skvoretz, J. Node centrality in weighted networks: Generalizing degree and shortest paths. *Social networks 32*, 3 (2010), 245–251.

[39] Palla, G., Farkas, I. J., Pollner, P., Derényi, I., and Vicsek, T. Directed network modules. *New journal of physics 9*, 6 (2007), 186.

[40] Rozemberczki, B., Allen, C., and Sarkar, R. Multi-scale attributed node embedding. *arXiv preprint arXiv:1909.13021* (2019).

[41] Rozemberczki, B., Davies, R., Sarkar, R., and Sutton, C. Gemsec: Graph embedding with self clustering. In *Proceedings of the 2019 IEEE/ACM international conference on advances in social networks analysis and mining* (2019), pp. 65–72.

[42] Traud, A. L., Mucha, P. J., and Porter, M. A. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications 391*, 16 (2012), 4165–4180.

[43] Ulanowicz, R. E., and DeAngelis, D. L. Network analysis of trophic dynamics in south florida ecosystems. *US Geological Survey Program on the South Florida Ecosystem* (1999), 114.

[44] Viswanath, B., Mislove, A., Cha, M., and Gummadi, K. P. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks* (2009), pp. 37–42.

[45] Watts, D. J., and Strogatz, S. H. Collective dynamics of 'small-world'networks. *nature 393*, 6684 (1998), 440–442.

[46] White, J. G., Southgate, E., Thomson, J. N., and Brenner, S. The structure of the nervous system of the nematode caenorhabditis elegans. *Philos Trans R Soc Lond B Biol Sci 314*, 1165 (1986), 1–340.

[47] Yang, J., and Leskovec, J. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems 42*, 1 (2015), 181–213.

[48] Zhang, B., Liu, R., Massey, D., and Zhang, L. Collecting the internet as-level topology. *ACM SIGCOMM Computer Communication Review 35*, 1 (2005), 53–61.