A Random Model of Supply Chain Networks

Philippe Blaettchen
Bayes Business School, City St George's, University of London, United Kingdom, EC1Y 8TZ, philippe.blaettchen@city.ac.uk

Andre P. Calmon
Scheller College of Business, Georgia Institute of Technology, Atlanta, GA, 30308, andre.calmon@gatech.edu

Georgina Hall
Decision Sciences, INSEAD, Fontainebleau, France, 77305, georgina.hall@insead.edu

Abstract. Supply chain problems are frequently formulated as optimization problems over graphs representing complex networks of interlinked input-output relationships. Frequently, these problems are hard, so researchers rely on analyzing stylized structures or developing heuristic solutions. Yet, a scarcity of real-world data has hindered our understanding of how these exact and heuristic solutions perform in practice and whether managerial insights carry over from these simplified settings. We address this critical gap by introducing *RG4SC*, a versatile random graph model for creating "test tracks" for supply chain management research. *RG4SC*'s simple micro-foundations and interpretable input parameters allow for systematically generating diverse and realistic network structures. We demonstrate its empirical validity and that it more adequately represents real-world supply chain networks than existing random models. We then showcase *RG4SC*'s utility for research through a case study on the Guaranteed Service Model, a widely used framework for safety stock optimization. Our analysis shows how *RG4SC* can be central to uncovering novel managerial insights, analyzing the computational complexity of algorithms, benchmarking heuristics, and training machine learning models. *RG4SC* is accessible through a user-friendly web interface at https://scngenerator.pythonanywhere.com/.

Key words: supply chain networks; random generative models; supply chain management; inventory management; data sets; algorithms; computational complexity

1. Introduction

Several popular supply chain management problems are modeled as optimization problems over *supply chain networks*—graphs where nodes correspond to firms or processing steps in supply chains, and edges represent material flows between them. Frequently, these problems are NP-hard, lacking tractable solution approaches for general network structures. As a result, researchers solve these problems in simplified structures, such as two-tier networks, or develop heuristics and data-driven algorithms. These solution approaches are then used to derive managerial insights.

Testing and validating such approaches requires real-world supply chain network data, but publicly available sources are limited.[1] For instance, the only public dataset containing detailed processing steps and covering multiple industries, Willems (2008), contains just 38 networks. Consequently, researchers face challenges in validating new algorithms and insights, often resorting to stylized examples that might not represent the complexity and variety of the networks encountered in practice. Our predicament is analogous to engineers designing ever more high-performance cars with access to only few test tracks that do not fully represent real-world driving conditions.

To bridge this gap, we adapt random generative graph models, common to fields like social network analysis and epidemiology (Newman et al. 2002, Keeling and Eames 2005), to supply chain optimization. We develop *RG4SC* (or *Random Graphs for Supply Chain Management*), which creates "test tracks" for supply chain management researchers. The model has simple micro-foundations and generates networks that empirically match those encountered in practice. It enables researchers to easily generate synthetic, but realistic, datasets of networks with controlled structural characteristics, or extrapolate from existing, limited datasets. Our results also allow researchers to link *RG4SC*'s input parameters, which determine how buyer-supplier relationships are formed, with the structure of the networks it generates.

Researchers can use *RG4SC* to systematically evaluate the different approaches for tackling hard optimization problems in supply chain management. First, they can analyze whether managerial insights hold up in practice by testing them on datasets with diverse network structures, or use *RG4SC* to systematically uncover new insights. Second, they can examine the performance of exact algorithms by exploring the relationships between the micro behavior of firms in *RG4SC*, the resulting network structures, and exact algorithms' complexity. Third, they can comprehensively evaluate heuristics by creating principled benchmarks. Finally, they can use *RG4SC* to improve the performance of data-driven algorithms, by bootstrapping existing data to fine-tune them.

We introduce *RG4SC* in § 3.1. The model generates networks consisting of nodes and edges. Nodes represent value-adding activities, and edges represent input-output relationships between nodes. For simplicity, we refer to nodes as *firms* but caution that nodes may reflect individual activities controlled by the same company. The model's input parameters include the number of nodes and tiers, probability distributions of node characteristics, and controls on the formation of input-output relationships. This design enables *RG4SC* to generate a wide variety of realistic supply chain network structures. Our model is available online with a user-friendly interface, allowing researchers to systematically generate networks to study optimization problems.

In § 3.2, we show how to adjust *RG4SC*'s inputs to shape structural characteristics of the generated supply chain networks, such as the degree distribution and modularity. The easily interpretable input parameters of *RG4SC* allow researchers to generate sets of networks with specific properties to assist in analyzing supply chain optimization problems. Notably, we prove an analytical relationship between *RG4SC*'s inputs and the treewidth—a measure of how closely a graph resembles a tree—of the generated networks. This relationship is valuable since many NP-hard optimization problems are fixed-parameter tractable in the treewidth of the underlying network (Bodlaender 2006).

We develop an estimation procedure to fit *RG*4*SC* to data in § 3.3. This enables us to validate *RG*4*SC* empirically, showing that it generates networks consistent with, among others, the real-world data in Willems (2008). We further demonstrate that *RG*4*SC* outputs more realistic networks than, for example, random attachment models (Erdős and Rényi 1959, Perera et al. 2017a). Importantly, our estimation procedure allows researchers to fit *RG*4*SC*'s inputs to a set of supply chain networks and then generate networks with similar structural characteristics to the ones in the set. Hence, researchers can use *RG*4*SC* to investigate, for example, the performance and behavior of an algorithm in supply chain networks that are structurally similar to examples from a given industry.

We then illustrate *RG*4*SC* as a foundational tool to systematically study hard supply chain management problems, using the Guaranteed Service Model (GSM) as a case study. The GSM, described in § 4.1, is a celebrated and widely employed optimization problem for determining safety stock placement in supply chain networks (Schoenmeyr and Graves 2022). It has been the focus of hundreds of research papers and is used by dozens of major corporations.[2]

In § 4.2, we show how *RG*4*SC* helps extend an important managerial insight about safety-stock placement in the GSM for simple two-tier networks—that safety-stock locations depend on the connectivity between the tiers—to complex, real-world-inspired supply chain networks. To do so, we solve the GSM on thousands of supply chain networks generated by *RG*4*SC* with varying degrees of connectivity. We find a new managerial insight: As a network's connectivity decreases, safety stock placement spreads upstream in the supply chain across multiple tiers and firms. Conversely, safety stock becomes concentrated near the demand nodes as connectivity increases.

In § 4.3, we introduce two methods to use *RG*4*SC* for characterizing the behavior and limitations of exact solution algorithms for the GSM, with the example of an optimization-based solution procedure recently developed in Blaettchen et al. (2024b). The first method is analytical: we prove a critical relationship between *RG*4*SC*'s inputs describing a firm's local behavior (such as how a firm connects to its neighbors) and the worst-case complexity of solving the GSM. The second method is a principled empirical approach, where we use *RG*4*SC* to generate a sequence of networks, with only one network structural parameter (e.g., the scale) varying, while others are fixed. This allows us to extrapolate insights about the solution algorithm's complexity from limited data.

In § 4.4, we turn to heuristics and data-driven algorithms for the GSM. We first use *RG*4*SC* to benchmark and study an existing heuristic. While previous studies, based on ad-hoc network structures, indicated that the heuristic could have a high performance, we show that this perfomance depends on the network's micro-structures. We uncover settings where both exact and heuristic

approaches face substantial challenges, highlighting supply chain networks for which new solutions methods may be needed. Finally, we demonstrate how *RG*4*SC* serves as a powerful tool for *improving* data-driven algorithms through two methods: creating synthetic data or bootstrapping existing data. We illustrate this by training a simple machine learning model to predict the characteristics of the optimal solution based on network structure and inventory costs. As a benchmark, we train the model on a part of the Willems (2008) dataset, and use the rest of the dataset as a test set. We show that training the model on synthetic networks or bootstrapped versions of the original training data, both created by *RG*4*SC*, significantly improves the model's performance.

*RG*4*SC* provides a principled way to generate supply chain management test tracks, and our case study underscores its value for supporting the main approaches for solving supply chain optimization problems. Researchers can use *RG*4*SC* to design supply chain network structures to derive new managerial insights, analyze exact solution approaches, and benchmark or improve heuristic solutions. Ultimately, *RG*4*SC* can accelerate replicable research in supply chain optimization, improving how researchers approach and solve complex supply chain management problems.

2. Literature review

Our paper bridges two literature streams: The extensive work on supply chain-related optimization problems, and research on random models, particularly in supply chain settings.

A standard approach to studying and solving *supply chain management* problems is modeling them as optimization problems over graphs. Examples of this approach include order quantity optimization (Eppen 1979), requirements planning (Graves et al. 1998), material flow costs minimization (Gamarnik et al. 2012), inventory placement (Ettl et al. 2000), contract design (Majumder and Srinivasan 2008), supplier auditing (Zhang et al. 2022), disruption risk mitigation (Bimpikis et al. 2019), and technology diffusion targeting (Blaettchen et al. 2024a).

A rich literature examines how supply chain strategies should be tailored to specific supply chain structures (see, e.g., Song and Yao 2002). However, many of the problems above are NP-hard, so a complete analysis of all practically relevant structures is usually infeasible. As a result, researchers assume simplified structures, such as serial systems (Gong et al. 1994), two- and three-tier networks (Zhang et al. 2022), or trees (Majumder and Srinivasan 2008), or they rely on heuristics (Ettl et al. 2000) and data-driven algorithms (Gijsbrechts et al. 2022). Even when general solutions exist, their characteristics and sensitivity in practice are of crucial managerial interest. However, if any empirical validation is conducted, this is limited to individual case studies (Gao et al. 2019).

*RG*4*SC* can help researchers create realistic supply chain networks with controlled characteristics, either from scratch or by bootstrapping and rescaling existing data. Generated networks can be used to verify managerial insights drawn from simplified structures, analyze the drivers of solution complexity, benchmark heuristics, or train machine learning algorithms.

With a case study on the Guaranteed Service Model (GSM; for a review, see Eruguz et al. 2016), our analysis also directly contributes new results to the multi-echelon inventory management literature. This is important, as de Kok et al. (2018) highlights that most inventory management research assumes simplified two-tier networks. Using *RG*4*SC* reveals new insights into how the supply chain network structure affects safety stock placement, how the complexity of solving the GSM scales with network size, and the approximability of the GSM using heuristics. Thus, our findings directly extend the works of Humair and Willems (2006), Magnanti et al. (2006), and Shu and Karimi (2009) on strategic safety stock placement in general acyclic networks.

Moreover, our work is deeply rooted in *random graph theory*, a field first introduced by Gilbert (1959) and Erdős and Rényi (1959). Using randomly generated graphs to model "typical" scenarios rather than worst-case scenarios has found applications across disciplines, from computer science to biology. However, despite their prevalence in many areas, random graphs have seen limited application in supply chain management. To the best of our knowledge, random graph models of supply chain networks appear primarily in two areas: in network design, to understand networks' resilience to external shocks, and in economics, for the study of network equilibria.

Our work extends their application in supply chain management in a novel direction: *RG*4*SC* is designed to analyze the solutions to supply chain optimization problems and the complexity of solving them. This approach fills a critical gap in the literature, as the few existing models of supply chain networks are inadequately equipped for this purpose.

Models from network design (e.g., Thadakamaila et al. 2004, Zhao et al. 2011, Mari et al. 2015, Perera et al. 2017c) tend to lack structural elements that commonly appear in supply chain management (such as tiers), make stylized assumptions that poorly reflect reality (such as preferential attachment connections; for a discussion, see Atalay et al. 2011), and have too few input parameters to create deep insights. While these models produce graphs that are easy to generate and analyze, they lack the richness required for complex optimization problems.

Conversely, models from economics (e.g., Atalay et al. 2011, Carvalho and Voigtländer 2014, Oberfield 2018, Lim 2018, Taschereau-Dumouchel 2020) tend to be overly complex, requiring

hard-to-obtain quantities such as wages, customization costs, or production and utility functions. Connections often result from solving optimization problems, making graph generation challenging.

RG4SC strikes a balance between these two extremes. It is sufficiently complex to reflect key concepts of supply chain emergence and incorporate the structural elements common to supply chain management, yet simple enough to be easily generated and analyzed. This balance allows for a more nuanced and realistic representation of supply chain networks while maintaining computational tractability, enabling new insights into supply chain optimization problems.

3. Random generative model

This section introduces our generative model, *RG4SC*. § 3.1 describes the model, emphasizing the underlying supply chain management principles of specialization and local connectivity and highlighting our online tool which makes our model readily available for researchers and practitioners. § 3.2 analyzes how *RG4SC*'s input parameters influence the structure of generated supply chain networks and underscores how the design principles enable generating realistic supply chain networks. Finally, § 3.3 provides empirical evidence of *RG4SC*'s ability to accurately represent real supply chain networks, and presents a novel, non-trivial, parameter estimation procedure.

3.1. An intuitive generative model of supply chain networks: *RG4SC*

RG4SC has five input parameters: (i) the expected number n of firms (or activities); (ii) the number k of supply chain tiers; (iii) a probability vector $\alpha = (\alpha_1, \ldots, \alpha_k) \in [0,1]^k$ such that $\sum_{\ell=1}^{k} \alpha_\ell = 1$ describing the distribution of firms across the k tiers; (iv) a set of k vectors of probabilities $q_\ell = (q_\ell^s, q_\ell^{tr}, q_\ell^t) \in [0,1]^3$ with $q_\ell^s + q_\ell^{tr} + q_\ell^t = 1$ for $\ell = 1, \ldots, k-1$, describing the role (source, transit, or sink) that a firm can take; and (v) pairs $(c_{\ell,\ell+1}, p_{\ell,\ell+1}) \in \mathbb{N} \times [0,1]$ for $\ell \in \{1, \ldots, k\}$, characterizing the procedure for forming input-output relationships between firms, described later.

The output, a supply chain network, is a directed graph $G = (\mathcal{N}, \mathcal{E})$ with set of nodes $\mathcal{N}$ and set of edges $\mathcal{E}$. We refer to nodes as firms but they can represent smaller entities, such as individual factories or divisions, or larger entities, such as countries. Edges represent input-output relationships between firms. We refer to the set of all possible graphs generated via *RG4SC* as $\mathcal{G}$.

We next provide an in-depth description of *RG4SC*, summarized in Algorithm 1.

Node generation in *RG4SC*. We generate N nodes, where $N \sim Poisson(n)$. We take N to be random, because this makes *RG4SC* more amenable to theoretical analysis without fundamentally changing the graphs *RG4SC* generates; see Remark 1 below.

Algorithm 1: Random generative process ($RG4SC$).

Data: $k, n \in \mathbb{N}$; $\alpha := (\alpha_1, \ldots, \alpha_k)$ with $\alpha \geq 0$ and $\sum_{\ell=1}^{k} \alpha_\ell = 1$; $q_\ell := (q_\ell^s, q_\ell^{tr}, q_\ell^t)$ with $q_\ell \geq 0$ and $q_\ell^s + q_\ell^{tr} + q_\ell^t = 1 \; \forall \ell = 1, \ldots, k$ and $q_1 = (0, 0, 1)$, $q_k = (1, 0, 0)$; $(c_{\ell,\ell^+1}, p_{\ell,\ell+1}) \in \mathbb{N} \times [0, 1] \; \forall \ell = 1, \ldots, k$

Result: A directed graph $G = (\mathcal{N}, \mathcal{E}) \in \mathcal{G}$.

$N \leftarrow Poisson(n)$;
for $i = 1, \ldots, N$ **do**
  $\mathcal{N} \leftarrow \mathcal{N} \cup \{i\}$;
  $\ell_i \leftarrow L_i \overset{\text{iid}}{\sim} categorical(\alpha_1, \ldots, \alpha_k)$;
  $x_i \leftarrow X_i \overset{\text{iid}}{\sim} U[0, 1]$;
  $r_i \leftarrow R_i \overset{\text{iid}}{\sim} categorical(q_{\ell_i}^s, q_{\ell_i}^{tr}, q_{\ell_i}^t)$;
for $\ell \in \{1, \ldots, k-1\}$ **do**
  for *i s.t.* $\ell_i = \ell$ *and* $r_i \neq t$ **do**
    for *j s.t.* $\ell_j = \ell + 1$ *and* $r_j \neq s$ **do**
      compute $C_i(j)$ and $C_j(i)$ as in (1)
      if $C_i(j) = 1$ *or* $C_j(i) = 1$ **then**
        $\mathcal{E} \leftarrow \mathcal{E} \cup (i, j)$
      else if $C_i(j) \leq c_{\ell,\ell+1}$ *or* $C_j(i) \leq c_{\ell,\ell+1}$ **then**
        $\mathcal{E} \leftarrow \mathcal{E} \cup (i, j)$ with probability $p_{\ell,\ell+1}$

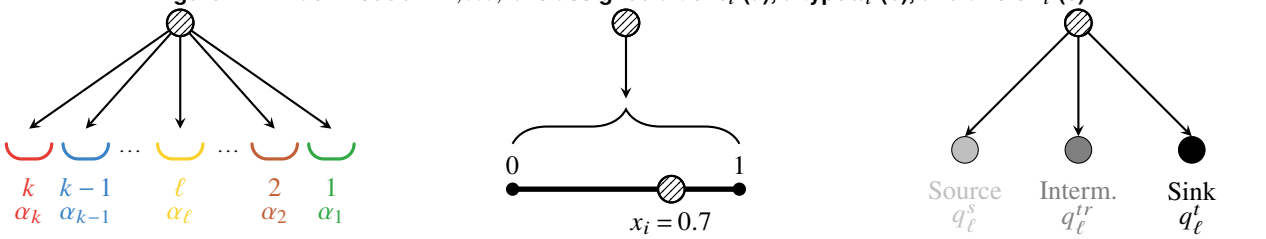
Each node $i \in \{1, \ldots, N\}$ is then assigned three characteristics: a *tier*, a *type*, and a *role*, as depicted in Figure 1. We model tiers, types, and roles through simple user-specified distributions.

The tier $\ell_i \in \{1, \ldots, k\}$ of node i is the production process stage in which i operates, or its *vertical specialization*. A low tier means node i is downstream in its supply chains, while a high tier means node i is upstream. We assume that tier ℓ_i of node i follows a categorical distribution over $\{1, \ldots, k\}$, governed by the user-specified parameter $\alpha = (\alpha_1, \ldots, \alpha_k)$ such that $\sum_{\ell=1}^{k} \alpha_\ell = 1$. In other words, $\ell_i = \ell$ with probability α_ℓ, independently of all other nodes; see Figure 1a.

The type $x_i \in [0, 1]$ of node i is its *horizontal specialization*. This horizontal specialization can, for example, model a firm's geographic location or technical know-how. Type x_i of node i follows a uniform distribution over $[0, 1]$, again, independently of other nodes; see Figure 1b. Assigning types and tiers to firms is consistent with the increased specialization of firms and production processes (Hummels et al. 2001, Baldwin and Freeman 2022), due to the clear economic benefits obtained from specialization, as well as the reduction of typical specialization barriers such as transportation costs, trade barriers, and complex task coordination (Baldwin 2012).

The role r_i of node i is either that of a *source* ($r_i = s$), a *sink* ($r_i = t$), or a *transit* ($r_i = tr$) node. A sink (resp. source) node only has incoming (resp. outgoing) edges and is at the end (resp. beginning)

Figure 1 **Each node i in $1,\dots,N$ is assigned a tier ℓ_i (a), a type x_i (b), and a role r_i (c).**



(a) Node i's tier ℓ_i follows a categorical distribution with parameter $\alpha = (\alpha_1,\dots,\alpha_k)$.

(b) Node i's type x_i follows a uniform distribution on $[0,1]$.

(c) Node i's role r_i conditioned on $\ell_i = \ell$ follows a categorical distribution with parameter $q_\ell = (q^s_\ell, q^{tr}_\ell, q^t_\ell)$.

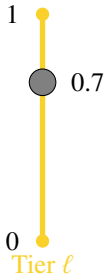
of a production process. A transit node is neither a sink nor a source node and has both incoming and outgoing edges. Node i takes role r_i with a tier-specific vector of probabilities $q_\ell = (q^s_\ell, q^{tr}_\ell, q^t_\ell)$. At each tier $\ell \in \{1,\dots,k\}$, $q^s_\ell + q^{tr}_\ell + q^t_\ell = 1$. Thus, r_i conditional on $\ell_i = \ell$ follows a categorical distribution with parameter q_ℓ; see Figure 1c. Quite naturally, for tier 1, $q^t_1 = 1$ (and $q^s_1 = q^{tr}_1 = 0$), i.e., all nodes are sinks. For tier k, $q^s_k = 1$, i.e., all nodes are sources. The other vectors are again user-specified. We refer to the set of sources as S^s and the set of sinks as S^t.

Figure 2a illustrates the combined tier, type, and role allocation process. Node i's position on the $[0,1]$ segment reflects its type: here, $x_i = 0.7$. The segment's color is its tier: here, $\ell_i = \ell$. Finally, the node's color is its role: here, $r_i = tr$. Figure 2b shows how this allocation process is repeated for all N nodes, with light grey corresponding to source nodes, medium grey to transit nodes, and dark grey to sink nodes. Algorithm 1, lines 1–6, formalizes the process.

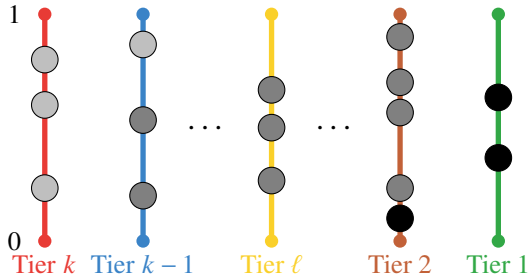
Remark 1. Assuming a random number of nodes N rather than a fixed number n allows us to use Poissonization (Penrose 2003) for our theoretical analyses while keeping very similar graphs (as N is centered around n). In particular, Poissonization enables us to view the number of nodes (with role r) at tier ℓ as independent Poisson random variables with mean $n\alpha_\ell$ ($n\alpha_\ell q^r_\ell$). If we had used a fixed number n, these random variables would have followed multinomial distributions and would not have been independent across tiers and roles. Poissonization also allows us to use two powerful results regarding a firm's type in our analysis: (i) the number of nodes with a type in any subinterval $[a,b] \subseteq [0,1]$ follows a Poisson random variable with mean $n(b-a)$; (ii) the number of nodes that have types in disjoint intervals of $[0,1]$ are independent.

Edge generation in *RG4SC*. Edges represent input-output relationships between firms. They are formed between consecutive tiers ℓ and $\ell+1$, based on the user-specified parameters $(c_{\ell,\ell+1}, p_{\ell,\ell+1}) \in \mathbb{N} \times [0,1]$. Informally, $c_{\ell,\ell+1}$ bounds the number of neighbors a node can connect to in an adjacent

Figure 2 **Graphic illustrations of the assignment process described in Figure 1 for one node (a) and all nodes (b).**



(a) An example of a node i at tier $\ell_i = \ell$ with type $x_i = 0.7$, and role $r_i = tr$.



(b) We repeat the process described in Figure 1 for all nodes $\{1, \ldots, N\}$. Each node has a tier, a type, and a role.

tier ("connectivity range"), while $p_{\ell,\ell+1}$ is the probability of a possible connection being realized. Before formalizing the edge generation process, we introduce a measure of "closeness".

Consider a tier $\ell \in \{1, \ldots, k-1\}$ and a node i on tier ℓ that is not a sink node. Consider also all nodes j on tier $\ell+1$ that are not source nodes and rank these nodes depending on how close they are to i in type. We denote by $C_i(j)$ the rank of node j relative to i in *type*. In other words, if $C_i(j) = u$, then j is the u^{th} closest (in terms of type) non-source node to i on tier $\ell+1$, i.e.,

$$C_i(j) = u \text{ if } j = \arg \min_{\substack{\{j' \mid \ell_{j'}=\ell+1, r_{j'} \neq s, \\ C_i(j') \neq 1, \ldots, C_i(j') \neq u-1\}}} |x_i - x_{j'}|, \tag{1}$$

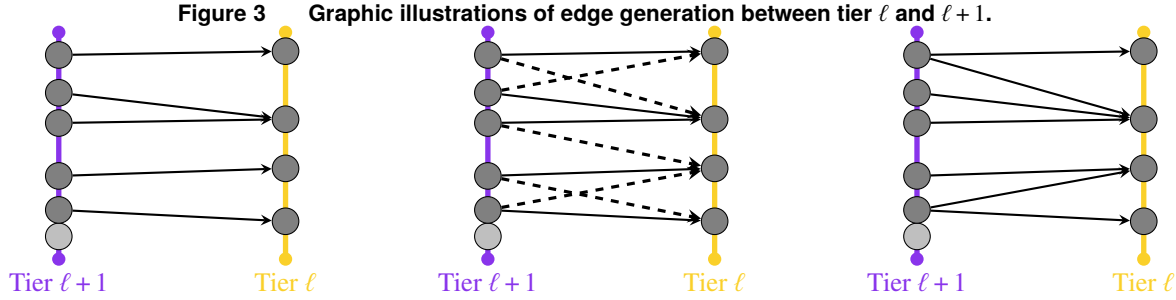
where we may have $C_i(j) \neq C_j(i)$.

It may be challenging for a firm to obtain complete information about all other firms in the supply chain network (Uzzi 2018, Sodhi and Tang 2019). Thus, firms joining a supply chain network are more likely to have knowledge of (and establish relationships with) similarly specialized firms (Gulati et al. 2000). This is particularly evident when the specialization reflects the geographic location. When firms are too far apart in terms of specialization, they will not connect, because it may imply high transaction costs, or because firms are not aware of each other's existence.

Hence, if $C_i(j) = 1$ or $C_j(i) = 1$, that is, node i is closest to j or j is closest to i, the nodes connect with probability 1. This guarantees that the graph does not have isolated nodes and reflects the following intuition: A firm that joins the network does so with the intent of forming input-output relationships with at least one node. This node is naturally the one with the closest specialization to it in an adjacent tier.[3] If $C_i(j) > 1$ and $C_j(i) > 1$ but $C_i(j) \leq c_{\ell,\ell+1}$ or $C_j(i) \leq c_{\ell,\ell+1}$, node i and j connect with probability $p_{\ell,\ell+1}$. If $C_i(j) > c_{\ell,\ell+1}$ and $C_j(i) > c_{\ell,\ell+1}$, node i and j do not connect.

The edge creation process is repeated for all pairs (i, j) of nodes in adjacent tiers, where i (resp. j) is not a sink (resp. source) node. It is formalized in Algorithm 1, lines 7–14 and illustrated in

Figure 3 Graphic illustrations of edge generation between tier ℓ and $\ell+1$.



(a) Nodes that are closest in type connect with probability 1.

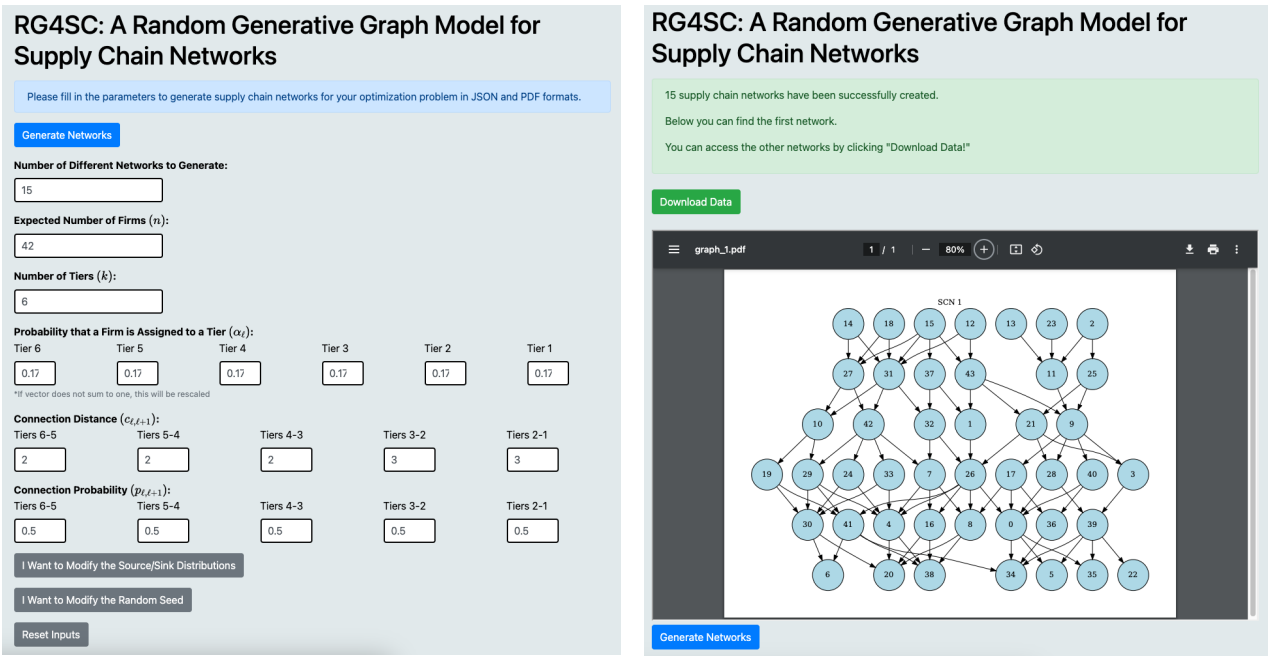
(b) As $c_{\ell,\ell+1} = 2$, nodes consider their two closest neighbors as possible connections.

(c) The dotted edges are added to the graph with probability $p_{\ell,\ell+1}$.

Figure 3. In Figure 3a, all edges between closest nodes on tiers ℓ and $\ell+1$ are added (see lines 11–12 of Algorithm 1). In Figure 3b, we assume that $c_{\ell,\ell+1} = 2$ and indicate with dotted lines the edges that would connect a node to its second-closest neighbor (if that edge has not already been added previously). Edges are sampled from the dotted lines with probability $p_{\ell,\ell+1}$, as indicated in lines 13–14. Figure 3c shows the resulting set of edges.

Accessible implementation of *RG4SC*. We make *RG4SC* easily available to researchers and practitioners, by providing a web interface at `https://scngenerator.pythonanywhere.com/`.[4] Users can input the parameters outlined above to efficiently generate large sets of networks.

Figure 4 Web interface to generate networks with *RG4SC*.



(a) Input interface.

(b) Output interface.

Table 1 Summary of relationships between input parameters and network characteristics.

Characteristic	Result type	n	k	α_ℓ	$c_{\ell,\ell+1}$	$p_{\ell,\ell+1}$
Node degree at ℓ	Asymptotic	=	=	↓	↑	↑
Treewidth	Upper bound	↑	↑	↓	↑	=
Modularity	Numerical	↑	/	/	↓	↓
Assortativity	Numerical	↑	↑	/	/	/

Changes in q_ℓ are omitted as they can be reformulated as changes in n and α. Here, ↑ (resp. ↓, =) means increasing (resp. decreasing, unchanged) and / indicates a more complex relationship.

Figure 4a shows the interface to set *RG4SC*'s input parameters and determine the number of networks to generate. The parameters are initially set to default values for ease of use. We also hide the input fields for the parameter choices that we expect users to adjust less frequently, but these additional fields can easily be displayed as needed by clicking the appropriate buttons.

Figure 4b shows a sample output. The first network generated is shown directly on the webpage. All generated networks can be accessed through a "download" button. This results in a ZIP file that contains (i) a PDF for each generated network, produced using Graphviz (Gansner and North 2000), and (ii) a single JSON file describing the input parameters, and the resulting list of networks, each in node-link format. That is, networks are represented by a dictionary of nodes and a dictionary of edges. Using the JSON format to represent networks makes storing, sharing, and analyzing networks with common graph analysis software packages straightforward.

3.2. Relationship between inputs and supply chain network characteristics

We now establish how *RG4SC*'s input parameters shape the structures of generated networks. This allows users to generate networks with specific characteristics, further supporting applicability.

We explore four crucial structural parameters of the output networks: expected node degree, treewidth, modularity, and assortativity. Node degree, modularity, and assortativity are arguably some of network theory's most fundamental and analyzed graph structural parameters. Modularity and assortativity, in particular, have been analyzed extensively in empirical supply chain networks, and we highlight how the notions of specialization and local connectivity, key to designing our model, help to create networks consistent with the resulting observations. While treewidth may seem a less obvious structural parameter, it plays a crucial role in driving the computational complexity of solution procedures to optimization problems over networks (Blaettchen et al. 2024a).

Table 1 summarizes our results. Proofs and additional figures are in Appendix A, which also includes results relating to other structural characteristics found in the recent supply chain management literature: centrality, accessibility, interconnectedness, and diamond scale.

Node degree. A node's *degree* counts the number of its edges. For directed graphs, the in-degree (resp. out-degree) counts the number of incoming (resp. outgoing) edges only. Besides being a key network metric (Perera et al. 2017a), the node degree is related to several supply chain management insights. For example, Agrawal and Osadchiy (2024) find that inventory productivity is higher for firms with a high degree (and high centrality, which we discuss in Appendix A.6).

Theorem 1 provides asymptotic results for the expected in- and out-degrees of nodes in a network generated by *RG4SC*, which will play a vital role in our estimation procedure discussed in § 3.3.

THEOREM 1. *Let $G \in \mathcal{G}$, and assume $n \to \infty$, while $\lim_{n\to\infty} \frac{c_{\ell,\ell+1}}{n} = 0 \ \forall \ell = 1, \dots, k-1$. Denote $n_\ell^{in} = \alpha_\ell \left(1 - q_\ell^s\right)$, resp. $n_\ell^{out} = \alpha_\ell \left(1 - q_\ell^t\right)$. The expected in-degree at tier $\ell = 1, \dots, k-1$ is*

$$\mathbb{E}\left[D_{in}^{\ell}\right] = (1-q_\ell^s)\left[\frac{n_{\ell+1}^{out}}{n_\ell^{in}}\left(1 + p_{\ell,\ell+1}(c_{\ell,\ell+1}-1)\right) + (1-p_{\ell,\ell+1})\frac{n_\ell^{in}}{n_\ell^{in}+n_{\ell+1}^{out}} + p_{\ell,\ell+1}\sum_{h=1}^{c_{\ell,\ell+1}}\left(1 - B_{\frac{n_{\ell+1}^{out}}{n_\ell^{in}+n_{\ell+1}^{out}}}(h, c_{\ell,\ell+1})\frac{(c_{\ell,\ell+1}+h-1)!}{(c_{\ell,\ell+1}-1)!(h-1)!}\right)\right]$$

where B is the incomplete Beta-function. Meanwhile, the expected out-degree at tier $\ell = 2, \dots, k$ is

$$\mathbb{E}\left[D_{out}^{\ell}\right] = (1-q_\ell^t)\left[\frac{n_{\ell-1}^{in}}{n_\ell^{out}}\left(1 + p_{\ell-1,\ell}(c_{\ell-1,\ell}-1)\right) + (1-p_{\ell-1,\ell})\frac{n_\ell^{out}}{n_\ell^{out}+n_{\ell-1}^{in}} + p_{\ell-1,\ell}\sum_{h=1}^{c_{\ell-1,\ell}}\left(1 - B_{\frac{n_{\ell-1}^{in}}{n_\ell^{out}+n_{\ell-1}^{in}}}(h, c_{\ell-1,\ell})\frac{(c_{\ell-1,\ell}+h-1)!}{(c_{\ell-1,\ell}-1)!(h-1)!}\right)\right].$$

Without assuming $n \to \infty$, for each tier, we would need to consider the distance between the outermost nodes in $[0,1]$ and the boundaries of this segment. This distance does not follow an exponential distribution (different from the distance between two adjacent nodes), preventing an explicit formulation of node degree. This type of "boundary effect" and the resulting asymptotic analysis is common in the random network literature (cf. Jackson 2010).

In Appendix A.2, we run numerical experiments to confirm that for finite n the impact of a boundary effect on the expected degree is small. We find that the above formulation is an adequate approximation, as long as $c_{\ell,\ell+1} < \frac{\min\{\alpha_\ell, \alpha_{\ell+1}\}n}{2}$. Moreover, when $c_{\ell,\ell+1}$ exceeds this threshold, the explicit formulation serves as a lower bound on the expected degree.

The explicit degree formulation allows us to directly identify the impact of varying the model's input parameters: In- and out-degrees both decrease in α_ℓ, but increase in $c_{\ell,\ell+1}$ and $p_{\ell,\ell+1}$. The parameters k and n do not influence the expected degrees due to the independent edge generation across tiers and the proportional impact of changing n on all tiers.

Treewidth. The *treewidth* describes the similarity of a graph to a tree: A tree graph has a treewidth of 1, while a fully connected graph with n nodes has a treewidth of $n-1$. Formally:

DEFINITION 1 (BODLAENDER 1994). Let $G = (\mathcal{N}, \mathcal{E})$ be an undirected graph. A *tree decomposition* of G is a pair $(T, \{X_z\}_{z \in T})$, with tree T and bags X_z for each node $z \in T$, such that:

(a) $\bigcup_{z \in T} X_z = \mathcal{N}$.

(b) If $(i, j) \in \mathcal{E}$, there must be a set X_z with $i, j \in X_z$.

(c) If a node $i \in \mathcal{N}$ appears in two distinct bags X_x and X_y, then it appears in all bags X_z such that z is on the (unique) path between x and y in T.

The tree decomposition's *width* is $\max_{z \in T} |X_z| - 1$. The *treewidth* $tw(G)$ of G is the minimum width over all possible tree decompositions of G. For a directed graph, $tw(G)$ is the treewidth of its undirected counterpart.

Treewidth is a critical structural parameter in optimization since many optimization problems on graphs that are NP-hard can be solved in polynomial time when the underlying graph has bounded treewidth.[5] Examples appear in several contexts, including, e.g., combinatorial optimization. A supply chain management example is Blaettchen et al. (2024a), where the authors analyze the adoption of traceability technologies within supply chain networks.

We provide probabilistic upper bounds on the treewidth of graphs generated from *RG4SC*, depending on the scale of the connectivity ranges $c_{\ell,\ell+1}$. For this purpose, let $\overline{c} = \max_{\ell=1,\dots,k-1} c_{\ell,\ell+1}$. Moreover, denote with $\tilde{\alpha}$ a positive constant that strictly lower-bounds the entries of α, that is, $0 < \tilde{\alpha} < \alpha_\ell \ \forall \ell = 1, \dots, k$. We let $A = \frac{1-\tilde{\alpha}}{\tilde{\alpha} W\left(-\frac{1-\tilde{\alpha}}{e}\right)}$, where W is the product logarithm.

THEOREM 2. *Assume* $G \in \mathcal{G}$.

(i) If $\lim_{n\to\infty} \frac{\overline{c} \log\log n}{\log n} = 0$, *then*

$$\lim_{n\to\infty} P\left(tw(G) \leq (1+\epsilon) A \left(\log n + \overline{c} \log\log n\right)\right) = 1 \ \forall \epsilon > 0.$$

(ii) If $\lim_{n\to\infty} \frac{\overline{c} \log\log n}{\log n} = \infty$, *but* $\lim_{n\to\infty} = \frac{\overline{c} \log\log n}{n} = 0$, *then*

$$\lim_{n\to\infty} P\left(tw(G) \leq (1+\epsilon)^{1/\tilde{\alpha}} \left(\log n + \overline{c} \log\log n\right)\right) = 1 \ \forall \epsilon > 0.$$

Theorem 2, case (i), shows that the treewidth of graphs generated by *RG4SC* grows with the logarithm of n, as long as the connectivity is not too large, for example, when $\overline{c} = \log(n)^{1-\epsilon}$ for some small $\epsilon > 0$. In this case, any procedure that is only exponential in the treewidth can be solved efficiently, in quasi-polynomial time. Case (ii) considers the case where $\overline{c}$ is of a similar magnitude

to n, for example, $\overline{c} = n^{1-\epsilon}$. Here, the treewidth is of that same magnitude, and we can only hope for sub-exponential procedures. We will discuss this in detail in the context of the GSM in § 4.3.

Our numerical experiments reveal that *RG4SC*'s input parameters impact the actual treewidth of generated networks in the same direction as the upper bound. That is, the treewidth decreases as the smallest value α_ℓ increases (e.g., for smaller k), and it increases with $c_{\ell,\ell+1}$ and n. However, the upper bound is less tight as the number of tiers increases (cf. Figure EC.4 in Appendix A.3). Moreover, the treewidth numerically increases with $p_{\ell,\ell+1}$, which does not affect the upper bound.

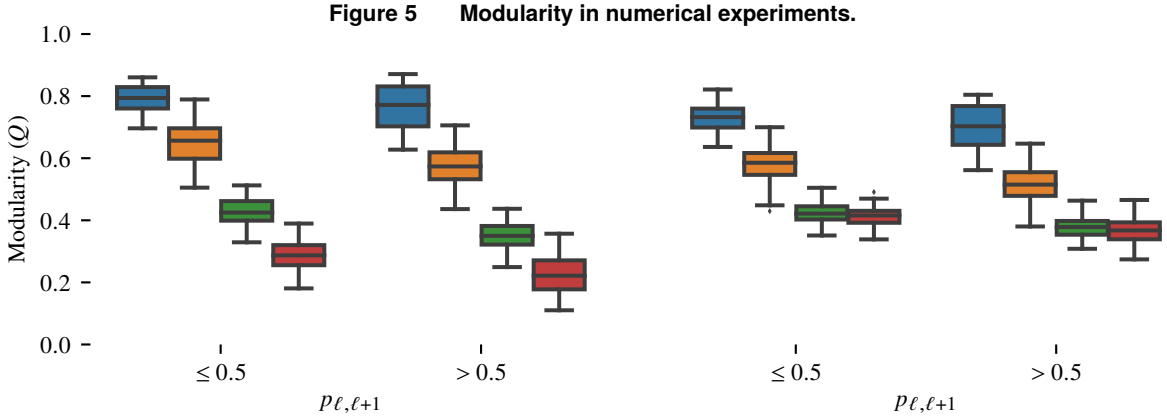
Modularity. A network's *modularity* measures how pronounced sub-communities of nodes are within the network. The most commonly used definition is given in Clauset et al. (2004): A number $Q \in [0, 1]$ that reflects the density of edges inside communities as compared to the density of edges between communities. High (resp. low) values of Q indicate a strongly (resp. weakly) pronounced community structure. Empirically, supply chains have been found to have high modularity, although this is not captured by extant generative models (Perera et al. 2017a).

Our model can generate networks with high modularity because connections are formed locally. In particular, when $c_{\ell,\ell+1}$ is low, nodes across tiers form distinct groups along their horizontal specialization. We illustrate this effect in Appendix A.4. Figure 5 plots the maximum modularity of graphs generated with *RG4SC*, clearly highlighting that local connectivity (low $c_{\ell,\ell+1}$) is crucial to obtaining networks with high modularity. As $c_{\ell,\ell+1}$ increases, we observe a distinct decrease in modularity. With high $p_{\ell,\ell+1}$, linkages between otherwise distant communities are more likely, so the decrease is more pronounced. On the other hand, the decrease is dampened when there are more tiers, because tiers themselves can serve as the basis of a community structure, leading to a complex impact of the number of tiers k on modularity. The impact of the values α_ℓ on modularity is yet more complex and heavily dependent on other parameters.

(Dis)assortativity. The *assortativity* of a network measures the similarity of its connections: a network will be highly assortative if low-degree nodes connect to low-degree nodes and high-degree nodes connect to high-degree nodes. Conversely, a network will be disassortative if high-degree nodes connect to low-degree nodes. Supply chain networks tend to be disassortative, but existing random models cannot easily create such topologies (Perera et al. 2017a,b).

In contrast, due to local connectivity by similarly specialized nodes, our model naturally generates disassortative networks, as we describe in detail in Appendix A.5. Figure 6 plots the undirected assortativity measure that correlates the overall node degrees of connected nodes, for different

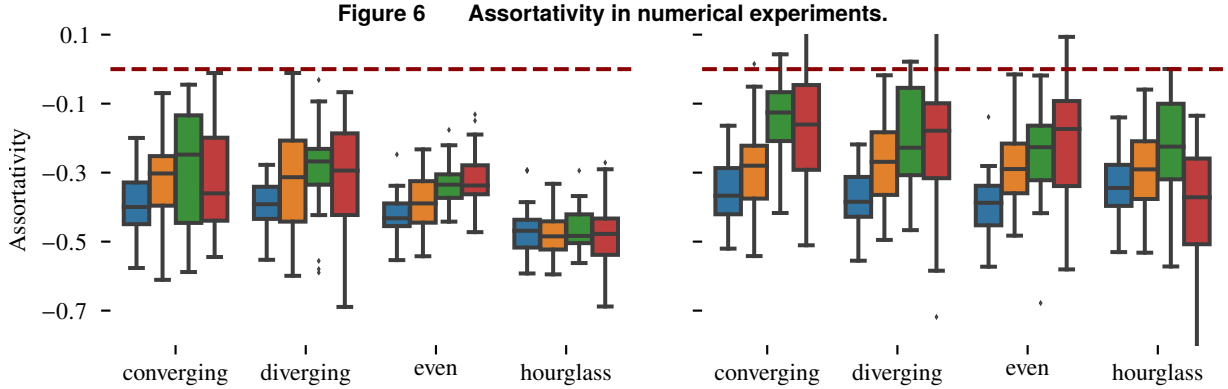
Figure 5 Modularity in numerical experiments.



(a) $k = 4$ (b) $k = 8$

Note. Data is generated as described in Appendix A.1. Bars from left to right indicate $c_{\ell,\ell+1} \in \{1,2\},\{3,4,5\},\{10\},\{15,20\}$.

Figure 6 Assortativity in numerical experiments.



(a) $k = 4$ (b) $k = 8$

Note. Data is generated as described in Appendix A.1, selecting $p_{\ell,\ell+1} = 1$. The appendix also describes the structural archetypes. Bars from left to right indicate $c_{\ell,\ell+1} \in \{1,2\},\{3,4,5\},\{10\},\{15,20\}$.

values of k and $c_{\ell,\ell+1}$, and different prototypical tier structures. A tier structure is a vector α that leads to prototypical networks found throughout the supply chain management literature. For example, if α is increasing from tier 1 to tier k leads to a "converging" or "assembly" structure.

Figure 6 highlights that the tier structure plays a crucial role in determining assortativity. However, differences between structures diminish as k increases. Meanwhile, the parameters $c_{\ell,\ell+1}$ and $p_{\ell,\ell+1}$ impact assortativity in an intricate manner, that is highly dependent on other parameters. Importantly, although assortativity is generally decreasing in $c_{\ell,\ell+1}$ in an even network with two tiers, when we consider multiple tiers, a more complex behavior emerges.

Since *RG4SC* generates directed graphs, assortativity can also be defined by correlating in-degrees, out-degrees with in-degrees, etc. While these assortativity measures behave differently,

all measures tend to be negative for networks output by *RG4SC*, consistent with the observed disassortativity of real-world supply chain networks (Perera et al. 2017a).

3.3. Fit with real-world supply chain network data

We now provide evidence for the hypothesis that *RG4SC* is a plausible model for the Willems (2008) data, describing real-world supply chain networks at the level of individual processing steps. This level of granularity is crucial for many optimization problems, and the data in Willems (2008) was collected to study safety stock placement models. Additionally, the dataset is known to contain relatively large networks, including nodes at "deep" tiers (Perera et al. 2017b). In Appendix B, we further validate *RG4SC* by showing that it is also a good fit for a different firm-level dataset of supply chain networks from the pharmaceutical industry.

We follow a systematic approach adapted from Clauset et al. (2009) to show that our hypothesis cannot be rejected. Central to this is the ability to estimate *RG4SC*'s parameters from empirical networks, such as the ones in Willems (2008). Hence, we first introduce an efficient estimation procedure. Proofs and additional details for this subsection can be found in Appendix B.

Parameter estimation. Suppose we are given a directed acyclic and tiered graph G. What estimated values of the input parameters $(n, k, \alpha, \{q_\ell\}_\ell, \{c_{\ell,\ell+1}\}_\ell, \{p_{\ell,\ell+1}\}_\ell)$ should we pick so that G is most likely to be a realization of *RG4SC*'s random generation process?

Conventional approaches to estimating random graph models rely on optimizing goodness-of-fit metrics, such as the likelihood of the degree distribution. However, this presents challenges for our model due to the inability to express the likelihood in closed form, stemming from the correlation between nodes (e.g., a node may have two suppliers that share the same supplier). Conversely, relying solely on simulated goodness-of-fit metrics incurs a substantial computational cost.

To strike a balance, we adopt a hybrid approach similar to Wan et al. (2017), combining the method of moments for most parameters with the optimization of a simulated goodness-of-fit metric for others. Following recent advances in network fitting (Shore and Lubin 2015, de Siqueira et al. 2021), we employ the spectral distance between the original and simulated networks as our metric.

Estimation of the node-generating process. We start by estimating $\hat{n}$, $\hat{k}$, $\hat{\alpha}$, and $\{\hat{q}_\ell\}_\ell$. We equate these values to the observed number of nodes and tiers, and the proportions of nodes per tier and nodes per tier with and without in- and out-connections, respectively. These are consistent estimators, as long as $\alpha_\ell > 0 \; \forall \ell = 1, \ldots, k$.

Estimation of the edge-generating process. We note that, conditional on the nodes and their specifications, the edge-generation process between a pair of tiers is independent of the edge-generation process between another pair of tiers, even if these pairs overlap. Hence, we can individually estimate $(c_{\ell,\ell+1}, p_{\ell,\ell+1})$ for each tier $\ell = 1, \ldots, k-1$.

Without access to the likelihood of the degree distribution, the method of moments on the expected in- and out-degrees—which we can define (approximately) in closed form based on our asymptotic analysis—seems an obvious choice. However, this is challenging to do in practice, because the connectivity range $c_{\ell,\ell+1}$ and the probability of connecting $p_{\ell,\ell+1}$ act as substitutes when it comes to a node's degree. Indeed, a high connectivity range but a low probability of realizing connections within the range can lead to a similar number of connections as a low range combined with a high probability of realizing connections. As it is hard to disentangle the effect of the two quantities on the degree of a node, the method of moments is likely to fail.

As a workaround, we first fix a value of $p_{\ell,\ell+1}$ and only estimate $c_{\ell,\ell+1}$ using the expected in- and out-degrees. Given the other parameters, and under the assumptions of Theorem 1, both moments are strictly increasing functions of $c_{\ell,\ell+1}$. Hence, we can identify $\hat{c}_{\ell,\ell+1}$ via bisection from either of the moments. Moreover, due to the Continuous Mapping Theorem, the estimators are consistent. Finally, as we have two estimates, we can average across them to improve robustness.

To formalize this, we denote with f_{in}^{ℓ} (resp. $f_{out}^{\ell+1}$) a continuous generalization of $\mathbb{E}\left[D_{in}^{\ell}\right]$ (resp. $\mathbb{E}\left[D_{out}^{\ell+1}\right]$) from Theorem, 1, i.e., assuming $n \to \infty$. The generalization takes arbitrary values of $c_{\ell,\ell+1} \geq 1$ by linearly interpolating between adjacent integers. We also denote with $\hat{D}_{in}^{\ell}$ (resp. $\hat{D}_{out}^{\ell+1}$) the sample average of the in-degree of nodes at tier ℓ (resp. the out-degree of nodes at tier $\ell+1$).

Proposition 1. *The function f_{in}^{ℓ} (resp. $f_{out}^{\ell+1}$) is strictly increasing in $c_{\ell,\ell+1}$. Let $\hat{c}_{\ell,\ell+1}$ be the unique solution to $\hat{D}_{in}^{\ell} = f_{in}^{\ell}$ (resp. $\hat{D}_{out}^{\ell+1} = f_{out}^{\ell+1}$). Under the conditions of Theorem 1, and given all other model parameters, $\hat{c}_{\ell,\ell+1}$ is a consistent estimator of $c_{\ell,\ell+1}$.*

To estimate $p_{\ell,\ell+1}$, we repeat this process for multiple possible values. We delay the details of the iteration and first discuss how we choose among multiple candidate values of $\hat{p}_{\ell,\ell+1}$. In particular, we minimize the *spectral distance* between networks produced by our estimated partial (two-tier) model and the (partial) original network. We follow Gu et al. (2015) to operationalize the spectral distance. First, we define the normalized Laplacian spectrum and the resulting *spectral measure*:

Definition 2. For a graph G with n nodes, take its undirected version and denote with A the adjacency matrix and with D the (diagonal) degree matrix. Let $L = D - A$ and $D^{-1/2}$ the result of

applying $1/\sqrt{\cdot}$ to all non-zero elements of D. Then, $\mathcal{L} = D^{-1/2}LD^{-1/2}$ is the normalized Laplacian matrix and the set of eigenvalues $\sigma(G) = \{\lambda_1, \ldots, \lambda_n\}$ its spectrum. Moreover, $\mu_{\sigma(G)} \overset{\text{Def.}}{=} \frac{1}{n}\sum_{i=1}^{n} \delta_{\lambda_i}$ is the spectral measure of G. Here, δ_{λ_i} is the Dirac measure concentrated on eigenvalue λ_i.

The normalized Laplacian is ideally suited when graphs have different numbers of nodes because eigenvalues are in $[0, 2]$ (Wills and Meyer 2020). We next define the *spectral distance*:

DEFINITION 3. Let G_1 and G_2 be two graphs. The spectral distance between G_1 and G_2 is the first Wasserstein distance between their spectral measures, that is, $d(G_1, G_2) = \inf_{\pi \in \Pi(\mu_{\sigma(G_1)}, \mu_{\sigma(G_2)})} \int_{[0,2]} \int_{[0,2]} |x - y| d\pi(x, y)$.

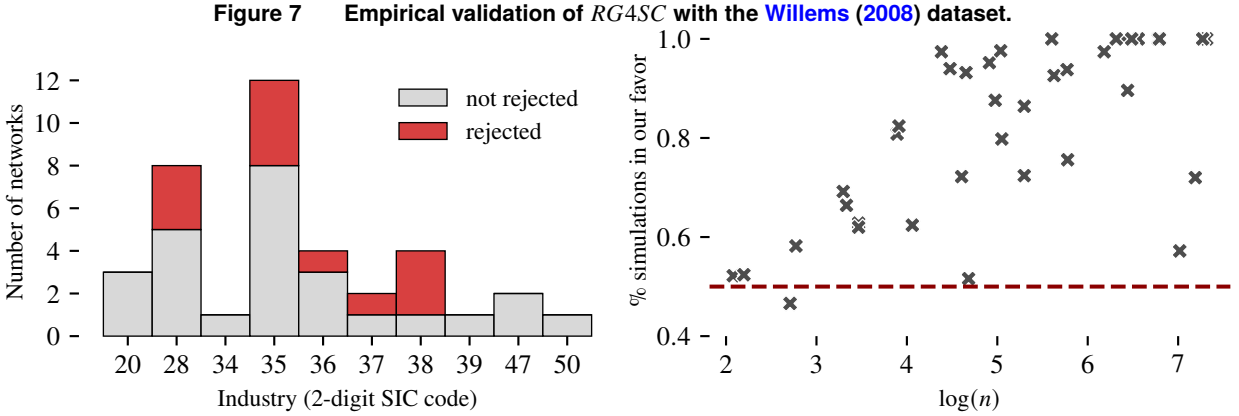
Finally, we could enumerate the interval $\hat{p}_{\ell,\ell+1} \in [0, 1]$ with a granular grid. However, we find numerically that the spectral distance is unimodular in $\hat{p}_{\ell,\ell+1}$, so we reduce the computational burden by applying the golden-section search, without reducing the estimation quality. We exemplify the unimodularity in Appendix B.2, where we also numerically compare our approach with a full iteration. The appendix also analyzes the quality of our estimation procedure for simulated networks.

Empirical validation. To establish that *RG4SC* is a plausible model for the networks in Willems (2008), we evaluate qualitatively that networks created by *RG4SC* — after estimating the parameters from the data — exhibit similar structural characteristics. We present an example in Appendix B.3.

Next, we establish this insight more robustly, similar to Clauset et al. (2009). For this, we slightly modify the datasets' networks, as outlined in Appendix B.3, where we also provide additional details of the procedure. For each network G we follow three steps:

1. *Estimation*. We estimate the input parameters to *RG4SC* from G.
2. *Hypothesis testing*. From the estimated model, we generate a set of networks $\mathcal{G}_G$. We then estimate the input parameters of those networks and compute their goodness-of-fit metric (that is, the spectral distance relative to simulations based on the newly estimated parameters). This results in a distribution of goodness-of-fit metrics. Then, we compute the goodness-of-fit metric of G (relative to a separate set of simulations) and compare this to the distribution of metrics, in order to derive a p-value. We reject the hypothesis that *RG4SC* is a plausible model for G when the p-value is smaller than 0.05.
3. *Eliminating alternative hypotheses*. We consider that G may alternatively be drawn from one of two random attachment models, similar to the one introduced in Erdős and Rényi (1959), which frequently serves as a baseline in network analysis (Newman et al. 2002). We compare *RG4SC* to an alternative model by fitting the latter model to the same data, then generating pairs of networks (one from *RG4SC*, the other from the alternative model). We compare each

Figure 7 Empirical validation of *RG4SC* with the Willems (2008) dataset.



(a) Hypothesis testing.

(b) Eliminating alternative hypothesis ER_T.

pair, by computing the spectral between the original network and either of the simulations, then deciding which model leads to a smaller distance.

The hypothesis that *RG4SC* is a plausible model cannot be rejected for 26 out of 38 networks. We analyze rejection cases to understand whether the cause may lie in systematic differences. We identify one network where nodes at the final tier have exactly one outgoing neighbor each, which in turn only has that node as its incoming neighbor. This structure cannot be captured accurately by our model by design. Otherwise, there are no distinct qualitative differences between the original networks in the data and the networks generated by *RG4SC*, even where the hypothesis is rejected. As shown in Figure 7a, rejections are not systematically tied to certain industries either. Moreover, we present an alternative testing procedure in Appendix B.3, with similar results.

While the results from the second step indicate that *RG4SC* is a plausible fit for the data, we also want to establish that it is better at explaining the data than alternative models. Figure 7b shows that networks generated by a fitted Erdős and Rényi (1959)-type model, adjusted to account for a tiered structure, are more dissimilar from the original network than networks generated by *RG4SC*.

This dissimilarity is particularly pronounced as the network scale (n) grows: Micro-structures become more distinct when there are more nodes. Erdős and Rényi (1959)-type models cannot capture such structures since they treat connections between two tiers equally. *RG4SC*, on the other hand, enables the emergence of local structures due to the horizontal specialization and local connectivity of nodes (see §3.2). Appendix B.3 discusses additional comparisons.

In Appendix B.4, we repeat the validation procedure for an alternative, firm-level dataset. Following Attari et al. (2023), we use supply chain networks derived from the ARCOS database. This publicly available database contains information on all transactions of controlled substances

between firms involved in drug supply chains (Rich et al. 2023). We find that the hypothesis that *RG*4*SC* is a plausible model for networks from the ARCOS dataset cannot be rejected in 83% of cases. Moreover, we can eliminate the alternative hypothesis of a tiered Erdős and Rényi (1959)-type model in 96% of cases. These findings further validate our model.

4. Case study: The Guaranteed Service Model

We now demonstrate *RG*4*SC*'s utility as a foundational tool for supporting the analysis of NP-hard optimization problems in supply chain management. We select the Guaranteed Service Model (GSM) as a case study, given its widespread practical application (Schoenmeyr and Graves 2022) and the rich academic literature dedicated to its analysis (Eruguz et al. 2016).

We review the GSM in § 4.1. Then, in § 4.2, we showcase how *RG*4*SC* generates varied networks that allow assessing how managerial insights from stylized structures carry over to complex structures, and which additional nuances emerge. In § 4.3, we revisit exact algorithmic approaches. We show how to evaluate their complexity, using the relationships between *SC*4*RG*'s input parameters and the resulting networks, and by extrapolating from limited data. Finally, in § 4.4, we discuss heuristics and data-driven algorithms. We illustrate that *RG*4*SC* provides sufficient levers to create rich benchmarks. Moreover, it can create data to effectively train machine learning models. This section's proofs and supplemental details are in Appendix C.

4.1. The Guaranteed Service Model

The GSM determines safety stock placement within a directed acyclic supply chain network $G = (\mathcal{N}, \mathcal{E})$. It can be formulated as follows (for a complete derivation, see Graves and Willems 2000):

$$\begin{aligned} \min_{S_i, SI_i,\, i\in\mathcal{N}} \quad & \sum_{i\in\mathcal{N}} h_i \left(D_i(SI_i + T_i - S_i) - (SI_i + T_i - S_i)\mu_i\right) \\ \text{s.t.} \quad & S_i - SI_i \le T_i, \ i \in \mathcal{N} \\ & SI_j - S_i \ge 0, \ (i,j) \in \mathcal{E} \\ & S_i \le s_i, \ i \in F \\ & S_i, SI_i \ge 0 \text{ and integer}, \ i \in \mathcal{N}. \end{aligned} \tag{2}$$

Here F is the set of demand nodes, S_i, SI_i are the decision variables ("outbound", resp. "inbound" service times), h_i is the per-unit holding cost for inventory at stage i, s_i is the maximum service time for the demand node i, T_i is a known deterministic production lead-time, and μ_i is the average demand per period. Finally, the functions D_i upper-bounding the demand for which node i guarantees fulfillment within time S_i, are assumed to be concave and increasing.

For the special case where G is a tree, Graves and Willems (2000) provide a pseudo-polynomial algorithm with complexity $O(|\mathcal{N}|M^2)$, where M is the maximum replenishment time along a path, that is, $M = \max_{p \in paths(G)} \sum_{i \in p} T_i \leq \sum_{i \in \mathcal{N}} T_i$. This is improved upon by Lesnaia (2004), who provides a polynomial-time algorithm without dependence on the maximum replenishment time. Humair and Willems (2006) provide a dynamic programming approach that extends the GSM to networks that are similar to trees, but where each tree node can consist of a bipartite subgraph, so-called *clusters of commonality*. The runtime of their algorithm is driven by the size of the largest of those subgraphs (and the maximum replenishment time). However, Lesnaia (2004) shows that the GSM is NP-hard for general acyclic networks. Several papers consider such a general acyclic network and aim to either improve the runtime of a mixed-integer program to solve the GSM (e.g., Magnanti et al. 2006), or provide effective heuristic approaches (e.g., Shu and Karimi 2009).

REMARK 2. Problem (2) assumes a single decision-maker. In practice, although a firm will usually control multiple stages in a supply chain, there are often several decision-makers. Schoenmeyr and Graves (2022) shows that the GSM can easily be adapted: They consider two decision-makers that each control a subset of nodes. As long as the decision-makers can agree on the "appropriate" service times of boundary nodes, each will manage their part as in the global optimum. Moreover, a simple contract will lead to the appropriate boundary service times under mild assumptions.

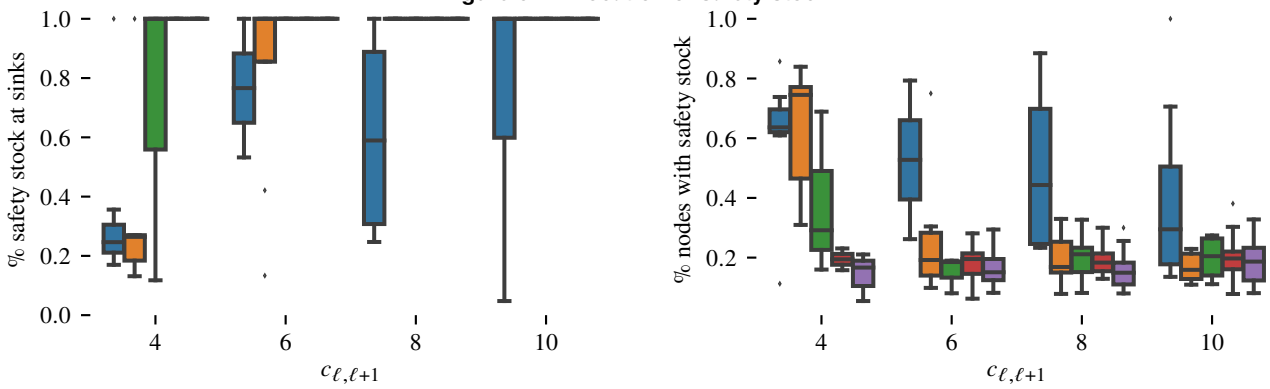
4.2. Managerial insights

Researchers often assume stylized supply chain structures to obtain managerial insights. For example, to explore complex supplier-buyer interactions, they typically consider a simple supply chain consisting of one manufacturer and one retailer (Cachon 2003). In fact, de Kok et al. (2018) find that 62% of papers studying multi-echelon inventory management assume two-tiered networks.

RG4SC is a valuable tool for investigating whether managerial insights *scale* and remain true for real-world networks with many tiers and firms, and to identify novel insights. Researchers can use *RG4SC* to systematically create network structures with controlled characteristics and numerically analyze optimization problems and their solutions. The principled approach *RG4SC* uses to generate large numbers of networks makes it easier to validate and find interpretable and replicable insights, compared to generating network structures ad-hoc. In addition, the insights' practical relevance is supported by *RG4SC*'s micro-foundations and validation with empirical data.

To illustrate how to use *RG4SC* for scaling existing managerial insights and uncovering novel ones, consider a fundamental question of the GSM: Can safety stock be placed in just a handful of

Figure 8 Location of safety stock.



(a) Safety stock at sinks.

(b) Nodes with safety stock.

Note. Only instances with an "even" tier structure, and with an inventory holding cost increase factor of 1.5 are shown. Bars from left to right indicate $p_{\ell,\ell+1} = 0.2, 0.4, 0.6, 0.8, 1.0$.

nodes? Placing safety stock in only a few nodes could be much simpler to implement and might benefit from economies of scale not explicitly modeled in the GSM.

For two-tier networks, the answer is straightforward: If demand nodes face independent external demands and inventory costs are the same in both tiers, all safety stock should be held in the demand nodes. Even if inventory holding costs are higher at the demand tier than upstream tiers, safety stock will be held at the demand nodes as long as each demand node has sufficiently many suppliers.

We use *RG*4*SC* to investigate if this insight extends to general network structures. Namely, for larger multi-tier networks, we study if it is still more likely that safety stock will be held at demand nodes when the network is more connected (i.e., for higher values of $c_{\ell,\ell+1}$ and $p_{\ell,\ell+1}$ in the case of *RG*4*SC*). To do so, we systematically generate network structures and solve the GSM based on the process described in Appendix C.1. We focus on instances where demand functions are generated with the "pooling" mechanism and where $c_{\ell,\ell+1} > 2$. We retain 7,406 instances.

Figure 8a indicates how increased connectivity indeed counteracts an increase in inventory costs moving downstream, driving safety stock away from demand. With all other parameters fixed, we are usually able to identify a threshold in $c_{\ell,\ell+1}$ such that instances above the threshold only see safety stock at sink nodes, while instances below the threshold see safety stock at other nodes.

In addition to validating the general idea that more connected networks are more likely to have all safety stock at the demand nodes, we also find that the solution for less connected networks may be particularly undesirable from a managerial perspective. As shown in Figure 8b, a shift in inventory away from the final tier (low $c_{\ell,\ell+1}$) can imply more nodes with safety stocks, although

tiers are of comparable size. This may be more complex to operationalize and reduces economies of scale. Our insight is particularly pronounced for diverging networks, which are not shown here.

4.3. Exact solution algorithms

A common approach for tackling NP-hard problems is to characterize specific structures where a solution can be found, e.g., in pseudo-polynomial time. Alternatively, one may take a mixed-integer programming (MIP) approach without any guarantee on the time for finding a solutions.

Researchers have used both of these approaches to solve the GSM. Pseudo-polynomial time algorithms were designed for GSM instances where the underlying graph G is a tree (Graves and Willems 2000), or an extension of a tree with clusters of commonality (Humair and Willems 2006). While MIP approaches are also common (see, e.g., Magnanti et al. 2006) they quickly become intractable as the graph density increases (Shu and Karimi 2009).

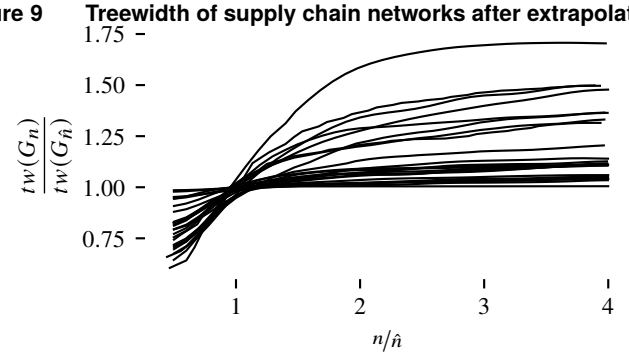
When optimization problems are defined over graphs, as in (2), there is frequently a structural parameter of the graph characterizing the computational complexity of exact solution algorithms. For example, an increasingly popular approach is to derive exact algorithms whose runtime is exponential in the graph's treewidth but linear in the number of nodes (Bodlaender 2006, Jagabathula and Rusmevichientong 2019, Blaettchen et al. 2024a). Such approaches allow users to determine how "hard" an optimization problem is by examining the underlying graph's structure.

Recently, Blaettchen et al. (2024b) used this approach to obtain a new solution procedure for the GSM in general networks. This procedure, which generalizes previous exact algorithms, solves (2) exactly and efficiently for practical cases where the underlying supply chain networks are not exact trees but have low treewidth. *RG*4*SC* is valuable for examining solution approaches such as the one in Blaettchen et al. (2024b), since it can generate supply chain networks with different structures, helping to test and characterize the limits of the proposed approach.

We discuss two methods to employ *RG*4*SC* in this context. First, we can use analytical results linking input parameters to resulting network structures, to establish a direct link between the micro behaviors reflected in *RG*4*SC* and the runtime of an exact algorithm. For example, we can upper-bound the runtime of the procedure in Blaettchen et al. (2024b) as a function of *RG*4*SC*'s input parameters by using Theorem 2. Recall that Theorem 2 shows that the treewidth is low when the connectivity range $c_{\ell,\ell+1}$ is low. This means that we can provide an explicit runtime guarantee for solving the GSM on networks generated by *RG*4*SC*, based on the connectivity range:

COROLLARY 1. *Consider a network $G \in \mathcal{G}$ generated by RG4SC with $\overline{c} = \max_{\ell=1,\dots,k-1} c_{\ell,\ell+1}$, and assume that the maximum replenishment time M is a constant.*

Figure 9 **Treewidth of supply chain networks after extrapolating n.**



Note. For clarity, only networks with the 1-digit SIC code "Heavy manufacturing" are shown. For each value of n, we simulate 96 networks, then take the average treewidth. The resulting lines are smoothed with a Gaussian kernel.

(i) If $\lim_{n\to\infty} \frac{\bar{c}\log\log n}{\log n} = 0$*, with high probability,* (2) *can be solved in quasi-polynomial time.*
(ii) If $\lim_{n\to\infty} \frac{\bar{c}\log\log n}{\log n} = \infty$*, but* $\lim_{n\to\infty} \frac{\bar{c}\log\log n}{n} = 0$*, with high probability,* (2) *can be solved in sub-exponential time.*

The first method can be particularly helpful to study the types of networks benefiting or impeding exact solution algorithms. For example, a low connectivity range speaks to networks where firms seek out very similarly specialized partners. This might be the case in industries with complex technological requirements and deep buyer-supplier relationships, such as the aerospace industry.

Naturally, the first method is limited by the ability to show analytical links between inputs and relevant structural parameters. However, we can also use *RG*4*SC* to empirically study the structural parameters driving algorithmic complexity, by extrapolating network structures from limited data. This is particularly relevant as there is a dearth of supply chain network data. For example, consider a (SIC code) industry with only few sample supply chain networks in the Willems (2008) dataset. We can use the procedure from § 3.3 to estimate the *RG*4*SC* input parameters that best describe the data available. We can then investigate how the complexity of solving the GSM changes as we vary one input parameter, keeping the others fixed. For example, we might change the scale of the networks in the dataset (n) and observe how the relevant structural parameters change.

We illustrate this in Figure 9, where we estimate *RG*4*SC* input parameters for networks from the "Heavy Manufacturing" industry codes in Willems (2008). We then replace the estimate $\hat{n}$ by some other value n to generate new supply chain networks of increasing size. We find that the treewidth of the resulting networks increases logarithmically as n increases. Importantly, while we are considering the treewidth based on the procedure by Blaettchen et al. (2024b), other structural parameters (for which we do not have analytical results) can be studied in the same manner.

4.4. Heuristics and Data-Driven Algorithms

Many supply chain optimization problems can only be solved for complex network structures through heuristic approaches. We differentiate here between two types: classic heuristics, usually derived from relaxing an otherwise intractable MIP formulation, and data-driven algorithms, usually machine learning models, that are trained on realistic network data to derive solutions.

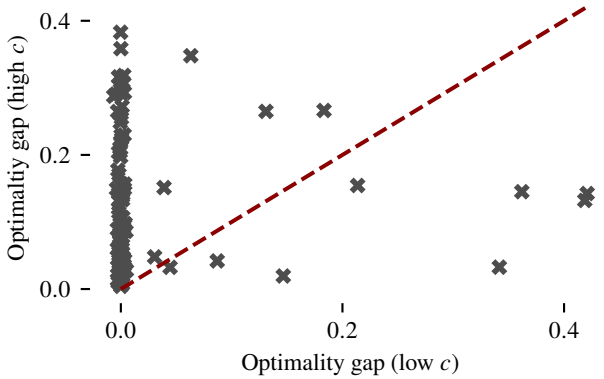
First, consider Shu and Karimi (2009), who propose multiple heuristics for (2), and show that they are particularly valuable for dense networks, with many edges per node. The authors introduce an LP-based heuristic that consistently achieves an optimality gap of 6% with negligible solving time for randomly generated networks with different densities. The exact solution approach introduced in Magnanti et al. (2006) is the benchmark in Shu and Karimi (2009), and the authors find that the time taken to solve the GSM exactly increases exponentially in the network's density.

As discussed in § 3.2, *RG*4*SC* can generate networks with a controlled level of different structural properties. These properties can be crucial dimensions for analyzing a heuristic's performance. We illustrate the value of *RG*4*SC* in generating powerful and varied benchmarks by examining the claim that an LP-based heuristic effectively approximates (2). In particular, we use *RG*4*SC* to generate supply chain networks on which we solve the GSM using the LP-based heuristic in Shu and Karimi (2009). To obtain exact solutions, we use the Baron solver, which dynamically chooses among different relaxation options (Zhou et al. 2018), thus generalizing the approach in Magnanti et al. (2006). We employ the instance generation process described in Appendix C.1 and define the "optimality gap" of the LP-based heuristic as the relative gap between the heuristic solution and the best upper bound from Baron. Because we allowed for solutions from Baron that are within 10% of optimality, the gap is negative in 590 instances. We eliminate these to retain 30,409 instances.

While our results confirm that the (fast) LP-based heuristic may achieve low optimality gaps even for dense networks, the use of *RG*4*SC* reveals a crucial novel insight: For networks with similar densities, the LP-based heuristic is effective when $c_{\ell,\ell+1}$ is low, but it can be highly ineffective when $c_{\ell,\ell+1}$ is high. Thus, the number of neighbors a node can connect to (determined by the connectivity range $c_{\ell,\ell+1}$) critically affects the LP heuristic's performance.

Figure 10 depicts this insight by matching instances. We consider all combinations of GSM-related parameters, k, and α. For each combination, we consider simultaneously all networks where the realized number of nodes (resp. edges) is within an interval of length ten. Then, we take the highest and lowest values of $c_{\ell,\ell+1}$ for any of the networks in a set (recalling that $c_{\ell,\ell+1}$ are identical

Figure 10 Optimality gap of iterative LP-heuristic as a function of c.



Note. Horizontal jitter, distributed with $Normal(0.0.002)$, is added to visually differentiate points at Optimality gap (low c) = 0.

across tiers ℓ). We compute the average optimality gap for the networks in the set that were generated with that highest (resp. lowest) value of $c_{\ell,\ell+1}$. Finally, we contrast these two gaps.

As Figure 10 shows, the average optimality gap is higher for networks with high $c_{\ell,\ell+1}$ even when instances are otherwise nearly perfectly comparable. In particular, 93% of cases lie above the diagonal line, and the average gap across sets is 12% for high $c_{\ell,\ell+1}$, and 2% for low $c_{\ell,\ell+1}$.

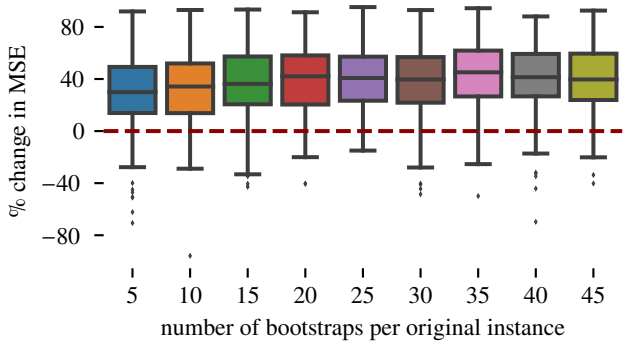
Our findings indicate a broader issue: When $c_{\ell,\ell+1}$ is low, *RG4SC* generates networks with low treewidth, which can be solved exactly (see § 4.3). Hence, a desirable heuristic approach should have a small optimality gap for high $c_{\ell,\ell+1}$, which is not the case for the LP heuristic.

We now turn to data-driven algorithms. For example, Gijsbrechts et al. (2022) evaluate deep reinforcement learning for inventory management. We are not aware of machine learning models for tackling the GSM specifically, so we create a novel experiment to illustrate how *SG4RC* can be used to create relevant training data, by providing synthetic datasets or bootstrapping real data.

Motivated by the managerial insights from § 4.2, we train machine learning models to predict the optimality gap between the GSM with the additional constraint that all inventory needs to be held at demand nodes, and the original formulation. Our objective is to make effective predictions for networks from the Willems (2008) dataset that have not been seen by the model.

We compare three training regimes. In the first regime, we train the model on other networks from Willems (2008). In the second regime, we train the model on a large number of synthetic networks generated by *RG4SC* with arbitrary input parameters. In the third regime, we train the model on bootstrapped versions of the networks from the first one. In particular, we estimate the input parameters for these networks with the procedure from § 3.3, then generate a large number of new networks from *RG4SC* using the input parameters. As we detail in Appendix C.3, we run several

Figure 11 **Change in MSE of training regime 3 compared to regime 1.**



Note. The change in MSE is relative to the higher value. That is, if the MSE of regime 3 is zero, but that of regime 1 is not, the relative change is 100%. Vice versa, if the MSE of regime 3 is positive, but that of regime 1 is zero, the relative change is −100%.

experiments, corresponding to different choices of the GSM parameters. For each experiment, we predict the optimality gap on the test set, then compare the resulting mean-square error (MSE).

We observe that the second regime leads to a lower MSE in 61% of cases. Note that this is although the model is trained using networks with arbitrary structures that may not resemble the testing data at all. When we bootstrap the existing data, likely to be more representative of the test set, we obtain a lower MSE in 84 % of the cases. We also analyze in detail how the extent of bootstrapping supports the training process. Figure 11 shows the relative change in MSE across experiments as the number of bootstraps per instance is increased. Noticeably, even a small number of bootstraps is sufficient to drastically improve performance.

5. Conclusion

Over the past decades, the supply chain management community has developed increasingly sophisticated theoretical tools, heuristics, and managerial insights to address complex, often NP-hard, supply chain optimization problems. However, the scarcity of real-world supply chain network data has constrained our ability to evaluate and refine these approaches.

We fill this gap by introducing *RG*4*SC*, a random generative model of supply chain networks, that provides "test tracks" for supply chain management. *RG*4*SC* has simple micro-foundations, yet outputs networks matching those encountered in practice. It allows creating an unlimited variety of networks, scaling up and extrapolating from existing data, and deriving novel analytical results based on relationships between its interpretable input parameters and the generated network structures. We make *RG*4*SC* readily accessible to researchers through a web interface and a Python package.

To enhance *RG*4*SC*'s usability, we establish clear relationships between its input parameters, reflecting the micro foundations of supply chain network formation, and various structural characteristics of the output networks, including treewidth, node degrees, modularity, and assortativity.

These relationships enable researchers to generate networks with specific structural properties, facilitating nuanced insights into the different solution approaches for hard optimization problems.

Our empirical study demonstrates that *RG*4*SC* is a plausible fit for real-world supply chain networks and describes those better than alternative models. As part of this study, we leverage contemporary network theory to introduce a novel estimation procedure for fitting *RG*4*SC*'s input parameters to empirical networks. This procedure, available as part of our Python package, allows researchers to generate supply chain networks structurally similar to example networks. This can be of particular importance when only few instances are available.

To showcase *RG*4*SC*'s utility in supporting research, we revisit the Guaranteed Service Model (GSM). The GSM is a classic and widely used NP-hard optimization problem for placing safety stock in supply chain networks. Our analysis showcases how *RG*4*SC* can serve as a foundational tool to systematically study supply chain optimization problems, while yielding several novel insights.

We first show how *RG*4*SC* can help to validate managerial insights from simplified structures and uncover practically relevant nuances when analyzing more complex structures. In particular, we uncover a critical relationship between the location of safety stock and the connectivity of firms in the network. In sparsely connected networks, safety stock is often distributed throughout the network. As connectivity increases, inventory becomes concentrated closer to demand points.

We then turn to how *RG*4*SC* can help evaluate exact solution algorithms. By linking the relationship between *RG*4*SC*'s input parameters and the resulting treewidth with recent advances in the literature, we provide new insights about tractability of the GSM based on the micro behaviors underlying a network. This idea is broadly applicable, because the treewidth has been shown to be a key determinant of complexity in many optimization problems. *RG*4*SC* can also support a principled empirical approach, by generating networks where only a single structural parameter (e.g., the scale) is varied, while others remain fixed. For example, we find that the complexity of solving the GSM (and other problems that are fixed-parameter tractable in the treewidth) grows logarithmically with the network size for networks from heavy manufacturing industries.

Finally, we demonstrate the potential of *RG*4*SC* to validate heuristics and improve data-driven algorithms. We use it to generate a benchmark dataset to provide nuanced insights into the performance of heuristics for the GSM. We show that an LP-based heuristic is suitable for networks with low treewidth—precisely the type of networks that can be solved efficiently using exact algorithms. Second, we use *RG*4*SC* to create and bootstrap data, both of which can be powerful tools for improving machine learning algorithms where supply chain network data is sparse.

While our case study focuses on the GSM, *RG*4*SC* serves as a versatile generator of test tracks for a wide array of optimization problems in supply chain management. We anticipate that it will advance our understanding of other crucial problems such as the stochastic service model, material flow management, product traceability, and supplier auditing. By providing a principled and replicable way of generating data, it bridges the gap between theoretical models and empirical analysis, accelerating our knowledge of managing complex, interlinked supply chains. We invite the research community to leverage *RG*4*SC* to drive further innovations in supply chain optimization.

Notes

[1]Publicly available datasets on supplier-buyer relationships broadly used in the literature, such as Bloomberg (e.g., Osadchiy et al. 2016, Zou et al. 2023) or FactSet Revere (e.g., Osadchiy et al. 2021, Wu et al. 2022, Charoenwong et al. 2023), have many "missing links" and primarily cover public companies (Perera et al. 2017b). For example, "Walmart" appears in only 30 buyer-supplier relationships in the FactSet Reverere dataset, while Walmart claims to have 100,000 suppliers (Walmart 2023). The level of analysis—companies or divisons, rather than individual processing steps within a supply chain—is also too coarse for many supply chain optimization applications. Meanwhile, commercial transaction data, while potentially allowing for supply chain structure mapping, are usually limited to cross-border transactions (see, e.g., Jain et al. 2014).

[2]The GSM has been implemented by Hewlett-Packard (Billington et al. 2004), Proctor & Gamble (Farasyn et al. 2011), Intel (Manary et al. 2019), and at least 19 other major corporations (2019 presentation based on an early draft of Schoenmeyr and Graves 2022). Three distinct GSM applications were selected as finalists for the *Franz Edelman Award* (Schoenmeyr and Graves 2022).

[3]Our model does not allow for connections skipping tiers, in line with the supply chain network literature (e.g., Bimpikis et al. 2019), or asymmetric connectivity parameters. It is straightforward to extend *RG*4*SC* to account for these aspects, but it would make the notation unnecessarily complex.

[4]This is a temporary anonymous link for review purposes. The final version will link to our repository with an implementation of *RG*4*SC* and our estimation procedure (see § 3.3) in Python.

[5]Usually, tailored dynamic programs can be used to derive polynomial-time algorithms for combinatorial problems on graphs with bounded treewidth (Bodlaender 2006). Such graphs can be separated into subgraphs that are only connected sparsely. Dynamic programs then solve the optimization problem on some subgraphs and recursively combine solutions to obtain a result for the original problem (Harvey and Wood 2017).

References

Agrawal D, Osadchiy N (2024) Inventory productivity and stock returns in manufacturing networks. *Manufacturing & Service Operations Management* 26(2):573–593.

Atalay E, Hortacsu A, Roberts J, Syverson C (2011) Network structure of production. *Proceedings of the National Academy of Sciences* 108(13):5199–5202.

Attari I, Helm JE, Mejia J (2023) Hiding behind complexity: Supply chain, oversight, race, and the opioid crisis. *Production and Operations Management* (OnlineFirst).

Baldwin R, Freeman R (2022) Risks and global supply chains: What we know and what we need to know. *Annual Review of Economics* 14:153–180.

Baldwin RE (2012) Global supply chains: Why they emerged, why they matter, and where they are going. CEPR Discussion Paper No. DP9103: https://tinyurl.com/2tzt72hc (Accessed Sep 24, 2024).

Billington C, Callioni G, Crane B, Ruark JD, Rapp JU, White T, Willems SP (2004) Accelerating the profitability of Hewlett-Packard's supply chains. *Interfaces* 34(1):59–72.

Bimpikis K, Candogan O, Ehsani S (2019) Supply disruptions and optimal network structures. *Management Science* 65(12):5504–5517.

Blaettchen P, Calmon AP, Hall G (2024a) Traceability technology adoption in supply chain networks. *Management Science* (Articles in Advance).

Blaettchen P, Calmon AP, Hall G, Tawarmalani M (2024b) From trees to closed loops: Inventory management in treewidth-bounded supply chain networks. https://tinyurl.com/48scn48e (Accessed Sep 24, 2024).

Bodlaender HL (1994) A tourist guide through treewidth. *Acta Cybernetica* 11(1-2):1.

Bodlaender HL (2006) Treewidth: Characterizations, applications, and computations. *Graph-Theoretic Concepts in Computer Science: 32nd International Workshop, WG 2006, Bergen, Norway, June 22-24, 2006 Revised Papers 32*, 1–14 (Springer).

Cachon GP (2003) Supply chain coordination with contracts. *Handbooks in Operations Research and Management Science* 11:227–339.

Carvalho VM, Voigtländer N (2014) Input diffusion and the evolution of production networks. *National Bureau of Economic Research: Working Paper* 20025.

Charoenwong B, Han M, Wu J (2023) Trade and foreign economic policy uncertainty in supply chain networks: Who comes home? *Manufacturing & Service Operations Management* 25(1):126–147.

Clauset A, Newman ME, Moore C (2004) Finding community structure in very large networks. *Physical Review E* 70(6):066111.

Clauset A, Shalizi CR, Newman ME (2009) Power-law distributions in empirical data. *SIAM Review* 51(4):661–703.

de Kok T, Grob C, Laumanns M, Minner S, Rambau J, Schade K (2018) A typology and literature review on stochastic multi-echelon inventory models. *European Journal of Operational Research* 269(3):955–983.

de Siqueira S, Fujita A, Matias C (2021) Spectral density of random graphs: Convergence properties and application in model fitting. *Journal of Complex Networks* 9(6).

Eppen GD (1979) Note—Effects of centralization on expected costs in a multi-location newsboy problem. *Management Science* 25(5):498–501.

Erdős P, Rényi A (1959) On random graphs I. *Publicationes Mathematicae* 6(3–4):290–297.

Eruguz AS, Sahin E, Jemai Z, Dallery Y (2016) A comprehensive survey of guaranteed-service models for multi-echelon inventory optimization. *International Journal of Production Economics* 172:110–125.

Ettl M, Feigin GE, Lin GY, Yao DD (2000) A supply network model with base-stock control and service requirements. *Operations Research* 48(2):216–232.

Farasyn I, Humair S, Kahn JI, Neale JJ, Rosen O, Ruark J, Tarlton W, Van de Velde W, Wegryn G, Willems SP (2011) Inventory optimization at Procter & Gamble: Achieving real benefits through user adoption of inventory tools. *Interfaces* 41(1):66–78.

Gamarnik D, Shah D, Wei Y (2012) Belief propagation for min-cost network flow: Convergence and correctness. *Operations Research* 60(2):410–428.

Gansner ER, North SC (2000) An open graph visualization system and its applications to software engineering. *Software: Practice and Experience* 30(11):1203–1233.

Gao SY, Simchi-Levi D, Teo CP, Yan Z (2019) Disruption risk mitigation in supply chains: The risk exposure index revisited. *Operations Research* 67(3):831–852.

Gijsbrechts J, Boute RN, Van Mieghem JA, Zhang DJ (2022) Can deep reinforcement learning improve inventory management? Performance on lost sales, dual-sourcing, and multi-echelon problems. *Manufacturing & Service Operations Management* 24(3):1349–1368.

Gilbert EN (1959) Random graphs. *The Annals of Mathematical Statistics* 30(4):1141–1144.

Gong L, de Kok T, Ding J (1994) Optimal leadtimes planning in a serial production system. *Management Science* 40(5):629–632.

Graves SC, Kletter DB, Hetzel WB (1998) A dynamic model for requirements planning with application to supply chain optimization. *Operations Research* 46(3):S35–S49.

Graves SC, Willems SP (2000) Optimizing strategic safety stock placement in supply chains. *Manufacturing & Service Operations Management* 2(1):68–83.

Gu J, Hua B, Liu S (2015) Spectral distances on graphs. *Discrete Applied Mathematics* 190:56–74.

Gulati R, Nohria N, Zaheer A (2000) Strategic networks. *Strategic Management Journal* 21(3):203–215.

Harvey DJ, Wood DR (2017) Parameters tied to treewidth. *Journal of Graph Theory* 84(4):364–385.

Humair S, Willems SP (2006) Optimizing strategic safety stock placement in supply chains with clusters of commonality. *Operations Research* 54(4):725–742.

Hummels D, Ishii J, Yi KM (2001) The nature and growth of vertical specialization in world trade. *Journal of International Economics* 54(1):75–96.

Jackson MO (2010) *Social and economic networks* (Princeton University Press).

Jagabathula S, Rusmevichientong P (2019) The limit of rationality in choice modeling: Formulation, computation, and implications. *Management Science* 65(5):2196–2215.

Jain N, Girotra K, Netessine S (2014) Managing global sourcing: Inventory performance. *Management Science* 60(5):1202–1222.

Keeling MJ, Eames KT (2005) Networks and epidemic models. *Journal of the Royal Society Interface* 2(4):295–307.

Lesnaia E (2004) *Optimizing safety stock placement in general network supply chains*. Ph.D. thesis, Massachusetts Institute of Technology.

Lim K (2018) Endogenous production networks and the business cycle. Working paper: https://tinyurl.com/mtcmymhb (Accessed Sep 24, 2024).

Magnanti TL, Shen ZJM, Shu J, Simchi-Levi D, Teo CP (2006) Inventory placement in acyclic supply chain networks. *Operations Research Letters* 34(2):228–238.

Majumder P, Srinivasan A (2008) Leadership and competition in network supply chains. *Management Science* 54(6):1189–1204.

Manary MP, Wieland B, Willems SP, Kempf KG (2019) Analytics makes inventory planning a lights-out activity at Intel Corporation. *INFORMS Journal on Applied Analytics* 49(1):52–63.

Mari SI, Lee YH, Memon MS, Park YS, Kim M (2015) Adaptivity of complex network topologies for designing resilient supply chain networks. *International Journal of Industrial Engineering* 22(1).

Newman ME, Watts DJ, Strogatz SH (2002) Random graph models of social networks. *Proceedings of the National Academy of Sciences* 99(suppl_1):2566–2572.

Oberfield E (2018) A theory of input–output architecture. *Econometrica* 86(2):559–589.

Osadchiy N, Gaur V, Seshadri S (2016) Systematic risk in supply chain networks. *Management Science* 62(6):1755–1777.

Osadchiy N, Schmidt W, Wu J (2021) The bullwhip effect in supply networks. *Management Science* 67(10):6153–6173.

Penrose M (2003) *Random geometric graphs*, volume 5 (OUP Oxford).

Perera S, Bell MG, Bliemer MC (2017a) Network science approach to modelling the topology and robustness of supply chain networks: A review and perspective. *Applied Network Science* 2(1):1–25.

Perera S, Perera HN, Kasthurirathna D (2017b) Structural characteristics of complex supply chain networks. *2017 Moratuwa Engineering Research Conference (MERCon)*, 135–140 (IEEE).

Perera S, Perera HN, Kasthurirathna D (2017c) Value chain approach for modelling resilience of tiered supply chain networks. *2017 Moratuwa Engineering Research Conference (MERCon)*, 159–164 (IEEE).

Rich S, Moody P, Schaul K (2023) How deeply did prescription opioid pills flood your county? See here. https://tinyurl.com/ya9z6wyx (Accessed Sep 24, 2024).

Schoenmeyr T, Graves SC (2022) Coordination of multiechelon supply chains using the guaranteed service framework. *Manufacturing & Service Operations Management* 24(3):1859–1871.

Shore J, Lubin B (2015) Spectral goodness of fit for network models. *Social Networks* 43:16–27.

Shu J, Karimi I (2009) Efficient heuristics for inventory placement in acyclic networks. *Computers & Operations Research* 36(11):2899–2904.

Sodhi MS, Tang CS (2019) Research opportunities in supply chain transparency. *Production and Operations Management* 28(12):2946–2959.

Song JS, Yao DD (2002) *Supply Chain Structures: Coordination, Information and Optimization*, volume 42 (Springer Science & Business Media).

Taschereau-Dumouchel M (2020) Cascades and fluctuations in an economy with an endogenous production network. *SSRN 3115854* .

Thadakamaila H, Raghavan UN, Kumara S, Albert R (2004) Survivability of multiagent-based supply networks: A topological perspect. *IEEE Intelligent Systems* 19(5):24–31.

Uzzi B (2018) Social structure and competition in interfirm networks: The paradox of embeddedness. *The Sociology of Economic Life*, 213–241 (Routledge).

Walmart (2023) Suppliers. https://tinyurl.com/25jd33yb (Accessed Sep 24, 2024).

Wan P, Wang T, Davis RA, Resnick SI (2017) Fitting the linear preferential attachment model. *Electronic Journal of Statistics* 11(2):3738 – 3780.

Willems SP (2008) Real-world multiechelon supply chains used for inventory optimization. *Manufacturing & Service Operations Management* 10(1):19–23.

Wills P, Meyer FG (2020) Metrics for graph comparison: A practitioner's guide. *PLoS ONE* 15(2):e0228728.

Wu J, Zhang Z, Zhou SX (2022) Credit rating prediction through supply chains: A machine learning approach. *Production and Operations Management* 31(4):1613–1629.

Zhang H, Aydin G, Parker RP (2022) Social responsibility auditing in supply chain networks. *Management Science* 68(2):1058–1077.

Zhao K, Kumar A, Harrison TP, Yen J (2011) Analyzing the resilience of complex supply network topologies against random and targeted disruptions. *IEEE Systems Journal* 5(1):28–39.

Zhou K, Kılınç MR, Chen X, Sahinidis NV (2018) An efficient strategy for the activation of MIP relaxations in a multicore global MINLP solver. *Journal of Global Optimization* 70:497–516.

Zou F, Dong Y, Song S, Rungtusanatham M (2023) Product recalls and supply base innovation. *Manufacturing & Service Operations Management* 25(5):1931–1946.

E-Companion to *"A Random Model of Supply Chain Networks"*

Appendix A: Proofs and additional results related to the characteristics of the generated supply chain networks

A.1. Description of instance generation for numerical results in Appendix A

We generate two types of simulations for this Appendix. First, to validate using the expected node degree derived under the assumption of $n \to \infty$ also when $n < \infty$ (Appendix A.2.2), we generate graphs using *RG4SC* with $k = 2$, $q_1 = (0,0,1)$, $q_2 = (1,0,0)$, $p_{1,2} \in \{0.3, 0.7\}$, and $c_{1,2} \in \{1,2,\ldots,30\}$. For the number of nodes and the distribution across tiers we consider two options: $n = 200$ and $\alpha = (0.5, 0.5)$, or $n = 250$ and $\alpha = (0.6, 0.4)$ (in either case, the expected number of nodes at tier 2 is 100). For each combination of parameters, we generate 100 instances.

Second, for all other numerical results, we generate graphs with $n \in \{50, 100, 150, 200, 250\}$, $k \in \{2,4,6,8\}$, $q_1 = (0,0,1)$, $q_\ell = (0,1,0)$ $\forall \ell = 2,\ldots,k-1$, $q_k = (1,0,0)$, $c_{\ell,\ell+1} \in \{1,2,3,4,5,10,15,20\}$ (identically for all $\ell = 1,\ldots,k-1$), and $p_{\ell,\ell+1} \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$ (identically for all $\ell = 1,\ldots,k-1$). We vary α by generating four structural archetypes; (i) "even" with $\alpha_\ell = 1/k$; (ii) "diverging" with $\alpha_\ell = 0.1 + 2\frac{1-0.1k}{k(k-1)}(\ell - 1)$; (iii) "converging" with $\alpha_\ell = 0.1 + 2\frac{1-0.1k}{k(k-1)}(k - \ell)$; and, if $k > 2$, (iv)"hourglass" with $\alpha_\ell = 0.1 + 2\frac{1-0.1k}{A}\max\{\frac{k+1}{2} - l, l - \frac{k+1}{2}\}$ for all $\ell = 1,\ldots,k$. In the last case, A is a normalizing constant. For each combination of options, we generate four graphs for a total of 12,000. Out of these, we remove 20 with at least one tier with zero nodes to arrive at the final set.

A.2. Proofs and additional results related to the node degree

A.2.1. Proof of Theorem 1. We consider a slightly modified graph generating process, where node positions are uniform over the interval $[0, n]$ and edges are generated as before. Clearly, there is a one-to-one mapping between the two processes: By inflating the distances between two adjacent nodes by the factor n, we arrive from the original graph to the modified one, and vice versa. We operate under the assumption that $n \to \infty$ which means that for any node i with position X_i at tier ℓ, there exist at least c neighboring nodes, $i+1,\ldots,i+c$ with positions $X_i < X_{i+1} < \ldots < X_{i+c}$ for any c such that $\lim_{n\to\infty} \frac{c}{n} = 0$.

As the node-generation (resp. edge-generation) is independent across tiers (resp. pairs of tiers), it suffices to consider two tiers, ℓ and $\ell+1$, and show the in-degree at ℓ and the out-degree at $\ell+1$. Hence, to simplify notation, we denote $c_{\ell,\ell+1} = c$ and $p_{\ell,\ell+1} = p$. Moreover, because the edge-generation process between tiers ℓ and $\ell+1$ is symmetric, we only show the out-degree at tier $\ell+1$. The in-degree at tier ℓ follows by appropriately switching the model parameters.

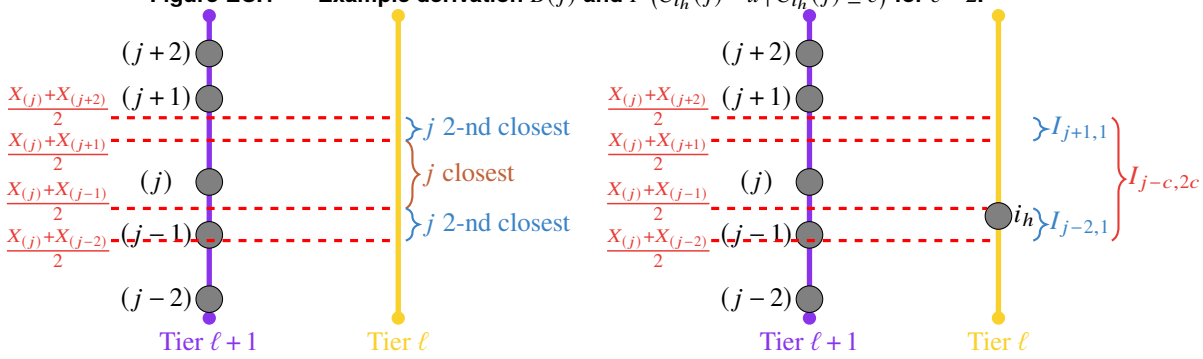
Sink nodes at tier $\ell+1$ have a zero out-degree. Hence, we consider a node j that is not a sink node at tier $\ell+1$. Let $B(j)$ be the set of non-source nodes at tier ℓ that are potential outgoing neighbors of j, that is,

$$B(j) = \{i_h \ : \ \ell_{i_h} = \ell, r_{i_h} \neq s, C_{i_h}(j) \leq c\}.$$

In Figure EC.1a, we graphically illustrate $B(j)$ for the case of $c = 2$.

Let $I(j) \subseteq B(j)$ be the subset of nodes where the connection to j is realized: these are outgoing neighbors of j. In addition, j may have have outgoing neighbors outside of $B(j)$: Any non-source node i_h at tier ℓ with $C_j(i_h) \leq c$, but where i_h is not part of $B(j)$, that is, $C_{i_h}(j) > c$. Denote the set of such nodes where a connection is realized with $I'(j)$. Then, $I(j) \cup I'(j)$ are the outgoing neighbors of j and $\mathbb{E}[D^{out}(j)] = \mathbb{E}[|I(j)|] + \mathbb{E}[|I'(j)|]$. In Lemma EC.1, we compute $\mathbb{E}[I(j)]$, and in Lemma EC.2, we compute $\mathbb{E}[I'(j)]$. By summing the expressions, we obtain our result.

Figure EC.1 Example derivation $B(j)$ and $P\left(C_{i_h}(j)=u \mid C_{i_h}(j)\le c\right)$ for $c=2$.



(a) All non-source nodes at tier ℓ within the marked intervals are in $B(j)$.

(b) If $|I_{a,b}| = l_{a,b}$, then $P\left(C_{i_h}(j)=2 \mid C_{i_h}(j)\le c\right) = \frac{l_{j-2,1}+l_{j+2-1,1}}{l_{j-c,2c}}$.

LEMMA EC.1. $\mathbb{E}\left[|I(j)|\right] = \frac{n_\ell}{m_{\ell+1}}\left[1+p(c-1)+(1-p)\frac{n_{\ell+1}}{n_\ell}\left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}-\left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^c\right)\right]$

Proof. Denote with Z_{i_h} a random variable that is one if i_h and j are indeed connected, and zero otherwise. Because node positions on tier $\ell+1$ are determined independently and j is within the closest c nodes to i_h for any h by definition of $B(j)$, Z_{i_h} are identically distributed and independent of the size of $B(j)$. Hence,

$$\begin{aligned}\mathbb{E}\left[|I(j)|\right] &= \mathbb{E}_{|B(j)|}\left[\mathbb{E}\left[Z_{i_1}+Z_{i_2}+\ldots+Z_{i_b} \mid |B(j)|=b\right]\right] = \mathbb{E}_{|B(j)|}\left[b\mathbb{E}\left[Z_{i_h} \mid |B(j)|=b\right]\right]\\ &= \mathbb{E}_{|B(j)|}\left[b\mathbb{E}\left[Z_{i_h}\right] \mid |B(j)|=b\right] = \mathbb{E}\left[|B(j)|\right]\mathbb{E}\left[Z_{i_h}\right].\end{aligned}$$

As nodes are generated according to a Poisson process, $\mathbb{E}\left[|B(j)|\right] = c\frac{n_\ell}{n_{\ell+1}}$ follows from Stegehuis and Weedage (2022, Theorem 3). To derive $\mathbb{E}\left[Z_{i_h}\right]$, we use the fact that i_h and j are connected when $C_{i_h}(j)=1$, when $C_j(i_h)=1$, or, with probability p, when $1<C_{i_h}(j)\le c$ and $1<C_j(i_h)$:

$$\begin{aligned}\mathbb{E}\left[Z_{i_h}\right] &= P\left(C_{i_h}(j)=1 \mid C_{i_h}(j)\le c\right)\\ &\quad+\sum_{u=2}^{c} P\left(C_{i_h}(j)=u \mid C_{i_h}(j)\le c\right)\left[P\left(C_j(i_h)=1 \mid C_{i_h}(j)=u\right)+pP\left(C_j(i_h)>1 \mid C_{i_h}(j)=u\right)\right]\\ &= P\left(C_{i_h}(j)=1 \mid C_i(j)\le c\right)\\ &\quad+\sum_{u=2}^{c} P\left(C_{i_h}(j)=u \mid C_{i_h}(j)\le c\right)\left[p+(1-p)P\left(C_j(i_h)=1 \mid C_i(j)=u\right)\right].\end{aligned}$$

First, we show that $P\left(C_{i_h}(j)=u \mid C_{i_h}(j)\le c\right) = 1/c\ \forall u\in\{1,\ldots,c\}$. Without loss of generality, we assume that node j is the j-th node in terms of position on tier $\ell+1$, i.e., $X_{(j)}=X_j$, otherwise we can reindex the nodes on tier $\ell+1$. Denote with $X_{(j-c)},\ldots,X_{(j-1)},X_{(j+1)},\ldots,X_{(j+c)}$ the positions of the closest nodes to j on the same tier. Then,

$$P\left(C_{i_h}(j)=u \mid C_{i_h}(j)\le c\right) = P\Bigg(X_{i_h}\in\underbrace{\left[\frac{X_{(j-u)}+X_{(j)}}{2},\frac{X_{(j-u+1)}+X_{(j)}}{2}\right]}_{I_{j-u,1}}\cup\underbrace{\left[\frac{X_{(j+u-1)}+X_{(j)}}{2},\frac{X_{(j+u)}+X_{(j)}}{2}\right]}_{I_{j+u-1,1}}\Bigg|\ X_{i_h}\in\underbrace{\left[\frac{X_{(j)}+X_{(j-c)}}{2},\frac{X_{(j)}+X_{(j+c)}}{2}\right]}_{I_{j-c,2c}}\Bigg).$$

Let $Y_{(u')} = X_{(u'+1)} - X_{(u')}$. By the properties of the Poisson distribution, $Y_{(m)} \overset{\text{i.i.d.}}{\sim} Exp(n_{\ell+1})\ \forall m \in j-c,\dots,j+c-1$. As a union of disjoint intervals, one can check that the length of $I_{j-u,1} \cup I_{j+u-1,1}$ is $\frac{Y_{(j-u+1)}+Y_{(j+u)}}{2}$, while the length of $I_{j-c,2c}$ is $\frac{Y_{(j-c+1)}+\ldots+Y_{(j)}+\ldots+Y_{(j+c)}}{2}$. Moreover, X_{i_h} is uniformly distributed within $I_{j-c,2c}$, which contains $I_{j-u,1} \cup I_{j+u-1,1}$. Hence, conditional on $|I_{j-u,1}| = l_{j-u,1}$, $|I_{j+u-1,1}| = l_{j+u-1,1}$, and $|I_{j-c,2c}| = l_{j-c,2c}$, the probability that X_{i_h} is within $I_{j-u,1} \cup I_{j+u-1,1}$ is $\frac{l_{j-u,1}+l_{j+u-1,1}}{l_{j-c,2c}}$. This is illustrated in Figure EC.1b. We can then rewrite

$$P\left(C_{i_h}(j) = u \mid C_{i_h}(j) \leq c\right) = \int_0^\infty \dots \int_0^\infty \frac{y_{(j-u+1)} + y_{(j+u)}}{y_{(j-c+1)} + \ldots + y_{(j+c)}} n_{\ell+1}^{2c} e^{-n_{\ell+1}\left(y_{(j-c+1)}+\ldots+y_{(j+c)}\right)} dy_{(j-c+1)} \dots dy_{(j+c)}$$
$$= 1/c,$$

where the second equality follows from standard calculus.

Next, we show that $P\left(C_j(i_h) = 1 \mid C_{i_h}(j) = u\right) = \left(\frac{n_{\ell+1}}{n_\ell + n_{\ell+1}}\right)^u \ \forall u \in \{2,\dots,c\}$. Let $X_{i_h} = x_{i_h}$. We denote with $j_{(m)}$ (resp. $j_{(-m)}$) the m-th closest node on the right (resp. left) of x_{i_h} on tier $\ell+1$. For ease of notation, we let $j_{(0)} = i_h$. Without loss of generality, we assume that the $u-1$-st closest node to i_h is on its right and that there are λ_r nodes between i_h and the $u-1$-st closest node (including the latter). Note that $1 \leq \lambda_r \leq u-1$ and that the $u-1$-st closest node corresponds to index $j_{(\lambda_r)}$. Moreover, $Y_{j_{(m)}} \overset{\text{Def.}}{=} X_{j_{(m)}} - X_{j_{(m-1)}} \overset{\text{i.i.d.}}{\sim} Exp(n_{\ell+1})\ \forall m = -(u-\lambda_r),\dots,\lambda_r+1$. We have two possible cases: (i) $j = j_{(\lambda_r+1)}$; (ii) $j = j_{(-(u-\lambda_r))}$. We make use of the notation i_h^r (resp. i_h^l) to denote the closest node to the right (resp. to the left) on tier ℓ to i_h.

Consider case (ii). The event $C_j(i_h) = 1$ is equivalent to

$$X_j - X_{i_h^l} > x_{i_h} - X_j \Leftrightarrow \frac{x_{i_h} - X_{i_h^l}}{2} > x_{i_h} - X_j \Leftrightarrow \frac{x_{i_h} - X_{i_h^l}}{2} > \sum_{m=-(u-\lambda_r)+1}^{0} (X_{j_{(m)}} - X_{j_{(m-1)}}) \Leftrightarrow \frac{\tilde{Y}}{2} > \sum_{m=-(u-\lambda_r)+1}^{0} Y_{j_{(m)}},$$

where $\tilde{Y} \sim Exp(n_\ell)$ is independent of $Y_{j_{(m)}}$ for all $m = -(u-\lambda_r),\dots,\lambda_r+1$.

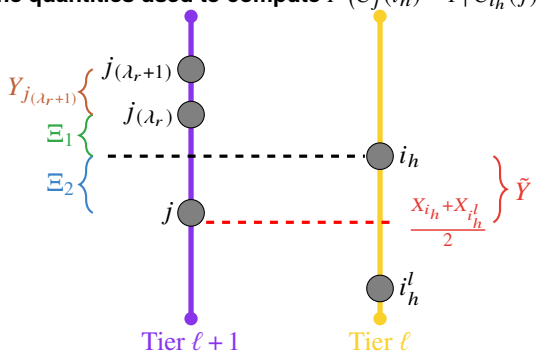
Meanwhile, the event $C_{i_h}(j) = u$ is equivalent to

$$\sum_{m=1}^{\lambda_r} (X_{j_{(m)}} - X_{j_{(m-1)}}) < \sum_{m=-(u-\lambda_r)+1}^{0} (X_{j_{(m)}} - X_{j_{(m-1)}}) < \min\Big\{ \sum_{m=1}^{\lambda_r+1} (X_{j_{(m)}} - X_{j_{(m-1)}}), \sum_{m=-(u-\lambda_r)}^{0} (X_{j_{(m)}} - X_{j_{(m-1)}}) \Big\}$$
$$\Leftrightarrow \sum_{m=1}^{\lambda_r} (X_{j_{(m)}} - X_{j_{(m-1)}}) < \sum_{m=-(u-\lambda_r)+1}^{0} (X_{j_{(m)}} - X_{j_{(m-1)}}) < \sum_{m=1}^{\lambda_r+1} (X_{j_{(m)}} - X_{j_{(m-1)}})$$
$$\Leftrightarrow \sum_{m=1}^{\lambda_r} Y_{j_{(m)}} < \sum_{m=-(u-\lambda_r)+1}^{0} Y_{j_{(m)}} < \sum_{m=1}^{\lambda_r+1} Y_{j_{(m)}} \Leftrightarrow \Xi_1 < \Xi_2 < \Xi_1 + Y_{j_{(\lambda_r+1)}},$$

where $\Xi_1 \overset{\text{Def.}}{=} \sum_{m=1}^{\lambda_r} Y_{j_{(m)}} \sim Erlang(\lambda_r, n_{\ell+1})$ and $\Xi_2 \overset{\text{Def.}}{=} \sum_{m=-(u-\lambda_r)+1}^{0} \sim Erlang(u-\lambda_r, n_{\ell+1})$ are two independent random variables. We exemplify the relevant quantities in Figure EC.2 and rewrite

$$P\left(C_j(i_h) = 1 \mid C_{i_h}(j) = u\right) = \frac{P\left(\Xi_1 < \Xi_2 < \min\{\tilde{Y}/2, \Xi_1 + Y_{j_{(\lambda_r+1)}}\}\right)}{P\left(\Xi_1 < \Xi_2 < \Xi_1 + Y_{j_{(\lambda_r+1)}}\right)}$$
$$= \frac{\int_0^\infty \int_0^\infty \left[\int_{\xi_1}^{\xi_1+y} \int_{\xi_1}^{\tilde{y}} K d\xi_2 d\tilde{y}\right] + \left[\int_{\xi_1+y}^{\infty} \int_{\xi_1}^{\xi_1+y} K d\xi_2 d\tilde{y}\right] d\xi_1 dy}{\int_0^\infty \int_0^\infty \int_{\xi_1}^{\xi_1+y} n_{\ell+1} e^{-n_{\ell+1} y} \frac{n_{\ell+1}^{\lambda_r} \xi_1^{\lambda_r-1} e^{-n_{\ell+1}\xi_1}}{(\lambda_r-1)!} \frac{n_{\ell+1}^{u-\lambda_r} \xi_2^{u-\lambda_r-1} e^{-n_{\ell+1}\xi_2}}{(u-\lambda_r-1)!} d\xi_2 d\xi_1 dy},$$

Figure EC.2 **Derivation of the quantities used to compute $P\left(C_j(i_h)=1 \mid C_{i_h}(j)=u\right)$ for any $\lambda_r \in \{0,\dots,u-1\}$.**



where $K = 2n_\ell e^{-2n_\ell \tilde{y}} n_{\ell+1} e^{-n_{\ell+1} y} \frac{n_{\ell+1}^{\lambda_r} \xi_1^{\lambda_r-1} e^{-n_{\ell+1}\xi_1}}{(\lambda_r-1)!} \frac{n_{\ell+1}^{u-\lambda_r} \xi_2^{u-\lambda_r-1} e^{-n_{\ell+1}\xi_2}}{(u-\lambda_r-1)!}$. Standard calculus allows us to simplify this into a term that is independent of λ_r:

$$\frac{\left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^u 2^{-u} \frac{(u-1)!}{\lambda_r!(u-\lambda_r-1)!}}{2^{-u}\frac{(u-1)!}{\lambda_r!(u-\lambda_r-1)!}} = \left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^u.$$

We now consider case (i). Here, the event $C_j(i_h)=1$ is equivalent to

$$X_{i_h^r} - X_j > X_j - x_{i_h} \Leftrightarrow \frac{x_{i_h} - X_{i_h^l}}{2} > x_{i_h} - X_j \Leftrightarrow \frac{X_{i_h^r} - x_{i_h}}{2} > \sum_{m=1}^{\lambda_r+1} (X_{j_{(m)}} - X_{j_{(m-1)}}) \Leftrightarrow \frac{\tilde{Y}}{2} > \sum_{m=1}^{\lambda_r+1} Y_{j_{(m)}},$$

while the event $C_{i_h}(j)=u$ is equivalent

$$\sum_{m=-(u-\lambda_r)+2}^{0} (X_{j_{(m)}} - X_{j_{(m-1)}}) < \sum_{m=1}^{\lambda_r+1} (X_{j_{(m)}} - X_{j_{(m-1)}}) < \min\Big\{ \sum_{m=-(u-\lambda_r)+1}^{0} (X_{j_{(m)}} - X_{j_{(m-1)}}), \sum_{m=1}^{\lambda_r+2} (X_{j_{(m)}} - X_{j_{(m-1)}}) \Big\}$$

$$\Leftrightarrow \sum_{m=-(u-\lambda_r)+2}^{0} Y_{j_{(m)}} < \sum_{m=1}^{\lambda_r+1} Y_{j_{(m)}} < \sum_{m=-(u-\lambda_r)+1}^{0} Y_{j_{(m)}} \Leftrightarrow \Xi_3 < \Xi_4 < \Xi_3 + Y_{j_{(u-\lambda_r)+1}},$$

where $\Xi_3 \overset{\text{Def.}}{=} \sum_{m=-(u-\lambda_r)+2}^{0} Y_{j_{(m)}} \sim Erlang(u-\lambda_r-1, n_{\ell+1})$ and $\Xi_4 \overset{\text{Def.}}{=} \sum_{m=1}^{\lambda_r+1} \sim Erlang(\lambda_r+1, n_{\ell+1})$ are again independent. By re-indexing λ_r to $u-\lambda_r-1$, we immediately see that the case follows equivalently with

$$P\left(C_j(i_h)=1 \mid C_{i_h}(j)=u\right) = \left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^u.$$

As the result is identical for any x_{i_h}, it is unchanged when we integrate over the condition $X_{i_h}=x_{i_h}$. Overall,

$$\begin{aligned}\mathbb{E}\left[Z_{i_h}\right] &= \frac{1}{c}\left[1 + \sum_{u=2}^{c}\left(p + (1-p)\left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^u\right)\right] = \frac{1}{c}\left[1 + p(c-1) + (1-p)\sum_{u=2}^{c}\left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^u\right] \\ &= \frac{1}{c}\left[1 + p(c-1) + (1-p)\frac{n_{\ell+1}}{n_\ell}\left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}} - \left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^c\right)\right],\end{aligned}$$

and the result is obtained by multiplying $\mathbb{E}\left[|B(j)|\right]\mathbb{E}\left[Z_{i_h}\right]$. □

LEMMA EC.2. $\mathbb{E}\left[|I'(j)|\right] = \left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^c + p\sum_{h=2}^{c}\left(1 - B_{\frac{n_\ell}{n_\ell+n_{\ell+1}}}(h,c)\frac{(c+h-1)!}{(c-1)!(h-1)!}\right)$.

Proof. Recall that $I'(j)$ is the set of additional neighbors of j, beyond the nodes in $B(j)$ that could have been neighbors (independent of whether a connection was actually realized). Consider the closest c (non-source) nodes at

tier ℓ to node j. Denote them, in order of distance, with $i_h, h = 1, \ldots, c$. Then, we need to consider node i_h if and only if it is not in $B(j)$, that is, if and only if $C_{i_h}(j) > c$. Thus,

$$\mathbb{E}\left[|I'(j)|\right] = P\left(C_{i_1}(j) > c \mid C_j(i_1) = 1\right) + p\sum_{h=2}^{c} P\left(C_{i_h}(j) > c \mid C_j(i_h) = h\right).$$

We will show that $P\left(C_{i_h}(j) \leq c \mid C_j(i_h) = h\right) = B_{\frac{n_\ell}{n_\ell+n_{\ell+1}}}(h,c)\frac{(c+h-1)!}{(c-1)!(h-1)!}\ \forall h \in \{1,\ldots,c\}$, where B is the incomplete Beta-function. We proceed as in the proof of Lemma EC.1, when computing $P\left(C_j(i_h) = 1 \mid C_{i_h}(j) = u\right)$. In particular, we use symmetric notation: We let $X_j = x_j$ and denote with $i_{(m)}$ (resp. $i_{(-m)}$) the m-th closest node on the right (resp. left) of x_j on tier ℓ. For ease of notation, we let $i_{(0)} = j$. We also assume, without loss of generality, that the $h-1$-st closest node to j is on its right and that there are $\lambda_r \in \{1,\ldots,h-1\}$ nodes between j and the $h-1$-st closest node, including the latter. We then have, as before, $Y_{i_{(m)}} = X_{i_{(m)}} - X_{i_{(-m)}} \overset{\text{i.i.d.}}{\sim} Exp(n_\ell)\ \forall m = -(h-\lambda_r), \ldots, \lambda_r + 1$. Similarly, we have two cases to consider: (i) $i_h = i_{(\lambda_r+1)}$; (ii) $i_h = i_{(-(h-\lambda_r))}$.

Take case (ii), and assume for now that $h \in \{2,\ldots,c\}$, which we relax later. By assumption, node i_h is to the left of node j, but at tier ℓ. We denote with $j^l_{(u)}, u = -1, \ldots, -c$ the $(-u)$-th node to the left of j at tier $\ell+1$. For ease of notation, we also let $j^l_{(0)} = j$. Hence, the event $C_{i_h}(j) \leq c$ is equivalent to

$$X_{i_h} > \frac{x_j + X_{j^l_{(c)}}}{2} \Leftrightarrow \frac{x_j - X_{j^l_{(c)}}}{2} > x_j - X_{i_h} \Leftrightarrow \frac{\sum_{u=-c+1}^{0}(X_{j^l_{(u)}} - X_{j^l_{(u-1)}})}{2} > \sum_{m=-(h-\lambda_r)+1}^{0}(X_{i_{(m)}} - X_{i_{(m-1)}}) \Leftrightarrow \frac{\tilde{\Xi}}{2} > \Xi_2,$$

where $\tilde{\Xi} \sim Erlang(c, n_{\ell+1})$, and we redefine $\Xi_2 \sim Erlang(h-\lambda_r, n_\ell)$.

By a symmetric argument to that in the proof of Lemma EC.1, the event $C_j(i_h)$ is equivalent to $\Xi_1 < \Xi_2 < \Xi_1 + Y_{i_{(\lambda_r+1)}}$, with $\Xi_1 \sim Erlang(\lambda_r, n_\ell)$ also redefined appropriately. It follows that

$$\begin{aligned}
P\left(C_{i_h}(j) \leq c \mid C_j(i_h) = h\right) &= \frac{P\left(\Xi_1 < \Xi_2 < \min\{\tilde{\Xi}/2, \Xi_1 + Y_{i_{(\lambda_r+1)}}\}\right)}{P\left(\Xi_1 < \Xi_2 < \Xi_1 + Y_{i_{(\lambda_r+1)}}\right)} \\
&= 1 - \left(\frac{n_{\ell+1}}{n_\ell + n_{\ell+1}}\right)^c \sum_{u=1}^{h}\left[\frac{(c-1+u-1)!}{(c-1)!(u-1)!}\left(\frac{n_\ell}{n_\ell+n_{\ell+1}}\right)^{u-1}\right] \\
&= B_{\frac{n_\ell}{n_\ell+n_{\ell+1}}}(h,c)\frac{(c+h-1)!}{(c-1)!(h-1)!},
\end{aligned}$$

where the last line follows from standard calculus. It remains to consider $h = 1$. Then, $\lambda_r = 0$ and we replace the random variable Ξ_1 by zero (since there is no node that is closer to j than i_h). The computation extends directly, with

$$P\left(C_{i_1}(j) \leq c \mid C_j(i_1) = 1\right) = \frac{P\left(\Xi_2 < \min\{\tilde{\Xi}/2, Y_{i_{(1)}}\}\right)}{P\left(\Xi_2 < Y_{i_{(1)}}\right)} = 1 - \left(\frac{n_{\ell+1}}{n_\ell + n_{\ell+1}}\right)^c.$$

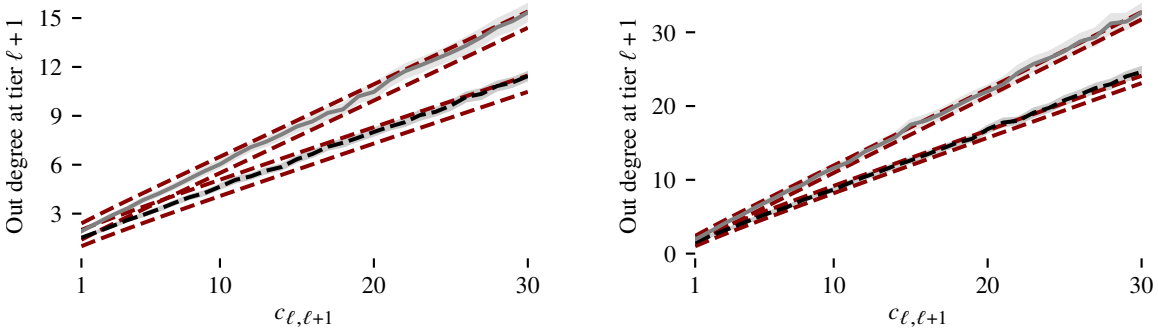
As before, case (i) with $i_h = i_{(\lambda_r+1)}$ leads to identical computations, so we omit it here for brevity. □

Putting everything together, we have

$$\begin{aligned}
\mathbb{E}\left[D^{\ell+1}_{out}(j) \mid r_j \neq t\right] = &\frac{n_\ell}{n_{\ell+1}}\left[1 + p(c-1)\right] + (1-p)\left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}} - \left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^c\right) \\
&+ \left(\frac{n_{\ell+1}}{n_\ell+n_{\ell+1}}\right)^c + p\sum_{h=2}^{c}\left(1 - B_{\frac{n_\ell}{n_\ell+n_{\ell+1}}}(h,c)\frac{(c+h-1)!}{(c-1)!(h-1)!}\right).
\end{aligned}$$

The proof is completed by applying the law of iterated expectations over r_j and rearranging the terms.

Figure EC.3 Expected degree (assuming $c_{\ell,\ell+1} \ll n \to \infty$) compared to simulated mean degree.



(a) $p_{\ell,\ell+1} = 0.3$ (b) $p_{\ell,\ell+1} = 0.7$

Note. Instances are generated as described in Appendix A.1. The black (resp. gray) lines display the average observed out-degree at tier $\ell+1$ for $n = 200$ and $\alpha = (0.5, 0.5)$ (resp. $n = 200$ and $\alpha = (0.5, 0.5)$). The shaded area displays three standard deviations across 100 experiments. The dashed red lines display the expected out degree, ± 0.5.

A.2.2. Validation for finite n. While the expected degree derived in Theorem 1 assumes that $n \to \infty$ and $c_{\ell,\ell+1} \ll n$, we can effectively use the expected degree also for finite n, as long as $c_{\ell,\ell+1}$ is not too large. In particular, when $n < \infty$, an increase in $c_{\ell,\ell+1}$ increases the observed degrees above the computed expectation.

We thus conduct extensive experiments to analyze the impact of this. We find that $c_{\ell,\ell+1} < \frac{\min\{\alpha_\ell, \alpha_{\ell+1}\}n}{4}$ is sufficient in most cases to avoid a deviation between the minimum and maximum degree by more than 0.5. This is also captured in Figure EC.3, where we present the out-degree (the in-degree follows symmetrically). We further experiment with the impact of the deviations on the estimation procedure described in § 3.3, and find that the broader assumption $c_{\ell,\ell+1} < \frac{\min\{\alpha_\ell, \alpha_{\ell+1}\}n}{2}$ is sufficient to avoid any noticeable impact.

A.3. Proofs and additional results related to the treewidth

A.3.1. Proof of Theorem 2. We show that we can upper-bound the treewidth on a random geometric graph with dimension one (Lemma EC.3). We then show that graphs generated by *RG4SC*, for a given number of nodes $N = n'$, are subgraphs of corresponding random geometric graphs (Lemma EC.4). As the treewidth of a graph is upper-bounded by the treewidth of the supergraph, we are able to show the result (Lemma EC.5). As a last step, we show that the result extends when removing conditioning. First, we recall the following definitions and properties:

Definition EC.1. If Ω is a sample space and X_n and X are random variables, we say

1. X_n converges to X *almost surely*, if $P(\omega \in \Omega : \lim_{n\to\infty} X_n(\omega) = X(\omega)) = 1$
2. X_n converges to X *in probability*, if $P(\omega \in \Omega : |X_n(\omega) - X(\omega)| > \epsilon) = 0 \; \forall \epsilon > 0$.

Definition EC.2 (Penrose 2003). Let $n' \in \mathbb{N}$ and $r : \mathbb{N} \to \mathbb{R}^+$. A graph over n' nodes is said to be a *random geometric graph* of radius $r(n')$ if node positions are randomly and independently drawn from $U[0,1]$ and there is an edge between two nodes if the distance between two nodes is less than $r(n')$.

Definition EC.3. A graph is said to be chordal if all cycles in the graph of four or more vertices have a chord, that is, an edge that is not part of the cycle but connects two vertices of the cycle.

Definition EC.4. A clique in a graph G corresponds to a subset of nodes such that any two nodes in the subset share an edge. The clique number of a graph, $\xi(G)$, is the size of its maximum clique.

As it turns out, there is a nice relationship between clique number and treewidth for chordal graphs:

PROPOSITION EC.1. *We have that*

(i) any random geometric graph is almost surely chordal.

(ii) in any chordal graph G, $tw(G) = \xi(G) + 1$.

Proof. The proof of (i) results from the following observation. Let $H_{n'}$ be a random geometric graph and let X_i be the random variable corresponding to the position of node i for $i = 1, \ldots, n'$. By definition, two nodes i and i' are connected if and only if $||X_i - X_{i'}|| \leq r(n')$. For any realization $\omega \in \Omega$, where ω is an infinite-dimensional vector, we can fully define $H_{n'}(\omega) = H(X_1(\omega_1), X_2(\omega_2), \ldots X_{n'}(\omega_{n'}); r(n'))$. To see that $H_{n'}$ is chordal, assume that there are edges (i_1, i_2), (i_2, i_3), (i_3, i_4), and (i_4, i_1) between four nodes $i_1, \ldots, i_4$ and, without loss of generality, further assume that $X_1(\omega_1) < X_2(\omega_2) < X_3(\omega_3) < X_4(\omega_4)$. Clearly, $r(n') \geq ||X_1(\omega_1) - X_4(\omega_4)|| > ||X_1(\omega_1) - X_3(\omega_3)||$ and $r(n') \geq ||X_1(\omega_1) - X_4(\omega_4)|| > ||X_2(\omega_2) - X_4(\omega_4)||$. Hence, there must be a chord for any four-cycle. Property (ii) is well-known (see, e.g., Eppstein 2002). □

LEMMA EC.3. *Let $H_{n'}$ be a random geometric graph on n' nodes with radius $r(n') = \frac{\log n' + (\overline{c} + \epsilon') \log\log n'}{n' \tilde{\alpha}}$ for some $\epsilon' > 0$. (i) If $\lim_{n' \to \infty} \frac{\overline{c} \log\log n'}{\log n'} = 0$, then*

$$\lim_{n' \to \infty} P\left(tw(H_{n'}) \leq \frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} n' r(n') + \epsilon \right) = 1 \ \forall \epsilon > 0,$$

where W is the product logarithm. (ii) If $\lim_{n' \to \infty} \frac{\overline{c} \log\log n'}{\log n'} = \infty$, but $\lim_{n' \to \infty} \frac{\overline{c} \log\log n'}{n'} = 0$, then

$$\lim_{n' \to \infty} P\left(tw(H_{n'}) \leq n' r(n') + \epsilon \right) = 1 \ \forall \epsilon > 0.$$

Proof. Take the first case. We have $\lim_{n' \to \infty} \frac{n' r(n')}{\log n} = \lim_{n' \to \infty} \frac{1}{\tilde{\alpha}} \left(1 + \frac{(\overline{c} + \epsilon') \log\log n'}{\log n'} \right) = \frac{1}{\tilde{\alpha}}$. This corresponds to case (iii) in McDiarmid and Müller (2011, Theorem 1.2), with $\sigma = \sup\{t : vol(\{x : f(x) > t\}) > 0\} = 1$ when $f(x)$ is the pdf of a uniform distribution on $[0, 1]$. In particular, the clique number $\zeta(H_{n'})$ converges almost surely to $\frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} n' r(n')$.

For the second case, note that $\lim_{n' \to \infty} r(n') = 0$, while $\lim_{n' \to \infty} \frac{n' r(n')}{\log n} = \lim_{n' \to \infty} \frac{\log n' + (\overline{c} + \epsilon') \log\log n'}{\alpha \log n}$ (i.e., $n' r(n') \gg \log n'$). This corresponds to case (iv) in McDiarmid and Müller (2011, Theorem 1.2), so the clique number $\zeta(H_{n'})$ converges almost surely to $n' r(n')$.

From Proposition EC.1 (i), $H_{n'}$ is chordal almost surely. Hence, from Proposition EC.1 (ii), it follows that $tw(H_{n'})$ also converges almost surely to $\frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} n' r(n')$, respectively $n' r(n')$. By Fatou's Lemma, this means that $tw(H_{n'})$ also converges in probability to $n' r(n')$, or

$$\lim_{n \to \infty} P\left(\omega \in \Omega : \left| tw\left(H_{n'}(\omega)\right) - \frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} n' r(n') \right| > \epsilon \right) = 0 \ \forall \epsilon > 0,$$

respectively

$$\lim_{n \to \infty} P\left(\omega \in \Omega : |tw\left(H_{n'}(\omega)\right) - n' r(n')| > \epsilon \right) = 0 \ \forall \epsilon > 0,$$

which directly implies the result. □

We next consider the set of graphs $\mathcal{G}_{n'}$ generated by *RG4SC* with exactly n' nodes. That is, conditional on $N = n'$, $\mathcal{G}_{n'} = \mathcal{G}$. Recall that $\overline{c} = \max_{\ell = 1, \ldots, k-1} c_{\ell, \ell+1}$, and $\tilde{\alpha}$ is a positive constant with $0 < \tilde{\alpha} < \alpha_\ell \ \forall \ell = 1, \ldots, k$.

LEMMA EC.4. *Let $G_{n'} \in \mathcal{G}_{n'}$. Furthermore, let $H_{n'}$ be a random geometric graph on n' nodes, whose positions correspond to the types of the nodes in $G_{n'}$, and with radius $r(n') = \frac{\log n' + (\overline{c}+\epsilon')\log\log n'}{\tilde{\alpha} n'}$ for some $\epsilon' > 0$. Then,*

$$\lim_{n\to\infty} P\left(G_{n'} \subseteq H_{n'}\right) = 1.$$

Proof. If for all X_i on tier ℓ, there are $c_{\ell,\ell+1}$ nodes contained within radius $r(n')$ of X_i on tier $\ell+1$ and $c_{\ell-1,\ell}$ nodes contained within $r(n')$ of X_i on tier $\ell-1$, then $G_{n'} \subseteq H_{n'}$. We define the following events, suppressing the dependence on $\omega \in \Omega$ for clarity:

- A_i^ℓ: the interval of length $r(n')$ at tier ℓ, centered at X_i, contains more than or equal to $\max\{c_{\ell-1,\ell}, c_{\ell,\ell+1}\}$ nodes. Here, extend c with the entires$c_{k,k+1} = 1$ and $c_{0,1} = 1$.
- C_ℓ: there are at least $\tilde{\alpha} n$ nodes at tier ℓ.

We have

$$\begin{aligned}
P\left(G_{n'} \subseteq H_{n'}\right) &\geq P(A_1^1 \cap A_2^1 \cap \ldots \cap A_{n'}^k) = 1 - P\left(\overline{A_1^1} \cup \overline{A_2^1} \cup \ldots \cup \overline{A_{n'}^k}\right) \\
&\geq 1 - \sum_{i=1}^{n'} P\left(\overline{A_i^1} \cup \ldots \cup \overline{A_i^k}\right) = \sum_{i=1}^{n'} P(A_i^1 \cap \ldots \cap A_i^k) - (n'-1) \\
&= \sum_{i=1}^{n'} \prod_{\ell=1}^{k} P(A_i^\ell) - (n'-1),
\end{aligned}$$

where the second line follows from Boole's inequality, and the third line follows because events A_i^ℓ and $A_i^{\ell'}$ for $\ell \neq \ell'$ are independent.

Consider the term $P(A_i^\ell) = P(A_i^\ell \mid C_\ell)P(C_\ell) + P(A_i^\ell \mid \overline{C_\ell})P(\overline{C_\ell})$ for some $\ell = 1, \ldots, k$. Note that $A_i^\ell \mid C_\ell$ is the event that a Binomial random variable with a number of draws greater or equal to $\tilde{\alpha} n'$ and success probability equal to the length of the subinterval $r(n')$ is greater or equal to $\max\{c_{\ell-1,\ell}, c_{\ell,\ell+1}\}$. Let $Z \sim Binomial(\tilde{\alpha} n', r(n'))$. We have:

$$P(A_i^\ell \mid C_\ell) \geq P(Z \geq \overline{c}).$$

Using the Chernoff bound for Binomial variables, we have $P(Z \geq \overline{c}) \geq 1 - exp\left(-\tilde{\alpha} n' D\left(\frac{\overline{c}}{\tilde{\alpha} n'} || r(n')\right)\right)$, where $D(\cdot||\cdot)$ denotes the Kullback-Leibler divergence. Denoting $f(n') \sim g(n')$ iff $\lim_{n'\to\infty} \frac{f(n')}{g(n')} = 1$, we have

$$\begin{aligned}
D\left(\frac{\overline{c}}{\tilde{\alpha} n'} || r(n')\right) =& \frac{\overline{c}}{\tilde{\alpha} n'} \log\left(\frac{\frac{\overline{c}}{\tilde{\alpha} n'}}{\frac{\log n' + (\overline{c}+\epsilon')\log\log n'}{\tilde{\alpha} n'}}\right) + \left(1 - \frac{\overline{c}}{\tilde{\alpha} n'}\right) \log\left(\frac{1 - \frac{\overline{c}}{\tilde{\alpha} n'}}{1 - \frac{\log n' + (\overline{c}+\epsilon')\log\log n'}{\tilde{\alpha} n'}}\right) \\
=& \frac{\overline{c}}{\tilde{\alpha} n'} \log(\overline{c}) - \frac{\overline{c}}{\tilde{\alpha} n'} \log\left(\log n' + (\overline{c}+\epsilon')\log\log n'\right) + \left(1 - \frac{\overline{c}}{\tilde{\alpha} n'}\right) \log\left(1 - \frac{\overline{c}}{\tilde{\alpha} n'}\right) \\
& - \left(1 - \frac{\overline{c}}{\tilde{\alpha} n'}\right) \log\left(1 - \frac{\log n' + (\overline{c}+\epsilon')\log\log n'}{\tilde{\alpha} n'}\right) \\
\sim& \frac{\overline{c}}{\tilde{\alpha} n'} \log(\overline{c}) - \frac{\overline{c}}{\tilde{\alpha} n'} \log\log n' - \frac{\overline{c}}{\tilde{\alpha} n'} + \frac{\log n' + (\overline{c}+\epsilon')\log\log n'}{\tilde{\alpha} n'} \\
\sim& \frac{\log n' + \epsilon' \log\log n'}{\tilde{\alpha} n'},
\end{aligned}$$

where the third line follows from the fact that $\log\left(1 - \frac{b}{n'}\right) \sim -\frac{b}{n'}$ for any constant b. Thus,

$$1 - exp\left(-\tilde{\alpha} n' D\left(\frac{\overline{c}}{\tilde{\alpha} n'} || r(n')\right)\right) \sim 1 - \frac{1}{n'(\log n')^{\epsilon'}}.$$

Next, let $Z' \sim Bin(n', \alpha_\ell)$ be another Binomial random variable. We have that $P(C_\ell) = P(Z' \geq \tilde{\alpha} n') \geq 1 - exp(-n' D(\tilde{\alpha}||\alpha_\ell))$ using the same Chernoff bound as above. As $\tilde{\alpha} \leq \alpha_\ell$, $D(\tilde{\alpha}||\alpha_\ell)$ is a nonnegative constant. These bounds, combined with the fact that probabilities cannot exceed one, give us

$$\begin{aligned}
&P(A_i^\ell \mid C_\ell) \sim 1 - \frac{1}{n'(\log n')^{\epsilon'}} \\
\Rightarrow &P(A_i^\ell) \sim 1 - \frac{1}{n'(\log n')^{\epsilon'}} \\
\Rightarrow &\sum_{i=1}^{n'} \prod_{\ell=1}^{k} P(A_i^\ell) - (n'-1) \sim 1 - n'\left(1 - \left(1 - \frac{1}{n'(\log n')^{\epsilon'}}\right)^k\right) \sim 1 - \frac{k}{(\log n')^{\epsilon'}} \overset{n'\to\infty}{\to} 1,
\end{aligned}$$

so the result follows directly. □

LEMMA EC.5. *Assume $G_{n'} \in \mathcal{G}_{n'}$. (i) If $\lim_{n'\to\infty} \frac{\bar{c}\log\log n'}{\log n'} = 0$, then*

$$\lim_{n'\to\infty} P\left(tw(G_{n'}) \leq \frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} \frac{\log n' + (\bar{c}+\epsilon)\log\log n'}{\tilde{\alpha}}\right) = 1 \ \forall \epsilon > 0,$$

where W is the product logarithm. (ii) If $\lim_{n'\to\infty} \frac{\bar{c}\log\log n'}{\log n'} = \infty$, but $\lim_{n'\to\infty} \frac{\bar{c}\log\log n'}{n'} = 0$, then

$$\lim_{n'\to\infty} P\left(tw(G_{n'}) \leq \frac{\log n' + (\bar{c}+\epsilon)\log\log n'}{\tilde{\alpha}}\right) = 1 \ \forall \epsilon > 0.$$

Proof. Let $H_{n'}$ be the random geometric graph with nodes as in $G_{n'}$ and radius $r(n') = \frac{\log n' + (\bar{c}+\epsilon')\log\log n'}{n'\tilde{\alpha}}$ for some $\epsilon' > 0$. Moreover, let $f(n')$ be some function of n'. Then,

$$\begin{aligned}
P(tw(G_{n'}) > f(n')) =& P(tw(G_{n'}) > f(n') \mid G_{n'} \subseteq H_{n'}) P(G_{n'} \subseteq H_{n'}) \\
&+ P(tw(G_{n'}) > f(n') \mid G_{n'} \nsubseteq H_{n'}) P(G_{n'} \nsubseteq H_{n'}) \\
\leq& P(tw(H_{n'}) > f(n')) + P(G_{n'} \nsubseteq H_{n'}),
\end{aligned}$$

where the inequality follows from the fact that probabilities are less than one, and $P(tw(G_{n'}) > f(n') \mid G_{n'} \subseteq H_{n'}) \leq P(tw(H_{n'}) > f(n'))$ as the treewidth of a graph is smaller than the treewidth of a supergraph.

For case (i), replace $f(n')$ by $\frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} n' r(n') + \epsilon$ for some $\epsilon > 0$. Taking limits on both sides, and applying Lemmas EC.3–EC.4, we have

$$\lim_{n'\to\infty} P\left(tw(G_{n'}) > \frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} \frac{\log n' + (\bar{c}+\epsilon')\log\log n'}{\tilde{\alpha}} + \epsilon\right) = 0.$$

Since this is true for any constants $\epsilon, \epsilon' > 0$, we can appropriately redefine ϵ to obtain the result. For case (ii), replace $f(n')$ by $n' r(n') + \epsilon$. Again, taking limits on both sides and applying the previous lemmas, we have

$$\lim_{n'\to\infty} P\left(tw(G_{n'}) > \frac{\log n' + (\bar{c}+\epsilon')\log\log n'}{\tilde{\alpha}} + \epsilon\right) = 0 \ \forall \epsilon > 0.$$

Again, we can appropriately redefine ϵ to obtain the result. □

Proof of Theorem 2. Note that by conditioning the distribution of $\mathcal{G}$ on the event $N = n'$, we recover the distribution of $\mathcal{G}_{n'}$ (Mitsche and Perarnau 2017). Let $f(n)$ be a function of n and let $\epsilon' > 0$. Then,

$$\begin{aligned}
&\lim_{n\to\infty} P(tw(G) > f(n)) \\
&\quad = \lim_{n\to\infty} \sum_{n'=0}^{\infty} P\left(tw(G) > f(n) \middle| N = n'\right) P(N = n')
\end{aligned}$$

$$
\begin{aligned}
&= \lim_{n\to\infty} \sum_{n'=0}^{\infty} P\left(tw(G_{n'}) > f(n)\right) P\left(N = n'\right) \\
&= \lim_{n\to\infty} \sum_{n'=0}^{\lceil (1+\epsilon')n \rceil} P\left(tw(G_{n'}) > f(n)\right) P\left(N = n'\right) \\
&\qquad + \sum_{n'=\lceil (1+\epsilon')n \rceil+1}^{\infty} P\left(tw(G_{n'}) > f(n)\right) P\left(N = n'\right) \\
&\leq \lim_{n\to\infty} P\left(tw(G_{\lceil (1+\epsilon')n \rceil}) > f(n)\right) + \sum_{n'=\lceil (1+\epsilon')n \rceil+1}^{\infty} P\left(N = n'\right).
\end{aligned}
$$

For the inequality, we upper-bound the first weighted sum by the largest term (multiplied with one), and we upper-bound the second weighted sum by one in the first term.

The second term is $P\left(N \geq \lceil (1+\epsilon')n \rceil + 1\right) \leq P\left(N > (1+\epsilon')n\right) \leq \inf_{t>0} \mathbb{E}\left[e^{tN}\right] e^{-t(1+\epsilon')n} = \left(\frac{(1+\epsilon')n}{n}\right)^{-(1+\epsilon')n} e^{(1+\epsilon')n-n} = (1+\epsilon')^{-(1+\epsilon')n} e^{\epsilon' n}$, based on the Chernoff bound of the Poisson distribution with rate n. Note that $\lim_{n\to\infty} (1+\epsilon')^{-(1+\epsilon')n} e^{\epsilon' n} = 0$.

For the first term, we note that there exists an n_0 such that, for all $n > n_0$,

$$
\log\lceil (1+\epsilon')n \rceil + (\overline{c}+\epsilon'') \log\log\lceil (1+\epsilon')n \rceil \leq \left(1+\epsilon'+\frac{1}{n}\right) \log n + \left(1+\epsilon'+\frac{1}{n}\right)\left(1+\frac{\epsilon''}{\overline{c}}\right) \overline{c} \log\log n.
$$

This is true for all $\epsilon', \epsilon'' > 0$. Hence, for any $\epsilon > 0$, there also exists an n_0' such that, for all $n > n_0'$,

$$
\log\lceil (1+\epsilon')n \rceil + (\overline{c}+\epsilon'') \log\log\lceil (1+\epsilon')n \rceil \leq (1+\epsilon)\left(\log n + \overline{c} \log\log n\right).
$$

Hence, for case (i), we replace $f(n)$ by $(1+\epsilon)\frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} n r(n)$. Then,

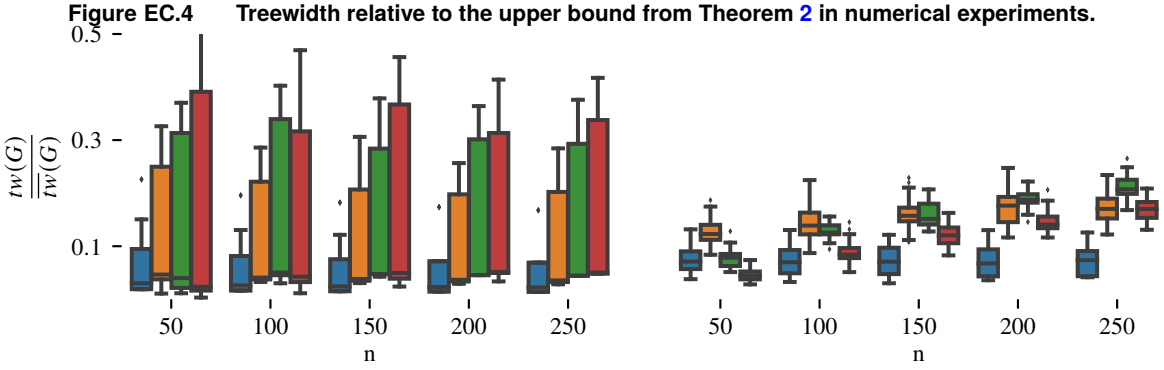
$$
\begin{aligned}
&\lim_{n\to\infty} P\left(tw(G_{\lceil (1+\epsilon')n \rceil}) > (1+\epsilon)\frac{-(1-\tilde{\alpha})}{W(-\frac{1-\tilde{\alpha}}{e})} \frac{\log n + \overline{c}\log\log n}{\tilde{\alpha}}\right) \\
&\leq \lim_{n\to\infty} P\left(tw(G_{\lceil (1+\epsilon')n \rceil}) > \frac{-(1-\tilde{\alpha})}{\tilde{\alpha} W(-\frac{1-\tilde{\alpha}}{e})} \left(\log\lceil (1+\epsilon')n \rceil + (\overline{c}+\epsilon'') \log\log\lceil (1+\epsilon')n \rceil\right)\right) \\
&= \lim_{n'\to\infty} P\left(tw(G_{n'}) > \frac{-(1-\tilde{\alpha})}{\tilde{\alpha} W(-\frac{1-\tilde{\alpha}}{e})} \left(\log n' + (\overline{c}+\epsilon'') \log\log n'\right)\right) = 0.
\end{aligned}
$$

Similarly, for case (ii), we replace $f(n)$ by $(1+\epsilon) n r(n)$. Then,

$$
\begin{aligned}
&\lim_{n\to\infty} P\left(tw(G_{\lceil (1+\epsilon')n \rceil}) > (1+\epsilon)\frac{\log n + \overline{c}\log\log n}{\tilde{\alpha}}\right) \\
&\leq \lim_{n\to\infty} P\left(tw(G_{\lceil (1+\epsilon')n \rceil}) > \frac{1}{\tilde{\alpha}} \left(\log\lceil (1+\epsilon')n \rceil + (\overline{c}+\epsilon'') \log\log\lceil (1+\epsilon')n \rceil\right)\right) \\
&= \lim_{n'\to\infty} P\left(tw(G_{n'}) > \frac{1}{\tilde{\alpha}} \left(\log n' + (\overline{c}+\epsilon'') \log\log n'\right)\right) = 0. \qquad \square
\end{aligned}
$$

A.3.2. Comparison between the realized treewidth and the upper bound. Theorem 2 provides an upper bound on the treewidth. Using our numerical experiments, we consider the tightness of this bound. As shown in Figure EC.4, the bound can be quite tight for low values of k and high values of $c_{\ell,\ell+1}$. However, as k increases, the bound's tightness generally decreases. Interestingly, if k is sufficiently high, the relative difference also starts displaying a non-monotonic relationship with $c_{\ell,\ell+1}$.

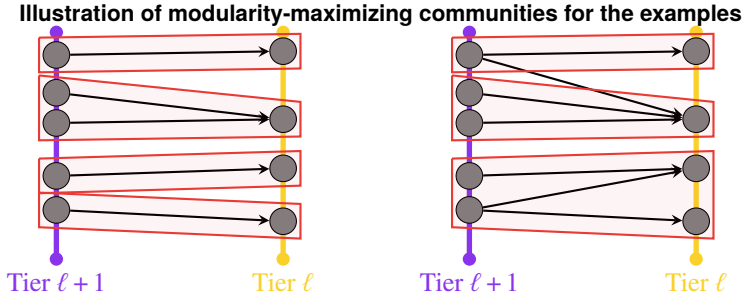
Figure EC.4 **Treewidth relative to the upper bound from Theorem 2 in numerical experiments.**



(a) $k = 2$ (b) $k = 8$

Note. Data is generated as described in Appendix A.1, selecting $p_{\ell,\ell+1} = 1$. Bars from left to right indicate $c_{\ell,\ell+1} \in \{1,2\},\{3,4,5\},\{10\},\{15,20\}$. We let $\overline{tw}(G) = 1/\tilde{\alpha}\,(\log n + \bar{c}\log\log n)$ as in case (ii), which only differs from case (i) in the constant.

Figure EC.5 **Illustration of modularity-maximizing communities for the examples in Figure 3.**



(a) $c_{\ell,\ell+1} = 1, Q = 0.72$. (b) $c_{\ell,\ell+1} = 2, Q = 0.5$.

A.4. Connection between local connectivity and high modularity

Figure EC.5 illustrates how local connectivity is crucial for obtaining networks with high modularity, by analyzing the simple (sub-)graph from Figure 3. In Figure EC.5a, we assume $c_{\ell,\ell+1} = 1$. Modularity is maximized with four distinct communities, marked with red boxes in the figure, in line with the overlaps between neighbors. With these communities, modularity is $Q = 0.72$: Although the communities are distinct, not all edges that could exist in a community actually exist, because we do not allow for connections within tiers.

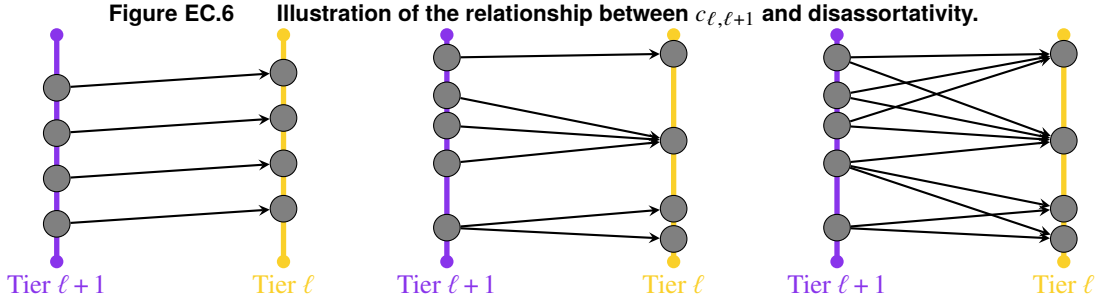
In Figure EC.5b, we assume $c_{\ell,\ell+1} = 1$ and $p_{\ell,\ell+1} < 1$. Here, modularity is maximized with three communities. Noticeably, there is an edge crossing between two communities, and the resulting moduarlity is $Q = 0.5$. Moreover, if we had $p_{\ell,\ell+1} = 1$, the maximum modularity would further decrease to $Q = 0.395$.

A.5. Connection between local connectivity and disassortativity

Disassortativity of the networks generated by *RG4SC* is driven by the random nature of the horizontal specialization, together with local connectivity. Assume first that $c_{\ell,\ell+1} = 1$, $\alpha_\ell = \alpha_{\ell+1} = 0.5$, and nodes are placed at even intervals across the tiers. Every node has exactly one neighbor, so the assortativity coefficient is undefined—see Figure EC.6a.

As a result of the random specialization, nodes vary in the number of neighbors they have. Consider Figure EC.6b: a node that has multiple incoming (resp. outgoing) neighbors can only have one outgoing (resp. incoming) neighbor. Hence, the assortativity coefficient, measuring the correlation between degrees, must be negative.

Figure EC.6 Illustration of the relationship between $c_{\ell,\ell+1}$ and disassortativity.



(a) $c_{\ell,\ell+1} = 1$, assort. coef. undefined. (b) $c_{\ell,\ell+1} = 1$, assort. coef. -0.62. (c) $c_{\ell,\ell+1} = 2$, assort. coef. -0.37.

Importantly, for a node at tier ℓ to have multiple incoming neighbors in the first place, there must be a sufficiently large gap to the adjacent nodes at ℓ, so that there are multiple nodes at tier $\ell+1$ filling this gap (and vice versa). As $c_{\ell,\ell+1}$ increases, this is no longer true, because the "baseline" number of connection increase. Generally speaking, the assortativity coefficient will increase, although it remains negative—see Figure EC.6c.

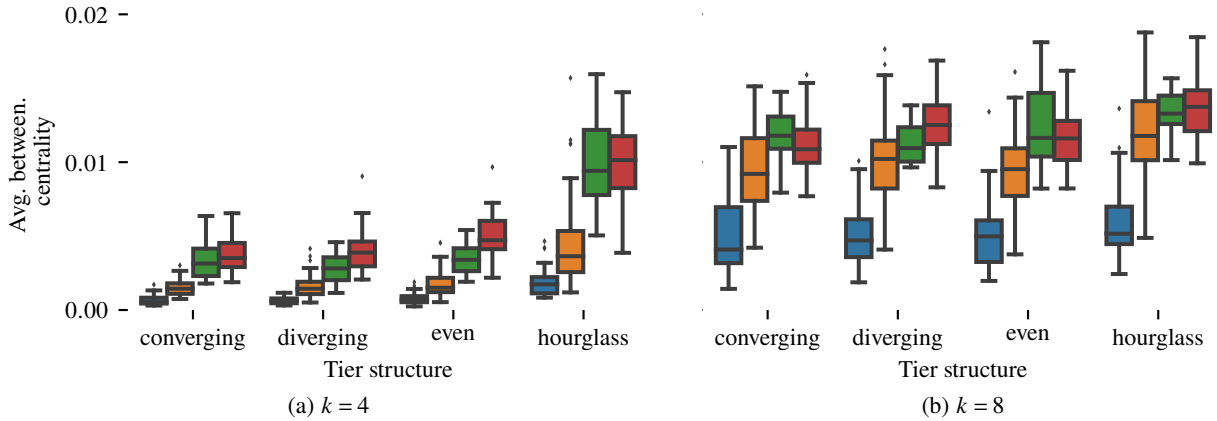
A.6. Additional structural characteristics

A.6.1. Centrality. A node's *centrality* indicates the influence this node has within the network, allowing to distinguish "important" nodes from less important ones. The notion of importance is naturally ambiguous, so centrality can be defined in a number of ways. Here, we focus on betweenness centrality, which measure the number of shortest paths between pairs of nodes through the focal node (see Jackson 2010, Chapter 2, for an exact definition).

We choose this measure, because it has been linked directly to operational outcomes. In particular, Agrawal and Osadchiy (2024) find that manufacturing firms with high degree and high betweenness centrality within their supply chain network tend to have higher inventory turnover, which is a crucial performance metric in any supply chain.

Figure EC.7 indicates that betweenness centrality increases with k and $c_{\ell,\ell+1}$, but may decrease again for sufficiently high $c_{\ell,\ell+1}$. It is also related to the tier structure in complex ways: for example, networks following an "hourglass" structure display higher centrality than other networks, but only when the number of tiers is small.

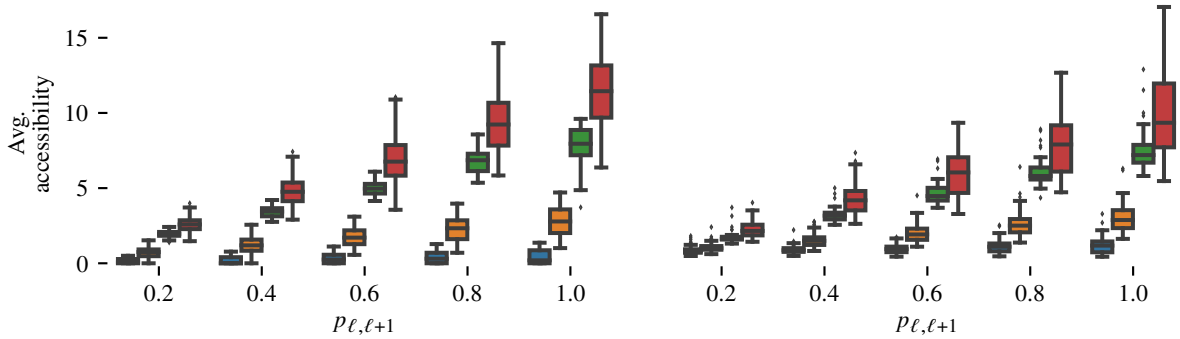
Figure EC.7 Average betweenness centrality per node, for tiers $\ell = 2, \ldots, k-1$.



(a) $k = 4$ (b) $k = 8$

Note. Data is generated as described in Appendix A.1, selecting $n = 100$. Bars from left to right indicate $c_{\ell,\ell+1} \in \{1,2\}, \{3,4,5\}, \{10\}, \{15,20\}$.

Figure EC.8 Average accessibility / interconnectedness per node, for tiers $\ell = 2, \ldots, k-1$.



(a) Accessibility (b) Interconnectedness

Note. Data is generated as described in Appendix A.1, selecting $n = 100$. Bars from left to right indicate $c_{\ell,\ell+1} \in \{1,2\}, \{3,4,5\}, \{10\}, \{15,20\}$.

Our model is able to create network structures with different degrees and centrality values by appropriately tuning the input parameters—for example, n and k do not influence the node degree, but do influence centrality. This opens the door to systematically experimenting with different structures before redesigning supply chain networks.

A.6.2. Accessibility and interconnectedness. We next consider two related measures: accessibility and interconnectedness, defined in Bellamy et al. (2014). Accessibility describes a node's access to information from different parts of the network. It is the harmonic mean length of paths that end at the node. Interconnectedness concerns the shared relationships between network partners, that is, the shared neighbors of two connected nodes. As nodes connect tier-by-tier, we cannot employ the original definition, so we consider nodes at the same tier that share neighbors as partners. Their interconnectedness, then, is the number of neighbors shared amongst them. Bellamy et al. (2014) find that both accessibility and interconnectedness of firms in a supply chain network positively influence their innovativeness.

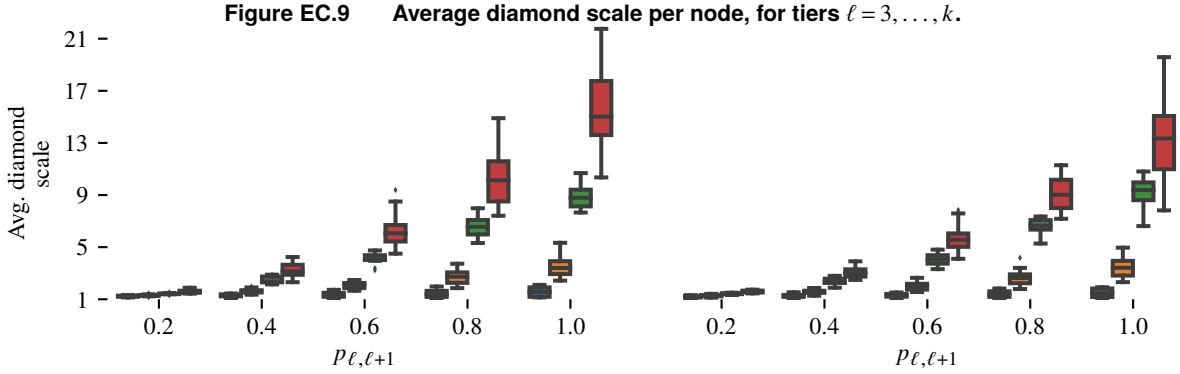
Figure EC.8a indicates a positive relationship between $c_{\ell,\ell+1}$ and $p_{\ell,\ell+1}$ and the accessibility of a node. The parameters are complimentary in increasing accessibility. Figure EC.8b displays similar relationships for interconnectedness. Not shown here is a detailed analysis of the interconnectedness at different tiers. For example, we observe a distinctly higher interconnectedness for the intermediary nodes in hourglass networks, which can be interpreted as bottlenecks.

As with inventory turnover, *RG4SC*'s ability to create networks with distinct differences in accessibility and interconnectedness allows users to systematically experiment with different network structures to effectively evaluate improvements in innovativeness.

A.6.3. Diamond scale. The overlap between suppliers' suppliers is a major concern in supply chain risk management (Ang et al. 2017). In particular, we consider the *diamond scale* Wang et al. (2021). It describes a focal node's second-tier suppliers (that is, incoming neighbors of the focal node's own incoming neighbors). The diamond scale indicates by how many direct suppliers the average second-tier supplier is shared. Wang et al. (2021) show that nodes with a large diamond scale are more prone to disruption risks due to the commonalities in their supply chains.

Consider a focal node at tier ℓ. Its diamond scale is expected to increase in $c_{\ell+1,\ell+2}$, because direct suppliers (at tier $\ell+1$) increasingly see overlaps between their own eligible suppliers (at tier $\ell+2$). Figure EC.9 indicates that the diamond scale indeed monotonically increases in the connectivity range. However, the figure also reveals that the

Figure EC.9 Average diamond scale per node, for tiers $\ell = 3, \ldots, k$.



(a) Tier structure "even" (b) Tier structure "hourglass"

Note. Data is generated as described in Appendix A.1, selecting $n = 100$. Bars from left to right indicate $c_{\ell,\ell+1} \in \{1,2\},\{3,4,5\},\{10\},\{15,20\}$.

realization of potential overlaps (controlled by the parameters $p_{\ell,\ell+1}$) has a more pronounced, super-linear effect. This is intuitive: if a firm at tier ℓ has two suppliers at tier $\ell+1$, diamond structures only emerge when both realize a connection with the same firm. Assuming that the latter firm is not the closest neighbor of either of the suppliers, but it is within the closest $c_{\ell+1,\ell+2}$ neighbors, this occurs with probability $p^2_{\ell,\ell+1}$.

This distinct relationship between input parameters and diamond scale is important to use *RG4SC* in the context of supply chain risk management. When supply chain network structures and, thus, risks, are not fully known, it is easy to envisage fitting *RG4SC* based on partial information. A user may then simulate possible complete networks in order to better grasp key risk metrics such as the diamond structure and assess weak points.

Appendix B: Proofs and additional results related to the empirical fit of the random model

B.1. Proof of Proposition 1.

We consider only $f^{\ell+1}_{out}$, as the result follow symmetrically for the in-degree. We simplify notation, letting $p_{\ell,\ell+1} = p$ and $c_{\ell,\ell+1} = c$. First, we show that the average out-degree for nodes at tier $\ell+1$, is a consistent estimator of the expected out-degree. Second, we show that $f^{\ell+1}_{out}$ is increasing, which enables us to define a continuous function, based on its inverse. Finally, we use that function to apply the continuous mapping theorem and show our result.

LEMMA EC.6. $\lim_{n\to\infty} P\left(|\hat{D}^{\ell+1}_{out} - \mathbb{E}\left[D^{\ell+1}_{out}\right]| > \epsilon\right) = 0 \; \forall \epsilon > 0.$

Proof. Let $\tilde{\alpha}$ be a constant such that $0 < \tilde{\alpha} < \alpha_\ell$ and C_ℓ the event that there are at least $\tilde{\alpha}n$ nodes at tier $\ell+1$.

$$P\left(|\hat{D}^{\ell+1}_{out} - \mathbb{E}\left[D^{\ell+1}_{out}\right]| > \epsilon\right) = P\left(|\hat{D}^{\ell+1}_{out} - \mathbb{E}\left[D^{\ell+1}_{out}\right]| > \epsilon \,\middle|\, C_\ell\right) P(C_\ell) + P\left(|\hat{D}^{\ell+1}_{out} - \mathbb{E}\left[D^{\ell+1}_{out}\right]| > \epsilon \,\middle|\, \overline{C_\ell}\right) P(\overline{C_\ell})$$
$$\leq P\left(\left|\frac{\sum_{i=1}^{\tilde{\alpha}n} D^{\ell}_{out}(i)}{\tilde{\alpha}n} - \mathbb{E}\left[D^{\ell+1}_{out}\right]\right| > \epsilon\right) P(C_\ell) + P\left(|\hat{D}^{\ell+1}_{out} - \mathbb{E}\left[D^{\ell+1}_{out}\right]| > \epsilon \,\middle|\, \overline{C_\ell}\right) P(\overline{C_\ell})$$

We have $\lim_{n\to\infty} P(\overline{C_\ell}) = 0$ (proof of Lemma EC.4). Moreover, because $\tilde{\alpha} > 0$, $\lim_{n\to\infty} P\left(\left|\frac{\sum_{i=1}^{\tilde{\alpha}n} D^{\ell+1}_{out}(i)}{\tilde{\alpha}n} - \mathbb{E}\left[D^{\ell+1}_{out}\right]\right| > \epsilon\right) = \lim_{n\to\infty} P\left(\left|\frac{\sum_{i=1}^{n} D^{\ell+1}_{out}(i)}{n} - \mathbb{E}\left[D^{\ell+1}_{out}\right]\right| > \epsilon\right) = 0$ by the weak law of large numbers. □

LEMMA EC.7. *$f^{\ell+1}_{out}$ is strictly increasing in c.*

Proof. Due to the linear interpolation, we only need to show that $f_{out}^{\ell+1}(c+1) > f_{out}^{\ell+1}(c)$ for all $c \in \mathbb{N}_{>0}$. We assume, w.l.o.g. that $q_{\ell+1} = (1,0,0)$ and $q_\ell = (0,0,1)$. Then $f_{out}^{\ell+1}(c) = \frac{\alpha_\ell}{\alpha_{\ell+1}}(1+p(c-1)) + (1-p)\frac{\alpha_{\ell+1}}{\alpha_\ell+\alpha_{\ell+1}} + p\sum_{h=1}^{c}\left(1 - B_{\frac{\alpha_\ell}{\alpha_\ell+\alpha_{\ell+1}}}(h,c)\frac{(c+h-1)!}{(c-1)!(h-1)!}\right)$. Hence,

$$\frac{1}{p}\left(f_{out}^{\ell+1}(c+1) - f_{out}^{\ell+1}(c)\right) = \frac{\alpha_\ell}{\alpha_{\ell+1}} + \sum_{h=1}^{c}\frac{(c+h-1)!}{(c-1)!(h-1)!}\left(B_{\frac{\alpha_\ell}{\alpha_\ell+\alpha_{\ell+1}}}(h,c) - B_{\frac{\alpha_\ell}{\alpha_\ell+\alpha_{\ell+1}}}(h,c+1)\frac{c+h}{c}\right) + \left(1 - B_{\frac{\alpha_\ell}{\alpha_\ell+\alpha_{\ell+1}}}(c+1,c+1)\frac{(2c+1)!}{c!c!}\right).$$

The final term is non-negative because it corresponds to 1 minus the regularized Beta function. Hence, it is sufficient to show that the first two terms together are positive.

We take the derivative of the first two terms to α_ℓ:

$$\frac{(\alpha_\ell\alpha_{\ell+1})^c}{(\alpha_\ell+\alpha_{\ell+1})^{2c+1}}\frac{(2c)!}{c!^2}\left[c + {}_2F_1\left(1, 2c+1; c+1; \frac{\alpha_\ell}{\alpha_\ell+\alpha_{\ell+1}}\right)\right],$$

where ${}_2F_1$ is the hypoergeometric function, which is positive everywhere, as is the entire derivative. Hence, we only need to check $\alpha_\ell = 0$: Here, all terms are zero, and the result follows. □

Because $f_{out}^{\ell+1}$ is continuous and strictly increasing, it has a continuous inverse, which we denote with f^{-1}. Note that we use $\hat{c} = f^{-1}\left(\hat{D}_{out}^{\ell+1}\right)$ as an estimator for c. We next show consistency:

LEMMA EC.8. $\lim_{n\to\infty} P\left(|f^{-1}\left(\hat{D}_{out}^{\ell+1}\right) - c| > \epsilon\right) = 0 \;\forall \epsilon > 0, \forall c = 1,2,\ldots$

Proof. Take any node i at tier $\ell+1$. From the proof of Theorem 1, it follows that the expected number of out-degrees for that node, $\mathbb{E}\left[D_{out}^{\ell+1}(i)\right]$, is only different from $f_{out}^{\ell+1}(c)$ if there are less than c nodes between X_i and either of the boundaries of $[0,1]$ on tier ℓ or tier $\ell+1$. Because the relative number of such nodes goes to zero as $n \to \infty$ (and because their out-degree is of the same order of magnitude as that of other nodes), there exists a function $\tilde{g}_{out}^{\ell+1}(c)$ with $\lim_{n\to\infty}\tilde{g}_{out}^{\ell+1}(c) = 0$, such that $\mathbb{E}\left[D_{out}^{\ell+1}\right] = f_{out}^{\ell+1}(c) + \tilde{g}_{out}^{\ell+1}(c)\;\forall c = 1,2\ldots$. We again extend $\tilde{g}_{out}^{\ell+1}$ to a continuous function $g_{out}^{\ell+1}$ by linear interpolation in-between the integer values of c.

Hence, we can applying the Continuous Mapping Theorem to the result of Lemma EC.6:

$$\lim_{n\to\infty} P\left(\left|f^{-1}\left(\hat{D}_{out}^{\ell+1} - g_{out}^{\ell+1}(c)\right) - f^{-1}\left(\mathbb{E}\left[D_{out}^{\ell+1}\right] - g_{out}^{\ell+1}(c)\right)\right| > \epsilon'\right) = 0 \;\forall \epsilon' > 0.$$

Note that $f^{-1}\left(\mathbb{E}\left[D_{out}^{\ell+1}\right] - g_{out}^{\ell+1}(c)\right) = c \;\forall c = 1,2,\ldots$. Moreover, because f^{-1} is continuous, for every $\epsilon'' > 0$, there exists some $\delta > 0$ such that

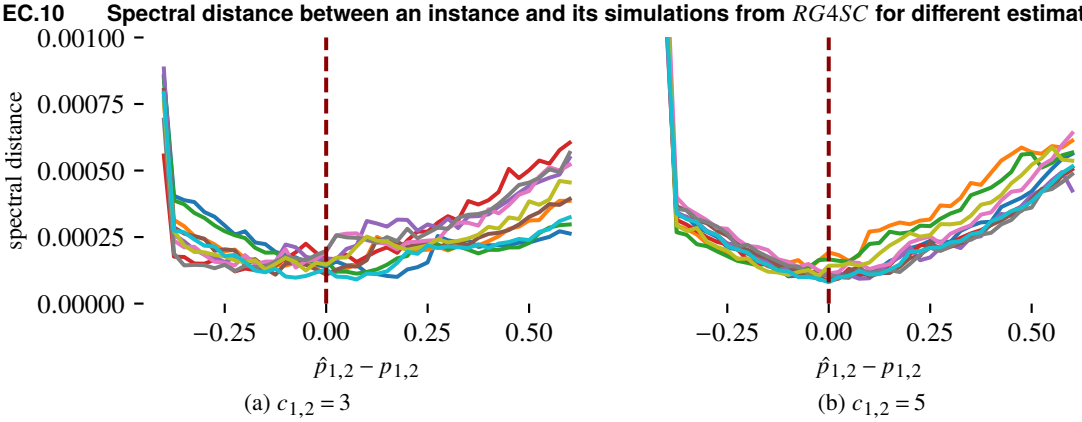
$$\left|f^{-1}\left(\hat{D}_{out}^{\ell+1} \pm \delta\right) - f^{-1}\left(\hat{D}_{out}^{\ell+1}\right)\right| < \epsilon''.$$

Finally, because $\lim_{n\to\infty}\tilde{g}_{out}^{\ell+1}(c) = 0$, there exists n_0 such that $|\tilde{g}_{out}^{\ell+1}(c)| < \delta \;\forall n > n_0$. It follows that

$$\lim_{n\to\infty} P\left(\left|f^{-1}\left(\hat{D}_{out}^{\ell+1}\right) - c\right| > \epsilon' + \epsilon''\right) = 0 \;\forall \epsilon', \epsilon'' > 0, c = 1,2,\ldots$$

The result follows by letting $\epsilon = \epsilon' + \epsilon''$ □

Figure EC.10 Spectral distance between an instance and its simulations from *RG4SC* for different estimates $\hat{p}_{1,2}$.



Note. Data is from the first set, with $p_{1,2} = 0.4$. Each solid line corresponds to one instance, while the vertical dotted line marks the point at which $\hat{p}_{1,2} = p_{1,2}$.

B.2. The estimation procedure in detail

B.2.1. Description of instance generation for numerical results in Appendix B.2. To validate our estimation procedure, we generate two sets of graphs. In either case, we have $k = 2$, $q_1 = (0, 0, 1)$, $q_2 = (1, 0, 0)$, $c_{1,2} \in \{1, 2, 3, 4, 5, 10, 15, 20\}$, and $p_{1,2} \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$.

For the first set, we additionally let $n = 200$ and $\alpha_1 = \alpha_2 = 0.5$. For each combination, we generate ten graphs for a total of 400 instances. For each instance, we estimate $\hat{c}_{1,2}$ for each candidate value $\hat{p}_{1,2} \in \{0, 0.025, \ldots, 1\}$.

For the second set, we have $n \in \{50, 100, 150, 200, 250\}$, $\alpha_1 \in \{0.15, 0.3, 0.45\}$, and $\alpha_2 = 1 - \alpha_1$. For each combination such that $c_{1,2} \leq \frac{\min\{\alpha_1, \alpha_2\} n}{2}$, we generate two times thirty graphs for a total of 30,600. For thirty instances, we estimate $\hat{c}_{1,2}$ and $\hat{p}_{1,2}$ with a grid search on $\hat{p}_{1,2} \in \{0, 0.025, \ldots, 1\}$. For the other thirty instances, we estimate $\hat{c}_{1,2}$ and $\hat{p}_{1,2}$ with a golden-section search on $\hat{p}_{1,2}$ with tolerance 0.001.

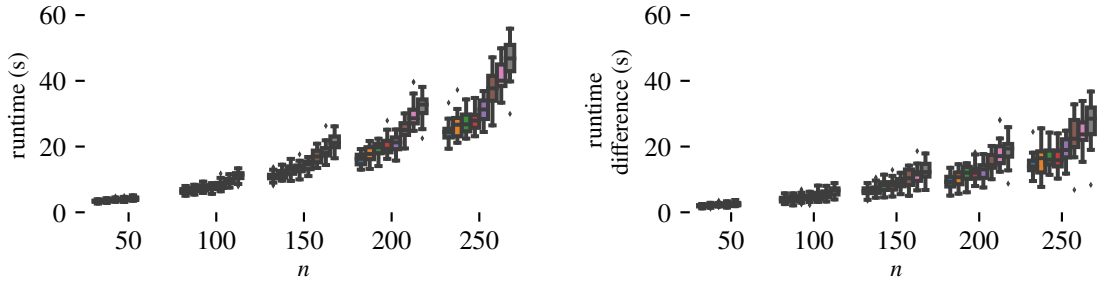
B.2.2. Performance improvements through a golden-section search. Here, we provide evidence that it is not necessary to consider a grid search over the values of $\hat{p}_{\ell,\ell+1}$. Recall that, for a given value of $\hat{p}_{\ell,\ell+1}$, we estimate a corresponding $\hat{c}_{\ell,\ell+1}$. We then search for the $\hat{p}_{\ell,\ell+1}$ that minimizes the spectral distance d between the partial network consisting of tiers $(\ell, \ell+1)$ and simulations of this partial network. These simulations are generated with *RG4SC* under the assumption of the current set of parameters, especially the current $\hat{p}_{\ell,\ell+1}$ and resulting $\hat{c}_{\ell,\ell+1}$.

We argue that d is unimodular in $\hat{p}_{\ell,\ell+1}$. This implies that a golden-section search is sufficient for identifying the optimal value of $\hat{p}_{\ell,\ell+1}$. Figure EC.10 displays exactly this behavior: for each instance displayed here, we consider all possible $\hat{p}_{1,2}$ and the corresponding spectral distance. We see that the spectral distance is minimized when $\hat{p}_{1,2} = p_{1,2}$ and that the lines are largely unimodular. While we observe some deviations from unimodularity for low $c_{1,2}$ and $p_{1,2}$, these deviations largely disappear with $c_{1,2} \geq 5$.

Using the golden-section search drastically improves the runtime of the simulation. Figure EC.11a displays the runtime with a full grid search, while Figure EC.11a displays the reduction achieved with the golden-section search.

B.2.3. Quality of the estimation procedure. We next analyze the performance of our estimation procedure, in particular the quality of the estimates $\hat{c}_{\ell,\ell+1}$ and $\hat{p}_{\ell,\ell+1}$. We note that differences between the grid search and golden-section search approaches are marginal, so we focus on the performance of the latter.

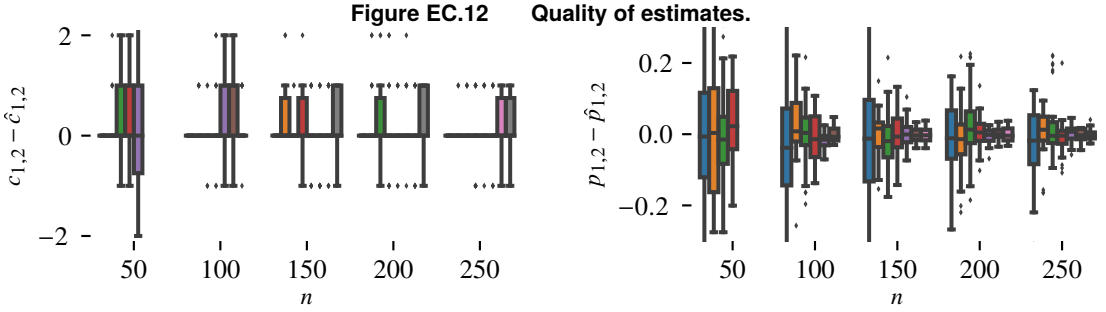
Figure EC.11 Estimation runtime using grid search and reduction from golden-section search.



(a) Grid search (b) Reduction with golden-section search

Note. Data is from the second set, with $p_{1,2} = 0.4$ and $\alpha = (0.3, 0.7)$. Bars from left to right indicate $c_{1,2} = 1, 2, 3, 4, 5, 10, 15, 20$. Not all bars may be displayed due to the condition $c_{1,2} \leq \frac{\min\{\alpha_1, \alpha_2\} n}{2}$.

Figure EC.12 Quality of estimates.



(a) Difference $c_{1,2} - \hat{c}_{1,2}$ (b) Difference $p_{1,2} - \hat{p}_{1,2}$

Note. Data is from the second set, with $p_{1,2} = 0.4$ and $\alpha = (0.3, 0.7)$. Bars from left to right indicate $c_{1,2} = 1, 2, 3, 4, 5, 10, 15, 20$, where 1 is omitted for Figure (b). Not all bars may be displayed due to the condition $c_{1,2} \leq \frac{\min\{\alpha_1, \alpha_2\} n}{2}$.

As seen in Figure EC.12, differences between the true value and the estimate are small and approach 0 as n or $c_{1,2}$ increases. Not shown here is that the performance of the estimator further increases with an increase in $p_{1,2}$, as well as an increase in the asymmetry of α_1 and α_2.

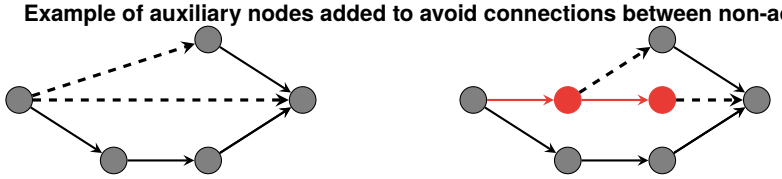
B.3. The empirical validation process in detail

B.3.1. Description of the dataset generation. We first describe how we modify the 38 supply chain networks published in Willems (2008) in order to derive a dataset for the empirical validation of *RG4SC*. For each network, we read in nodes and edges from the original data. We then sort nodes based on their topological generation, that is, nodes without incoming edges are added to tier k, nodes that only have incoming edges from tier k are added to tier $k-1$, etc. As a result, if a node may be in multiple tiers based on its connections, it is placed as far upstream as possible.

We further adjust the tier assignments: For tier $\ell = 2, 3, \ldots, k$, we check for all nodes i with $\ell_i = \ell$, whether the node can be moved to a tier further downstream without needing to move its outgoing neighbors. Denote the set of nodes J^i such that $j \in J^i$ if and only if $(i, j) \in \mathcal{E}$. Let $\hat{\ell} = \min_{j \in J_h} \ell_j$. If $\hat{\ell} < \ell - 1$, then we can move i further downstream and assign it to tier $\hat{\ell} + 1$. This allows us to add fewer auxiliary nodes in the following step.

Next, we ensure that nodes only connect to neighbors in adjacent tiers, as specified by *RG4SC*. While we could extend *RG4SC* to allow for edges skipping tiers instead, we believe that the additional complexity in notation and in

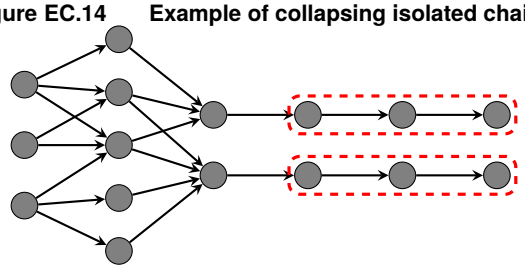
Figure EC.13 Example of auxiliary nodes added to avoid connections between non-adjacent tiers.



(a) Dashed edges connect across more than one tier.

(b) Auxiliary nodes are added (red) to reconfigure the dashed edges.

Figure EC.14 Example of collapsing isolated chains.



Note. The nodes marked with a dashed red outline are replaced by a single node in our procedure, as they have only one incoming neighbor and at most one outgoing neighbor.

selecting parameters when using *RG4SC* does not outweigh the benefits of a slightly more comprehensive model. This is because any acyclic graph can be converted into a graph without skips with the procedure outlined below.

In particular, for each tier $\ell = k, k-1, \ldots, 2$, consider each node i with $\ell_i = \ell$. For any $h = 2, 3, \ldots$, let J_h^i denote the set of nodes such that $j \in J_h^i$ if and only if $(i,j) \in \mathcal{E}$ and $\ell_j = \ell + h$. If $|J_h| > 0$, remove the connecting edges from i to this set of nodes. Let $\hat{h}$ be the largest h such that $|J_{\hat{h}} > 0|$. Then, add auxiliary nodes $i_2, \ldots, i_{\hat{h}}$ and edges $(i, i_2), (i_2, i_3), \ldots, (i_{\hat{h}-1}, i_{\hat{h}})$. For each $j \in J_h, h = 2, 3, \ldots, \hat{h}$, add an edge (i_h, j) to $\mathcal{E}$. This process is exemplified in EC.13. Adding such auxiliary nodes is only required in 10 cases. Within those 10 cases, auxiliary nodes account for less than 6% of the total nodes at the median.

Finally, we collapse what we term as "isolated chains." That is, the main network is followed by sequences of nodes where each node in a sequence only has a single incoming and outgoing neighbor. Figure EC.14 provides an example. Naturally, *RG4SC* is highly unlikely to generate such sequences based on the random assignment of node positions. It is easy to define an extension that would allow to recreate them. We omit this for the sake of brevity, and for two conceptual reasons: First, it is unclear whether such structures are common more broadly, or whether adjusting *RG4SC* accordingly would be akin to overfitting to the Willems (2008) data. Second, in many optimization application, collapsing these long chains into a single node leads to an equivalent formulation, as long as the problem parameters are adjusted accordingly. The Guaranteed Service Model presented in § 4 is one example.

We collapse isolated chains as follows: If there is a sequence of nodes $i_\ell, i_{\ell-1}, \ldots, i_1$ such that $(i_\ell, i_{\ell-1}), \ldots, (i_2, i_1) \in \mathcal{E}$, but $D^{in}(i_h) = 1 \; \forall h = \ell, \ldots, 2, 1$ and $D^{out}(i_h) = 1 \; \forall h = \ell, \ldots, 2$, remove nodes $i_{\ell-1}, \ldots, i_1$ (and their edges) from the graph. This occurs in 21 cases. For the median network within those cases, the procedure removes 23% of nodes.

B.3.2. Qualitative comparison between original and generated graphs. As a first step to validating our model using the data in Willems (2008), we analyze instances produced by *RG4SC* (after fitting the parameters to a specific network in the dataset). We generally observe a close resemblance, in particular in regards to the presence of hierarchical

Figure EC.15 Supply Chain 16 from Willems (2008).

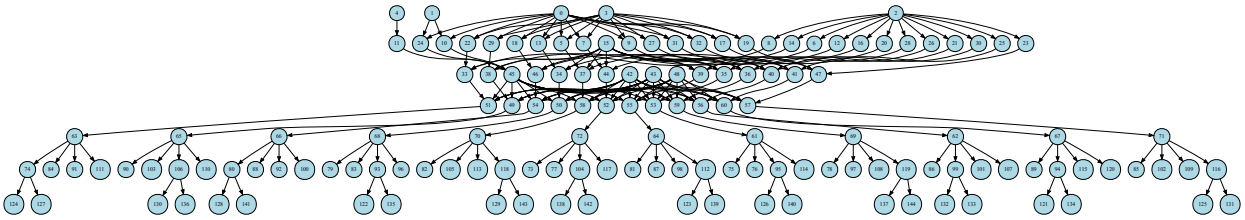
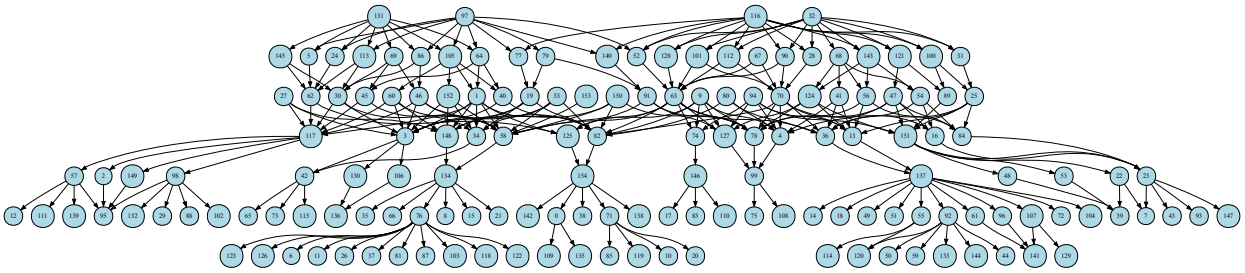


Figure EC.16 Network generated with *RG4SC*, after fitting to Supply Chain 16 from Willems (2008).



and micro-structures. For brevity, we provide one example below: Supply Chain 16, which reflects a network within the "Electromedical and electrotherapeutic apparatus" SIC classification.

Figure EC.15 presents the original network, with tier 1 (resp. k) at the bottom (resp. top) for better visibility. This network is reflected in the original data, as no modifications occur according to the process outlined in Appendix B.3.1.

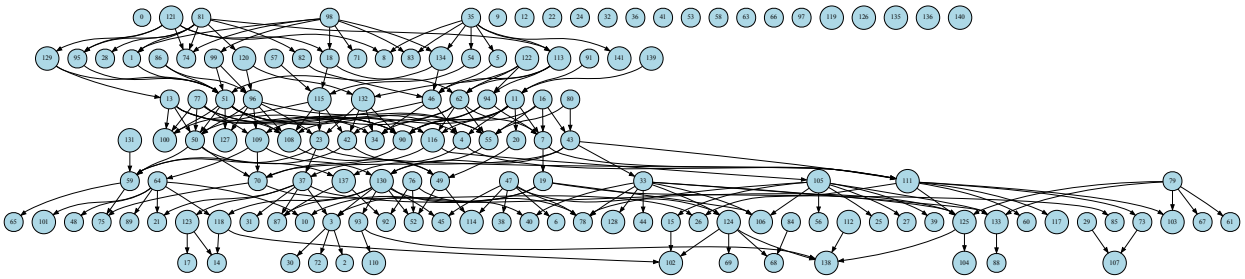
Figure EC.16 displays a typical instance generated by *RG4SC* after estimating the parameters from the original network, using the procedure outlined in § 3.3. As in the original network, there is a pronounced change from the globally connected deeper tiers (at the top), to the locally connected demand-facing tiers (at the bottom). Similar to the original network, the latter also display a pronounced modular and somewhat repetitive structure.

For comparison, we also generate instances with the model ER_T (described in Appendix B.3.4). Figure EC.17 displays an example. While the model is able to capture the globally connected structure of the deeper tiers, it fails to create a modular structure in the demand-facing tiers. This is because there is no concept of local connectivity in the model. As the network is relatively sparse (i.e., there are few edges per node), the probability of any individual connection is small, and some nodes are not connected at all. This further reduces the fit with the original network.

B.3.3. Details of the hypothesis testing step. The challenge in evaluating the plausibility of an analytical model is to distinguish between deviations due to the random nature of sampling and those due to a lack of fit. Clauset et al. (2009) provide one of the most widely cited approaches to evaluate the fit between network data and an analytical model. In their case, they analyse the plausibility that node degrees follow a power-law model. The authors propose the following approach to hypothesis testing:

1. Fitting the data to the analytical model. In our case, we fit *RG4SC* by estimating the parameters from a single (original) network G.

Figure EC.17 Network generated with ER_T, after fitting to Supply Chain 16 from Willems (2008).



2. Generating a goodness-of-fit metric. We derive a (simulated) goodness-of-fit metric: In particular, we create 30 networks from the fitted model $RG4SC$, and for each simulated network G_h, we compute the spectral distance $d(G, G_h)$. The average across the spectral distances serves as our goodness-of-fit metric: $m_G = \frac{\sum_{h=1}^{30} d(G,G_h)}{30}$.
3. Generating synthetic data sets from the fitted model. We generate 96 datasets,[1] consisting of one network each, from the fitted model. We denote the set of these networks with $\mathcal{G}_G$.
4. Fitting each data set independently to the model in order to generate another goodness-of-fit metric. For each graph $G' \in \mathcal{G}_G$, we independently repeat steps 1 and 2 in the same way as we did for graph G. Hence, we obtain a different goodness-of-fit metric $m_{G'}$ for each $G' \in \mathcal{G}_G$.
5. Comparing m_G with the distribution of $m_{G'}$ to obtain a p-value, and rejecting the hypothesis when the p-value is too large. In particular, the p-value is taken as the percentage of cases for which $m_G > m_{G'}, G' \in \mathcal{G}_G$. We reject the hypothesis when the p-value exceeds 0.05.

A key feature of this procedure is that $m_{G'}$ is relative to a distribution of networks generated with $RG4SC$ fitted to G' rather than G. In this way, the goodness-of-fit metrics $m_{G'}$ serve as unbiased estimates of the goodness-of-fit metric of G, if the latter were indeed generated by $RG4SC$.

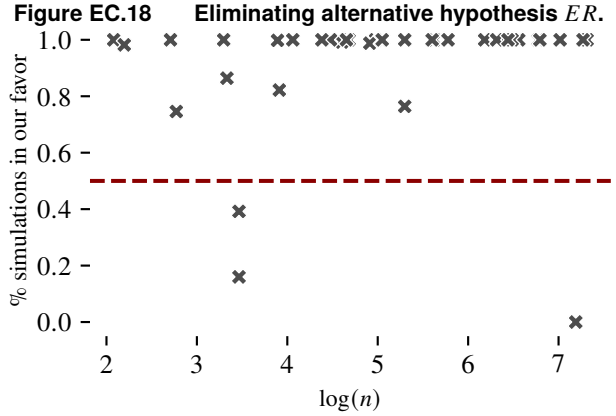
Note that we choose the threshold 0.05 for the p-values, rather than 0.1, as proposed by Clauset et al. (2009). This is because we only have one datapoint per hypothesis test, and because our model is more specific than the power-law model (which specifies only the type of output, but not the generating process).

We also perform an alternative test, in order to further establish the fit between the model and the data. The central idea of this test is to evaluate whether the original network G is as similar to networks generated by $RG4SC$ as if it were indeed generated by the model. We proceed as follows:

1. Fitting the data to the analytical model. In our case, we fit $RG4SC$ by estimating the parameters from a single (original) network G.
2. Generating a set of networks to compare with G. We generate 500 networks from the fitted model $RG4SC$ and add them to the set $\mathcal{G}_1$.
3. Generating a set of networks to derive similarity. We generate another 500 networks from the fitted model $RG4SC$ and add them to the set $\mathcal{G}_2$,.

[1] We choose 96 here for computational reasons, as we are working with a cluster with 48 cores per node.

Figure EC.18 Eliminating alternative hypothesis ER.



4. Establishing a similarity distribution. For each $G' \in \mathcal{G}_1$, we derive $m_{G'} = \frac{\sum_{G'' \in \mathcal{G}_2} d(G',G'')}{500}$. These metrics provide an empirical distribution of similarity to be expected when the network is indeed generated by $RG4SC$.

5. Comparing the similarity of G to the distribution to obtain a p-value, and rejecting the hypothesis when the p-value is too extreme. We compute $m_G = \frac{\sum_{G'' \in \mathcal{G}_2} d(G,G'')}{500}$ and compare this to the previous distribution. We reject the hypothesis if m_G is outside the (symmetric) 90% confidence interval.

With this test, we cannot reject the hypothesis in 29 out of 38 networks, a slightly higher number than with the previous test (26). A potential issue is that we could avoid rejection here by inflating the variance of the graphs produced by $RG4SC$. However, we observe that G fits more closely to the networks $\mathcal{G}_2$ than the comparison networks in $\mathcal{G}_1$ in five cases, while the distance is greater in four cases. This symmetry of rejections supports the applicability of the test.

B.3.4. Details of the elimination of alternatives step. The random attachment model by Erdős and Rényi (1959) is one of the most widely applied models in network analysis (Newman et al. 2002). Hence, we evaluate the model, denoted ER, as an alternative explanation of the Willems (2008) data. The ER model consists of n_{ER} nodes. Each of the potential edges is independently generated with probability p_{ER}. For the comparison, we proceed as follows:

1. Fitting the data to two alternative models. For $RG4SC$, we proceed as outlined in § 3.3. For ER, we estimate $\hat{n}_{ER} = |\mathcal{N}|$ and $\hat{p}_{ER} = \frac{|\mathcal{E}|}{|\mathcal{N}|(1-|\mathcal{N}|)}$.

2. Generating pairs of networks from the fitted models, with one network per model. We generate a network each from the fitted $RG4SC$ and the fitted ER model, and repeat this 500 times. Denote the resulting tuples with $(G_h^{SCRM}, G_h^{ER}), h = 1, \ldots, 500$.

3. Identifying the network in each pair with the smaller spectral distance to the original network. In particular, for $h = 1, \ldots, 500$, we let $m_h = SCRM$ if and only if $d(G, G_h^{SCRM}) < d(G, G_h^{ER})$. Otherwise, $m_h = ER$.

4. Computing the percentage of pairs for which the comparison is in favor of $RG4SC$ rather than the alternative model. That is, we compute $\frac{\sum_{h=1}^{500} \mathbb{1}\{m_h = ER\}}{500}$, where $\mathbb{1}$ is the indicator function.

The result is presented in Figure EC.18, with one point per network from the Willems (2008) dataset. We see that the vast majority of networks are better explained by $RG4SC$. For the networks where ER performs better, this can be explained by the fact that the latter generates networks with exactly the same number of nodes as found in the original network. Meanwhile, for $RG4SC$, only the expected number of nodes is equal to that found in the original network.

Networks in the data are inherently tiered, and this might favor our model. Hence, we next compare $RG4SC$ with the adjusted model ER_T. Here, nodes are (randomly) generated and assigned to tiers and roles as in $RG4SC$. Next, edges are generated according to the random attachment logic: For each edge between nodes at adjacent tiers, the edge is added with a probability $p^{ER_T}_{\ell,\ell+1}$ specific to the tier. We estimate the node-related parameters of ER_T in the same way as we estimate them for $RG4SC$. For the parameters $p^{ER_T}_{\ell,\ell+1}$, we simply identify the proportion of edges that exist between tiers ℓ and $\ell+1$, out of all the possible edges between nodes at those tiers, $|\mathcal{N}_\ell||\mathcal{N}_{\ell+1}|$, where $\mathcal{N}_\ell = \{i \in \mathcal{N} : \ell_i = \ell, r_i \neq t\}$ and $\mathcal{N}_{\ell+1} = \{i \in \mathcal{N} : \ell_i = \ell+1, r_i \neq s\}$. Otherwise, we proceed as above. The comparison is presented in Figure 7b.

Besides the model by Erdős and Rényi (1959), the broader network literature typically uses two further model families as benchmarks (Perera et al. 2017a): 'Small-world' models initially presented by Watts and Strogatz (1998), and 'preferential attachment' models developed by Barabási and Albert (1999). We omit a comparison for conceptual reasons. Small-world models construct networks based on a regular ring lattice, with some edges randomly rewired. There is no obvious extension to generate tiered graphs. As a further sign that these networks are not well suited to represent supply chain networks, we were not able to identify applications in the empirical supply chain literature.

This is different from preferential attachment models, which can be found more broadly in the supply chain literature. However, several properties of networks in practice are inconsistent with the predictions of such a model (Perera et al. 2017b). In particular, the original model produces networks where the degree distribution follows a power law with scaling parameter three. Empirical works predict a significantly smaller scaling parameter. Additionally, the edge generating process does not account for important features found in practice, such as heterogeneity between nodes. As a result, researchers have proposed a number of different extensions (Perera et al. 2017a). However, even with these extensions, a central challenge remains: The preferential attachment model is an evolving model, where nodes are added sequentially, and their attachments are based on the network structure up to this point. Hence, accurately estimating model parameters requires data about network evolution. While it is possible to estimate the parameters from a single snapshot for a simple preferential attachment model (Wan et al. 2017), extending this procedure to a more complex model, for example, one that accounts for tiers, is not obvious.

B.4. Empirical validation with firm-level data.

We next establish that $RG4SC$ is also a plausible model for networks at the level of firm-to-firm relationships. For this purpose, we identify a dataset that provides complete supply chain information for a large number of supply chains. In particular, we follow Attari et al. (2023) in using the Automation of Reports and Consolidated Orders System (ARCOS) database. This dataset, regulated by the Drug Enforcement Administration of the United States, contains information on all transactions of controlled substances between firms involved in drug supply chains and has received significant attention in the context of the Opioid Crisis (Rich et al. 2023). The complete data related to opioids, from January 1, 2016 to December 31, 2019, is freely available through the ARCOS Data Portal at `https://arcos.nd.edu/`.

We focus on transactions related to the substances oxycodone or hydrocodone, which are by far the most relevant substances in the opioid crisis (Rich et al. 2024). Then, for each product containing either of the two substances, we generate an acylic and tiered supply chain network G as follows:

1. For any pair of firms reporting a transaction involving this product between January 1 and December 31, 2019, we add both firms as nodes to G, and we add a directed edge between them.
2. We assign all nodes without outgoing edges to tier $\ell = 0$.

3. We consider tiers $\ell = 1, 2, \ldots$ sequentially. Any node not assigned to a tier is assigned to ℓ if and only if all its outgoing edges connect to a firm assigned to tiers $\ell' = 0, 1, \ldots, \ell - 1$.
4. Nodes not assigned so far are part of directed cyles. We sort these nodes by the percentage of outgoing edges to nodes already assigned, from lowest to highest. We then, sequentially, add nodes to tiers, by removing their edges to unassigned nodes. Say, node i is being added, and that it has a set of outgoing edges $\mathcal{E}_i$ after edges to unassigned nodes are removed. Then, we let $\ell_i = \max_{j \in \mathcal{E}_i} \ell_j + 1$.
5. Any remaining unassigned nodes are isolated and, thus, removed.
6. We again add auxiliary nodes to avoid edges between non-adjacent tiers (see Appendix B.3.1).
7. We remove all nodes assigned to tier $\ell = 0$. These nodes are all dispensing outlets, in particular pharmacies and hospitals. As the number of such outlets exceeds the number of other nodes in G by multiple orders of magnitude, retaining them heavily biases any judgment about the quality of our model to represent the overall network.

We omit any network G with less than 10 nodes or three supply chain tiers. As a result, we retain 135 supply chain networks with a median (interquartile range) of 124 (71–161) nodes. We then apply the same validation process as before, outlined in § 3.3. We find that the hypothesis that *RG4SC* is a plausible model for a network G from the ARCOS dataset cannot be rejected in 83% of cases. Moreover, we can eliminate the alternative hypothesis of a tiered Erdős and Rényi (1959)-type model in 96% of cases. These findings further strenghten the validity of our model.

Appendix C: Proofs and additional results related to the GSM

C.1. Description of instance generation for numerical results in Appendix C

To support our numerical insights in § 4, we generate networks using *RG4SC*, then solve (2) on each instance. For § 4.2 and the first part of § 4.4, we generate networks with $n \in \{60, 80\}$, $k \in \{4, 6\}$, $q_1 = (0, 0, 1)$, $q_\ell = (0, 1, 0)$ $\forall \ell = 2, \ldots, k-1$, $q_k = (1, 0, 0)$, $c_{\ell,\ell+1} \in \{2, 4, 6, 8, 10\}$ and $p_{\ell,\ell+1} \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$ $\forall \ell = 1, \ldots, k-1$. Moreover, we consider the four structural archetypes and set the resulting vector α as in Appendix A.1.

In addition to the *RG4SC* parameters, we also specify a number of parameters for the GSM. First, we let the processing time $T_i = 4$ (resp. inventory holding cost $h_i = 1$) for all nodes i at tier k. Then, the processing time (resp. inventory holding cost) for each subsequent tier is reduced (resp. increased) by a factor in $\{1.0, 1.2\}$ (resp. in $\{1.0, 1.1, 1.2, 1.3, 1.4, 1.5\}$). We also set the service bounds s_i to zero for all sink nodes, that is, the nodes at tier 1.

Second, we assume that $D_i(\tau) = \mu_i \tau + \sigma_i \sqrt{\tau}$ for each node i, and we generate standard deviations in two possible ways. In the first case ("pooling"), we draw $\sigma_j \sim \max\{Normal(10, 2), 0\}$ for all sink nodes j (at tier 1). Then, for tier $k = 2$, we set $\sigma_i = \sqrt[p]{\sum_{i \in \mathcal{N} \,:\, (i,j) \in \mathcal{E}} \sigma_j^p}$, where $p \in \{2, 3\}$. We proceed tier-by-tier, until all values σ_i have been set.

In the second case ("simulation"), we again draw $\sigma_j \sim \max\{Normal(10, 2), 0\}$ for all sink nodes j, but also define $\mu_j = 4\sigma_j$. Assume the sink nodes are denoted $1, \ldots, n_1$. We then define the matrix

$$A = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & \vdots \\ \vdots & \cdots & \ddots & 0 \\ 0 & \cdots & \cdots & \sigma_{n_1} \end{bmatrix}$$

Next, we let $\rho \in \{-0.2, 0.0, 0.6\}$, although we eliminate cases where $\rho < \frac{-1}{\max\{n\alpha_1, 1\}}$, as these can lead to covariance matrices that are not positive semi-definite. Next, we define the covariance matrix for the sink nodes as follows:

$$\Sigma = A \begin{bmatrix} \rho & \rho & \cdots & \rho \\ \rho & \rho & \cdots & \vdots \\ \vdots & \cdots & \ddots & \rho \\ \rho & \cdots & \cdots & \rho \end{bmatrix} A$$

Using the covariance matrix and the previously defined μ_j, we draw demand realizations for the sink nodes. These demands are propagated through the network by assuming that demand at node i equals the sum of demands at nodes j with $(i, j) \in \mathcal{E}$. We simulate 1,000 demand propagations, to derive the 90-th percentile d_i^{90} and mean $\overline{d}_i$ of demand at each node $i \in \mathcal{N}$. Finally, we define $\sigma_i = d_i^{90} - \overline{d}_i$ (also overwriting the previous definition for the sink nodes).

For each combination of options, we generate four graphs for a total of 79,200 and attempt to solve (2) with a time limit of 180 seconds with the Baron solver (Zhou et al. 2018) run on 48 cores. We retain 30,999 instances, where the solver is able to successfully identify a solution guaranteed to be within 10% of optimality. We also solve (2) approximately in each of these instance, using the LP-based iterative heuristic presented in Shu and Karimi (2009).

For the second part of § 4.4, we generate four sets of networks and corresponding optimization solutions. For the first training regime (and for testing), we use the networks form Willems (2008), adjusted as outlined in Appendix B.3.1, that have at most 200 nodes. There are 22 such networks. For each network, we consider each of the GSM parameter combinations outlined above, adding network-GSM parameter pairs to a dataset $\mathcal{D}_1$. We use Baron to solve both (2), as well as a modified version where only demand nodes can have inventory, with a time limit of 180 seconds on 48 cores. We retain any network where the best identified upper bound on the original problem is smaller than the best identified upper bound on the constrained problem. This gives us 1,087 instances, for which we define the optimality gap as the relative gap between the second upper bound and the first.

For the second training regime, we generate "synthetic networks" similar to before, with a few modifications: We extend the range of n to $\{40, 60, 80, 100, 120\}$. For any tier $\ell \notin \{1, k\}$, we let $q_\ell^r = 0.1$ with 20% probability for $r \in \{s, t\}$. Moreover, we select $c_{\ell,\ell+1}$ and $p_{\ell,\ell+1}$ independently and at random for each tier, rather than using the same values for all tiers, to generate a broader variety of structures. The possible values are as before, as are the different GSM parameter options. For each combination of parameters, we generate nine instances, and add the network-GSM parameter pairs to dataset $\mathcal{D}_2$. In total, this leads to approximately 20 times the number of instances generated for the first training regime. After solving the unconstrained and constrained versions of problem (2) as before, we retain 23,892 instances.

For the third regime, we generate bootstrapped versions of the Willems (2008) data, limited to networks with at most 200 nodes. In particular, for each network, we estimate its input parameters, then randomly generate networks from *RG*4*SC*. For each input parameter-GSM parameter pair, we generate 20 networks (i.e., approximately the same total number as synthetic networks), and add the resulting network-GSM parameter pairs to dataset $\mathcal{D}_3$. After solving the unconstrained and constrained versions of problem (2) as before, we retain 21,293 instances.

Finally, to study the evolution of the model performance based on the number of bootstraps, we repeat the first and third regime, limiting the GSM parameters further by only considering the "pooling" mechanism for demand, and inventory cost increase factors in $\{1.0, 1.3, 1.5\}$. Moreover, in the third regime, we generate 50 networks for each input parameter-GSM parameter pair. In the resulting datasets, $\mathcal{D}_1'$ (resp. $\mathcal{D}_3'$), we retain 258 instances (resp. 12,897 instances) after solving the GSM as before.

C.2. Proof of Corollary 1

Recall the following definitions:

DEFINITION EC.5. $f(n) \in O(g(n))$ if and only if $\exists \tilde{M} > 0$ and some value n_0 such that $|f(n)| \leq \tilde{M} g(n)\ \forall n \geq n_0$.

DEFINITION EC.6. An algorithm for a problem with input size n runs in quasi-polynomial time if its worst-case running time can be upper-bounded by $2^{O((\log n)^c)}$ for some constant c.

DEFINITION EC.7. An algorithm for a problem with input size n runs in sub-exponential time if its worst-case running time can be upper-bounded by $O\left(2^{n^\epsilon}\right)$ for all $\epsilon > 0$.

From Blaettchen et al. (2024, Theorem 2), we know that (2) can be solved through a linear program with $O(nM^{tw(G)+1})$ variables and constraints. Hence, following Jiang et al. (2020), there is an algorithm that can solve (2) with worst-case running time $O(n^{2.5})M^{O(tw(G)+1)}$.

Consider the first case and assume $G \in \mathcal{G}$. From Theorem 2, with high probability, $tw(G) \in O(\log n)$. Hence, also with high probability, there is an algorithm that can solve (2) with worst-case running time in $O(n^{2.5})M^{O(\log n+1)}$. Moreover, because M is assumed constant, for any function $f(n) \in O(n^{2.5})M^{O(\log n+1)}$, $f(n) \in 2^{O((\log n)^c)}$ for a sufficiently large constant c.

Consider now the second case. From Theorem 2, with high probability, $tw(G) \in O(n^\epsilon)$ for all $\epsilon > 0$. Hence, also with high probability, there is an algorithm that can solve (2) with worst-case running time in $O(n^{2.5})M^{O(n^\epsilon+1)}$. Moreover, because M is assumed constant, for any function $f(n) \in O(n^{2.5})M^{O(n^\epsilon+1)}$, $f(n) \in O\left(2^{n^\epsilon}\right)$, and the result follows.

C.3. Details of the machine learning experiment

Using the data from Appendix C.1, we generate experiments as follows: We select a test percentage $t \in \{0.3, 0.5\}$ and, for each GSM parameter, a subset of feasible options. From the datasets $\mathcal{D}_b$ $b \in \{1, 2, 3\}$ we then take all instances whose GSM parameters fall within the selected subsets. From the 22 supply chain networks selected from Willems (2008) (the ones with at most 200 nodes), we split off the percentage t. Instances selected from $\mathcal{D}_1$ corresponding to any of these networks form the test set. Instances selected from $\mathcal{D}_3$ corresponding to any of these networks are dropped.

Next, we build three training datasets. For the first regime, we take the remaining networks selected from $\mathcal{D}_1$. For the second regime, we take all networks selected from $\mathcal{D}_2$. For the third regime, we take the remaining networks selected from $\mathcal{D}_3$. We use the training datasets to independently train a gradient boosting regressor to predict the relative optimality gap of the constrained solution to (2). As inputs, we use both network information (number of all nodes, demand nodes, edges, and tiers, mean and standard deviation of degrees, modularity), and GSM information (the parameters used to set up the model). We optimize the hyperparameters of the gradient boosting regressor with the Optuna framework, using 3-fold cross-validation and 100 iterations (Akiba et al. 2019). We then train the best identified model on the entire training set. Finally, we evaluate the trained model on the test set, which contains only instances with networks that have not been seen by the model previously.

To generate different experiments, we consider the power sets of all the GSM parameters described in Appendix C.1. To speed up computations, we omit some subsets. In particular, for parameter ρ, where there are three options, we only consider subsets of size two or three. For the inventory cost increase factor, we consider three subsets of size two, two subsets of size three, and the full set of size six. We also eliminate any experiment where the test set has less than 30 instances. In total, this results in 611 experiments.

Note that we verify our results on the comparison of MSEs with all test set size thresholds above 5. In the worst case, the second regime leads to a lower MSE in 57% of cases, and the third regime leads to a lower MSE in 80% of cases.

Finally, we repeat the process with $\mathcal{D}_1'$ (resp. $\mathcal{D}_3'$) for the first (resp. third) regime, omitting the second regime. Based on a limited selection of GSM parameters, we arrive at 140 experiments. We repeat the training process for the third regime nine times, with a subset of training data that is $5, 10, \ldots, 45$ times the size of the first regime's training data.

References for the E-Companion

Agrawal D, Osadchiy N (2024) Inventory productivity and stock returns in manufacturing networks. *Manufacturing & Service Operations Management* 26(2):573–593.

Akiba T, Sano S, Yanase T, Ohta T, Koyama M (2019) Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Ang E, Iancu DA, Swinney R (2017) Disruption risk and optimal sourcing in multitier supply networks. *Management Science* 63(8):2397–2419.

Attari I, Helm JE, Mejia J (2023) Hiding behind complexity: Supply chain, oversight, race, and the opioid crisis. *Production and Operations Management* (OnlineFirst).

Barabási AL, Albert R (1999) Emergence of scaling in random networks. *Science* 286(5439):509–512.

Bellamy MA, Ghosh S, Hora M (2014) The influence of supply network structure on firm innovation. *Journal of Operations Management* 32(6):357–373.

Blaettchen P, Calmon AP, Hall G, Tawarmalani M (2024) From trees to closed loops: Inventory management in treewidth-bounded supply chain networks. https://tinyurl.com/48scn48e (Accessed Sep 24, 2024).

Clauset A, Shalizi CR, Newman ME (2009) Power-law distributions in empirical data. *SIAM Review* 51(4):661–703.

Eppstein D (2002) Subgraph isomorphism in planar graphs and related problems. Tamassia R, Tollis IG, eds., *Graph Algorithms and Applications I*, 283–309 (World Scientific).

Erdős P, Rényi A (1959) On random graphs I. *Publicationes Mathematicae* 6(3–4):290–297.

Jackson MO (2010) *Social and economic networks* (Princeton University Press).

Jiang S, Song Z, Weinstein O, Zhang H (2020) Faster dynamic matrix inverse for faster LPs. *arXiv:2004.07470* .

McDiarmid C, Müller T (2011) On the chromatic number of random geometric graphs. *Combinatorica* 31(4):423–488.

Mitsche D, Perarnau G (2017) On treewidth and related parameters of random geometric graphs. *SIAM Journal on Discrete Mathematics* 31(2):1328–1354.

Newman ME, Watts DJ, Strogatz SH (2002) Random graph models of social networks. *Proceedings of the National Academy of Sciences* 99(suppl_1):2566–2572.

Penrose M (2003) *Random geometric graphs*, volume 5 (OUP Oxford).

Perera S, Bell MG, Bliemer MC (2017a) Network science approach to modelling the topology and robustness of supply chain networks: A review and perspective. *Applied Network Science* 2(1):1–25.

Perera S, Perera HN, Kasthurirathna D (2017b) Structural characteristics of complex supply chain networks. *2017 Moratuwa Engineering Research Conference (MERCon)*, 135–140 (IEEE).

Rich S, Ba Tran A, Williams A (2024) Arcos. https://tinyurl.com/2ws27v8k (Accessed Sep 24, 2024).

Rich S, Moody P, Schaul K (2023) How deeply did prescription opioid pills flood your county? See here. https://tinyurl.com/ya9z6wyx (Accessed Sep 24, 2024).

Shu J, Karimi I (2009) Efficient heuristics for inventory placement in acyclic networks. *Computers & Operations Research* 36(11):2899–2904.

Stegehuis C, Weedage L (2022) Degree distributions in AB random geometric graphs. *Physica A: Statistical Mechanics and its Applications* 586:126460.

Wan P, Wang T, Davis RA, Resnick SI (2017) Fitting the linear preferential attachment model. *Electronic Journal of Statistics* 11(2):3738 – 3780.

Wang Y, Li J, Wu D, Anupindi R (2021) When ignorance is not bliss: An empirical analysis of subtier supply network structure on firm risk. *Management Science* 67(4):2029–2048.

Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. *Nature* 393(6684):440–442.

Willems SP (2008) Real-world multiechelon supply chains used for inventory optimization. *Manufacturing & Service Operations Management* 10(1):19–23.

Zhou K, Kılınç MR, Chen X, Sahinidis NV (2018) An efficient strategy for the activation of MIP relaxations in a multicore global MINLP solver. *Journal of Global Optimization* 70:497–516.