

Neighborhood Structure Configuration Models

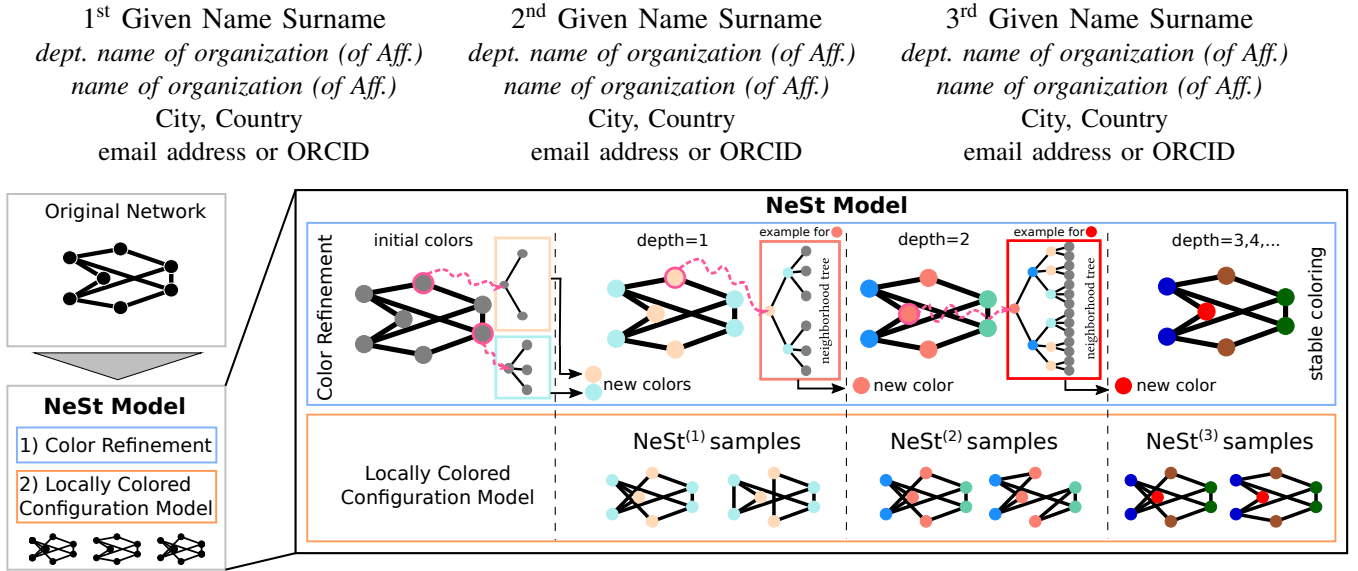


Fig. 1: Illustration of sampling networks with the NeSt model. Starting from an original network we use the Color Refinement (CR) algorithm to extract node colors for different depths d (blue box). At each d , nodes with an isomorphic neighborhood tree of depth d are assigned the same color. We then 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 k -hop neighborhood structure of a given network for any given k . The proposed algorithm allows trading off the variety in network samples for the amount of neighborhood structure that is guaranteed to be preserved. Our key innovation is to employ a colored Configuration Model with colors created from iterations of the so-called Color Refinement algorithm. We prove that, as the amount of structural information that is preserved increases, the generated synthetic networks and the given original network become more and more similar and are finally indistinguishable in terms of centrality measures such as PageRank, Katz centrality, eigenvector centrality and HITS. Our work enables to more efficiently generate samples with adjustable similarity to the original network, especially for large networks.

Index Terms—network generation, color refinement, colored configuration model, network null models, network sampling

I. INTRODUCTION

Current network null models preserve only very local information about nodes, such as degree, but preserving the larger neighborhood structure of nodes is necessary when null models should resemble a given network more closely. Current null models either lack that ability or are hard or even impossible to fit and sample for large networks.

In this paper, we introduce a simple and tunable network null model which preserves the neighborhood trees around each node up to a specific depth d . Even for large networks, the model is 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 [1] (an approximate graph isomorphism test) with a locally Colored Configuration Model [2], to obtain network models we call NeSt models. 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, select 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 [3], [4]. 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 list 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 [5] (IRG) generalize ER-random graphs by allowing varying edge probabilities. This enables the popular stochastic block model [6] (edge probability depends on group membership) and the Chung-Lu model [7] (degree distribution is preserved on expectation).

Configuration models The configuration model [8], [9] (CM) keeps degree of each node fixed in each sample. In the globally colored configuration model [10] we associate a color to each node and fix the total number of edges between colors. Alternatively, in locally-colored configuration models [2] we fix the color of each nodes' neighbors.

Exponential Random Graph models (ERGMs) [11] 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 [12], and sampling from these network models is often problematic [13].

Machine learning based network generators [14] 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 of networks, 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.

II. 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 distinguish matrix powers (A^k) from matrices with superscripts $(A^{(k)})$ by parenthesis. 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 $c: V \rightarrow \{1, \dots, k\}$ that assigns each node one out of $k \in \mathbb{N}$ colors. Each coloring induces a partition \mathcal{C} of the nodes into equivalence *classes* of equal color $\mathcal{C}_i = \{v \in V | c(v) = i\}$. 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* Γ_X , which compute the centrality scores of the nodes as the dominant eigenvector w of certain matrices M_X :

$$\Gamma_X = w, \text{ where } M_X w = \lambda w \text{ and } \lambda = \underset{\lambda_i \in \text{spec}(M_X)}{\text{argmax}} \lambda_i$$

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:

$$w = \lim_{m \rightarrow \infty} \frac{1}{m} \sum_{i=0}^m (\lambda^{-1} M_X)^i \mathbb{1}$$

For the *Eigenvector Centrality* [15], the relevant matrix is

$$M_{\text{EV}} = A^\top. \quad (\Gamma_{\text{EV}})$$

Similarly, the well-studied *PageRank* [16] measure corresponds to the dominant eigenvector of the following matrix:

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

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 $D = \text{diag}(\bar{A} \mathbb{1})$ is invertible.

HITS [17] assigns each node both a hub-score h_v and an authority score a_v . These are the dominant eigenvectors of:

$$M_{\text{auth}} = A^\top A \quad (\Gamma_{\text{auth}}), \quad M_{\text{hub}} = A A^\top \quad (\Gamma_{\text{hub}})$$

Katz centrality [18] is defined as:

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

with $\frac{1}{a} > \max_{\lambda_i \in \text{spec}(A)} |\lambda_i|$ being a parameter. Small/large values of a put more weights on shorter/longer paths.

A. Color refinement

The color refinement algorithm (CR) also known as Weisfeiler Lehman algorithm and originally proposed in [1] 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 \mathcal{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:

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

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. $c^{(d+1)} \subseteq c^{(d)}$. As there can be at most n strict refinements $c^{(d+1)} \subset c^{(d)}$, 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 [19]. 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 $c_{\text{in}}^{(d)}$ or $c_{\text{out}}^{(d)}$ respectively. We may further distinguish nodes by both their in- and out-neighborhood:

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

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 Fig. 1. The *neighborhood tree* can be seen on the right. It has all paths of length $\leq d$ as its nodes and two of these are connected if one of the corresponding paths extends the other by exactly one node. The neighborhood tree shows exactly what the CR colors encode in terms of neighborhood structure. The color of the root node in the next iteration directly encodes the structure of the whole tree in the sense they are the same if and only if their neighborhood trees are isomorphic [20].

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

The CR algorithm has a worst-case runtime of $\mathcal{O}((|E| + |V|) \cdot \log(|V|))$ [21]. However, we use a variant that has worst-case runtime $\mathcal{O}(d \cdot |V| \cdot \deg_{\max} \cdot \log(\deg_{\max}))$, 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 .

III. THE NESt MODEL

In this section we introduce the **Neighborhood Structure 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 easy to fit and sample on a computer.

For a given labeled graph G and initial node colors $c^{(0)}$, the set $\mathcal{N}_G^d(c^{(0)})$ 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 $\mathcal{N}_G^d(c^{(0)})$ 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 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) $\mathcal{N}_G^{(1)}(\text{const})$ recovers the standard configuration model with degree sequence identical to G
- 2) The graphs $G' \in \mathcal{N}_G^{(d)}(\text{const})$ and G are identically colored during the first d steps of the CR-algorithm
- 3) The set $\mathcal{N}_G^{(d)}$ contains *all* (simple) graphs that agree with G on the first d steps of the employed CR-algorithm
- 4) Structure preserved in $\mathcal{N}_G^{(d)}$ is also preserved in $\mathcal{N}_G^{(d+1)}$
- 5) $\mathcal{N}_G^{(d)}(c_0) \subseteq \mathcal{N}_G^{(d)}(\text{const})$

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.

A. 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 $g_{\text{green}, \text{red}}$ while edges connecting red to red nodes are put into $g_{\text{red}, \text{red}}$ (see Fig. 2 for such a partition). In the case of directed networks, we distinguish $g_{\text{green}, \text{red}}$ and $g_{\text{red}, \text{green}}$ 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

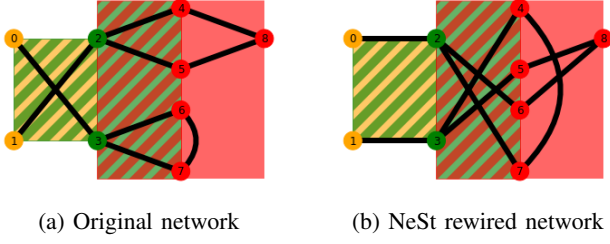


Fig. 2: Illustration of the rewiring procedure employed when sampling from the NeSt⁽²⁾ model. First, the edges are partitioned into subgraphs according to the colors of depth 1 of their endpoints. The striped regions with two colors (yellow-green, green-red) correspond to bipartite networks connecting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

are bipartite. To ensure a consistent neighborhood preserving sampling we can thus only allow edge 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 VIII).

The entire algorithm is displayed 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 [22], [23] — the strategy we apply in this work — although other strategies are also possible [24]. We assert the utility of the sampling procedure with the following theorem:

Theorem 1. *The sampling procedure shown in algorithm 1 samples 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 VIII). 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,

Input: Graph G , initial colors $c^{(0)}$, depth d

Output: Sample Graph G' distributed as $\text{NeSt}_G^d(c^{(0)})$

```

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  $g_{C_i, C_j} \in \mathcal{G}$  do
5   for  $r \cdot |E(g_{C_i, C_j})|$  steps do
6     choose edges  $u_1v_1, u_2v_2 \in E(g_{C_i, C_j})$ 
       uniformly at random
7     if  $u_1v_2 \notin E(g_{C_i, C_j})$  and  $u_2v_1 \notin E(g_{C_i, C_j})$ 
       then
8       remove  $u_1v_1, u_2v_2$ 
9       add  $u_1v_2, u_2v_1$ 
10    if  $c_i = c_j$  then
11      randomly choose node triplet  $u_1, u_2, u_3$ 
12      if  $u_1, u_2, u_3$  constitute a directed triangle
       then
13        flip the direction of the triangle
           $u_1, u_2, u_3$ 
14 Return  $G' = \bigcup_{g_{C_i, C_j} \in \mathcal{G}} g_{C_i, C_j}$ 

```

Algorithm 1: Sampling from $\text{NeSt}_G^d(c^{(0)})$ using edge switches. The number of steps can be set by the user, potentially also as a function of g . We use a number of steps proportional to the number of edges in each subgraph, which yields a runtime of $\mathcal{O}(r \cdot |E(G)| \cdot \deg_{\max})$ - disregarding the computation time needed for the CR algorithm. While for undirected graphs simple edge switches (lines 6-9) are sufficient, for the directed case an additional directed triangle switch (lines 10-13) is required for uniform sampling (see the proof of Theorem 1 in Section VIII).

it is sufficient to preserve the node color and the multiset of its 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 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 [10].

B. 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 1.

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 $\text{NeSt}^{(d)}$. 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 $c^{(d)} \supseteq c^{(d_*)} \supseteq c^{(d+1)}$. Note: The second refinement relation is not strict only if *all* nodes associated with the selected depth d colors are ‘pushing’.

IV. THE ROLE OF THE DEPTH PARAMETER d

A. 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 2. *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) $\Gamma_{\text{Katz}}(u) = \Gamma_{\text{Katz}}(v)$
- 2) $\Gamma_{\text{EV}}(u) = \Gamma_{\text{EV}}(v)$.

- 3) If $c^{(0)} \sqsubseteq \text{deg}_{\text{out}}$, then $\Gamma_{\text{PR}}(u) = \Gamma_{\text{PR}}(v)$.
- 4) If $c_{\text{in}}^{(\infty)}$ is computed for $A_i^\top A_i$, then $\Gamma_{\text{auth}}(u) = \Gamma_{\text{auth}}(v)$
- 5) If instead CR aggregates over both in and out neighbors then $\Gamma_{\text{auth}}(u) = \Gamma_{\text{auth}}(v)$ and $\Gamma_{\text{hub}}(u) = \Gamma_{\text{hub}}(v)$

This theorem 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 such that stable coloring is reached, many eigenvector-based 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 [25].

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 $\mathcal{C}_i = \{v \in V \mid c(v) = i\}$ a node belongs to:

$$H_{i,j}^{(d)} = \mathbb{I}[i \in \mathcal{C}_j^{(d)}] = \begin{cases} 1 & \text{if } i \in \mathcal{C}_j^{(d)} \\ 0 & \text{else} \end{cases} \quad (3)$$

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:

$$(A^\top H^{(d)})_{u,i} = \sum_{v \in N_{\text{in}}(u)} \mathbb{I}[v \in \mathcal{C}_i^{(d-1)}] \quad (4)$$

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 to express this matrix as

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

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:

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

Proof: (Theorem 2) The main idea to the proof is that each eigenvector of the A^π matrix ($A^\pi w^\pi = \lambda w^\pi$) can be scaled up by multiplication with the partition indicator matrix H (defined in Eq. (3) indicating the coarsest equitable partition) such that Hw^π is an eigenvector of A to the same eigenvalue:

$$AHw^\pi = HA^\pi w^\pi = \lambda Hw^\pi$$

In [26] it is shown that the dominant eigenpair of A is shared in this way if it is unique, which is the key insight toward proving the theorem, seeing as both A_i have the same H and

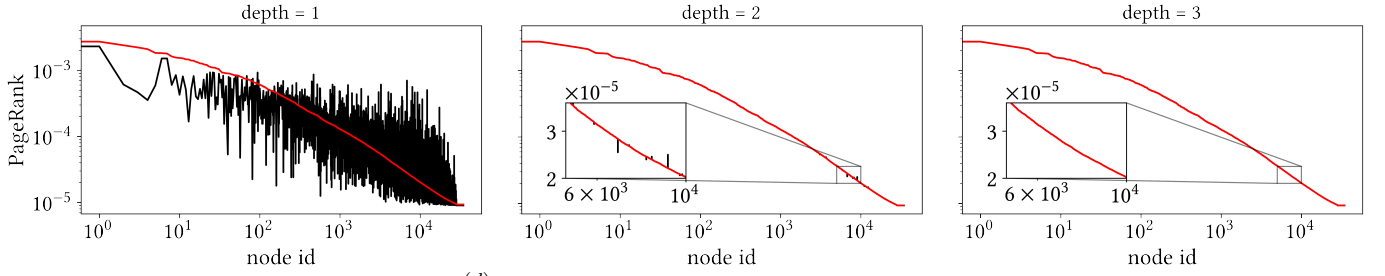


Fig. 3: PageRank distribution of the $\text{NeSt}^{(d)}(\text{deg}_{\text{out}})$ model for different values of the depth d . We show the distribution of PageRank for the HepPh-network (red) and one sample from the fitted $\text{NeSt}^{(d)}(\text{deg}_{\text{out}})$ model (black). We can see that at depth of 1 (left) there still is a large difference of the original PageRank score compared the PageRank of the sampled network. With increasing depth this error decreases and almost vanishes (max error $< 10^{-16}$) for depth 3. This shows that multi-hop neighborhood information can be necessary to accurately preserve the PageRank score.

A_π by assumption. 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:

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

(2) For eigenvector centrality, we have:

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

(3) For PageRank, notice that $c_{1,\text{in}}^{(\infty)}, c_{2,\text{in}}^{(\infty)} \subseteq c^{(0)} \subseteq d_{\text{out}}$ 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:

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

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 then be scaled up to see that $H w_\pi$ 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 found by the WL algorithm through aggregating both in- and out-neighbourhood, then: $A_1 H = H A^\pi = A_2 H$ as well as $A_1^\top H = H A_{\text{in}}^\pi = A_2^\top H$. We prove by induction that all iterations $a_i^{(k)}, h_i^{(k)}$ are the same for both graphs and are of the form $a_i^{(k)} = H a_\pi^{(k)}, h_i^{(k)} = H h_\pi^{(k)}$. 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 $a_i^{(\infty)} = H a_\pi^{(\infty)}, h_i^{(\infty)} = H h_\pi^{(\infty)}$ and the statement follows. \square

B. 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 in which cases preserving smaller neighborhood depths is already sufficient to maintain centralities. 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 the NeSt model. The following result formalizes the consequences of this for PageRank centrality.

Lemma 1. *Let G be a graph with adjacency matrix A and let \tilde{G} be sampled from $\text{in-NeSt}_G^d(\text{deg}_{\text{out}})$ with adjacency matrix \tilde{A} . Consider the iterative computation of PageRank:*

$$x^{(t+1)} = A^\top D_{\text{out}}^{-1} x^{(t)} + \frac{1-\alpha}{n} \mathbb{1}_n \quad (6)$$

and let $\tilde{x}^{(t+1)}$ be defined accordingly for \tilde{G} , then:

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

The proof of this lemma 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 2.

The algebraic view for intermediate depth We extend the previously established perspective to intermediate colors. That is, we derive statements akin to Eq. (5) for the intermediate colors. As noted in [27], 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:

$$A H^{(d)} = H^{(d+1)} X_{d+1}^\pi \quad (7)$$

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-

$\text{NeSt}^d(c_0)$ have identical colorings for $t \leq d$, 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.

Proof: For a proof of Lemma 1 we now consider the PageRank iteration Eq. (6) starting with $x^{(0)} = \mathbb{1}_n$, which is equivalent to the definition given in Section II [28] up to normalization. First notice, that $H_{\text{in}}^{(t)}$ and D_{out}^{-1} are similarly related as the adjacency matrix $D_{\text{out}}^{-1} H_{\text{in}}^{(t)} = H_{\text{in}}^{(t+1)} D_{t+1}^\pi$ for every t because D_{out} is encoded in the initial colors. We now proceed by induction on t . We show that $x^{(t)} = H_{\text{in}}^{(t)} x_\pi^{(t)}$, which holds for $t = 0$ and $x^{(0)} = \text{const} \cdot \mathbb{1}$. Assuming the induction statement, it follows that:

$$x^{(t+1)} = \alpha A^\top D_{\text{out}}^{-1} x^{(t)} + \frac{(1-a)}{n} \mathbb{1}_n \quad (8a)$$

$$= \alpha A^\top D_{\text{out}}^{-1} H_{\text{in}}^{(t)} x_\pi^{(t)} + \frac{(1-a)}{n} \mathbb{1}_n \quad (8b)$$

$$= H_{\text{in}}^{(t+1)} \left(\alpha X_{t+1}^\pi D_{t+1}^\pi x_\pi^{(t)} + \frac{(1-a)}{n} \mathbb{1}_k \right) \quad (8c)$$

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. \square

Guaranteed similarity in PageRank Lemma 1 shows that graphs sampled from $\text{in-NeSt}_G^d(\text{deg}_{\text{out}})$ are constrained in their first power iterations. Combining this observation with convergence guarantees of the PageRank power iteration we can be sure that two samples from the NeSt model are bound to have centralities that are no further apart than the following:

Lemma 2. *Let x and \tilde{x} be the PageRank vectors of two graphs sampled from $\text{in-NeSt}_G^d(\text{deg}_{\text{out}})$. Let α be the PageRank damping factor used. It holds that:*

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

Informally, the 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.

Proof: From Lemma 1 the iterates $x^{(t)} = \tilde{x}^{(t)}$ are the same for $t \leq d$. We use Theorem 6.1 in [29] that states:

$$\|x - x^{(i)}\|_1 \leq \alpha^i \|x - x^{(0)}\|_1 \leq 2\alpha^i \quad (9)$$

We can slightly strengthen this bound as follows [28]. If we start with the vector $y^{(0)} = 0$ in Eq. (8), the next iteration is $y^{(1)} = x^{(0)} = (1-\alpha)\mathbb{1}$ for which we already know that $\|x - x^{(0)}\|_1 \leq \alpha$. It follows that:

$$\|x - x^{(i)}\|_1 \leq \alpha^i \|x - x^{(0)}\|_1 \leq \alpha^{i+1} \quad (10)$$

We conclude the proof using the triangle inequality:

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

\square

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 2 yields a bound of 1.4 (using typical $\alpha=0.85$), while we know from Theorem 2 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 worst case. We thus establish a second bound which relates the convergence in one network to a guaranteed similarity in PageRank.

Corollary 1. *With assumptions as in Lemma 1 and 2:*

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

Colloquially speaking, Corollary 1 states that if the PageRank iterations converge quickly, then the PageRank vectors of synthetic and original network are not too far apart.

Proof: Mirroring the reasoning of [28], we have:

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

Now, we have $x^{(t)} = \tilde{x}^{(t)}$ for $t \leq k$ (Lemma 1). Therefore:

$$\begin{aligned} \|x - \tilde{x}\|_1 &\leq \|x - \tilde{x}^{(k)}\|_1 + \|\tilde{x}^{(k)} - \tilde{x}\|_1 = \|x - x^{(k)}\|_1 + \|\tilde{x} - \tilde{x}^{(k)}\|_1 \\ &= \frac{1}{1-\alpha} (\|x^{(k-1)} - x^{(k)}\|_1 + \|\tilde{x}^{(k-1)} - \tilde{x}^{(k)}\|_1) = \frac{2}{1-\alpha} \|x^{(k-1)} - x^{(k)}\|_1 \end{aligned}$$

\square

V. 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 [30]), and a network of phonecalls [31]. For details on the networks see Fig. 6.

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 initial condition we chose the normalized (1-norm) / normal all-ones vector for PageRank / others.

We exemplify detailed convergence results for the PageRank distribution on the HepPh-network in Fig. 3. To increase visibility, nodes are sorted in decreasing 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

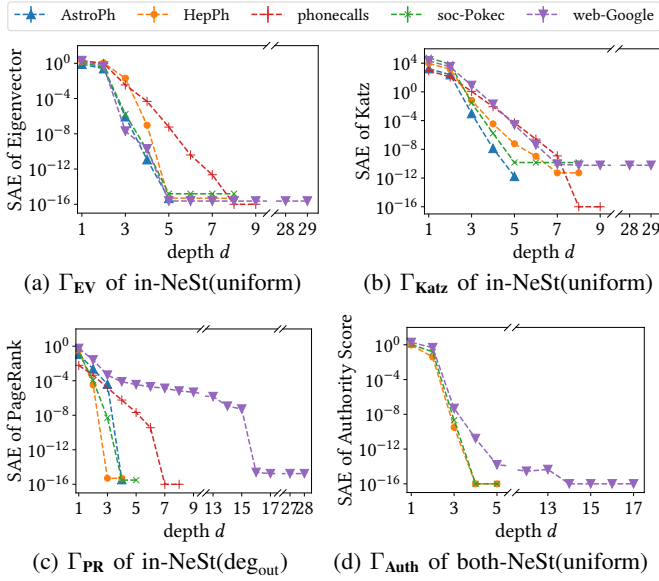


Fig. 4: Convergence of network centralities. We show the sum absolute error (SAE) for a sample in relation to the original network. Points are medians with 16%/84% quantile error bars for 100 samples. Values are capped below by 10^{-16} . Legend is on top. From left to right increasingly deeper neighborhood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

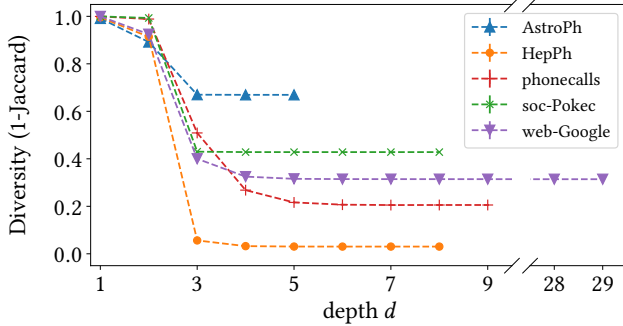


Fig. 5: Diversity in NeSt(const) samples. We measure Diversity as one minus the Jaccard-Similarity of the edges of the original and sampled network. For a diversity of 1 the original and sampled network share no edges, for a diversity of 0 the networks agree perfectly. We see that for most networks, 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.

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

Fig. 6: Statistics of the real world networks used. EV indicates whether the dominant eigenvector is unique.

plot ($d=3$) the sample reflects the PageRank almost perfectly (MAE drops by a factor of 10^{-10}).

Convergence results for other networks are summarized in Fig. 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 (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 (4d) shows the Authorities (HITS) score for the NeSt which preserves both the in- and out-neighborhood trees starting from uniform colors.

Better preserved neighborhood structure usually means diminished diversity in network samples, we show the extend of this in Fig. 5. Finally we compare the NeSt model with other random graph models in Fig. 7. For details on the ERGM used see appendix. The ERGM is very slow because the PageRank needs to be recalculated for each flip.

Implementation details For the CR-algorithm we use an implementation suggested by Kersting et al. [27]. The code is available at <https://github.com/SomeAuthor123/NestModel>.

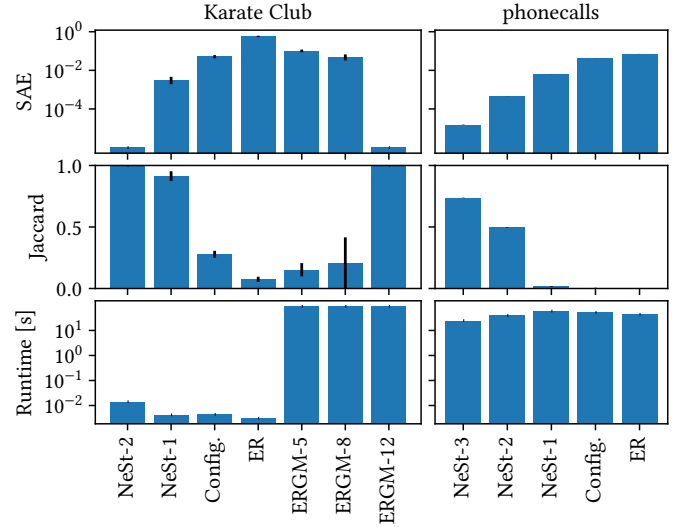


Fig. 7: Comparison of the NeSt model with other random graph models. The left/right column correspond to the Karate Club/ phonecalls network with 34/36k nodes. The rows show sum absolute error (SAE) of the PageRank (top), Jaccard similarity (mid) and runtime in seconds for 1000 samples (bottom). Lower is better on all scales. Besides NeSt we include Exponential Random Graph Models (ERGM), the Configuration model and the Erdős-Rényi (ER) network. ERGMs are excluded on phonecalls due to poor runtime. On the small network (left) both ERGM and NeSt allow a trade-off of similarity for SAE but ERGM offering 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 feasible (runtime).

Preserving centralities As shown in Section IV-B, 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 later NeSt of larger depth. However in applications, it might be undesirable to distinguish nodes with 1000 and 1001 neighbors. At the cost of not preserving neighborhood trees exactly, we may thus limit the minimal number of nodes within each color class by employing a clustering algorithm rather than a hash function.

VII. CONCLUSION

In this paper, we have introduced NeSt models which enable the creation of synthetic networks that preserve the neighborhood structure of nodes in a given network up to a specified depth. NeSt models thus represent a versatile generalization of existing configuration models. 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 preservation of neighborhood structure by applying NeSt models for preserving centralities, the capabilities of the NeSt model extend to the preservation of many eigenvectors. 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 [32], consensus dynamics [33], or questions related to controllability [34].

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 [20], [35], [36]. As all samples from the 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.

- [1] B. Weisfeiler and A. Leman, "The reduction of a graph to canonical form and the algebra which appears therein," *NTI, Series*, 1968.
- [2] B. Söderberg, "Random graphs with hidden color," *Physical Review E*, vol. 68, no. 1, p. 015102, 2003.
- [3] G. Cimini *et al.*, "The statistical physics of real-world networks," *Nature Reviews Physics*, vol. 1, no. 1, pp. 58–71, 2019.
- [4] T. Coolen, A. Annibale, and E. Roberts, *Generating random networks and graphs*. Oxford university press, 2017.
- [5] B. Söderberg, "General formalism for inhomogeneous random graphs," *Physical Review E*, vol. 66, no. 6, p. 066121, 2002.
- [6] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social networks*, vol. 5, no. 2, pp. 109–137, 1983.
- [7] F. Chung and L. Lu, "The average distances in random graphs with given expected degrees," *PNAS*, vol. 99, no. 25, pp. 15 879–15 882, 2002.
- [8] M. Newman, S. Strogatz, and D. Watts, "Random graphs with arbitrary degree distributions and their applications," *Phys Rev E*, 2001.
- [9] B. Fosdick *et al.*, "Configuring random graph models with fixed degree sequences," *Siam Review*, vol. 60, no. 2, pp. 315–355, 2018.
- [10] M. E. Newman, "Mixing patterns in networks," *Physical review E*, vol. 67, no. 2, p. 026126, 2003.
- [11] D. Lusher *et al.*, *Exponential random graph models for social networks: Theory, methods, and applications*. Cambridge Univ. P., 2013.
- [12] S. Chatterjee and P. Diaconis, "Estimating and understanding exponential random graph models," *The Annals of Statistics*, 2013.
- [13] T. A. Snijders *et al.*, "Markov chain monte carlo estimation of exponential random graph models," *Journal of Social Structure*, 2002.
- [14] X. Guo and L. Zhao, "A systematic survey on deep generative models for graph generation," *arXiv preprint arXiv:2007.06686*, 2020.
- [15] P. Bonacich, "Technique for analyzing overlapping memberships," *Sociological methodology*, vol. 4, pp. 176–185, 1972.
- [16] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Tech. Rep., 1999.
- [17] J. M. Kleinberg *et al.*, "Authoritative sources in a hyperlinked environment," in *SODA*, vol. 98. Citeseer, 1998, pp. 668–677.
- [18] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.
- [19] R. Paige and R. E. Tarjan, "Three partition refinement algorithms," *SIAM Journal on Computing*, vol. 16, no. 6, pp. 973–989, 1987.
- [20] P. Barceló, E. Kostylev, M. Monet, J. Pérez, J. Reutter, and J. Silva, "The logical expressiveness of graph neural networks," in *ICLR 2020*.
- [21] M. Grohe *et al.*, "Dimension reduction via colour refinement," in *European Symposium on Algorithms*. Springer, 2014, pp. 505–516.
- [22] Y. Artzy-Randrup and L. Stone, "Generating uniformly distributed random networks," *Physical Review E*, vol. 72, no. 5, p. 056708, 2005.
- [23] P. Erdos *et al.*, "The mixing time of the swap (switch) markov chains: a unified approach," *arXiv preprint arXiv:1903.06600*, 2019.
- [24] C. Carstens, "Proof of unif. sampling of binary matrices with fixed row and column sums for the curveball algorithm," *Phys Rev E*, 2015.
- [25] P. Boldi, V. Lonati, M. Santini, and S. Vigna, "Graph fibrations, graph isomorphism, and pagerank," *RAIRO-ITA*, 2006.
- [26] M. Scholkemper and M. T. Schaub, "Blind extraction of equitable partitions from graph signals," in *IEEE ICASSP*, 2022, pp. 5832–5836.
- [27] K. Kersting, M. Mladenov, R. Garnett, and M. Grohe, "Power iterated color refinement," in *28th AAAI*, 2014.
- [28] D. F. Gleich, "Pagerank beyond the web," *Siam Review*, vol. 57, no. 3, pp. 321–363, 2015.
- [29] M. Bianchini, M. Gori, and F. Scarselli, "Inside pagerank," *ACM Transactions on Internet Technology (TOIT)*, 2005.
- [30] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," <http://snap.stanford.edu/data>, Jun. 2014.
- [31] A. Barabási, "Networkscience book datasets," accessed: 2022-10-01. [Online]. Available: <http://networksciencebook.com/resources/data.html>
- [32] M. T. Schaub *et al.*, "Graph partitions and cluster synchronization in networks of oscillators," *Chaos*, vol. 26, no. 9, p. 094821, 2016.
- [33] N. O'Clery *et al.*, "Observability and coarse graining of consensus dynamics through the external equitable partition," *Phys Rev E*, 2013.
- [34] D. Cardoso *et al.*, "Laplacian eigenvectors and eigenvalues and almost equitable partitions," *Jour. of combinatorics*, 2007.
- [35] C. Morris *et al.*, "Weisfeiler and leman go neural: Higher-order graph neural networks," in *AAAI*, vol. 33, no. 01, 2019, pp. 4602–4609.
- [36] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *ICLR*, 2018.

VIII. APPENDIX

A. Proof of Theorem 1

Throughout this proof, we prove the statement for a subgraph 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 [9]. 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:

$$c^{(d)}(u_1) = \text{hash}\left(c^{(d-1)}(u_1), M_{\text{in}}(u_1), \left(M_{\text{out}}(u_1) \setminus \{\{c^{(d-1)}(v_1)\}\} \cup \{\{c^{(d-1)}(v_1)\}\}\right)\right)$$

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 :

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

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 $t \leq d$. 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 $\lfloor \frac{|D|}{2} \rfloor - 1$ edits, where $D = (E(G) \cup E(G')) \setminus (E(G) \cap E(G'))$ 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 $e'_1 = (u_1, v_2)$, $e'_2 = (u_2, v_1)$ (apart from renaming). As G

and G' are both in $\mathcal{N}_G^d(c^{(0)})$ we have that $c_G^{(d)}(u_1) = c_{G'}^{(d)}(u_1)$, 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: $n \rightarrow n - 1$. 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. $|D^*| \leq 2(n - 3)$. \square

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.

B. The ERGM used

In this work we used an ERGM with probabilities

$$p(\tilde{G}) \propto \exp(-100 \cdot \theta |\Gamma_{PR}(\tilde{G}) - \Gamma_{PR}(G)|)$$

The parameter θ controls how strongly graphs should resemble the PageRank score of the original graph G . We sampled using the dyad flip Markov chain.