Neighborhood Structure Configuration Models

Anonymous Author(s)*

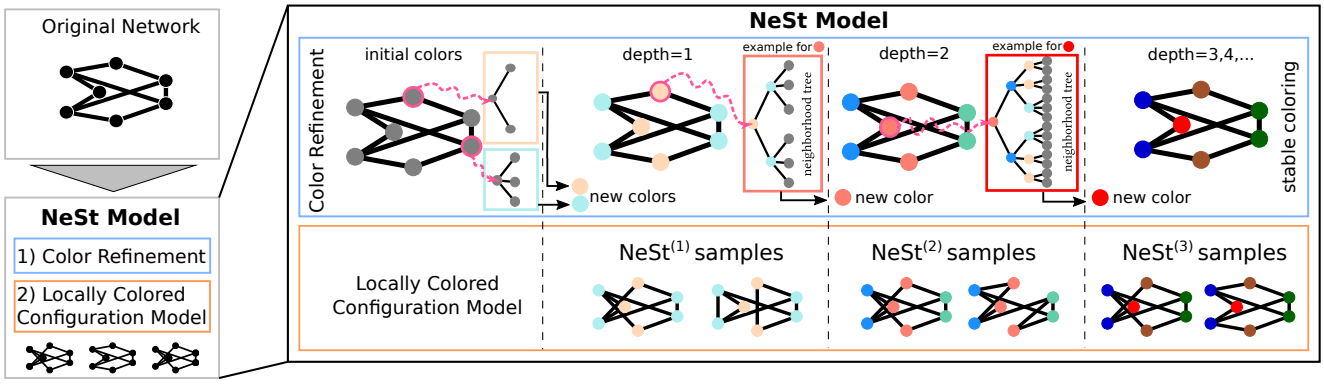


Figure 1: Network sampling with the NeSt model. Starting from an original network we use the Color Refinement (CR) algorithm to extract node colors for different depths d (blue box). At each depth d, nodes with an isomorphic neighborhood tree of depth d are assigned the same color. We then sample graphs via the locally Colored Configuration Model such that the CR colors of the original network are preserved. This ultimately preserves neighborhood trees as well.

ABSTRACT

We develop a new method to efficiently sample synthetic networks that preserve the d-hop neighborhood structure of a given network for any given d. The proposed algorithm trades off the diversity in network samples against the depth of the neighborhood structure that is preserved. Our key innovation is to employ a colored Configuration Model with colors derived from iterations of the so-called Color Refinement algorithm. We prove that with increasing iterations the preserved structural information increases: the generated synthetic networks and the original network become more and more similar, and are eventually indistinguishable in terms of centrality measures such as PageRank, HITS, Katz centrality and eigenvector centrality. Our work enables to efficiently generate samples with a precisely controlled similarity to the original network, especially for large networks.

1 INTRODUCTION

Sampling networks with predefined properties is a fundamental problem in network science. For instance, in order to assess whether a specific network statistic is significant, we need to compare its value against a baseline value, i.e., a reference statistic computed from a "typical" network drawn from some null model. While a plethora of models exist, many models suffer from one of the following deficiencies: On the one hand there exist null models that are efficient to sample from, but merely preserve local information about nodes such as their degrees or simple global network statistics such as the overall link density (examples include Erdos-Renyi random graph models, or configuration models). Yet, in many applications preserving the meso-scale neighborhood structure of nodes is necessary for null models to resemble a given network more closely. On the other hand, null models that can preserve certain mesoscale structures and motifs, such as exponential random graph model, are notoriously difficult to fit to data and difficult to sample, especially for larger networks.

In this paper, we address this research gap by introducing the NeSt models that both preserve mesoscale network structure and can be efficiently sampled. More specifically, our models preserves the neighborhood trees around each node up to a specific depth d, and even for large networks are easy to fit, can be efficiently sampled, and well-approximates a given network on a range of centrality measures. We achieve this by combining the so-called Color Refinement or Weisfeiler-Lehman algorithm [35] (an approximate graph isomorphism test) with a locally Colored Configuration Model [34]. With the depth parameter d, we can tune how deep the multi-hop neighborhood structure of each node in the original network is preserved in the null model (See Fig. 1 for an illustration). Ultimately, certain spectral properties of the original network are *exactly* preserved in samples from our network model. We demonstrate the utility of NeSt by generating null networks which better and better preserve 'spectral' centrality measures such as PageRank, Katz centrality, HITS, and eigenvector centrality with increasing depth.

Contributions. We introduce a new class of network null models, the $\text{NeSt}_G^d(c^{(0)})$ model, whose samples mimic the original network G in its neighborhood tree structure up to depth d with starting colors $c^{(0)}$. We further present an algorithm that allows efficient Markov Chain Monte Carlo sampling from the introduced model. We prove that NeSt samples exactly preserve popular centrality measures of G for an appropriate choice of $c^{(0)}$ and a large enough value of d. For lower values of d, we show that early convergence of the PageRank power iteration in the original network carries over to corresponding NeSt samples. Concluding, we illustrate empirically that similar low d convergence observations can be made across a range of centralities and real-world networks.

Network null models. There exists a broad range of network (null) models, including mechanistic models of network formation, models for growing networks, and many more. In this work we are concerned with *null models* for static, fixed networks described by (potentially) directed graphs. The purpose of these network models is typically to preserve relevant properties (observables) of empirically measured real-world data, while otherwise maximizing randomness [11, 12]. Depending on the chosen model, the desired network properties may either be preserved only in expectation, i.e., on average across the whole ensemble of generated networks or exactly in each sample. We provide a non-exhaustive list of some of the most commonly used null models in the following.

Erdős-Rényi type networks In Erdős-Rényi (ER) random graphs, edges exist with equal probability, which preserves network density on expectation. Inhomogenous random graphs [33] (IRG) generalize ER-random graphs by allowing varying edge probabilities. This enables the popular stochastic block model [18] (edge probability depends on group membership) and the Chung-Lu model [10] (degree distribution is preserved on expectation).

Configuration models In the configuration model [14, 25] (CM) the degree of each node is fixed in each sample. In the globally colored configuration model [26] we associate a color to each node and fix the total number of edges between colors. Alternatively, in locally-colored configuration models [34] we fix the colors of all nodes' neighbors.

Exponential Random Graph models (ERGMs) [23] are popular in the social sciences. They specify structures to be preserved in expectation by including them in an "energy term" of a Gibbs distribution from which networks are sampled. While the ERGM formulation is very flexible, fitting the parameters of the model is in general a difficult task [9], and sampling from these network models is often problematic [32].

Machine learning based network generators [17] are gaining popularity in scenarios where multiple samples from one family of networks (e.g. enzymes) are available. Deep learning methods like Variational Auto Encoders or Generative Adversarial Networks can then be employed to learn the distribution of this family of networks. While these methods can learn arbitrary distributions, they are not easily employed when learning from large graphs.

Outline. We start by introducing some prerequisites related to centrality measures and the color refinement algorithm. Subsequently, we introduce the NeSt model, discuss how we can sample from this model and investigate its mathematical properties. Finally, we show empirically how certain network centrality measures converge to those of the original network, even before the exact neighborhood tree structure of the original network is preserved (i.e., the final colors in the CR algorithm are reached). We conclude the paper by highlighting limitations and potential further impact of our work.

2 PRELIMINARIES AND NOTATION

Graphs. A *graph* or *network* $G=(V,E)$ consists of a set of nodes V and edges $E\subseteq\{uv\,|\,u,v\in V\}$. We always assume $V=\{1,\dots,n\}$, thus graphs are *labeled* and have an *adjacency matrix* $A\in\{0,1\}^{n\times n}$ with $A_{i,j}=1$ if $ij\in E$ and 0 otherwise. We use parenthesis to distinguish matrix powers (A^k) from matrices indexed by superscripts ($A^{(k)}$). A graph G is *undirected* if $uv\in E\Leftrightarrow vu\in E$, otherwise G is *directed*. For directed graphs the *in/out-neighborhood* is $N_{\text{in/out}}(v)=\{x\,|\,xv/vx\in E\}$ while for undirected graphs the *neighborhood* $N(v)=N_{\text{in}}(v)=N_{\text{out}}(v)$. The degree deg and in/out-degree $\deg_{\text{in/out}}$ is the cardinality of the respective neighborhood sets.

Colorings. A *graph coloring* is a function $c:V\to\{1,\dots,k\}$ that assigns each node one out of $k\in\mathbb{N}$ colors. Each coloring induces a partition C of the nodes into equivalence *classes* of equal color $C_i=\{v\in V\,|\,c(v)=i\}$. Given colorings c_1,c_2, we say c_1 refines c_2, denoted by $c_1\sqsubseteq c_2$, if $c_1(v)=c_1(u)\Rightarrow c_2(v)=c_2(u)$. Similarly, if $c_1\sqsubseteq c_2$ and $c_2\sqsubseteq c_1$, c_1 and c_2 are *equivalent*.

Centrality measures. Centrality measures assign importance scores to nodes such that important nodes have high centrality values. In this work, we mostly consider *eigenvector-based centralities* $\Gamma_{\mathbf{X}}$, which compute the centrality scores of the nodes as the dominant eigenvector w of certain matrices M_X, i.e.

$$w,\ \text{where } M_X w=\lambda_{\max}w \text{ and } \lambda_{\max}=\operatorname*{argmax}_{\lambda_i\in\operatorname{spec}(M_X)}|\lambda_i|$$

This is ill-defined when there are multiple dominant eigenvectors. We use a definition that ensures a unique centrality and agrees with the above if the dominant eigenvector is unique:

$$\Gamma_{\mathbf{X}}=\lim_{m\to\infty}\frac{1}{m}\sum_{i=0}^{m}(|\lambda_{\max}|^{-1}M_X)^i\mathbb{1}$$

This ensures unique centralities even when the graph is *non primitive*. For *Eigenvector Centrality* [6], the relevant matrix is

$$M_{\mathbf{EV}}=A^\top. \qquad (\Gamma_{\mathbf{EV}})$$

Similarly, the well-studied *PageRank* [28] measure corresponds to the dominant eigenvector of the following matrix:

$$M_{\mathbf{PR}}=\alpha\bar{A}^\top D^{-1}+(1-\alpha)\frac{1}{n}\mathbb{1}_n\mathbb{1}_n^\top, \qquad (\Gamma_{\mathbf{PR}})$$

where $\alpha\in[0,1]$ is a so-called damping parameter and $\bar{A}$ is the adjacency matrix of an augmented graph in which we connect zero out-degree nodes to all nodes in the graph. Thus the diagonal matrix of out-degrees $D=\operatorname{diag}(\bar{A}\mathbb{1})$ is invertible.

HITS [21] assigns each node both a hub-score h_v and an authority score a_v. These are the dominant eigenvectors of:

$$M_{\mathbf{auth}}=A^\top A \qquad (\Gamma_{\mathbf{auth}}), \qquad M_{\mathbf{hub}}=AA^\top \qquad (\Gamma_{\mathbf{hub}})$$

Lastly, *Katz centrality*[19] is defined as:

$$\Gamma_{\mathbf{Katz}}=\sum_{k=0}^{\infty}\sum_{j=1}^{n}a^k(A^k)_{j,\_}=\sum_{k=0}^{\infty}a^k(A^k)^\top\mathbb{1} \qquad (\Gamma_{\mathbf{Katz}})$$

with $\frac{1}{a}>\max_{\lambda_i\in\operatorname{spec}(A)}|\lambda_i|$ being a parameter.

2.1 Color refinement

The color refinement algorithm (CR), also known as Weisfeiler Lehman algorithm and originally proposed in [35], is a simple and efficient algorithm that is frequently used in the context of the graph isomorphism problem. CR iteratively assigns colors to the nodes of the graph. Starting from an initial coloring $c^{(0)}$ the colors are updated by distinguishing nodes that have a different number of colored neighbors. CR stops once the color partition C no longer changes. The initial coloring is typically chosen as constant over all nodes, but can also incorporate prior knowledge about the nodes provided.

The exact CR procedure follows the iteration:

$$c^{(d+1)}(v) = \text{hash}\left(c^{(d)}(v),\ \{\!\{c^{(d)}(x) \mid x \in N(v)\}\!\}\right) \tag{1}$$

where the doubled brackets indicate a multi-set (a set in which an element can appear multiple times) and hash denotes some injective function mapping the pair onto a color space. This injectivity of the hash function implies that distinct inputs are assigned distinct colors. Since the injective hash function takes the previous color as the first argument, the colorings are iteratively refined, i.e. $c^{(d+1)} \sqsubseteq c^{(d)}$. As there can be at most n strict refinements $c^{(d+1)} \sqsubset c^{(d)}$, eventually the algorithm converges to a stable partition $c^{(d^*)} \equiv c^{(d^*+1)}$.

Once a stable partition $c^{(\infty)}$ is reached, the partition will not change. At this point, the nodes' colors induce an *equitable partition*, i.e., all nodes within one class have the same number of connections to another class. In fact, the CR algorithm converges to the *coarsest* equitable partition of the graph [29]. As an example consider the graph in Figure 1. The partition at depth 3 is stable. There all nodes of a specific color have the same number of colored neighbors as any other node of that color. In contrast, the partition at depth 2 is not stable. There the central red node has two blue neighbors while the top and bottom red nodes have one blue and one teal neighbor.

Though typically used for undirected graphs, the CR algorithm can be extended to directed graphs by replacing the neighborhood $N(v)$ in Eq. 1 with either the in- or the out-neighborhood. We refer to the resulting colorings as $c^{(d)}_{\text{in}}$ or $c^{(d)}_{\text{out}}$ respectively. We may further distinguish nodes by both their in- and out-neighborhood:

$$c^{(d+1)}_{\text{both}}(v) = \text{hash}\left(c^{(d)}_{\text{both}}(v),\ \{\!\{c^{(d)}_{\text{both}}(x) \mid x \in N_{\text{in}}(v)\}\!\},\ \{\!\{c^{(d)}_{\text{both}}(x) \mid x \in N_{\text{out}}(v)\}\!\}\right) \tag{2}$$

Note that after d iterations of the algorithm, the colors encode information about the d-hop (in-/out-)neighborhood of the nodes. To illustrate what we mean, once again consider Figure 1, where we highlight a few *neighborhood trees* with colored boxes. The neighborhood trees that correspond to the colors used at depth=1 are to the left of the depth=1 area. For instance, observe that the teal color corresponds to nodes that have three neighbors while the lightblue color corresponds to nodes that have two neighbors. Similar at depth two the lightred nodes correspond to nodes that have two neighbor both which have three neighbors. Overall, the color of a node directly encodes the structure of the neighborhood tree (up to a specific depth) in the sense that colors are the same if and only if their neighborhood trees are isomorphic [3].

For directed graphs, the in- or out-neighborhood are isomorphic depending on the employed CR variant, e.g., for $c^{(\infty)}_{\text{both}}$ both in- and out-neighborhood are isomorphic, whereas for $c^{(\infty)}_{\text{in}}$ only in-neighborhood trees of the nodes are isomorphic.

The CR algorithm has a worst-case runtime of $O((|E| + |V|) \cdot \log(|V|))$ [16]. However, we use a variant that has worst-case runtime $O(d \cdot |V| \cdot \deg_{\max} \cdot \log(\deg_{\max}))$, which is preferable on most real-world graphs for which typically $d \ll |V|$ and we often only care about colorings corresponding to small d.

3 THE NEST MODEL

In this section we introduce the **Ne**ighborhood **St**ructure Configuration model, short the NeSt model. This model preserves neighborhood structure information of an original network up to a specified depth d as encoded with the Color Refinement Algorithm. The locally Colored Configuration Model is then used to flexibly generate surrogate networks for a given network. Importantly, due to its design, the NeSt model is computationally easy to fit and sample.

For a given labeled graph G and initial node colors $c^{(0)}$, the set $\mathcal{N}^d_G(c^{(0)})$ contains all labeled graphs whose nodes have the same d-round CR colors as the original graph. The NeSt model is the uniform probability distribution over $\mathcal{N}^d_G(c^{(0)})$ for $d \in \mathbb{N}^+$. We think of initial colors $c^{(0)}$ and depth d as the models' hyper parameters, while the remaining parameters are learned from the graph G. Note that preserving the neighborhood tree structure at depth d also preserves the neighborhood structure at depth $d' \leq d$, which implies that the sets of possible networks generated by the NeSt model are nested.

Before embarking on a more detailed technical discussion, we state here several noteworthy properties of the NeSt model:

(1) $\mathcal{N}^{(1)}_G(\text{const})$ recovers the standard configuration model with degree sequence identical to G
(2) The graphs $G' \in \mathcal{N}^{(d)}_G(\text{const})$ and G are identically colored during the first d steps of the CR-algorithm
(3) The set $\mathcal{N}^{(d)}_G$ contains *all* (simple) graphs that agree with G on the first d steps of the employed CR-algorithm
(4) Structure preserved in $\mathcal{N}^{(d)}_G$ is also preserved in $\mathcal{N}^{(d+1)}_G$
(5) $\mathcal{N}^{(d)}_G(c_0) \subseteq \mathcal{N}^{(d)}_G(const)$

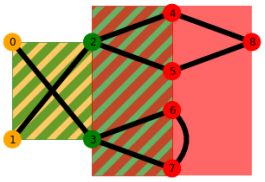
In the following we describe how we can efficiently sample from NeSt, and outline several variations to enrich the standard NeSt model and tailor it to specific scenarios.

3.1 Sampling from the NeSt model

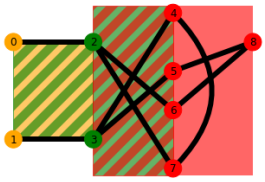
In this section we outline how we can sample efficiently from the NeSt model to generate networks which preserve the neighborhood structure of the original network up to a specified depth d.

To sample from the NeSt model we proceed as follows. First, we partition the edges of the initial graph according to the colors of their endpoints into disjoint subgraphs g_{C_i,C_j} with $V(g_{C_i,C_j}) = C_i \cup C_j$ and $E(g_{C_i,C_j}) = \{xy \mid x \in C_i, y \in C_j, xy \in E(G)\}$. As an example, all edges connecting green and red nodes are put into $g_{\text{green,red}}$ while edges connecting red to red nodes are put into $g_{\text{red,red}}$ (see fig. 2 for such a partition). In the case of directed networks, we distinguish $g_{\text{green,red}}$ and $g_{\text{red,green}}$ by the direction of the edge.

Second, after we have partitioned the edges into subgraphs, we can randomize each subgraph via edge swaps. In such an edge swap we choose two edges at random and swap their endpoints with one another. A few points in this context are worth remarking upon. For unicolored subgraphs all edge swaps that do not introduce multiedges are acceptable. The subgraphs can thus be rewired as in the normal configuration model. The subgraphs containing edges with endpoints of different color are bipartite. To ensure a consistent neighborhood preserving sampling we can thus only allow edge



(a) Original network



(b) NeSt rewired network

Figure 2: Illustration of the rewiring procedure employed when sampling from the NeSt$^{(2)}$ model. First, the edges are partitioned into subgraphs according to the colors of depth 1 of their endpoints. The striped regions with two colors (yellow-green, green-red) correspond to bipartite networks connecting nodes of different colors while the unicolored regions (red) connect same color nodes. Second, we rewire each subgraph (colored regions) while respecting bipartivity as required. Overall this procedure preserves the CR-colors at depth 2.

swaps that preserve this bipartite structure. These subgraphs are thus rewired as a bipartite configuration model.

For directed graphs a similar scheme can be used: The unicolored and two-color subgraphs are randomized according to the directed configuration model. Unlike in the undirected case, the bipartivity of the two-color subgraphs is preserved automatically, since the edges are directed from C_i to C_j, swapping endpoints always preserves the color at the end. For unicolored directed configuration models an additional triangle swap is required for correct rewiring (see section 8).

The entire procedure is detailed in Algorithm 1. We note that all edges in distinct subgraphs are independent, and thus the sampling can run in parallel for different subgraphs.

As outlined, the sampling procedure can be decomposed into drawing samples from well established configuration models. We can thus draw upon the established research on sampling from these models via MCMC using random edge switches [1, 13], which is the strategy we apply in this work, However, other strategies are also possible [8]. We assert the utility of the sampling procedure with the following theorem:

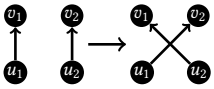
Theorem 3.1. *The sampling procedures shown in algorithm 1 and algorithm 2 sample uniformly at random from all labeled graphs in $\mathcal{N}_G^d(c^{(0)})$ for a given graph G, initial colors $c^{(0)}$ and depth d.*

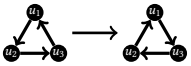
The proof can be found in the appendix (section 8). Note that the above result not only establishes the correctness of the algorithm but also that the samples drawn are *maximally random* in the sense that we draw uniformly from the space of all graphs that respect the desired neighborhood constraints. This also means the NeSt model is the maximum entropy distribution over the set $\mathcal{N}_G^d(c^{(0)})$.

We briefly comment on the intuition that edge swaps of this fashion do not affect the colors obtained by CR. To preserve the colors $c^{(d)}$ of a node from the original graph, it is sufficient to preserve the node color and the multiset of it's neighboring nodes' colors of the previous CR iteration ($c^{(d-1)}$). By performing the outlined edge swaps, the degree of each node within the subgraphs

Input: Graph G, initial colors $c^{(0)}$, depth d
Output: Sample Graph G' distributed as $\mathrm{NeSt}_G^d(c^{(0)})$

1 use CR to obtain depth $d-1$ colors $c^{(d-1)}$
2 partition edges ij of G into subgraphs $g_{C_i^{(d-1)},C_j^{(d-1)}}$
3 let $\mathcal{G}$ be a list of all those subgraphs
4 **for** *subgraph* $g \in \mathcal{G}$ **do**
5 **for** $r \cdot |E(g)|$ *steps* **do**
6 choose two edges u_1v_1 and u_2v_2 unif. at random from $E(g)$
7 **if** $|\{u_1, v_1, u_2, v_2\}| = 4$ **then**
8 **if** $u_1v_2 \notin E(g)$ *and* $u_2v_1 \notin E(g)$ **then**
9 remove u_1v_1, u_2v_2
10 add u_1v_2, u_2v_1
11 **if** $c^{(d-1)}(u_1) = c^{(d-1)}(v_1)$ **then**
12 randomly choose three nodes u_1, u_2, u_3
13 **if** u_1, u_2, u_3 *constitute a directed triangle* **then**
14 reverse direction of triangle u_1, u_2, u_3





15 return $G' = \bigcup_{g \in \mathcal{G}} g$

Algorithm 1: Sampling from $\mathrm{NeSt}_G^{(d)}(c^{(0)})$ using edge switches. Firstly the CR colors are computed and edges are partitioned into smaller subgraphs according to the colors of their endpoints (lines 1-3). We then visit each subgraph in turn and perform a number of switching attempts proportional to the number of edges in the subgraph (lines 4-5). For undirected graphs simple edge switches (lines 6-10) are sufficient to achieve randomization. For the *directed* case an additional directed triangle switch (lines 11-14) is required for uniform sampling (see the proof in section 8). This overall switching strategy (lines 6-14) is well established see [1, 13]. The runtime of the entire procedure is $O(r \cdot |E(G)| \cdot \deg_{\max})$ - disregarding the computation time needed for the CR algorithm. The loop in line 4 is independent making the bulk of the computation easy to parallelize.

stays the same. Thus for any color, the number of neighboring nodes with that color stays the same. Consequently, the multiset of neighboring nodes' colors stays the same. Note that preserving the local configuration of neighborhood colors of each node is *not* the same as preserving merely the total number of edges between color classes as done in other configuration models [26].

A different way to sample from the NeSt model is displayed in algorithm 2 which can be found in the appendix.

3.2 Variations of the NeSt model

To emphasize the flexibility of our proposed scheme we discuss some variations of the NeSt model in this section.

The initial node colors in the CR algorithm are a powerful way to incorporate additional constraints to the network ensemble. For concreteness, let us consider two simple examples.

- Assume the original network is bipartite and we want to maintain this property. In this case, one can choose the initial colors to reflect the bipartition.

- Assume a network comprises several connected components we want to keep separated. In this case, we can choose the initial node colors to reflect the components.

A more elaborate use of the initial colors would be the following. Assume we want to preserve the early steps of the PageRank power iteration. One possible strategy to achieve this is to color our graph with the out-degree of the nodes and then perform the CR algorithm using the in-neighborhood. In fact, this idea is formalized in lemma 4.2.

Incorporating externally defined node colors If external node colors are available, we can use them to initialize the CR algorithm. This implies that the external colors are used throughout the CR process, i.e., the external-colors of nodes at any depth in the neighborhood are preserved. This can be a strong restriction on the set of possible graphs in the NeSt models. An alternative, less restrictive way to introduce external colors is to use them as function arguments in later iterations of the CR algorithm. As an example consider injecting the external colors at iteration $d-1$ when sampling from $\mathrm{NeSt}^{(d)}$. In this case only direct node neighbors are additionally identified by their external color while two-hop neighbors are not identified by their external color.

Samples in between depth d and $d+1$ For certain graphs, the cardinality of the set of possible surrogate samples can shrink drastically when moving from depth d to depth $d+1$. In those cases, it can be useful to preserve more neighborhood structure than depth d but less than depth $d+1$. To illustrate how this can be achieved, observe that the CR algorithm as described above, can alternatively be understood as a two step procedure: 1) each node 'pushes' its depth d color to all its neighbors and itself; 2) each node uses the so collected multi-set of colors to create the round $d+1$ color. To retain the neighborhood structure in between d and $d+1$, one can adjust the first pass in this alternative view of CR to only use all those nodes belonging to a selected (e.g., random) subset of depth d colors. This has the effect that only parts of the neighborhood tree are being expanded, while other parts remain at the previous depth. By employing the above procedure, we obtain colors $c^{(d_*)}$ with $c^{(d)} \sqsupseteq c^{(d_*)} \sqsupseteq c^{(d+1)}$. Note: The second refinement relation is not strict only if *all* nodes associated with the selected depth d colors are 'pushing'.

4 THE ROLE OF THE DEPTH PARAMETER d

4.1 Maximum depth preserves centralities exactly

The following theorem exemplifies how the (full) neighborhood information contained in the CR-induced node colors preserves many observables of a network. Specifically, we show how a range of centrality measures are preserved if appropriate choices are made for the color refinement procedure.

Theorem 4.1. *Let $G_1, G_2 \in \mathcal{N}_G^\infty(c^{(0)})$ be samples from the NeSt model with CR aggregating over the in-neighbors and let A_1, A_2 be their adjacency matrices. If two nodes $u, v \in V(G_1 \cup G_2)$ have the same color $c^{(\infty)}(u) = c^{(\infty)}(v)$, then:*

(1) $\Gamma_{\boldsymbol{Katz}}(u) = \Gamma_{\boldsymbol{Katz}}(v)$
(2) $\Gamma_{\boldsymbol{EV}}(u) = \Gamma_{\boldsymbol{EV}}(v)$.
(3) *If $c^{(0)} \sqsubseteq deg_{out}$, then $\Gamma_{\boldsymbol{PR}}(u) = \Gamma_{\boldsymbol{PR}}(v)$.*
(4) *If $c_{in}^{(\infty)}$ is computed for $A_i^\top A_i$, then $\Gamma_{\boldsymbol{auth}}(u) = \Gamma_{\boldsymbol{auth}}(v)$*
(5) *If instead CR aggregates over both in and out neighbors then $\Gamma_{\boldsymbol{auth}}(u) = \Gamma_{\boldsymbol{auth}}(v)$ and $\Gamma_{\boldsymbol{hub}}(u) = \Gamma_{\boldsymbol{hub}}(v)$*

We emphasize again that for unique centralities a corresponding *primitive* matrix is not required, see our centrality definition in section 2.

Theorem 4.1 shows, that the centrality of a node is completely determined by the node's stable color, i.e, the neighborhood tree encoded by the color implies the value of the centrality score — even for completely unrelated graphs. This means, that for sufficiently large d, many centrality measures of the original graph are preserved exactly in NeSt samples.

We remark that the HITS score factors in both A and $A^\top$ which makes it necessary to regard both in- and outgoing neighbors in the CR computation (see eq. (2)). The statement for PageRank has previously been established in [5].

The algebraic view of CR To prove this result, we do an established switch of our perspective from the combinatorial view of CR to an algebraic one. Consider the following partition indicator matrix whose columns indicate the color class $C_i = \{v \in V \mid c(v) = i\}$ a node belongs to:

$$H_{i,j}^{(d)} = \mathbb{I}\left[i \in C_j^{(d)}\right] = \begin{cases} 1 & \text{if } i \in C_j^{(d)} \\ 0 & \text{else} \end{cases} \tag{3}$$

This indicator matrix H can be used to count the number of neighbors of color class i for node u by simply multiplying H with the adjacency matrix as follows:

$$\left(A^\top H^{(d)}\right)_{u,i} = \sum_{v \in N_{\text{in}}(u)} \mathbb{I}\left[v \in C_i^{(d-1)}\right] \tag{4}$$

Once a stable coloring is reached, the partition described by H is equitable which means each node from the same color class has the same number of colored neighbors. Thus the rows of the matrix $AH^{(\infty)}$ are the same for all nodes of the same color class. This allows us the express this matrix as

$$AH^{(\infty)} = H^{(\infty)} A^\pi, \tag{5}$$

where $A^\pi = (H^\top H)^{-1} H^\top A H$ is the adjacency matrix of the so-called *quotient graph*. We omit the superscript ∞ when referring to the indicator matrix H of the equitable partition.

The above considerations can be generalized for directed graphs, in which case the neighbourhood has to be replaced with either the out- or the in-neighbourhood:

$$AH_{out} = H_{out} A_{out}^\pi \quad \text{or} \quad A^\top H_{in} = H_{in} A_{in}^\pi$$

4.2 Intermediate d approximates centralities

In the previous section, we showed that it is possible to preserve enough structure in samples from the NeSt model to keep centralities like PageRank invariant. However for some purposes, it may suffice to only approximately preserve centrality scores while allowing for a richer set of possible network samples. In the following, we thus consider cases in which preserving smaller neighborhood depths is already sufficient. With PageRank as a running example, we show that convergence guarantees in the power iteration can be converted into approximation guarantees for samples drawn from

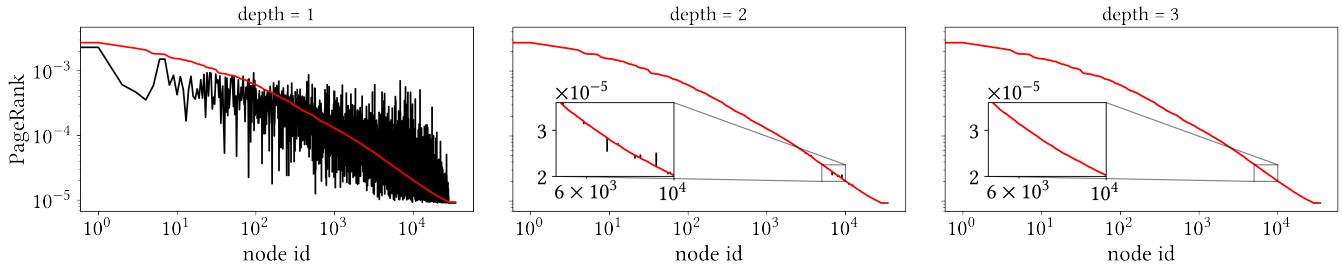


Figure 3: PageRank distribution of the NeSt$^{(d)}$(deg$_{\text{out}}$) model for different values of the depth d. We show the distribution of PageRank for the HepPh-network (red) and one sample from the fitted NeSt$^{(d)}$(deg$_{\text{out}}$) model (black). We can see that for a depth of one (left) the PageRank score of the original network varies a lot compared with the PageRank of the sampled network. With increasing depth this error decreases and almost vanishes (max error $< 10^{-16}$) for depth 3.

the NeSt model. Remember the PageRank iteration for a graph G with adjacency matrix A is defined as

$$x^{(t+1)}(x^{(0)}) = \alpha \bar{A}^\top D_{out}^{-1} x^{(t)}(x^{(0)}) + \frac{1-\alpha}{n}\mathbb{1}_n$$

when starting from starting vector $x^{(0)}$ with no negative entries. Remember $\bar{A}$ is an augmented adjacency matrix. The explicit dependency on $x^{(0)}$ is omitted when it is clear from context.

The following result formalizes the consequences for samples from the NeSt model for PageRank centrality.

Lemma 4.2. *Let G be a graph and let $\tilde{G}$ be sampled from in-NeSt^d_G(deg$_{out}$). Then the first d terms in the PageRank power iteration agree for G and $\tilde{G}$:*

$$x^{(t)} = \tilde{x}^{(t)} \quad \forall t \leq d$$

Where $x^{(t)}$ is a power iteration for G and $\tilde{x}^{(t)}$ be defined accordingly for $\tilde{G}$. Both power iterations start with $x^{(0)} = \tilde{x}^{(0)} = const \cdot \mathbb{1}_n$.

The proof of this lemma (found in the appendix) also implies an equivalent result for $x^{(t+1)} = A^\top x^{(t)}$ used when computing eigenvector centrality, and can be adapted for any power iteration. Notice that if d is large enough such that the coloring is stable, the lemma implies statement (3) in theorem 4.1.

The algebraic view for intermediate depth Towards a proof of Lemma 4.2, we extend the previously established algebraic view of CR to intermediate colors. That is, we derive statements akin to eq. (5) for the intermediate colors. As noted in [20], nodes that have the same rows in $A^\top H^{(d)}$ are in the same color class $C_i^{(d+1)}$ and vice versa. We can thus establish the following connection between consecutive iterations of CR:

$$A^\top H^{(d)} = H^{(d+1)} X_{d+1}^{\pi} \tag{6}$$

where X_{d+1}^{π} reflects the relationships between the colors at depth d and $d+1$. Now, because G and $\tilde{G}$ sampled from in-NeSt$^d(c_0)$ have identical colorings for $t \leq d$, thus they share the same $H^{(t)}$ and X_t^{π} matrices. A and $\tilde{A}$ are thus related

$$A^\top H^{(t-1)} = H^{(t)} X_t^{\pi} = \tilde{A}^\top H^{(t-1)}$$

for all $t \leq d$. These observations carry over to directed graphs.

Guaranteed similarity in PageRank We now extend the result that an original graph and samples from in-NeSt^d_G(deg$_{\text{out}}$) are constrained in their first power iterations (Lemma 4.2) to the finding, that the final PageRank values of both graphs cannot be arbitrarily apart. We achieve this by combining Lemma 4.2 with convergence guarantees of the PageRank power iteration. We can thus be sure that two samples from the NeSt model are bound to have centralities that are no further apart than the following:

Lemma 4.3. *Let x and $\tilde{x}$ be the PageRank vectors of two graphs sampled from in-NeSt^d_G(deg$_{out}$). Let α be the PageRank damping factor used. It holds that:*

$$\|x - \tilde{x}\|_1 \leq 2\alpha^{d+1}$$

Informally, the probability mass of the PageRank vector, for which the iterates of the original and the synthetic network do not yet agree upon, is of magnitude at most α^{d+1}. Hence, the final PageRank vectors are at most twice this magnitude apart. For a proof, see the appendix.

These theoretical considerations provide only worst-case bounds which are typically not tight for many (real-world) graphs — as one can see by considering the case that for regular graphs an equitable partition can already be reached at $d = 1$. Then Lemma 4.3 yields a bound of ≈ 1.4 (using typical α=0.85), while we know from Theorem 4.1 that the actual difference is 0. The next bound provides better guarantees in these cases.

Convergence of iteration implies similarity in PageRank In many real-world networks, the PageRank iteration converges faster than the worst case considered in lemma 4.3 (compare eq. (8) in the appendix). We thus establish a second bound which relates the convergence in one network to a guaranteed similarity in PageRank.

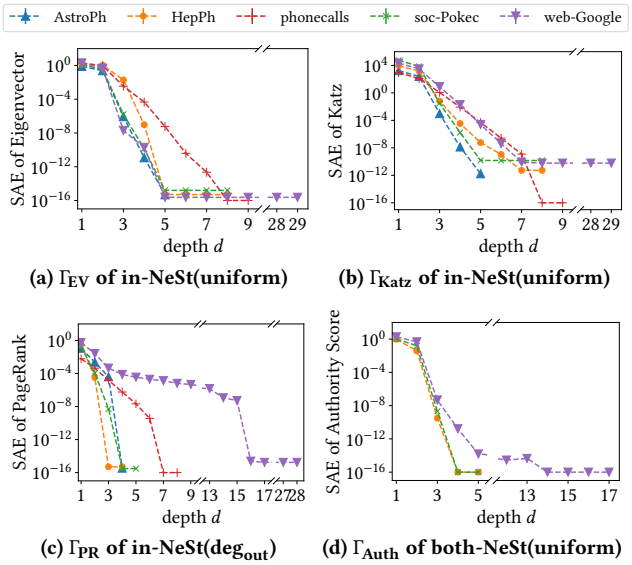
Corollary 4.4. *With assumptions as in Lemmas 4.2 and 4.3:*

$$\|x - \tilde{x}\|_1 \leq \frac{2}{1-\alpha}\|x^{(k-1)} - x^{(k)}\|_1$$

Colloquially speaking, Corollary 4.4 states that if the PageRank iterations in the original network converge quickly, then the PageRank vectors of synthetic and original are not far apart. For a proof see appendix.

5 EMPIRICAL ILLUSTRATION

We augment our theoretical considerations with an empirical evaluation on a variety of real-world networks (both directed and undirected) from different domains. We include the citation network HepPh, the web graph web-Google, the social network soc-Pokec, the collaboration network AstroPh (all previous are from [22]), and a network of phonecalls [2]. For details on the networks see Table 1.



(a) Γ_{EV} of in-NeSt(uniform) (b) Γ_{Katz} of in-NeSt(uniform)

(c) Γ_{PR} of in-NeSt($\deg_{out}$) (d) Γ_{Auth} of both-NeSt(uniform)

Figure 4: Centralities are better and better preserved with increasing depth. We show the sum absolute error (SAE) for a sampled network in relation to the original network. Points are medians with 16%/84% quantile error bars for 100 samples. Values are capped below by 10^{-16}. Legend is on top. From left to right increasingly deeper neighborhood trees are preserved which leads to centrality measures being better and better preserved.

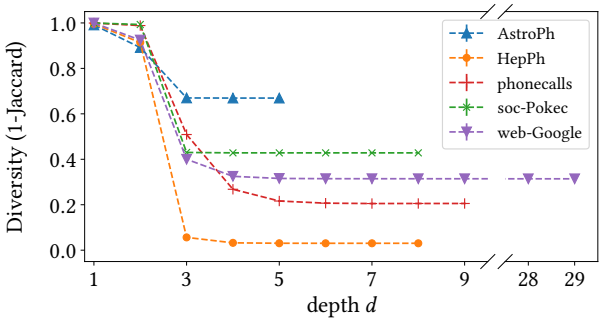


Figure 5: Diversity in NeSt(const) samples. We measure Diversity as one minus the Jaccard-Similarity of the edge sets of the original and sampled network. For a diversity of 1 the original and sampled network share no edges, for a diversity of 0 the networks agree perfectly. We see that for most networks, except for the HepPh network, there is still quite some diversity in the network samples. That is despite those networks better and better preserving centrality measures compare fig. 4.

We compute the CR colors using the indicated aggregation strategy and starting colors. We then generate samples from the NeSt model for each of the centralities, and compute the centrality measures for the sampled networks. Centralities are computed using a suitable iterative method until the SAE between subsequent iterations falls below a convergence threshold of 10^{-15}. As starting vector we chose the normalized (1-norm) vector for PageRank and the normal all-ones vector for other centralities.

We exemplify detailed convergence results for the PageRank distribution on the HepPh-network in Figure 3. To increase visibility, nodes are sorted in descending order by their original PageRank score. In the left plot at depth $d = 1$ (one step of CR starting from out-degree colors), we see that there is quite a large difference in PageRank. While the maximum absolute error (MAE) is below 0.002, the relative error can be an order of magnitude. In the middle plot ($d = 2$), the sampled network closely approximates the true PageRank with the MAE dropping three orders of magnitude. In the rightmost plot ($d = 3$) the sample reflects the PageRank almost perfectly (MAE drops by a factor of 10^{-10}). This shows, that meso scale properties are indeed relevant well recover the PageRank centrality in network samples.

Convergence results for other networks are summarized in Figure 4. Here we no longer show the individual distributions but the sum of absolute error (SAE) of the centrality of a sample in comparison to the original network averaged over 100 samples. The first two plots (Figures 4a and 4b) show eigenvector centrality and Katz centrality for the NeSt model which preserves in-neighborhood trees starting from uniform colors. The third plot shows PageRank for the NeSt which preserves in-neighborhood trees starting from a coloring that reflects the out degree. The last plot (Figure 4d) shows the Authorities (HITS) score for the NeSt which preserves both the in- and out-neighborhood trees starting from uniform colors.

As expected from out theoretical consideration, for sufficient depth d the sampled networks and original network are identical in their centrality scores. We further find that for smaller d networks already start to become more similar to the original network in terms of their centralities, which we proved for PageRank only. We finally note, to achieve significant reduction in SAE preserving neighborhood trees of depth of at least three seems necessary.

Better preserved neighborhood structure usually means diminished diversity in network samples, we show the extend of this in Figure 5. We see, that for depth 1 and 2 the diversity is usually not diminished much, while for higher depth the diversity goes down for all networks settling at some final diversity. This can be pretty high for networks like AstroPh or low as in the case of HepPh.

Finally we compare the NeSt model with other random graph models for the task of generating networks with similar PageRank as an original network. We compare the different models on three different scales (compare Figure 6) that highlight different dimensions of the task: SAE is the main objective measuring similarity in terms of their PageRank, Jaccard similarity measures similarity of the edge sets of the sample to the original network and is a proxy for sample diversity and lastly rewire time measures how long it takes for one sample to be generated. Lower is better on all scales. Shown are averages and their standard deviation obtained for 100 runs of the algorithms with different seeds. To have a fairer comparison of runtime, algorithm 2 was used to sample form NeSt.

The relatively fast ER model offers a large variety of network samples (low Jaccard) but does not well reflect the PageRank (high SAE). Comparing runtime of ER to ERGM or NeSt should be avoided because it is not an apples to apples comparison but included for completeness. The ERGM used (details see appendix) allows to trade Jaccard similarity for similarity in PageRank on the karate network

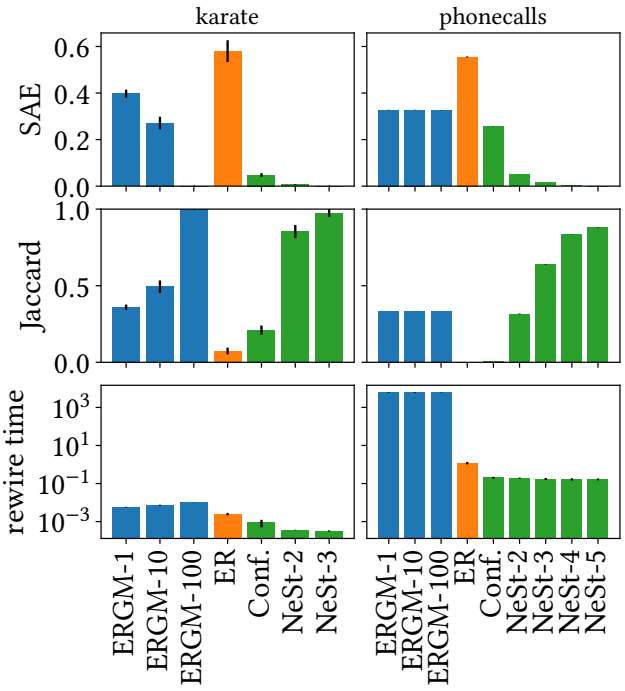


Figure 6: Comparison of the NeSt model with other random graph models. The left/right column correspond to the Karate Club/ phonecalls network with 34/ 36k nodes. The rows show sum absolute error (SAE) of the PageRank (top), Jaccard similarity (mid) and rewire time in seconds on a logarithmic scale (bottom). We used algorithm 2 with $2|E|$ steps. Lower is better on all scales. Besides NeSt we include Exponential Random Graph Models (ERGM), the Configuration model and the Erdős-Rényi (ER) network. On the small network (left) both ERGM and NeSt allow a trade-off of similarity for SAE but ERGM has poor runtime. On the larger network (right) we still have the same tradeoff while NeSt shows fast runtimes similar to Config./ER, but ERGMs are no longer feasable (runtime).

but fails to do so on the phonecalls network. Unfortunately, sampling from the ERGM is already impractical slow on the phonecalls network. The ERGM is slow because the PageRank needs to be recalculated for each step.

The NeSt model and the Configuration model (or alternatively $\text{NeSt}^{d=1}(\text{const})$) allow to trade SAE for Jaccard on both the small network and the larger phonecalls network while maintaining a reasonable sampling time. On the karate network, sampling NeSt models seems to become faster with increasing depth which goes against first intuition. But there are two effects at play: Firstly with increasing depth parts of the edges are frozen and are not considered for rewiring. Secondly, rewiring many smaller graphs (higher d) is faster than rewiring fewer bigger graphs (lower d).

Implementation details For the CR-algorithm we use an implementation suggested by Kersting et al. [20]. The code is available at https://github.com/SomeAuthor123/NestModel.

6 DISCUSSION

Preserving centralities As shown in Section 4.2, the NeSt models not only (approximately) preserve centrality measures, but also the first iterates (up to depth d) of a corresponding power iteration. Because we keep these early iterates invariant as well, NeSt is not sampling from *all* networks with the same centrality as the original one. Stated differently, there can be networks with different neighborhood trees that have the same centralities. Thus our model is not an attempt to exactly preserve centrality scores. In fact, we believe that the ability to maintain the neighborhood tree structure is more meaningful, as the network structure itself is the fundamental data we observe, whereas network statistics such as centrality measures are *derived* from the network structure.

Limiting the number of colors The CR algorithm can lead to color classes that contain just a single node, e.g., if there is a node with a unique degree. As a consequence, the connectivity pattern to that node is frozen in NeSt of larger depth. However in applications, it might be undesirable to distinguish nodes with 1000 and 1001 neighbors. Consequently it might not be worth to preserve neighborhood trees exactly but rather approximately. This may be achieved by employing a clustering algorithm rather than a hash function when deciding for new colors.

7 CONCLUSION

In this paper, we have introduced NeSt models which enable the creation of synthetic networks that preserve the neighborhood tree structure of nodes in a given network up to a specified depth. This allows to adjust the amount of structural information preserved in samples from null models to the task at hand. NeSt models thus represent a versatile generalization of existing configuration models which can only preserve the degrees of nodes. We demonstrate that NeSt models are efficient to fit through the Color Refinement Algorithm and easy to sample from, even for large networks.

While we illustrate the utility of preserving neighborhood structure by applying NeSt models for preserving centralities, the capabilities of the NeSt model extend to the preservation of many eigenvectors (e.g. the main eigenvalues [30] are preserved for sufficient d). This could open up NeSt models as possible candidate null models for other spectral properties of a given network. In fact, such spectral properties are important for a range of dynamical processes on networks such as (cluster) synchronization [31], consensus dynamics [27], or questions related to controllability [7].

Further, an interesting connection between NeSt models to message passing Graph Neural Networks (GNNs) exists: It has been shown that GNNs with d layers and uniform node initialization are no more expressive than the first d iterations of the CR-algorithm [3, 24, 36]. As all samples from the $\text{NeSt}^{(d)}$ model are identically colored during the first d CR iterations, they are thus indistinguishable by those GNNs. This opens up potential applications of the NeSt model and its variants for GNNs, e.g., to create (difficult) benchmark data sets. In summary, we believe that NeSt models can provide a versatile and efficient instrument for network scientists that aim to produce network null models that conserve the larger neighbourhood structure of networks.

REFERENCES

[1] Yael Artzy-Randrup and Lewi Stone. 2005. Generating uniformly distributed random networks. *Physical Review E* 72, 5 (2005), 056708.

[2] AL Barabási. [n.d.]. Networkscience book datasets. http://networksciencebook.com/resources/data.html Accessed: 2022-10-01.

[3] P Barceló, E Kostylev, M Monet, J Pérez, J Reutter, and JP Silva. [n.d.]. The logical expressiveness of graph neural networks. In *ICLR 2020*.

[4] Monica Bianchini, Marco Gori, and Franco Scarselli. 2005. Inside pagerank. *ACM Transactions on Internet Technology (TOIT)* (2005).

[5] P Boldi, V Lonati, M Santini, and S Vigna. 2006. Graph fibrations, graph isomorphism, and PageRank. *RAIRO-ITA* (2006).

[6] Phillip Bonacich. 1972. Technique for analyzing overlapping memberships. *Sociological methodology* 4 (1972), 176–185.

[7] DM Cardoso et al. 2007. Laplacian eigenvectors and eigenvalues and almost equitable partitions. *Jour. of combinatorics* (2007).

[8] CJ Carstens. 2015. Proof of unif. sampling of binary matrices with fixed row and column sums for the curveball algorithm. *Phys Rev E* (2015).

[9] S Chatterjee and P Diaconis. 2013. Estimating and understanding exponential random graph models. *The Annals of Statistics* (2013).

[10] Fan Chung and Linyuan Lu. 2002. The average distances in random graphs with given expected degrees. *PNAS* 99, 25 (2002), 15879–15882.

[11] G Cimini et al. 2019. The statistical physics of real-world networks. *Nature Reviews Physics* 1, 1 (2019), 58–71.

[12] Ton Coolen, Alessia Annibale, and Ekaterina Roberts. 2017. *Generating random networks and graphs*. Oxford university press.

[13] PL Erdos et al. 2019. The mixing time of the swap (switch) Markov chains: a unified approach. *arXiv preprint arXiv:1903.06600* (2019).

[14] B.K. Fosdick et al. 2018. Configuring random graph models with fixed degree sequences. *Siam Review* 60, 2 (2018), 315–355.

[15] David F Gleich. 2015. PageRank beyond the Web. *Siam Review* 57, 3 (2015), 321–363.

[16] M Grohe et al. 2014. Dimension reduction via colour refinement. In *European Symposium on Algorithms*. Springer, 505–516.

[17] Xiaojie Guo and Liang Zhao. 2020. A systematic survey on deep generative models for graph generation. *arXiv preprint arXiv:2007.06686* (2020).

[18] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. 1983. Stochastic blockmodels: First steps. *Social networks* 5, 2 (1983), 109–137.

[19] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (1953), 39–43.

[20] Kristian Kersting, Martin Mladenov, Roman Garnett, and Martin Grohe. 2014. Power iterated color refinement. In *28th AAAI*.

[21] Jon M Kleinberg et al. 1998. Authoritative sources in a hyperlinked environment.. In *SODA*, Vol. 98. Citeseer, 668–677.

[22] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[23] D Lusher et al. 2013. *Exponential random graph models for social networks: Theory, methods, and applications*. Cambridge Univ. P.

[24] Christopher Morris et al. 2019. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, Vol. 33. 4602–4609.

[25] MEJ Newman, SH Strogatz, and DJ Watts. 2001. Random graphs with arbitrary degree distributions and their applications. *Phys Rev E* (2001).

[26] Mark EJ Newman. 2003. Mixing patterns in networks. *Physical review E* 67, 2 (2003), 026126.

[27] N O'Clery et al. 2013. Observability and coarse graining of consensus dynamics through the external equitable partition. *Phys Rev E* (2013).

[28] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report.

[29] Robert Paige and Robert E Tarjan. 1987. Three partition refinement algorithms. *SIAM J. Comput.* 16, 6 (1987), 973–989.

[30] Peter Rowlinson. 2007. The main eigenvalues of a graph: a survey. *Applicable Analysis and Discrete Mathematics* (2007), 455–471.

[31] Michael T Schaub et al. 2016. Graph partitions and cluster synchronization in networks of oscillators. *Chaos* 26, 9 (2016), 094821.

[32] Tom AB Snijders et al. 2002. Markov chain Monte Carlo estimation of exponential random graph models. *Journal of Social Structure* (2002).

[33] Bo Söderberg. 2002. General formalism for inhomogeneous random graphs. *Physical Review E* 66, 6 (2002), 066121.

[34] Bo Söderberg. 2003. Random graphs with hidden color. *Physical Review E* 68, 1 (2003), 015102.

[35] Boris Weisfeiler and Andrei Leman. 1968. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series* (1968).

[36] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How Powerful are Graph Neural Networks?. In *ICLR*.

8 APPENDIX

8.1 Data on the networks used

Name	directed	EV	#nodes	#edges
Karate	✗	✓	34	78
AstroPh	✗	✓	18.772	198.110
phonecalls	✗	✓	36.595	45.680
HepPh	✓	✓	34.546	421.578
web-Google	✓	✓	875.713	5.105.039
soc-Pokec	✓	✓	1.632.803	30.622.564

Table 1: Statistics of the real world networks used. EV indicates whether the dominant eigenvector is unique.

8.2 Alternative way to sample NeSt

Instead of rewiring each subgraph independent from another as in algorithm 1 we are performing a randomisation one flip at a time. For each flip we choose a new graph g in which we perform the flip. This makes sampling more similar to other MCMC algorithms such as those used when sampling from ERGM models. The random choice of the subgraph g in line 5 doesn't matter as long as all graphs have a non zero chance to be selected and that selection chance doesn't change throughout execution. But the choice could matter for the mixing time of the Markov chain i.e. the number of steps required to achieve approximately uniform sampling.

1 Input/Output and lines 1-3 are the same as in algorithm 1
2 **for** *a number of steps* **do**
3 randomly choose a subgraph g from $\mathcal{G}$
4 do lines 6-14 of Algorithm 1
5 Return $G' = \bigcup_{g\in\mathcal{G}} g$

Algorithm 2: Alternative way to sample from $\mathrm{NeSt}_G^{(d)}(c^{(0)})$. Instead of attempting a fixed number of switches in each subgraphs g we allow for a variable number of switches by randomly choosing the subgraph to rewire (line 5). This makes sampling more similar to other MCMC sampling algorithms like those used for ERGMs. We note that this algorithm is much more sequential than algorithm 1 as the loop in line 4 is no longer independent.

8.3 Proof of theorem 3.1

Throughout this proof, we prove the statement for a sub-graph g_{C_i,C_j}, as that directly implies the statement for the whole graph. Toward this end, we must prove that the Markov chain used to sample the graphs is connected, aperiodic and doubly stochastic [14]. We start with the proof of connectedness. Let G be a graph, $c^{(0)}$ the initial colors and $d \in \mathbb{N}^+$. Further, let $O(G, d, c^{(0)})$ be the set of possible outputs of algorithm 1 with these input parameters.

Claim 1.

$$O(G, d, c^{(0)}) = \mathcal{N}_G^d(c^{(0)})$$

Proof. "$\subseteq$" The sampling procedure only edits edges in $g_{c_i^{(d-1)},c_j^{(d-1)}}$. Consider a single edge flip involving u_1, u_2, v_1, v_2 as in algorithm 1 with $c^{(d-1)}(u_1) = c^{(d-1)}(u_2)$ and $c^{(d-1)}(v_1) = c^{(d-1)}(v_2)$ by definition. We prove the most restrictive case where colors are aggregated in both directions as in eq. (2) and edges are directed from $c_i^{(d-1)}$ to $c_j^{(d-1)}$.

Let $M_X^{(d-1)}(v) = \{\!\{c^{(d)}(x) \mid x \in N_X(v)\}\!\}$ be the multi-set of colors of neighbouring nodes for $X \in \{\text{in}, \text{out}\}$. Then eq. (2) can be rewritten as:

$$c^{(d)}(u_1) = \text{hash}\Big(c^{(d-1)}(u_1), M_{\text{in}}(u_1), \Big(M_{\text{out}}(u_1)\backslash\{\!\{c^{(d-1)}(v_1)\}\!\}\Big) \cup \{\!\{c^{(d-1)}(v_1)\}\!\}\Big)$$

We prove by induction that colors $\tilde{c}^{(t)}$ in the new graph $\tilde{G}$ remain unchanged for all involved nodes. The base case for the initial colors holds per definition. For the induction step, assume the statement holds for t and consider node u_1:

$$\begin{aligned}\tilde{c}^{(t+1)}(u_1) &= \text{hash}\Big(\tilde{c}^{(t)}(u_1), \tilde{M}_{\text{in}}(u_1), \tilde{M}_{\text{out}}(u_1)\backslash\{\!\{\tilde{c}^{(t)}(v_1)\}\!\} \cup \{\!\{\tilde{c}^{(t)}(v_2)\}\!\}\Big)\\ &= \text{hash}\Big(c^{(t)}(u_1), M_{\text{in}}(u_1), M_{\text{out}}(u_1)\backslash\{\!\{c^{(t)}(v_1)\}\!\} \cup \{\!\{c^{(t)}(v_2)\}\!\}\Big)\\ &= \text{hash}\Big(c^{(t)}(u_1), M_{\text{in}}(u_1), M_{\text{out}}(u_1)\backslash\{\!\{c^{(t)}(v_1)\}\!\} \cup \{\!\{c^{(t)}(v_1)\}\!\}\Big) = c^{(t+1)}(u_1)\end{aligned}$$

Here the first equality is the result of the induction statement and the second equality holds because, in the original graph, $c^{(d)}(v_1) = c^{(d)}(v_2)$ implies that $c^{(t)}(v_1) = c^{(t)}(v_2)$ for $t \le d$. The same reasoning applies to the remaining nodes u_2, v_1, v_2.

"$\supseteq$" The backward direction is somewhat less intuitive. We show that any graph G' with the same CR colors of depth d can be reached by a sequence of at most $\frac{|D|}{2} - 1$ edits, where $D = (E(G) \cup E(G'))\backslash(E(G) \cap E(G'))$ is the set of edges G and G' don't agree upon. We again concern ourselves with the subgraphs g_{C_i,C_j} as using edge flips within these sub-graphs is sufficient to convert G into G', since if all edges in all subgraphs agree then G and G' are the same. Let g, g' be the subgraphs to the same pair of colors for G, G' respectively.

Base case: $|D| = 4$. Let $e_1 \neq e_2 \in D \cap E(g)$, $e_1' \neq e_2' \in D \cap E(g')$ and $e_1 = (u_1, v_1), e_2 = (u_2, v_2)$. Then it must be that $e_1' = (u_1, v_2), e_2' = (u_2, v_1)$ (besides renaming). As G and G' are both in $\mathcal{N}_G^d(c^{(0)})$ we have $c_G^{(d)}(u_1) = c_{G'}^{(d)}(u_1)$, which implies that $\deg_g^{\text{out}}(u_1) = \deg_{g'}^{\text{out}}(u_1)$. As G and G' agree on all other edges, this means $e_1' = (u_1, \cdot)$ or $e_2' = (u_1, \cdot)$. Similar reasoning applied to u_2 yields $e_1' = (u_1, \cdot)$ and $e_2' = (u_2, \cdot)$. Repeating the same for the in-degree of the v_i's and noting that $(u_1, v_1) \notin E(G')$ yields the statement.

Induction step: $n \to n - 1$. Let $|D| \cap (E(g) \cup E(g')) = 2n$. Let $e_1 = (u_1, v_1) \in D \cap E(g)$ be an edge that g and g' do not agree on. Since the degree in the subgraphs must be the same(see base case), there must be at least one edge $e_1' = (u_1, v_1') \in D \cap E(g')$ that g and g' also do not agree on. Since the in-degree of v_1' must also obey this, there exists an edge $e_2 = (u_2, v_1') \in D \cap E(g)$. In the case that $g = g_{C_i,C_j}$ for $C_i \neq C_j$, we have that all 4 nodes mentioned here are distinct. It could however be the case, that the edge $(u_2, v_1) \in E(g)$, which would prevent the edge flip. This implies that $\deg_g^{\text{in}}(v_1) > \deg_{g'}^{\text{in}}(v_1)$ and there exists another node x and an edge $(x, v_1) \in D \cap E(g')$. If this edge flip is also prevented by a edge, then the in degree of v_1' is again higher and we find another node. Since this cannot continue forever, as the graph is finite, we eventually find a node that we can use as u_2, i.e. where the edge flip is allowed. Performing an edge flip on e_1, e_2 yields the new edges $(u_1, v_1'), (u_2, v_1)$. Thus after this flip, e_1 has been transformed into e_1', so the graphs now agree on one more edge compared to

before. It is possible that $(u_2, v_1) \in D \cap E(g')$, in which case the edge flip relieved two disagreements simultaneously. In any case, $|D^*| \leq 2(n-1)$, where D^* is the disagreement set between the new graph created from G by the edge flip and G'.

However, in the case that g is not bipartite, it may be the case, that $u_2 = v_1$. If we can choose any other configuration to get 4 distinct nodes, then we do so and treat it like the previous case. If we cannot, then there is no $(v_1, x) \in E(g')$ and no $(x, v_1') \in D \cap E(g)$ for $x \neq u_2$. Then, u_1, v_1, v_1' constitute a directed triangle in g and g' with opposite directions. For this corner case, we need a new move, that turns one into the other, decreasing $|D|$ by 6, i.e. $|D^*| \leq 2(n-3)$. □

We have now shown that all graphs that have the same CR colors of depth d are a possible output of algorithm 1. Or in other words, the markov chain is connected. It is also aperiodic as choosing $e_1 = e_2$ is allowed and introduces a self loop. Finally, the Markov chain to sample the subgraph g_{C_i,C_j} for any fixed C_i, C_j is row stochastic, since the number of edge pair choices (= number of possible edits) is the same for all allowed subgraphs. Note that some of the possible edits may be invalid and make up a self-loop. Finally, every edit can also be reverted, meaning the markov chain is symmetric and thus doubly stochastic.

8.4 Proof of Theorem 4.1

Proof. *(theorem 4.1)* For (1) through (5) we always find that $\Gamma_X = H\Gamma_X^\pi$. Noticing that the blown-up vector has the same value for all nodes that are in the same WL-class (as indicated by the columns of H) yields that the nodes have the same centrality score.

(1) For Katz centrality, consider the definition:

$$\Gamma_{\mathbf{Katz}} = \sum_{k=1}^{\infty} a^k A^\top \mathbb{1}_n = \sum_{k=1}^{\infty} a^k A^\top H \mathbb{1}_k = H \sum_{k=1}^{\infty} a^k A_{\text{in}}^\pi \mathbb{1}_k = H\Gamma_{\mathbf{Katz}}^\pi$$

(2) For eigenvector centrality, we have:

$$\Gamma_{\mathbf{EV}} = \lim_{m\to\infty} \tfrac{1}{m} \sum_{i=0}^{m} (\lambda^{-1} A^\top)^i H \mathbb{1} = H \lim_{m\to\infty} \tfrac{1}{m} \sum_{i=0}^{m} (\lambda^{-1} A_{\text{in}}^\pi)^i \mathbb{1}$$

(3) For PageRank, notice that $c_{1,\text{in}}^{(\infty)}, c_{2,\text{in}}^{(\infty)} \sqsubseteq c^{(0)} \sqsubseteq d_{\text{out}}$ implies that $D_{\text{out}}^{-1} H = H D_{\text{out}}^\pi$. Consider the page rank matrices M_1, M_2 for both graphs. It holds that:

$$\begin{aligned} M_i H &= \alpha A_i^\top D_{\text{out}}^{-1} H + \tfrac{(1-\alpha)}{n} \mathbb{1}_n \mathbb{1}_n^\top H \\ &= H\Big(\alpha A^\pi D_{\text{out}}^\pi + \tfrac{(1-\alpha)}{n} \mathbb{1}_k (|C_1|, ..., |C_k|)\Big) = H M^\pi \end{aligned}$$

As the graph G_{M^π} that has M^π as its adjacency matrix, is strongly connected and aperiodic, M^π is primitive with perron (dominant) eigenvector w_π. This can be scaled up to see that Hw_π is shared between M_1 and M_2.

(4) Is similar to (2) but with A being replaced by $A_i^\top A_i$.

(5) Let G be a graph with adjacency matrix A.

Let H indicate the coarsest equitable partition when aggregating both in- and out-neighbourhood, then: $A_1 H = HA^\pi = A_2 H$ as well as $A_1^\top H = HA_{\text{in}}^\pi = A_2^\top H$. We prove by induction that all iterations $a_i^{(k)}, h^{(k)}$ are the same for both graphs and are of the form $a_i^{(k)} = Ha_\pi^{(k)}, h_i^{(k)} = Hh_\pi^{(k)}$. For the base case, $h_i^{(0)} = \mathbb{1} = H\mathbb{1}$. For the induction step:

$$a_i^{(k+1)} = \frac{A_i^\top H h_\pi^{(k)}}{\|A_i^\top H h_\pi^{(k)}\|} = \frac{H A_{\text{in}}^\pi h_\pi^{(k)}}{\|H A_{\text{in}}^\pi h_\pi^{(k)}\|}$$

The same statement for $h_i^{(k)}$ can be shown accordingly. The final iterates have the form $a_i^{(\infty)} = Ha_\pi^{(\infty)}, h_i^{(\infty)} = Hh_\pi^{(\infty)}$ and the statement follows. □

8.5 Proof of Lemma 4.2

Proof. For a proof of lemma 4.2 first notice that $H_{\text{in}}^{(t)}$ and D_{out}^{-1} are similarly related as the adjacency matrix, i.e. $D_{\text{out}}^{-1} H_{\text{in}}^{(t)} = H_{\text{in}}^{(t+1)} D_{t+1}^\pi$ for every t because D_{out} is encoded in the initial colors. We now proceed by induction on t. We show $x^{(t)} = H_{\text{in}}^{(t)} x_\pi^{(t)} = \tilde{x}^{(t)}$, which holds for $t = 0$ and $x^{(0)} = \text{const} \cdot \mathbb{1}_n$. Assuming the induction statement, it follows that:

$$\begin{aligned} x^{(t+1)} &= \alpha A^\top D_{\text{out}}^{-1} x^{(t)} + \tfrac{(1-a)}{n} \mathbb{1}_n && \text{(7a)} \\ &= \alpha A^\top D_{\text{out}}^{-1} H_{in}^{(t)} x_\pi^{(t)} + \tfrac{(1-a)}{n} \mathbb{1}_n && \text{(7b)} \\ &= H_{\text{in}}^{(t+1)} \big(\alpha X_{t+1}^\pi D_{t+1}^\pi x_\pi^{(t)} + \tfrac{(1-a)}{n} \mathbb{1}_k\big) && \text{(7c)} \end{aligned}$$

The last line is $x^{(t+1)} = H_{\text{in}}^{(t+1)} x_\pi^{(t+1)}$ which completes the induction. Repeating the same for $\tilde{x}^{(t+1)}$ concludes the proof. □

8.6 Proof of Lemma 4.3

Proof. We use theorem 6.1 in [4] that states:

$$\|x - x^{(i)}(y)\|_1 \leq \alpha^j \|x - x^{(i-j)}(y)\|_1 \tag{8}$$

for any non negative y with $\|y\|_1 \leq 1$. Similar to [15] we find the relationship $x^{(i+1)}(0) = x^{(i)}(\frac{1-\alpha}{n}\mathbb{1}_n)$. From this we get:

$$\begin{aligned} \|x - x^{(i)}(\tfrac{1-\alpha}{n}\mathbb{1}_n)\|_1 &= \|x - x^{(i+1)}(0)\|_1 \leq \alpha^{i+1} \|x - x^{(0)}(0)\|_1 \\ &= \alpha^{i+1} \|x\|_1 = \alpha^{i+1} \end{aligned}$$

We conclude the proof using the triangle inequality (1), Lemma 4.2 (2), and the directly above relationship (3).

$$\begin{aligned} \|x - \tilde{x}\|_1 &\overset{(1)}{\leq} \|x - \tilde{x}^{(d)}\|_1 + \|\tilde{x}^{(d)} - \tilde{x}\|_1 \\ &\overset{(2)}{=} \|x - x^{(d)}\|_1 + \|\tilde{x} - \tilde{x}^{(d)}\|_1 \overset{(3)}{\leq} \alpha^{d+1} + \alpha^{d+1} \end{aligned}$$ □

8.7 Proof of Corollary 4.4

Proof. Mirroring the reasoning of [15], we have:

$$\|x^{(k-1)} - x\|_1 \leq \tfrac{1}{1-\alpha} \|x^{(k-1)} - x^{(k)}\|_1$$

Now, we have $x^{(t)} = \tilde{x}^{(t)}$ for $t \leq k$ (lemma 4.2). Therefore:

$$\begin{aligned} &\|x - \tilde{x}\|_1 \leq \|x - \tilde{x}^{(k)}\|_1 + \|\tilde{x}^{(k)} - \tilde{x}\|_1 = \|x - x^{(k)}\|_1 + \|\tilde{x} - \tilde{x}^{(k)}\|_1 \\ &= \tfrac{1}{1-a}\Big(\|x^{(k-1)} - x^{(k)}\|_1 + \|\tilde{x}^{(k-1)} - \tilde{x}^{(k)}\|_1\Big) = \tfrac{2}{1-a} \|x^{(k-1)} - x^{(k)}\|_1 \end{aligned}$$ □

8.8 The ERGM used

In this work we used an ERGM with probabilities

$$p(\tilde{G}) \propto \exp(-10 \cdot \theta |\Gamma_{PR}(\tilde{G}) - \Gamma_{PR}(G)|)$$

The parameter θ controls how strongly graphs should resemble the PageRank score of the original graph G. We sample using the dyad flip Markov chain, i.e.from a current state a new graph is proposed which is obtained by flipping one dyad uniform at random from all dyads.