Traceability Technology Adoption in Supply Chain Networks

Philippe Blaettchen
Bayes Business School (formerly Cass), City, University of London, London EC1Y 8TZ, United Kingdom, 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, 77305 Fontainebleau, France, georgina.hall@insead.edu

Modern traceability technologies promise to improve supply chain management by simplifying recalls, increasing visibility, or verifying sustainable supplier practices. Initiatives leading the implementation of traceability technologies must choose the least-costly set of firms—or *seed set*—to target for early adoption. Choosing this seed set is challenging because firms are part of supply chains interlinked in complex networks, yielding an inherent *supply chain effect*: benefits obtained from traceability are conditional on technology adoption by a subset of firms in a product's supply chain. We prove that the problem of selecting the least-costly seed set in a supply chain network is hard to solve and even approximate within a polylogarithmic factor. Nevertheless, we provide a novel linear programming-based algorithm to identify the least-costly seed set. The algorithm is fixed-parameter tractable in the supply chain network's treewidth, which we show to be low in real-world supply chain networks. The algorithm also enables us to derive easily-computable bounds on the cost of selecting an optimal seed set. Finally, we leverage our algorithms to conduct large-scale numerical experiments that provide insights into how the supply chain network structure influences diffusion. These insights can help managers optimize their technology diffusion strategy.

Key words: supply chain traceability; sustainability; technology adoption; network diffusion; computational complexity; fixed-parameter tractability; treewidth

History: This paper has been accepted for publication by *Management Science* on August 27, 2023.

1. Introduction

Modern consumer goods supply chains form complex networks spanning dozens of countries and actors. As a result, most firms cannot reliably trace the products they produce and source beyond a few upstream and downstream supply chain tiers. This limited traceability—or ability to trace the processing history, origin of materials, and final destination of products (ISO 2005)—has several negative consequences. First, a lack of traceability is a major barrier to building sustainable, disruption-resilient supply chains (The White House 2022), limits supply chain coordination, and increases transaction costs (Wilson 2014). Second, firms lacking traceability are prone to extensive recalls (Wowak et al. 2016), with adverse consequences (Lee 2022). Finally, customers increasingly value traceability, so a lack of it can negatively affect demand (Retail Leader 2016).

To counter this traceability deficiency, many firms are leading the development and deployment of new traceability technologies, protocols, standards, and initiatives (henceforth technologies).[1] In the food industry, for instance, these traceability initiative leaders are typically large retailers or processors (e.g., Walmart or Tyson Foods), often in collaboration with IT companies (e.g., IBM and its "Food Trust" traceability solution) or industry consortia (see Naidu and Irrera 2017, Youngdahl and Hunsaker 2018, Haig 2020, Hiba 2023, for more examples). Similarly, in the fashion industry, fast-growing IT companies (such as Textile Genesis) engage with large retail companies (such as H&M, Lenzing, or Bestseller) to lead traceability initiatives (Ahmed and MacCarthy 2021). Traceability initiative leaders aim to have all players in target supply chains adopt their traceability technology or, more ambitiously, have their technology become the industry standard for specific product categories. However, they often find disseminating their technology across supply chains a daunting and "painfully complex" task (Saenz and Hinkel 2022).

This complexity is due to two key reasons. The first is a network effect unique to supply chain technologies called the *supply chain effect*. To illustrate this effect, consider a producer of chocolate-based products who wishes to use a traceability technology to trace the origins of its cocoa and the final destination of its products. While traceability may benefit all players in the chocolate supply chains (e.g., by improving the detection of supplier malpractice, increasing demand visibility, or enabling sustainability certifications), benefits are only obtained if products are traceable throughout their entire supply chains. If a subset of firms in the supply chain does not adopt the technology, the product produced by that supply chain is no longer fully traceable, and technology adoption benefits are drastically diminished for all firms. Thus, the supply chain effect requires most or even all firms involved in the product's supply chain to adopt the traceability technology for firms to benefit (see, e.g., Behnke and Janssen 2020, Sternberg et al. 2021). The second reason is the complex structure of modern supply chains. Traceability initiative leaders interact with thousands of firms in hundreds of partially overlapping supply chains, thus forming a *supply chain network*. While these overlapping supply chains help alleviate the supply chain effect, they complicate the design of traceability technology dissemination strategies.

Traceability initiative leaders often proactively engage with a set of early adopter firms in the supply chain network to jumpstart technology diffusion. We refer to this set as the network's *seed set*. Engagement with the seed set is costly and usually includes pilot programs, subsidies, or cost-sharing incentives. The leader's goal is to have seed set firms adopt the technology and then influence other firms into adopting it, triggering broad technology diffusion (World Economic

[1] In a recent survey of 150 senior supply chain leaders, 68% identify traceability as a "very or extremely" important issue (World Economic Forum 2021). The value of the traceability technology industry is estimated to grow to US$23 bn. by 2025 (Bhandalkar and Das 2019).

Forum 2021, Saenz and Hinkel 2022). Engaging with the "best" seed set is a critical decision for traceability leaders, and, to build an effective technology dissemination strategy, they must answer a few vital managerial questions: *(i) What is the lowest-cost seed set that ensures the whole network eventually adopts the technology? (ii) How does the size of the seed set depend on the network structure?* and *(iii) What are the different roles that seed set firms play in the diffusion process?*

The existing network diffusion literature addresses these questions through various mathematical models and frameworks (e.g. Rogers 2010). However, none of these models are tailored to supply chain networks, nor the specificities of traceability technology. In particular, there is no research on how the supply chain effect influences technology diffusion. Our paper aims to fill this gap by introducing a new model which incorporates the supply chain effect and can be used to optimize the dissemination of traceability technology in supply chain networks. This model enables us to answer the strategic questions above and can guide the design of traceability technology diffusion strategies. From a theoretical perspective, our framework extends and applies recent results from Integer Programming to technology diffusion, building a new bridge between these two fields.

More specifically, our theoretical contributions are as follows. In Section 3, we introduce our new technology diffusion model, the Supply Chain Traceability Model (SCTM), and formalize the seed set selection problem ($MIN\text{-}SCTM$). In Section 4, we prove that $MIN\text{-}SCTM$ is hard to solve and approximate and that the supply chain effect drives this complexity. In light of this result, any exact solution algorithm for $MIN\text{-}SCTM$ must be parametrized by a structural parameter of the network. We propose such an algorithm in Section 5, more specifically, a fixed-parameter tractable (FPT) linear programming-based algorithm with parameter *treewidth* of the supply chain network. This parameter measures how "tree-like" the network is and is low for real-world supply chains. We further provide two approximation schemes for $MIN\text{-}SCTM$. One is a principled heuristic that returns upper and lower bounds on the optimal cost, explicitly trading off accuracy with computational time (Section 5). The other is a simple heuristic based on our managerial insights (Section 6). These algorithms and heuristics collectively answer question *(i)* above.

We then conduct a series of large-scale numerical experiments using our optimization framework to answer questions *(ii)* and *(iii)*. In Section 6.1, we address *(ii)* and find that the *Jaccard clustering* of a supply chain network is a crucial predictor of the seed set size. This measure can thus estimate the effort required to disseminate a traceability technology. Section 6.2 examines *(iii)*. We observe two types of early adopter firms in the seed set: *starter* and *helper* firms. Starter firms are positioned within supply chains that are made traceable early in the diffusion process and help "jumpstart" diffusion. Conversely, helper firms are part of supply chains that become traceable at later stages of the diffusion process. These firms help circumvent the supply chain effect and "transfer" diffusion across different network parts. We show that the ratio of starter-to-helper nodes in the seed set

has a non-linear relationship with a network's *modularity*. Our insights can help managers tailor their diffusion strategy to a supply chain network's structure.

2. Literature Review

Most operations management papers on traceability technologies focus on the tools and IT infrastructure supporting traceability, ranging from RFID (Dutta et al. 2007, Heese 2007, Whang 2010) to data management systems such as blockchain technology (Babich and Hilary 2020, Chod et al. 2020, Cui et al. 2023). In contrast, our paper abstracts away specific technological details and focuses on diffusion across supply chains, an issue that affects all traceability technologies.

Our approach contributes to the network diffusion literature by building on the Linear Threshold Model (LTM) (Granovetter 1978). In the LTM, a node in a network adopts an innovation (such as a new technology) after a certain fraction of its neighbors have adopted the same innovation. The model we develop, the SCTM, relates to the LTM in two ways. First, the SCTM is a generalization of the LTM, allowing for interactions through hyperedges, not only direct neighbors. Thus, the SCTM encodes the supply chain effect—where *sets* of firms in a supply chain must adopt the technology for adoption benefits to become available—which the LTM cannot do directly. Second, as described in Section 3.3, by introducing an auxiliary graph to our supply chain network hypergraph, one can view the SCTM as a weighted generalization of the LTM on a highly structured graph. The auxiliary graph enables us to relate our results to existing results for the target set selection (TSS) problem in the LTM. For example, our hardness and inapproximability results for $MIN\text{-}SCTM$ add to the results on the hardness of TSS under additional structural assumptions (Kempe et al. 2003, Chen 2009). We also provide an extension of the dynamic programming-based fixed-parameter tractable algorithm for TSS by Ben-Zwi et al. (2011) to our setting.

This paper also builds on recent results from the integer programming literature. Specifically, Laurent (2009) and Bienstock and Muñoz (2018) show that binary linear programs are amenable to linear programming reformulations that are FPT in the treewidth of a graph related to the original formulation. We employ these results to derive an LP-based FPT algorithm for $MIN\text{-}SCTM$. By doing so, we open up a new application area for these integer programming techniques in network diffusion while providing an innovative approach to TSS in the LTM.[2] Namely, our LP formulation overcomes the implementation difficulties of the existing dynamic programming approach and provides a principled way for designing heuristics and obtaining managerial insights.

3. The Supply Chain Traceability Model (SCTM)

We introduce our model in Section 3.1 and show how it describes different supply chain relationships in Section 3.2. In Section 3.3, we present an auxiliary graph that is key to solving $MIN\text{-}SCTM$.

[2] While we do not apply our algorithm to this specific problem, we could easily extend our approach.

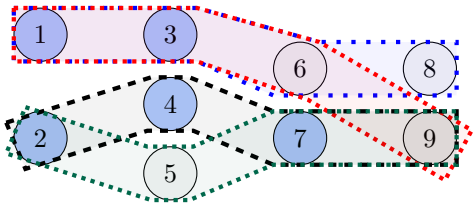


Figure 1 **Example of a hypergraph G with $n = 9$ and $m = 4$. Here, $N_F = \{1, \ldots, 9\}$ and $E = \{e_{blue}, e_{red}, e_{green}, e_{black}\}$ with $e_{blue} = \{1, 3, 6, 8\}$, $e_{red} = \{1, 3, 6, 9\}$, $e_{green} = \{2, 5, 7, 9\}$, $e_{black} = \{2, 4, 7, 9\}$.**

3.1. Model Description

Consider a hypergraph G with nodes $N_F = \{1, \ldots, n\}$ and hyperedges[3] $E = \{e_1, \ldots, e_m\}$. Figure 1 provides an example. Nodes could represent firms and hyperedges subsets of firms collaborating to produce a product (see Section 3.2). The size of e_j is k_j, and k_j is upper-bounded by a constant k. Without loss of generality, we assume G is connected—otherwise, we repeat our analysis on each connected component independently.

State of the Network. Each node $i \in N_F$ has a state $x_{it} \in \{0, 1\}$ at the end of period $t \in \{0, 1, 2, \ldots\}$. A node in state 1 is said to be *active* or have *adopted the technology*. Once active, a node remains in state 1 for all future periods. The *state of the network* at time t, given by S_t, is the set of active nodes at the end of period t, that is $S_t = \{i \in N_F \ : \ x_{it} = 1\}$.

Parameters of the Network. For each node i and hyperedge e_j, there is a *benefit* r_{ji} provided by e_j to i. For each node i, there is also an *adoption* cost, c_i, and a *seeding* cost, w_i. Each hyperedge e_j has an *adoption threshold* θ_j, which is the minimum number of nodes in e_j that need to be active for e_j to become *active* or *traceable*.

Activation Process and State Equation. In periods $t \in \{1, 2, \ldots\}$, a node $i \in N_F$ in state 0 decides whether to become active and switch to state 1. Node i switches states if the adoption cost c_i is outweighed by the adoption benefit $b_i(S_t)$, computed in the following way. Consider the set of all hyperedges e_j that i belongs to and which are traceable *after node i decides to become active*. We let the set of indexes of these hyperedges be $\mathcal{B}_i(S_t)$. Then, $\mathcal{B}_i(S_t) = \{j \ : \ i \in e_j, \ |S_t \cap e_j| \geq \theta_j - 1\}$.[4]

Each hyperedge e_j with $j \in \mathcal{B}_i(S_t)$ generates traceability benefit $r_{ji} \geq 0$ for node i once the node is active. Thus, the benefit i obtains from becoming active is $b_i(S_t) = \sum_{j \in \mathcal{B}_i(S_t)} r_{ji}$. We define the activation process's state equations, which we call the *SCTM activation process*, as

$$S_{t+1} = S_t \cup \{i \in N_F \setminus S_t \ : \ b_i(S_t) \geq c_i\}, \ \forall t = 0, 1, \ldots.$$

Given a hypergraph G and some initial set $S_0 \subseteq N_F$, this process is well-defined: A unique final set of adopters $S_\infty \subseteq N_F$ exists and can be attained in a finite number of steps.

[3] A hyperedge $e_j, j = 1, \ldots, m$ in hypergraph G is a subset of nodes in N_F.

[4] This definition comes from the following argument: Let $j \in \mathcal{B}_i(S_t)$. Then, $i \in e_j$ and $\sum_{i' \in e_j} x_{i't} \geq \theta_j$. If i decides to activate, then $x_{it} = 1$. Therefore, $\sum_{i' \in e_j} x_{i't} = \sum_{i' \neq i, i' \in e_j} x_{i't} + 1 \geq \theta_j$, which is equivalent to $|S_t \cap e_j| \geq \theta_j - 1$.

Activation Process Example. Figure 1 illustrates the activation process. We focus on the network structure's impact and set $r_{ji} \geq c_i \ \forall i \in N_F, e_j \in E$. Let the initial set of active nodes that have adopted the technology be $S_0 = \{1,2,3,4,7\}$ and assume we are in period $t = 1$. We further assume that traceability benefits only kick in when all nodes in a hyperedge have adopted the technology, that is, $\theta_j = 4 \ \forall e_j \in E$. In the case of e_{blue}, e_{red}, and e_{green}, there are two active nodes, so no node can obtain a benefit from these hyperedges by becoming active. However, hyperedge e_{black} has three active nodes, with only Node 9 inactive. By becoming active, Node 9 ensures that e_{black} is active or traceable and obtains benefits. In particular, $\mathcal{B}_9(S_0) = \{j : 9 \in e_j, \ |S_0 \cap e_j| \geq 3\} = \{black\}$ and $b_9(S_0) = r_{black,9}$. Because $r_{black,9} \geq c_9$, Node 9 becomes active, and $S_1 = \{1,2,3,4,7,9\}$. In period $t = 2$, hyperedges e_{red} and e_{green} are one node away from becoming active via Nodes 6, respectively 5. Consider Node 6: $\mathcal{B}_6(S_1) = \{j : 6 \in e_j, \ |S_1 \cap e_j| \geq 3\} = \{red\}$, so its benefit is $b_6(S_1) = r_{red,6} \geq c_6$. Similarly for Node 5: $\mathcal{B}_5(S_1) = \{j : 5 \in e_j, \ |S_1 \cap e_j| \geq 3\} = \{green\}$, so the benefit is $b_5(S_1) = r_{green,5} \geq c_5$. Both nodes become active, and $S_2 = \{1,2,3,4,5,6,7,9\}$. This leaves Node 8 as the only inactive node at the end of period $t = 2$. Since $\mathcal{B}_8(S_2) = \{j : 8 \in e_j, \ |S_2 \cap e_j| \geq 3\} = \{blue\}$, we have $b_8(S_2) = r_{blue,8} \geq c_8$ and Node 8 becomes active in period 3. Thus, $S_3 = S_\infty = N_F$.

Decision. At time 0, the decision-maker chooses the initial set of adopters, or *seed set* $S_0 \subseteq N_F$. Other nodes are in state 0. The decision-maker could be, for example, a retail chain interested in tracing the origins of a set of products it sells. We provide a detailed discussion in Section 3.2.

Problem Formulation. The decision-maker chooses the lowest-cost seed set S_0 so all nodes $i \in N_F$ eventually become active. We call this problem *MIN-SCTM*:

$$\begin{aligned} OPT = \min_{S_0 \subseteq N_F} \quad & \sum_{i \in S_0} w_i \\ \text{s.t.} \quad & S_\infty = N_F \\ & S_{t+1} = S_t \cup \{i \in N_F \setminus S_t \ : \ b_i(S_t) \geq c_i\}, \ \forall t = 0,1,\ldots \end{aligned} \tag{1}$$

Here, given a budget constraint, we minimize the seed set cost to achieve a final set of adopters rather than maximize the size of the final set of adopters. This is because the former is more appealing in practice: network effects and economies of scale inherent to traceability technology will eventually lead to a single industry-wide standard. For a decision-maker to reap benefits in the long term, broad adoption is needed, even at a high initial cost. In addition, we constrain this final set of adopters to be N_F, as it leads to an upper bound on the seeding cost of any subset of N_F. Our results can be directly extended to any subset of N_F.

Assumptions on the Parameters. We assume that all nodes can become active, that is, $\sum_{j:i \in e_j} r_{ji} \geq c_i \ \forall i \in N_F$. We also assume that all values r_{ji} and c_i are integers—values can be scaled if they are initially rational. For a given $i \in N_F$, if the greatest common divisor g_i of $\{r_{ji}\}_j$ and c_i

is larger than one, we consider benefits $\{r_{ji}/g_i\}_j$ and cost c_i/g_i. This is a useful preprocessing step, as the runtime of our subsequent algorithm will scale with the maximum of costs and benefits. Moreover, we assume $\theta_j \in \{2, 3, \ldots, k_j\}$. If $\theta_j > k_j$, hyperedge e_j never provides traceability benefits, and we remove it from G. If $\theta_j = 1$, traceability benefits prevail when just one node is active, so we can remove the hyperedge and reduce the adoption cost of each node $i \in e_j$ by the node's benefit, that is, $c_i := c_i - r_{ji}$. Finally, we assume that $c_i > 0$: If $c_i \leq 0$ for a node i, we can remove the node and let $\theta_j := \theta_j - 1$ for all j with $i \in e_j$. If this leads to $\theta_j < 2$, we repeat the previous simplification.

3.2. Model Discussion and Examples

We assume that G is a *supply chain network*. A node $i \in N_F$ represents a *firm*, and a hyperedge $e_j \in E$ a *supply chain*, that is, a subset of firms collaborating to produce product j. Suppose G is the supply chain network of a given product category (such as fruits, vegetables, or dairy). Then, nodes may not be entire firms but rather divisions handling this product category. Thus, our model allows for *parts of firms* to adopt traceability technology rather than firms in their entirety.

The benefit r_{ji} obtained by firm i if the supply chain of product j is traceable can correspond to a guarantee of continuing or expanding demand for the product or other more intangible benefits such as improved supply chain resiliency, better coordination, or lower quality costs. As traceability benefits are manifold and take different forms at different supply chain stages, they can be challenging to estimate, and traceability initiatives spend significant efforts on this task. Absent detailed information on the value of r_{ji} for each firm, an initial estimate for r_{ji} can be obtained by considering the total traceability benefits to a supply chain j (which are simpler to estimate than the individual benefits) and prorating them to r_{ji} according to the value added by each firm.

The adoption threshold θ_j is the number of firms in hyperedge e_j that must adopt the traceability technology for product j to become traceable. We assume that all nodes in e_j contribute equally to this threshold, though, in practice, contributions may be unequal. We can extend our model to this setting, but the notation and analysis become more complex, so we leave this to future research. If θ_j is unknown, one can set $\theta_j = k_j$, which implies that *all* firms in the supply chain of product j must be active to obtain traceability benefits. This worst-case scenario is a natural assumption for several traceability applications and provides an upper bound on the optimal seeding cost.

The adoption cost c_i of the traceability technology by firm i can represent IT, auditing, and training costs and all discounted future costs of operating the technology. If firms receive technology adoption benefits independently of other firms' adoption, we can consider c_i as the adoption cost net of such benefits. Our adoption process assumes that this cost is a one-time cost for each firm and that, once the technology has been adopted by firm i, the technology is available for all other products (if any) processed by firm i. Without detailed firm-level information on c_i, one can set it to reflect the size of i plus any fixed technology adoption costs.

Decision-Makers. The most common decision-maker for (1) is a large firm wishing to make products it processes traceable, i.e., it is a node of G. We refer to such a node as the network's *lead firm*.[5] Alternatively, the decision-maker can be a traceability initiative external to the supply chain network, such as a large IT company or certification organization, or even a mixture of internal and external players (Naidu and Irrera 2017, Youngdahl and Hunsaker 2018).

Our model can deal with both of these scenarios. If the decision-maker is an external initiative, it can solve (1) directly to obtain the optimal seed set. If the decision-maker is internal to the network, i.e., it is a lead firm, it can also solve (1) to obtain the optimal seed set. However, the hypergraph G over which (1) is solved will be slightly different. The set N_F of nodes will include an additional node, corresponding to the lead firm, and this node will have both its seeding and adoption costs set to zero.[6] To avoid having to distinguish between both cases for the remainder of the paper, we note that solving (1) over G as defined this way is equivalent to solving (1) over a slightly modified graph obtained by deleting the lead firm node from G and setting $\theta_j := \theta_j - 1$ for all hyperedges e_j to which this node belongs (see Section 3.1). In other words, in settings where the decision-maker is internal, one can modify the underlying hypergraph G to revert to the external decision-maker case, so we do not need to distinguish between the two cases.

Assembly Networks and Aggregators. Our model can represent an assembly network, where multiple inputs are needed to produce an output. An example is given in Figure 2a, with six nodes and two hyperedges, $e_{blue} = \{1, 2, 4, 6\}$ and $e_{red} = \{3, 5, 6\}$. Here, Node 4 converts the inputs from Nodes 1 and 2 into a single output. Thus, traceability benefits from e_{blue} only kick in if *both* Nodes 1 and 2 (as well as 4) adopt, accounted for by $\theta_{blue} = 4$: if only Nodes 1 and 4 have adopted, then Node 6 will not get any benefits. This example can be generalized to more complex settings.

Power Dynamics in Supply Chain Networks. Hyperedges can also encode more intangible relationships, such as power dynamics, where different firms within a supply chain may have asymmetric abilities to influence technology adoption. For example, a large retailer such as Walmart can have considerable influence over technology adopted by its suppliers (Nash 2018). Our model can encode such dynamics through additional hyperedges:

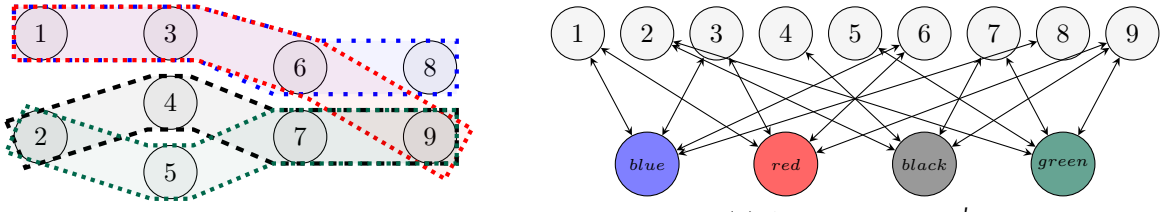
Consider first the hypergraph in Figure 2a. Assuming that $w_i = w_{i'} \ \forall i, i' \in N_F$, that $r_{ji} \geq c_i$, $\forall i \in N_F, e_j \in E$, and that $\theta_{blue} = 4$ and $\theta_{red} = 3$, then $S_0 = \{1, 2, 4, 5\}$ is an optimal seed set. Now, say that Node 6, upon adoption, forces Node 5 to adopt the same technology. We can model this effect by introducing an additional hyperedge $e_{black} = \{5, 6\}$ in Figure 2b, with threshold $\theta_{black} = 2$, benefits to Node 6 of zero ($r_{black,6} = 0$), and benefits to Node 5 corresponding to its adoption costs

[5] We assume we only have one lead firm, but our discussion can easily be adapted for multiple lead firms.

[6] Through its role, the lead firm would have adopted the technology at the start of the diffusion process.

(a) Assembly network. (b) Additional hyperedge for power dynamics.

Figure 2 Representing assembly networks and power dynamics using our model.



(a) Hypergraph G. (b) Auxiliary graph G'.

Figure 3 The supply chain network from Figure 1, as well as the associated auxiliary graph G'.

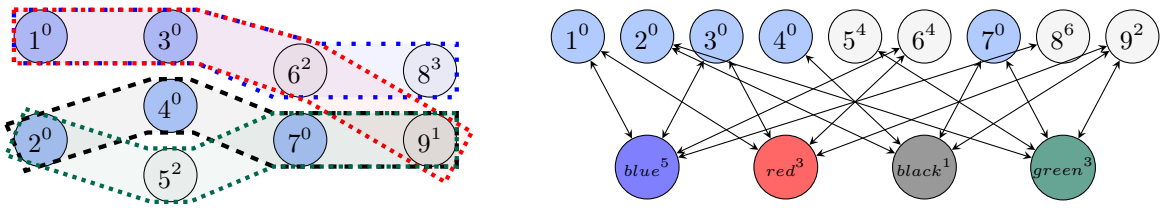
($r_{black,5} = c_5$). If, for example, Node 5 adopts before Node 6, then the additional hyperedge has no effect. If, however, Node 6 adopts at time t, while Node 5 has not adopted, then $black \in \mathcal{B}_5(S_t)$. But $r_{black,5} \geq c_5$, so Node 5 will adopt. Thus, the seed set $S_0 = \{1, 2, 4\}$ with one less node is optimal.

Model Limitations. While *SCTM* can describe rich supply chain relationships, it has a few limitations. For example, the model does not describe potential changes in supply chain relationships and firms' sourcing strategies due to technology adoption decisions. Furthermore, it assumes firms act myopically in each period (a common assumption in network diffusion models) and does not describe more sophisticated strategic games between firms. Enriching *SCTM* to model evolving supply chain dynamics and strategic behavior could be a fruitful source of research and insights.

3.3. The Auxiliary Graph

A crucial construct for solving *MIN-SCTM* is the *auxiliary graph* $G' = (N', E')$ of hypergraph G, which will help connect the SCTM to the linear threshold model (LTM). As mentioned in Section 2, the LTM is a popular model of diffusion in networks where a node becomes active if the number of active neighbors exceeds the node's threshold (Kempe et al. 2003).

Define G' to be a bipartite graph with a node for each firm $i \in N_F$ ("firm-node") and a node for each supply chain $e_j \in E$ ("SC-node", denoted with j). We let N_{SC} be the set of SC-nodes, so $N' = N_F \cup N_{SC}$. Each node $i \in N_F$ (resp. $j \in N_{SC}$) has a threshold $c'_i = c_i$ (resp. $c'_j = \theta_j - 1$). The graph is *weighted* and *directed*. Edges in E' are added as follows: if $i \in e_j$ in G, then $(i, j) \in E'$ with weight $w'_{ij} = 1$ and $(j, i) \in E'$ with weight $w'_{ji} = r_{ji}$. Figure 3 recalls graph G from Figure 1 and displays the corresponding auxiliary graph G' (without weights for legibility).



(a) The SCTM activation process. (b) The LTM activation process.

Figure 4 Equivalence between the SCTM and LTM activation processes. Superscripts represent the period in which a node becomes active.

We next define the activation process on G': A (firm or SC) node i becomes active at time t ($x'_{it} = 1$) if the sum of the incoming edge weights from active nodes exceeds its threshold. In other words, if $i \in N_F$, i becomes active if $\sum_{\{(j,i)\in E' \,:\, j \text{ is active}\}} r_{ji} \geq c_i$. If $j \in N_{SC}$, j becomes active if $|\{(i,j) \in E' \;:\; i \text{ is active}\}| \geq \theta_j - 1$. The activation process on the auxiliary graph can be viewed as a weighted-edge version of the LTM, so we refer to it as the *LTM activation process*. In an analogous way to the SCTM activation process on G, we define state equations, given a set $S'_0 \subseteq N'$:

$$S'_{t+1} = S'_t \cup \left\{ i \in N' \setminus S'_t \;:\; \sum_{j:(j,i)\in E'} w'_{ji} x_{jt} \geq c'_i \right\}, \;\; \forall t = 0, 1, \ldots,$$

where S'_t corresponds to the set of nodes active at time t in G'. As before, a unique final set of adopters S'_∞ is attained in a finite number of steps. Our first result shows that any SCTM activation process on G can be replicated via an LTM activation process on G':

PROPOSITION 1. *Let* $S_0 = S'_0 \subseteq N_F$. *Then,* $S_t = S'_{2t} \cap N_F \;\; \forall t = 0, 1, 2, \ldots,$ *and* $S_\infty = S'_\infty \cap N_F$.

Proofs for this section are in Appendix A. Figure 4 exemplifies the processes' equivalence. In both graphs, we start with the seed set $S_0 = S'_0 = \{1, 2, 3, 4, 7\}$. Under the SCTM, Node 9 becomes active in $t = 1$ because it can make e_{black} active, Node 5 (resp. 6) in $t = 2$, because it can make e_{green} (resp. e_{red}) active, and Node 8 in $t = 3$, because it can make e_{blue} active. Consider now the LTM: At time $t = 1$, Node *black* corresponding to e_{black} has three incoming active neighbors, while its threshold is three. Hence, it becomes active. With Node *black* active, Node 9 has one incoming active neighbor. Because $w'_{black,9} = r_{black,9} \geq c_9 = c'_9$, it becomes active, and $S'_2 \cap N_F = \{1, 2, 3, 4, 7\} = S_1$. Now, both of the Nodes *red* and *green* have three incoming active neighbors, and, again, their thresholds are three, so they become active at time $t = 3$, followed by Nodes 5 and 6 at time $t = 4$. Again, $S'_4 \cap N_F = \{1, 2, 3, 4, 5, 6, 7\} = S_2$. Repeating this one last time, we see that $S'_6 \cap N_F = N_F = S_3$.

Our next result formalizes the minimum cost seed set problem on the auxiliary graph.

COROLLARY 1. *Optimization problem (1) is equivalent to the optimization problem*

$$\begin{aligned} OPT = \min_{S_0' \subseteq N_F} \quad & \sum_{i \in S_0'} w_i \\ s.t. \quad & S_\infty' = N' \\ & S_{t+1}' = S_t' \cup \left\{ i \in N' \setminus S_t' \ : \sum_{j:(j,i)\in E'} w_{ji}' x_{jt} \geq c_i' \right\}, \ \forall t = 0, 1, \ldots \end{aligned} \tag{2}$$

Problem (2) has similarities with the *target set selection (TSS)* problem under the LTM, but also several differences (see Section 2). In a nutshell, the LTM activation process we consider is a weighted version of the regular LTM activation process. Moreover, the underlying graph in the TSS problem is generic. Here, G' has some additional structure: It is bipartite and has constraints on the nodes' degrees and thresholds. Hence, one cannot leverage existing results from the literature to show the problem's hardness (see Section 4). However, one can typically adapt algorithms for solving the TSS to (2) though they can be improved by leveraging the additional structure.

4. Computational Complexity

We turn to the computational complexity of *MIN-SCTM*. To isolate the effects of the graph's structure on the complexity of *MIN-SCTM*, we consider the simplest setting with trivial cost-benefit analysis. Namely, we let $r_{ji} = c_i = 1 \ \forall i = 0, \ldots, n, \ j = 1, \ldots, m$. Thus, any node whose activation would make a supply chain traceable will activate as the benefits will automatically outweigh the costs. This simplified setting can also be of interest in its own right, for example, in traceability systems that aim to restrict counterfeit drugs (see, e.g., Lock 2019). We further assume that $k_j = k$ and $\theta_j = \theta \ \forall j = 1, \ldots, m$. We define the decision version of Problem (1) under these assumptions and provide a *full characterization* of the difficulty of answering it.

DEFINITION 1. *DEC-SCTM* is the decision version of *MIN-SCTM*, with simplified G:
INPUT: Integer h; hypergraph G as defined in Section 3 with benefits $r_{ji} = 1$, adoption costs $c_i = 1$, edges with $k_j = k$ and $\theta_j = \theta$, and rational seeding costs w_i, for all $i = 0, \ldots, n, \ j = 1, \ldots, m$.
QUESTION: Is there a seed set S_0 of cost $\sum_{i \in S_0} w_i \leq h$ leading to full (SCTM) activation of G?

THEOREM 1. *The hardness of answering DEC-SCTM depends on k and θ as follows:*

θ / k	$k = 1$	$k = 2$	$k = 3$	$k \geq 4$
$\theta = 1$	*in P*	*in P*	*in P*	*in P*
$\theta = 2$		*in P*	*in P*	*in P*
$\theta = 3$			*NP-hard*	*NP-hard*
$\theta \geq 4$				*NP-hard*

This section's proofs are in Appendix B. The theorem states that if supply chains in G contain three or more firms, and at least three are needed for traceability benefits, *DEC-SCTM* is hard to answer. Then, if $P \neq NP$, there is no hope of a polynomial-time algorithm for *MIN-SCTM*.

From a managerial perspective, Theorem 1 shows that optimizing technology dissemination when the benefits to a firm only depend on one supplier's or buyer's adoption decision ($\theta \leq 2$) is "easy" (in P). However, once the supply chain effect is in place and the benefits of a traceability technology depend on adoption by second-tier suppliers and buyers ($\theta \geq 3$), there is a phase shift, and optimization becomes complex (NP-hard).

Relationship to Hardness of Target Set Selection under the LTM. Due to the specific weights and thresholds used in Theorem 1 and Corollary 1, Theorem 1 is equivalent to the following result: The decision version of target set selection under the LTM is NP-hard if the underlying graph G' is bipartite, with one set of nodes, N_F, having threshold 1, and the other set of nodes, N_{SC}, having threshold $\theta - 1 \leq k - 1$ and degree k. TSS under the LTM remains NP-hard under various assumptions on the graph and its thresholds (Chen 2009, Ben-Zwi et al. 2011, Centeno et al. 2011, Nichterlein et al. 2013, Chopin et al. 2014). However, none of the existing results cover our specific case. Indeed, the proofs of these results rely on reductions from vertex cover and require the construction of an instance where each node's threshold and degree are equal. This reduction is not feasible in our case due to our structural assumptions (nodes in N_{SC} have degree k and threshold $\theta - 1 \leq k - 1$), so we must resort to a more complex proof. Thus, Theorem 1 also contributes to the LTM literature, more specifically, to understanding which problem structures make TSS hard to solve. This is difficult to determine a priori. For example, TSS is fixed-parameter tractable with respect to treewidth, which tends to be low for sparse graphs, *and* with respect to cluster edge deletion number, which tends to be low for dense graphs (Chen 2009, Nichterlein et al. 2013).

Not only can we not answer *DEC-SCTM* in polynomial time if $k \geq \theta \geq 3$, we cannot provide a meaningful approximation of the true solution under a slightly stronger assumption than $P \neq NP$:

PROPOSITION 2. *For any $k \geq \theta \geq 3$, there exists an $\alpha > 1$ such that, unless $NP \subseteq DTIME(n^{polylog(n)})$, the optimal value to* (1) *with $r_{ji} = c_i = 1 \ \forall i = 0, \ldots, n, \ j = 1, \ldots, m$ cannot be approximated in polynomial time within the ratio of $O\left(\alpha^{\log^{1-\xi} n}\right)$ for any fixed constant $\xi > 0$.*

We have shown that *MIN-SCTM* is hard to solve or approximate when $k \geq \theta \geq 3$, even if the adoption costs are negligible compared to the traceability benefits. These results suggest that complexity is not mainly driven by costs and benefits but rather by the structure of G and the supply chain effect. Any hope of obtaining a polynomial-time algorithm for *MIN-SCTM* must rely on assuming additional structure. We discuss this next.

5. An Exact Solution Algorithm for *MIN-SCTM*

We now provide a linear programming-based *fixed-parameter tractable* algorithm for *MIN-SCTM* with respect to the *treewidth* of G. Precise definitions of these concepts are in Section 5.1, but

at a high level, this means that the complexity of solving $MIN\text{-}SCTM$ is significantly reduced when the treewidth of the supply chain network is small. In light of this, Section 5.1 provides evidence that real-world supply chain networks have small treewidth. Thus, solving $MIN\text{-}SCTM$ on real-world networks is not as hopeless as one may be led to believe by the results in Section 4.

Moving forward, Section 5.2 builds an integer programming formulation of $MIN\text{-}SCTM$ that exploits tree decompositions of G'. While the formulation allows for solving the problem in many practical instances, we go further and leverage the formulation to derive an LP-based FPT algorithm in Section 5.3. The techniques we use come from the integer programming literature and, to our knowledge, have never been applied in the context of network diffusion. Finally, Section 5.4 introduces a hierarchy of LPs that directly trades off computational complexity and approximation quality to obtain lower bounds on the optimal value of $MIN\text{-}SCTM$. We also use this hierarchy of LPs to obtain upper bounds (and corresponding feasible sets).

5.1. Treewidth and FPT algorithms

We define the concept of treewidth (Bodlaender 1994), central to the rest of the paper.

Definition 2. Let G be a (hyper)graph with nodes N and (hyper)edges E. A *tree decomposition* of G is a pair $\mathcal{T} = (T, \{X_z\}_{z \in T})$, with tree T and bags X_z for each node $z \in T$, such that:

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

(b) If $\{i_1, \ldots, i_h\} \in N$ belong to (hyper)edge $e \in E$, there must be a set X_z with $\{i_1, \ldots, i_h\} \in X_z$.

(c) If a node $i \in N$ 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 simply the minimum width over all tree decompositions of G.

A (hyper)graph G always admits a trivial tree decomposition with a single node T containing N, so $tw(G) \leq n - 1$. However, $tw(G)$ is much smaller when G is "tree-like" — trees have treewidth 1.

Two graphs play important roles in our model: the original hypergraph G and the auxiliary graph G'. Figure 5 displays a tree decomposition of G' from Figure 3b with treewidth two. The uppermost bag contains Firm-node 9 and SC-nodes *red* and *black*. Following (b), as SC-node *red* also appears in a bag at the bottom left, it must appear at the intermediate level on the left-hand side. A natural question is how the treewidth of G and G' relate: this is the focus of our next result.

Proposition 3. *Let G be a hypergraph as defined in Section 3.1 and let G' be its auxiliary graph as defined in Section 3.3. Then, $tw(G') \leq tw(G) + 1$.*

The proof of Proposition 3 is in Appendix C.1. Our analysis uses a tree decomposition of G' with treewidth $tw(G') = \omega'$, so we related our results to $tw(G) = \omega$ using the upper bound on ω'.

Another central concept to our work is *fixed-parameter tractability* (Downey and Fellows 2012):

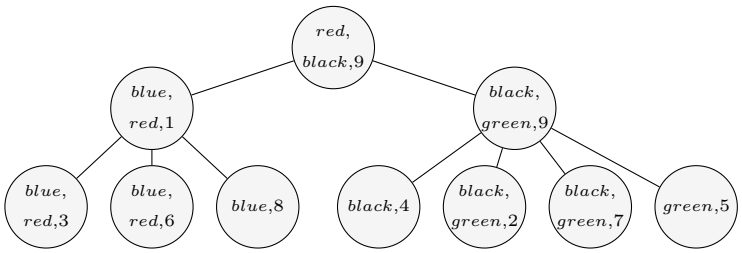


Figure 5 **Tree decomposition of the auxiliary graph in Figure 3b. Each bag in the tree corresponds to a set of nodes from the auxiliary graph.**

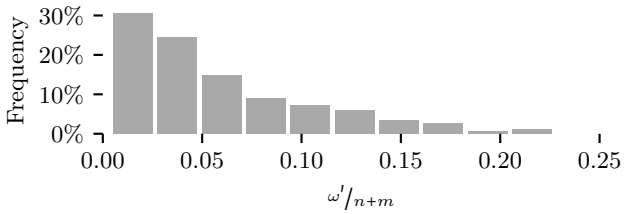


Figure 6 **Histogram of $\frac{\omega'}{n+m}$ for 657 randomly generated supply chain networks based on Willems (2008).**

DEFINITION 3. A problem parameterized by θ is *fixed-parameter tractable (FPT)* with respect to θ if it can be solved in $f(\theta)n^{O(1)}$ time, where the function f does not depend on n.

The algorithms we develop are fixed-parameter tractable in the treewidth of G. Thus, they are particularly valuable in settings where the treewidth of the supply chain network is small. Data from Willems (2008) provides evidence that supply chain networks have small treewidth. The data represents 38 acyclic network structures gathered from companies in 22 industries, from which we randomly generate 657 supply chain networks. The generation process details are in Appendix E.1. Figure 6 displays upper bounds on the auxiliary graph treewidths[7] relative to the networks' sizes for this dataset. We focus on the auxiliary graph treewidth ω', rather than the possibly larger hypergraph treewidth ω, as our results (including Corollary 2) hold for ω'. We observe that ω' is much smaller than the supply chain network size $n+m$. The mean (resp. median) treewidth is 9 (resp. 6), which is less than 3% of the mean size of 310 (resp. 151). More broadly, we note that the supply chain management literature frequently assumes supply chain network graphs to be trees (see, e.g., Graves and Willems 2000), which have a treewidth of one.

5.2. Binary Linear Programming Reformulations of *MIN-SCTM*

The goal of this section is to reformulate *MIN-SCTM* as a binary linear program (BiLP), that is, an optimization problem of the following form:

$$\begin{aligned} \min_{x\in\{0,1\}^n} \; & c^T x \\ \text{s.t. } & Ax \geq b, \end{aligned} \tag{3}$$

[7] To compute a minimal tree decomposition, we use the Flow Cutter algorithm from the *PACE 2017 Parameterized Algorithms and Computational Experiments Challenge* (Dell et al. 2018).

where $c \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}$, and $b \in \mathbb{R}^m$. The next result clarifies why this could be of interest.

Linear Programming Formulations of BiLPs. We first define an *intersection graph*:

Definition 4. The intersection graph of (3) is an undirected graph with a vertex for each variable $x_i,\ i = 1, \ldots, n$ and an edge for each pair (x_i, x_j) that feature in the same constraint.

Proposition 4 (Laurent 2009, Bienstock and Muñoz 2018). *If the intersection graph of (3) has treewidth ω, then there is an equivalent reformulation of (3) as a linear program with $O(2^{\omega} n)$ variables and constraints.*

Since LPs with h variables and constraints can be solved in $O(h^{2.5})$ time (Jiang et al. 2020), solving this LP is an FPT algorithm for (3) with parameter treewidth of the BiLP's intersection graph. Thus, our goal moving forward is to reformulate $MIN\text{-}SCTM$ as a BiLP, whose intersection graph's treewidth is upper-bounded by a function involving ω'. From Proposition 4, one can construct an equivalent linear program, which constitutes our FPT algorithm for $MIN\text{-}SCTM$. As mentioned, an LP approach of this type is new to the network diffusion literature.

A First BiLP Formulation of $MIN\text{-}SCTM$. Ackerman et al. (2010) provide a BiLP formulation of the target set selection problem in the linear threshold model, which—to the best of our knowledge—is the only such formulation in the literature. By virtue of Corollary 1, we adapt the formulation in Ackerman et al. (2010) to our setting as follows. Let

$$E_{F,SC} = \{(i,j) \in E' : i \in N_F, j \in N_{SC}\} \text{ and } E_{SC,F} = \{(j,i) \in E' : i \in N_F, j \in N_{SC}\}.$$

Proposition 5 (Adapted from Ackerman et al. 2010). *Let $G' = (N', E')$ represent the auxiliary graph of hypergraph G. Consider the following BiLP:*

$$\begin{aligned}
\min \quad & \sum_{i \in N_F} w_i s_i \\
s.t. \quad & \sum_{\{j \mid (j,i) \in E_{SC,F}\}} r_{ji} \ell_{ji} \geq c_i (1 - s_i), \ \forall i \in N_F, && \text{(4a)} \\
& \sum_{\{i \mid (i,j) \in E_{F,SC}\}} \ell_{ij} \geq \theta_j - 1, \ \forall j \in N_{SC}, && \text{(4b)} \\
& \ell_{ij} + \ell_{ji} = 1, \ \forall i \in N_F, \forall j \in N_{SC}, && \text{(4c)} \\
& \ell_{i_1 j_1} + \ell_{j_1 i_2} + \ell_{i_2 j_2} + \ell_{j_2 i_1} \leq 3, \ \forall i_1, i_2 \in N_F, \ \forall j_1, j_2 \in N_{SC}, && \text{(4d)} \\
& s_i \in \{0,1\}, \ \forall i \in N_F, \ \ell_{ij}, \ell_{ji} \in \{0,1\}, \ \forall i \in N_F, \ \forall j \in N_{SC}.
\end{aligned}$$

The set $S_0^* = \{i \in N_F : s_i^* = 1\}$ is a solution to the problem $MIN\text{-}SCTM$ on hypergraph G.

The proof of this and the following result are in Appendix C.1. The BiLP constructs a directed acyclic graph (DAG) on N', encoding the activation sequence of G'. In other words, there is an edge

($\ell_{ij} = 1$) between node $i, j \in N'$ if node i contributes to node j's activation. The seed set nodes are then simply the sources of the DAG and are encoded via the variables $s_i,\ i \in N_F$. Constraints (4a) and (4b) enforce that activation can only proceed as described in Section 3, whereas constraints (4c) and (4d) ensure that the final graph is, respectively, directed and acyclic.

Although (4) solves *MIN-SCTM*, the treewidth of its intersection graph is at least $nm - 1$, which precludes us from using Proposition 4 to derive an FPT algorithm from this BiLP.

PROPOSITION 6. *The treewidth of the intersection graph of (4) is at least equal to* $nm - 1$.

Proposition 6 makes it clear that one cannot simply use any BiLP formulation of *MIN-SCTM* to leverage Proposition 4: great care must be taken to formulate the BiLP appropriately.

A Second BiLP Formulation of *MIN-SCTM*. The next formulation is closely tied to a tree decomposition $\mathcal{T}' = (T', \{X'_z\}_{z \in T'})$ of G' with treewidth ω'. For the remainder of this section, we assume wlog that T' is binary. That is, each node $z \in T'$ has at most two children, $c_1(z)$ and $c_2(z)$.[8] To avoid the issues described in Proposition 6, we first replace constraints (4c) and (4d) by constraints that are tree decomposition-dependent. We then obtain

$$
\begin{aligned}
\min \quad & \sum_{i \in N_F} w_i s_i \\
\text{s.t.} \quad & \sum_{(j,i) \in E_{SC,F}} r_{ji} \ell_{ji} \geq c_i(1 - s_i), \ \ \forall i \in N_F, && \text{(5a)} \\
& \sum_{(i,j) \in E_{F,SC}} \ell_{ij} \geq \theta_j - 1, \ \ \forall j \in N_{SC}, && \text{(5b)} \\
& \ell_{ij} + \ell_{ji} = 1, \ \ \forall i, j \in N' \cap X'_z, \forall z \in T', && \text{(5c)} \\
& \ell_{ij} + \ell_{jk} + \ell_{ki} \leq 2, \ \ \forall i, j, k \in N' \cap X'_z, \forall z \in T', && \text{(5d)} \\
& s_i \in \{0,1\}, \ \ \forall i \in N_F, \ \ell_{ij}, \ell_{ji} \in \{0,1\}, \ \ \forall i, j \in N' \cap X'_z, \forall z \in T'.
\end{aligned}
$$

As it turns out, (5) and (4) are equivalent, which we formally state next.

PROPOSITION 7. *Problems* (5) *and* (4) *are equivalent. In particular, let* $(s_i^*, \ell_{ji}^*, \ell_{ij}^*)$ *be a solution to (5). The set* $S_0^* = \{i \in N_F \mid s_i^* = 1\}$ *is a solution to the problem MIN-SCTM on hypergraph* G.

The proof is in Appendix C.2. Although (4) and (5) are equivalent, (5) can have much fewer constraints than (4) if the treewidth of G' is small. To see this, note that (4) has $O\left(n^2 m^2\right)$ constraints, driven by (4d), which makes the activation sequence acyclic. In contrast, since there always exists a (binary) tree decomposition T' of G' with at most $4(n+m)$ bags (see Lemma 13.1.2. of Kloks

[8] If T' is not binary, we choose an arbitrary node as the root and proceed top-down. If a node z has $\tilde{n} > 2$ children $c_1(z), \ldots, c_{\tilde{n}}(z)$, we create a new node z' with bag $X'_{z'} = X'_z$ and add z' and $c_1(z)$ as children of z and $c_2(z), \ldots, c_{\tilde{n}}(z)$ as children of z', and repeat as necessary. This process returns a binary tree whose bags are exactly those in G'.

1994) and since T' has bags of size at most $\omega+1$, Formulation (5) has $O\left((n+m)\cdot\omega^3\right)$ constraints. When ω is much smaller than m and n, (5) will be a much smaller optimization problem than (4).

This difference directly translates to a difference in solving time for (4) and (5). To illustrate, we draw a sample of 150 supply chain networks with $m+n\geq 100$ from the set of networks described in Section 5.1, for which we can solve (4) and (5) within three hours using Gurobi (2022) on a computing cluster with 24 cores. The average time for solving (5) on these instances is 50% less than that for (4), even when accounting for the calculation time of the tree decomposition of G'. When the decomposition is not accounted for, solving (5) takes 72% less time than solving (4) on average. For several instances, we can solve (5) to optimality within three hours but not (4).

Despite the encouraging computational results, it is not clear that the treewidth of the intersection graph of (5) can be upper-bounded by a function of ω'. In fact, the intersection graph's treewidth can be as large as $m+1$. To see this, suppose that $i\in N_F$ belongs to m supply chains. Then, constraint (5b) leads to a clique of size $m+1$ in the intersection graph, containing $\{\ell_{ji}\}_{(j,i)\in E_{SC,F}}$ and s_i. This implies that the treewidth is at least $m+1$, and we can still not use Proposition 4 to derive an LP-based FPT algorithm. Hence, we next introduce new variables representing partial sums of terms appearing in constraints (5a) and (5b). The idea is to replace a constraint such as $v+w+y+z\geq x$ with three equivalent constraints $x_1+x_2\geq x$, $v+w\geq x_1$, $y+z\geq x_2$, leading to an intersection graph of smaller treewidth, at the expense of more constraints and variables. This is similar in spirit to Bienstock and Muñoz (2018). However, our formulation explicitly leverages the tree decomposition of G' as well as the specificities of our problem.

A Third (and Final) BiLP Formulation of $MIN\text{-}SCTM$. Before proceeding, we introduce some new notation. For $i\in N_F$ and $j\in N_{SC}$, we let $J^i=\{j\mid (j,i)\in E_{SC,F}\}$ and $I^j=\{i\mid (i,j)\in E_{F,SC}\}$ be the sets that appear in constraints (5a) and (5b). As mentioned above, our goal is to group the variables appearing in these constraints into partial sums in an effective way. We do this by leveraging T', with the idea that variables that appear in the same partial sum should have indices in the same bag in T' so that the resulting intersection graph has low treewidth. We describe how to do this based on a constraint in (5a). Let $i\in N_F$: we look for a *partition* of J^i into sets $\{J^i_z\}$ such that $J^i_z\subseteq X'_z$ for $z\in T'$. We can then rewrite $\sum_{j\in J^i} r_{ji}\ell_{ji}=\sum_z\left(\sum_{j\in J^i_z} r_{ji}\ell_{ji}\right)$, where $\sum_{j\in J^i_z} r_{ji}\ell_{ji}$ are partial sums with the variables' indices all belonging to the same bag.

Let T'_i be the subtree of T' when restricted to bags containing i, z^i_0 an arbitrarily chosen root node of T'_i, and $|T'_i|$ the number of tree nodes in T'_i. By construction, this tree's bags contain all of J^i, which we next partition. We split tree nodes of T'_i into two sets: a "useful" set T^G_i to build the partition, and a "useless" set $T^{\tilde{G}}_i$ to keep track of over-counting. Set T^G_i is obtained by sequentially adding tree nodes from T'_i while ensuring that the associated bag contains at least one j that is

not already present in the bags of T_i^G. We stop when T_i^G contains all of J^i in its bags. The set $T_i^{\tilde{G}}$ contains the remaining tree nodes of T_i'. We are now ready to build our partition $\{J_z^i\}_{z\in T_i^G}$ of J_i: we arbitrarily number the bags $\{X_z'\}_{z\in T_i^G}$ and let $J_z^i = X_z' \cap N_{SC}$ for the first tree node. For other tree nodes, we let $J_z^i = X_z' \cap N_{SC} \setminus \{j \in J_i \mid j \text{ present in previous bags}\}$. For constraints (5b), we similarly define T_j' as the subtree of T' when restricted to bags containing j, with z_0^j an arbitrary root of T_j' and $|T_j'|$ the number of tree nodes in T_j'. Within T_j', we construct analogous concepts T_j^G and $T_j^{\tilde{G}}$, replacing J^i by I^j. We also let $\{I_z^j\}_{z\in T_j^G}$ be the counterpart of $\{J_z^i\}_{z\in T_j^G}$. Letting

$$n_u = \lfloor \log_2(\max\{c_1,\dots,c_n\}+1)\rfloor - 1 \text{ and } n_v = \lfloor \log_2(\max\{\theta_1,\dots,\theta_m\}+1)\rfloor - 1, \tag{6}$$

we can formulate our final BiLP. As this BiLP is quite cumbersome to write out, we place it in Appendix C.2 as equation (10). Note that (10) is not precisely in the form of (3) to keep the formulation legible. However, it can be obtained by appropriately substituting variables using equations (10d), (10h), and (10i) in constraints (10d) and (10h). This reformulation is a common technique for replacing integer variables with binary ones (see, e.g., Watters 1967). Furthermore, we have not differentiated between nodes with different numbers of children in the constraints. With a slight abuse of notation, we assume that if $c_1(z) = \varnothing$ and/or $c_2(z) = \varnothing$, the corresponding sum is dropped. We now show two results regarding (10), with proofs in Appendix C.2. These enable us to propose an LP-based FPT algorithm for *MIN-SCTM* in the next section.

Proposition 8. *Optimization problems (5) and (10) are equivalent. In particular, let* $\left(s_i^*, \ell_{ij}^*, u_{iz}^{r*}, \tilde{u}_{iz}^{r*}, v_{jz}^{r*}, \tilde{v}_{jz}^{r*}\right)$ *be an optimal solution to (10). The set* $S_0^* = \{i \in N_F \mid s_i^* = 1\}$ *is a solution to the MIN-SCTM problem on hypergraph G.*

Proposition 9. *Let* $\vartheta_{\max} = \max\left\{\max_{i=1,\dots,n}\{c_i\}, \max_{j=1,\dots,m}\{\theta_j\} - 1\right\}$. *The treewidth of the intersection graph of (10) is at most* $O(\omega'^2 + \omega' \log_2(\vartheta_{\max}))$, *where* ω' *is the treewidth of* G'.

5.3. A Linear Programming FPT Algorithm for *MIN-SCTM*

We use Propositions 3, 8, and 9 to show one of our main results.

Theorem 2. *Let G be a hypergraph with* $tw(G) = \omega$. *Then, there is an equivalent reformulation for MIN-SCTM as a linear program with a number of constraints and variables in*

$$O\left(2^{\omega^2} \cdot \vartheta_{\max}^{\omega}\left(n + (n+m)\omega^3 + 8\log_2\left(\max\{c_1,\dots,c_n\}\right)(n+m) + \log_2\left(\max\{\theta_1,\dots,\theta_m\}\right)(n+m)\right)\right).$$

The proof of Theorem 2 can be found in Appendix C.2. Theorem 2 indicates that an LP reformulation of *MIN-SCTM* with $2^{O(\omega^2)}\vartheta_{\max}^{O(\omega)}O(n+m)$ constraints and variables exists. Following Jiang et al. (2020), we then have the following:

Corollary 2. *There is a linear programming-based FPT algorithm for solving MIN-SCTM with parameters* ω *and* $\vartheta_{\max}$, *running in time at most* $2^{O(\omega^2)}\vartheta_{\max}^{O(\omega)}O\left((n+m)^{2.5}\right)$.

Up to this point, we emphasized that our FPT algorithms are with respect to the treewidth ω. Technically speaking, they are FPT algorithms with respect to the parameters ω and $\vartheta_{\max}$. However, one can reasonably assume a bound, independent of n and m, on c_i and θ_j, for $i = 1, \dots, n$ and $j = 1, \dots, m$, and so on $\vartheta_{\max}$. This is because c_i is a firm-dependent parameter that describes the adoption costs. At the same time, θ_j is a "local" supply chain-dependent parameter, describing interactions between a fixed set of firms that does not change as the network scales.[9] Under this assumption, one can remove the dependency of the algorithm's runtime on $\vartheta_{\max}$.

We now present the construction of the LP that Corollary 2 relies on. This formulation differs from the one in, e.g., Bienstock and Muñoz (2018). The main advantage of our formulation is that one can easily use it to derive smaller LPs that provide good-quality lower bounds for $MIN\text{-}SCTM$ (see below for a detailed discussion). It is based on the tree decomposition of the intersection graph of (10), which we denote by $\mathcal{S} = (S, \{W_z\}_{z \in S})$. We also let $\omega_z = |W_z|$. Each bag W_z contains variables that we rename $x_1^z, \dots, x_{\omega_z}^z$, which are subsets of the decision variables in (10). Furthermore, we associate a set of l_z constraints from (10) with each bag W_z: the constraints that only feature variables in W_z. As these are linear in the decision variables, we can write them as $g_{\varnothing}^l + \sum_{i=1}^{\omega_z} g_{\{x_i^z\}}^l \cdot x_i^z \geq 0$ for $l = 1, \dots, l_z$. We are ready to state our LP formulation of $MIN\text{-}SCTM$:

$$
\begin{aligned}
\min_{Y_{\mathbb{S}}} \quad & \sum_i w_i Y_{\{s_i\}} \\
\text{s.t.} \quad & Y_{\varnothing} = 1, \quad \sum_{\{\mathbb{S} \in 2^{W_z} \mid \mathbb{T} \subseteq \mathbb{S}\}} (-1)^{|\mathbb{S}|-|\mathbb{T}|} Y_{\mathbb{S}} \geq 0, \ \forall \mathbb{T} \in 2^{W_z}, \ \forall z \in S, \\
& \sum_{\{\mathbb{S} \in 2^{W_z} \mid \mathbb{T} \subseteq \mathbb{S}\}} (-1)^{|\mathbb{S}|-|\mathbb{T}|} \left(g_{\varnothing}^l Y_{\mathbb{S}} + \sum_{i=1}^{\omega_z} g_{\{x_i^z\}}^l \cdot Y_{\{x_i^z\} \cup \mathbb{S}} \right) \geq 0, \ \forall \mathbb{T} \in 2^{W_z}, \ \forall l = 1, \dots, l_z, \forall z \in S,
\end{aligned}
\tag{7}
$$

where 2^{W_z} corresponds to all possible subsets of variables in W_z. For small values of ω_z, (7) can be solved exactly as commercial solvers allow for millions of variables and constraints. It can be the case, however, that when ω_z becomes larger, this problem becomes difficult to solve due to memory constraints. One advantage of the LP approach is that it provides a principled way of deriving less computationally-intense heuristics, as seen in Section 5.4.

Comparison to a Dynamic Programming-Based FPT Algorithms. We also propose a dynamic programming (DP)-based FPT algorithm for $MIN\text{-}SCTM$ in Appendix D, which is a non-trivial generalization of an algorithm introduced by Ben-Zwi et al. (2011) for the target set selection problem in the LTM. We derive an analogous statement to Corollary 2 for this algorithm and show that its runtime is at most $2^{O(\omega \log_2(\omega))} \vartheta_{\max}^{O(\omega)} (n + m)$. Thus, theoretically, the DP-based

[9] Such an assumption does not preclude $MIN\text{-}SCTM$ from being a hard problem to solve and approximate as evidenced in Section 4 where $\vartheta_{\max}$ is equal to 2.

algorithm has a slightly lower run time than the LP-based algorithm.[10] However, it has the usual caveat of being difficult to implement, requiring specific coding of the algorithmic procedure. Such an ad-hoc approach typically does not optimize for memory or processing capabilities. In contrast, widely available commercial solvers can solve linear programs with thousands of variables and millions of constraints in a matter of minutes. One can further speed up the solving time by using techniques such as set-up parallelization and warm-starting, which are tried-and-tested techniques for LPs. Last but not least, unlike the DP-based algorithm, the LP-based algorithm allows us to systematically derive good-quality lower and upper bounds on $MIN\text{-}SCTM$. We see this now.

5.4. Bounds from a Hierarchy of Linear Programs.

We now derive principled upper and lower bounds on the optimal value of $MIN\text{-}SCTM$.

Lower Bounds on the Optimal Value of $MIN\text{-}SCTM$. Any seed set leading to full activation of the graph directly implies an upper bound on the optimal value of $MIN\text{-}SCTM$. Obtaining lower bounds, on the other hand, that go above and beyond the simple lower bound obtained by considering the continuous relaxation of (10), or equivalently (5), can be trickier. Problem (7) provides us with a process to generate increasingly powerful lower bounds via a *hierarchy* of LPs, that is, a family $\{LP_\kappa\}_{\kappa=1,\ldots,\omega}$, where LP_ω is equal to (7) (i.e., LP_ω solves $MIN\text{-}SCTM$ exactly) and where the objective value of LP_κ is an increasingly tight lower bound on the objective value of (7) as κ grows. Recall the notation given in Section 5.3. The linear program LP_κ is given thus:

$$
\begin{aligned}
\min_{Y_{\mathbb{S}}} \quad & \sum_i w_i Y_{\{s_i\}} \\
\text{s.t.} \quad & Y_{\varnothing} = 1, \\
& \sum_{\{\mathbb{S}\in 2^{W_z} \;\mid\; \mathbb{T}\subseteq\mathbb{S}\subseteq\mathbb{U}\}} (-1)^{|\mathbb{S}|-|\mathbb{T}|} Y_{\mathbb{S}} \geq 0, \;\; \forall \mathbb{T},\mathbb{U}\in 2^{W_z}, \mathbb{T}\subseteq\mathbb{U}, |\mathbb{U}| = \min\{\kappa+1, n\}, \forall z\in S, \\
& \sum_{\{\mathbb{S}\in 2^{W_z} \;\mid\; \mathbb{T}\subseteq\mathbb{S}\subseteq\mathbb{U}\}} (-1)^{|\mathbb{S}|-|\mathbb{T}|} \left(g^l_{\varnothing} Y_{\mathbb{S}} + \sum_{i=1}^{\omega_z} g^l_{\{x_i^z\}} \cdot Y_{\{x_i^z\}\cup\mathbb{S}} \right) \geq 0, \;\; \forall \mathbb{T},\mathbb{U}\in 2^{W_z}, \mathbb{T}\subseteq\mathbb{U}, |\mathbb{U}| = \kappa, \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \forall l = 1,\ldots,l_z, \forall z\in S.
\end{aligned}
\tag{8}
$$

The objective value of (8) increases with κ. When $\kappa = \omega$, it equals the solution to (7) (see Laurent 2003). The LP's size also increases in κ, providing an explicit trade-off between accuracy and computation time. Interestingly, when $\kappa = 0$, we obtain the simple lower bound mentioned above.

PROPOSITION 10. *When $\kappa = 0$, (8) is equivalent to (10), where the binary variables have been replaced by continuous variables on $[0,1]$.*

[10] Recall the definition of an FPT algorithm in Section 5.1. The quality of an FPT algorithm is based on how small one can make $f(\omega)$, rather than the exponent of $(n+m)$ (see, e.g., Lokshtanov et al. 2011). This is important here as our LP-based algorithm is close to its DP counterpart for $f(\omega)$ but not as close for the exponent of $(n+m)$.

Experimentally, we compute lower bounds for a sample of 100 supply chain network instances with $m \geq 50$.[11] We measure the relative gap between the costs when solving LP_κ, and an upper bound obtained when attempting to solve (5) using Gurobi, as described in Section 5.2. When $\kappa = 0$, the median gap (resp. IQR) between our lower bound and the upper bound is -51% (resp. -62% – -42%). When $\kappa = 1$, the median gap dramatically improves to -19% (resp. -31% – -11%), indicating that for small values of κ, we already obtain powerful lower bounds that considerably outperform those that can be obtained from a simple relaxation.

Upper Bounds on the Optimal Value of MIN-$SCTM$. We can further leverage solutions to (8) to obtain upper bounds on the optimal value of MIN-$SCTM$ and corresponding feasible seed sets. The most direct approach would be to sort nodes according to their score $Y^*_{\{s_i\}}$ from the solution. We could then add them to the seed set one-by-one until the entire network activates. However, this does not consider supply chain dependencies and has poor numerical performance. Noting that activation proceeds along supply chains and that some seed nodes are only relevant late in the activation process, our algorithm instead sequentially selects supply chains based on their constituents' scores. This sequential process is motivated by our insights in Section 6.2.

The heuristic, formalized in Algorithm 1, involves three steps. First, for fixed κ, it solves (8). Second, it uses the optimal solution $Y^*_{\{s_i\}} \in [0,1]$, as a "score" for each firm i. The heuristic chooses the (relevant) subset of nodes with the highest average score for each supply chain. Namely, suppose a given supply chain requires h nodes for full activation and has inactive nodes I. Then, the heuristic examines $\binom{|I|}{h}$ combinations of nodes in that supply chain and chooses the combination with the highest average score as a candidate for the seed set. Finally, the heuristic compares the average scores of candidate combinations across supply chains, greedily choosing the combination (and, thus, the next supply chain to become active) with the highest average.

We apply our heuristic to the same 100 instances, using solutions to LP_0 and LP_1. This time, we measure the relative gap between the resulting upper bound and either the previous lower bound or the one obtained when attempting to solve (5) in Gurobi, whichever is higher. When $\kappa = 0$, the median gap (resp. IQR) is 28% (resp. 21% – 42%). When $\kappa = 1$, the gap dramatically improves to 12% (resp. 8% – 20%). Recall that the heuristic provides a feasible solution, so the choice of $\kappa = 1$ already enables high-quality approximations to the optimal seed set. Increasing κ allows us to improve upon those even more. In the case where we take $\kappa = \omega$ and (8) returns the optimal solution $Y_{s_i^*} = s_i^*$, the heuristic always seems (empirically) to return the optimal solution.

[11] We use a dataset similar to the Willems (2008) networks introduced in Appendix E.1. However, to make solving more challenging, we only standardize $r_{ji} = 1$ and randomly vary w_i, c_i, and θ_j for all $i \in N_F$, $j \in N_{SC}$.

Algorithm 1: An LP relaxation-based upper bound.

Data: A hypergraph $G = (N_F, E)$ as defined in Section 3.1.
Result: A feasible seed set S_0 of G.

Initialize: $S_0 = \varnothing$; $\{Y^*_{s_i}\}_{i \in N_F} \leftarrow$ solution to LP_κ applied to G;
while $x_{i,\infty} = 0$ *for some* $i \in N_F$*, given* S_0 *as seed set* **do**
  $s_{\max} \leftarrow 0$; $O_{\max} \leftarrow \varnothing$;
  for $e_j \in E$ **do**
    $I \leftarrow \{i \in e_j : x_{i,\infty} = 0 \text{ given } S_0 \text{ as seed set}\}$;
    $h \leftarrow \theta_j - 1 - \sum_{i \in e_j} x_{i,\infty}$;
    for $O \in choose(I, h)$ **do**
      $s \leftarrow \frac{1}{|O|} \sum_{i \in O} Y^*_{s_i}$;
      if $s \geq s_{\max}$ **then**
        $s_{\max} \leftarrow s$; $O_{\max} \leftarrow O$;
  $S_0 \leftarrow S_0 \cup O_{\max}$;

6. Managerial Insights and a Simple Heuristic for Solving $MIN\text{-}SCTM$

We now show how our optimization framework sheds light on two critical questions: *How does the size of the seed set depend on the network structure?* and *What are the different roles that seed set firms play in the diffusion process?* Answering these questions can help traceability initiative leaders estimate the effort required to disseminate their technology and how to engage with different firms in the supply chain network (World Economic Forum 2021, Sandoval et al. 2022).

Section 6.1 addresses the first question and finds that supply chain networks with a higher degree of *Jaccard clustering*—a measure of how much overlap there is between firms' neighborhoods—tend to have larger seed set sizes. Section 6.2 addresses the second question and shows that networks with intermediate levels of *modularity*—a measure of whether there are communities in the network and how tightly knit these communities are (Newman 2006)—give rise to *helper* nodes in the seed set. These nodes belong to supply chains that become traceable in later stages of the diffusion process and "help" diffusion move between different network parts. Interestingly, modularity and clustering frequently appear in supply chain network analyses (Perera et al. 2017).

While the LTM literature has studied the influence of modularity and clustering on network diffusion, there is no consensus on the direction of that influence. For instance, Acemoglu et al. (2011) suggest that network diffusion may be more widespread on networks with a smaller degree of clustering, in contrast to the conclusions given in Centola et al. (2007) and Centola (2010). Similarly, Shakarian and Paulo (2012) find that highly modular graphs tend to have a lower diffusion rate, contradicting the conclusions of Nematzadeh et al. (2014). Our numerical experiments will resolve these conflicts for traceability technology diffusion in supply chain networks.

6.1. The Relationship Between Seed Set Size and Network Structure

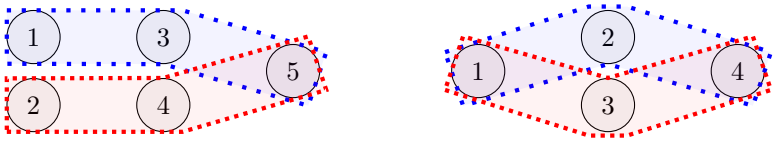
We examine which *structural characteristics* of supply chain networks influence seed set *size* and ultimately show that the higher the *Jaccard clustering* of a supply chain network (which is defined below), the larger the seed set. Identifying the drivers of the seed set size has important practical implications for managers. First, it can help them evaluate the effort required to disseminate a technology in a particular network. For example, traceability initiative leaders (such as Walmart) interested in increasing the traceability of a set of supply chain networks might, *ceteris paribus*, prefer to focus on product categories or industries whose supply chain networks display low Jaccard clustering. Second, firms interested in increasing their influence over their supply chain network should not only consider costs and risks when expanding their supply chains. They should also examine how their expansion strategy influences the network's degree of clustering.

Jaccard clustering was introduced in Latapy et al. (2008) and is frequently used in hypergraph-based models (e.g., Klamt et al. 2009). At a high level, this metric reflects whether the nodes in G tend to belong to the same sets of hyperedges or not. Formally, let E_i be the set of hyperedges that a node i is a part of, i.e., $E_i = \{j \in N_{SC} : i \in e_j\}$ and denote by $|E_i|$ its cardinality. Then, let the *neighborhood similarity* between two nodes i and k of the hypergraph be the relative overlap between E_i and E_k. Namely, $ns(i,k) = \frac{|E_i \cap E_k|}{|E_i \cup E_k|}$. This measure of set overlap, also known as the *Jaccard index* (Jaccard 1912), is commonly used in network theory, machine learning, and biology. If E_i and E_k overlap perfectly, then $ns(i,k) = 1$. Conversely, if E_i and E_k are very different—for example if i belongs to many hyperedges that do not contain k—then $ns(i,k)$ will be low. We denote the set of nodes that share at least one hyperedge with i as the *neighborhood* of i, defined as $\mathcal{N}_i = \{i' \in N_F : |E_i \cap E_k| \geq 1\}$. The *Jaccard clustering coefficient* of i is the average neighborhood similarity between i and its neighbors. Formally,

$$J_i = \frac{\sum_{k \in \mathcal{N}_i} ns(i,k)}{|\mathcal{N}_i|}. \tag{9}$$

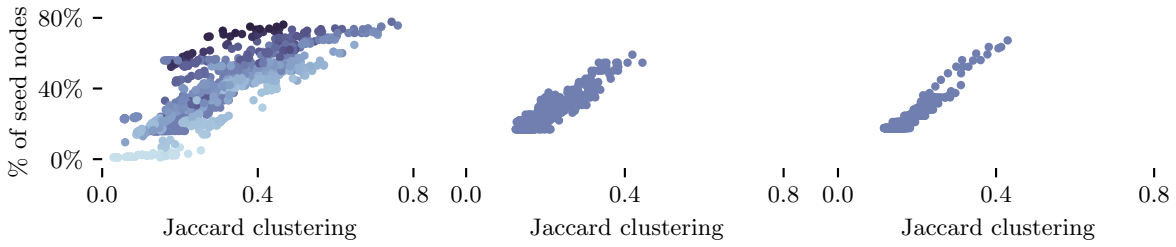
If all nodes in the neighborhood of i only belong to a single hyperedge, i.e., all firms belong to a single supply chain (to which i also belongs), then $J_i = 1$. Conversely, in a star graph with each node sharing one hyperedge with the central node i, J_i converges to zero as the number of nodes grows. To obtain a clustering measure for a hypergraph G, we compute the average clustering coefficient of all its nodes: $J = \frac{1}{n} \sum_{i \in N_F} J_i$, which we denote as the *Jaccard clustering* of the network.

We illustrate this definition through the example in Figure 7. In Figure 7a, we have $ns(1,5) = ns(2,5) = ns(3,5) = ns(4,5) = 1/2$ and $ns(1,3) = ns(2,4) = 1$. Thus, $J_5 = \frac{1}{2}$ and $J_i = \frac{1/2+1}{2} = 0.75$ for $i \neq 5$, so $J = 0.7$. In Figure 7b, we have $ns(1,2) = ns(1,3) = ns(2,4) = ns(3,4) = 1/2$ while $ns(1,4) = 1$. Thus, $J_1 = J_4 = \frac{2}{3}$ and $J_2 = J_3 = \frac{1}{2}$, giving $J = 0.5833$. The Jaccard clustering of the



(a) Higher Jaccard clustering (b) Lower Jaccard clustering

Figure 7 Illustration of Jaccard clustering.



(a) Willems (2008) (b) Random (c) Modular

Figure 8 Relationship between Jaccard clustering and the seed set size for different datasets. Colors vary with the average length of supply chains in the networks, from 2 (light) to 8 (dark).

supply chain network in Figure 7a is higher than in Figure 7b since there are fewer overlaps between the two supply chains and more firms that share the same neighborhoods.

The higher Jaccard clustering, the larger the seed set. Take Figure 7, assume that $w_i = c_i = r_{ij} = 1$ and $\theta_j = 3$ for all $i \in N_F, j \in N_{SC}$, and recall that Figure 7a has higher clustering than Figure 7b. Correspondingly, Figure 7a requires a larger percentage of firms to be seeded (60% vs. 50%). This is because the two supply chains in Figure 7a have little overlap and, due to the supply chain effect, activation from one does not entirely transfer to the other. More generally, the higher the Jaccard clustering, the more likely there will be clusters of supply chains with limited overlap, and the more difficult it will be for the diffusion process to enter regions of the network with non-traceable supply chains without adding additional active firms from within that region. We use problem (5) to further formalize this reasoning through an optimization lens in Appendix E.2.

We present several numerical experiments confirming that Jaccard clustering accurately predicts seed set size and has more predictive power than alternative clustering metrics and other structural network characteristics. Our experiments use three sets of supply chain networks, described in detail in Appendix E.1, with more than 1,600 instances across them. The first set is derived from the real-world supply chain networks in Willems (2008). The second set consists of randomly generated networks with structural characteristics similar to supply chains found in practice. The third set contains random networks used in Section 6.2. Our results are consistent across the three datasets.

We first analyze the relationship between Jaccard clustering and the percentage of total firms in the seed set, depicted in Figure 8. As one would expect from the supply chain effect, the longer

Table 1 Correlation coefficients between the percentage of seed nodes in the optimal solution and different clustering metrics. The highest for each dataset is highlighted in bold.

	Willems (2008)	Random	Modular
Jaccard clustering	**0.7743**	**0.9114**	**0.9446**
Projection clustering	0.7196	0.4819	0.4359
Projection clustering (weighted)	0.7081	0.3729	0.4482
Hourglass clustering	0.3265	0.5110	0.0950
Repetition of partners	0.2327	-0.8301	-0.8052

the length of the supply chains in the network, the more seed nodes are required. In addition, Jaccard clustering is highly correlated with the percentage of firms in the seed set, especially when considering supply chain networks with the same average supply chain length. This is clearly visible in Figures 8b and 8c, where all supply chains have length five.

The relationship between Jaccard clustering and the percentage of firms in the seed set is further confirmed by the correlation coefficient, which we compare to that of alternative clustering metrics[12] (see Appendix E.3 for definitions). In Table 1, we observe that Jaccard clustering is an excellent predictor of seed set size, its performance surpassing other clustering metrics across all datasets.

Next, we create predictive models of the seed set percentage as a function of a large set of key network measures, including various clustering metrics. Appendix E.3 states these metrics, and our analysis is in Appendix E.4. When we use random forest models to predict the seed set percentage, we achieve root mean square errors of 0.016–0.024 on unseen test data. Jaccard clustering is the first- or second-most predictive variable on each dataset. We then conduct an experiment using a penalized regression model and vary the penalty parameter. As we increase the penalty term, i.e., as we force the regression model to rely on fewer explanatory variables, the importance of Jaccard clustering increases and consistently reaches the first position among different metrics.

Our results indicate that, for supply chain networks, the effect discussed in Acemoglu et al. (2011), that higher clustering[13] leads to clusters that the diffusion process has trouble "entering" unless there are active nodes, dominates the effect observed in Centola et al. (2007) and Centola (2010), that nodes in highly-clustered networks share more neighbors and more opportunities for diffusion. This is a consequence of the supply chain effect.

6.2. Two Kinds of Seed Set Firms: Starters and Helpers

We now investigate the different roles played by seed set nodes in the diffusion process. To illustrate these different roles, consider the example in Figure 1. The seed set consists of Firms 1, 2, 3, 4,

[12] In hypergraphs, there is no consensus on the definition of clustering. Jaccard clustering is perhaps not an immediate choice as it requires two rounds of averaging to obtain a measure over graphs and does not reduce to the most common graph clustering metric when applied to hypergraphs with hyperedges of size 2 (i.e., graphs).

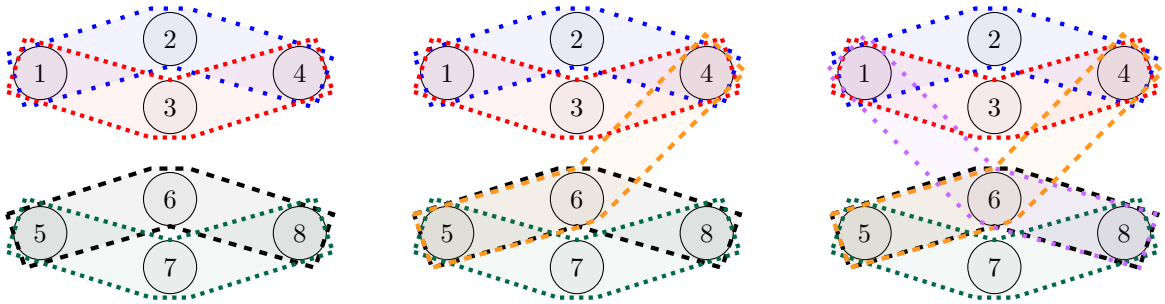
[13] We note that Acemoglu et al. (2011) uses a form of hourglass clustering.

and 7. While these firms are active at the beginning of the diffusion process, the hyperedges (i.e., supply chains) they belong to become active at different times. Hyperedge e_{black} containing Firms 2, 4, and 7 becomes active in the first period of the diffusion process (as we have $\theta_{black} = 4$) while hyperedges e_{red} and e_{blue} containing Firms 1 and 3 become active in the second and third periods, respectively. This example highlights a pattern that we observe across instances of $MIN\text{-}SCTM$. Namely, firms in the seed set can be of two types: firms that belong to hyperedges that become active at the *start* of the diffusion process, or firms that belong to hyperedges that only become active at *later* stages of the diffusion process. We refer to the former category as *starter firms* and to the latter as *helper firms* because these firms "help" diffusion after the first period. Firms 2, 4, and 7 in Figure 1 are starter firms, while 1 and 3 are helper firms.

From a managerial perspective, starter and helper firms play very different roles in the diffusion process. Starter firms immediately lead to the traceability of a set of supply chains. Making these supply chains (or, equivalently, the products they produce) traceable "jumpstarts" diffusion in the network. Conversely, helper firms are targeted without the explicit intent to make their supply chains traceable early in the diffusion process but to help keep the diffusion process "moving" along the network. As a consequence, helper firms are strategically placed, typically at the juncture between different network parts, as they "transfer" the activation process from one part to another. Interestingly, the strategic importance of "helper" firms has been picked up on in the practitioner literature (for example, World Economic Forum 2021 call them "alliance brokers"). To illustrate the different roles played by the seed set firms, consider the example in Figure 9b. Seeding Firms 1 and 2 at the top of the network will not be enough to ensure that the four firms at the bottom become active. One helper firm, such as 6, must be added to the seed set for diffusion to complete.

Managers may follow a strategy where starter firms are seeded at the beginning of the diffusion process while helper firms are seeded as the diffusion process "approaches" their supply chains. For the network in Figure 1, this would mean seeding Firms 2, 4, and 7 early on while seeding Firms 1 and 3 after e_{black} and e_{green} are active. Our IP formulation for $MIN\text{-}SCTM$ outputs the order in which hyperedges become active, which can be used to identify starter and helper firms.

We now examine how the supply chain network structure influences the seed set's proportion of starter and helper firms. Our main observation is that the proportion of starter firms in the seed set is *a V-shaped function* of modularity. Given a set of groups of nodes within a network, modularity measures how connected different groups are. Thus, a very modular graph has "tightly connected" groups of nodes that are "loosely connected" with other groups. Introducing a modularity metric for hypergraphs is the first step toward establishing this V-shaped relationship. Appendix E.3 discusses how we specialize the commonly-used definition in Newman (2006) to our setting.



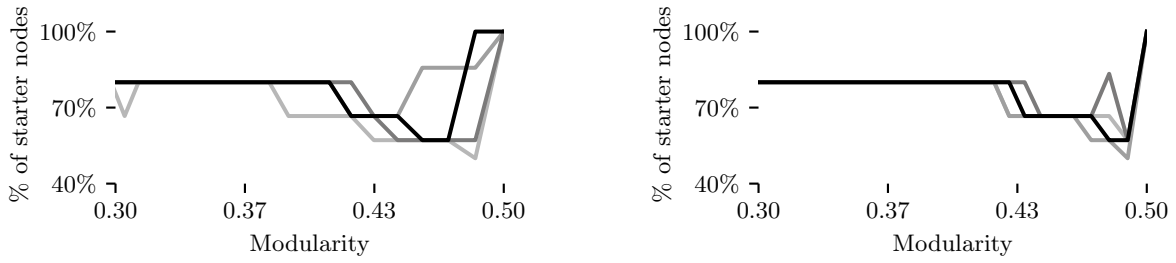
(a) High modularity (b) Intermediate modularity (c) Low modularity

Figure 9 Illustration of modularity. The parameters are $w_i = c_i = r_{ij} = 1$ and $\theta_j = 3$ for all $i \in N_F, j \in N_{SC}$.

Consider the networks in Figure 9 to develop intuition around this relationship. When a supply chain network has high modularity, there are near-disconnected groups in the network, and there is little possibility for helper firms to transfer activation across groups. In Figure 9a, modularity is the highest among the three networks (0.5), and we require four starter firms out of four seed firms (e.g., 1, 2 and 5, 6). When modularity is intermediate, groups of nodes are somewhat connected, and helper firms emerge to transfer activation across these groups. In Figure 9b, modularity is intermediate (0.431), and we require two starter firms and one helper firm out of three seed firms (e.g., 5, 6, and 4). Finally, when modularity is low, there are not many loosely-connected groups. As a result, there is less of a need for helper firms to transfer activation across groups. In Figure 9c, modularity is the lowest (0.401), and both seed set firms are starter firms (e.g., 5, 6). In consequence, the percentage of starters in the seed set is a V-shaped function of modularity.

To depict the emergence of helper firms, we randomly generate supply chain networks with varying modularity (see Appendix E.1). Simply put, we first generate a network with $\bar{m}$ hyperedges, which we duplicate to obtain two identical but disconnected groups of nodes. We then sequentially add $\bar{m}$ hyperedges across the groups, calculating the modularity, starter firms, and helper firms of the resulting network each time we add a hyperedge. As the number of cross-group hyperedges increases, the modularity of the overall supply chain network decreases. Figure 10 shows the percentage of starter firms in the seed set as a function of the network's modularity. Each line of the plot corresponds to an instance of this cross-group hyperedge addition process with a randomly generated initial network and with the number of cross-group hyperedges varying from 0 to $\bar{m}$.

On the left-hand side of the plots, there are $\bar{m}$ cross-group hyperedges, and modularity is low, so activation flows unhindered through the network, and the seed set consists primarily of starter firms. On the right-hand side of the plots, there are zero cross-group hyperedges, and modularity is high, so there are few or no helper firms since there is limited diffusion across the two network parts. In the middle of the plots, there are some cross-group hyperedges, and modularity is intermediate. Helper firms emerge to "help" transfer the diffusion process across the two network parts. Thus,



(a) $\overline{m} = 20$ (b) $\overline{m} = 30$

Figure 10 Starter firm percentage for varying numbers of connecting hyperedges on otherwise identical two-part networks, as described in Appendix E.1. Each line corresponds to different starting network.

the percentage of starter firms in the seed set decreases. Note that other network metrics heavily correlated with modularity, such as graph density, would describe the same effect.

Our insights indicate that traceability initiative leaders should be mindful of the modularity of their supply chain networks when searching for helper firms to target. If the network's modularity is intermediate, helper firms at the juncture of loosely connected network parts are likely needed to ensure broad technology diffusion. These firms help circumvent the supply chain effect and can be targeted later in the diffusion process.

6.3. A Simple Heuristic

We use the insights derived in Sections 6.1 and 6.2 to propose a simple seeding heuristic for managers. This heuristic first computes a *score* for each combination of firms that, when active, lead to a supply chain becoming active: the score includes the model parameters and Jaccard clustering. Once these scores have been computed, the heuristic seeds the set of firms with the highest score and lets diffusion propagate until it stops. It then repeats this process on the remaining non-active parts of the graph. By proceeding sequentially, the heuristic mimics the strategy mentioned in Section 6.2 of first seeding starter firms and then seeding helper firms as diffusion propagates.

The heuristic follows Algorithm 1, but instead of scores derived from an LP, it scores a node's Jaccard clustering. Specifically, in the heuristic, Line 8 of the algorithm becomes

$$s \leftarrow \frac{1}{|O|} \frac{\left(\sum_{i \in O} J_i\right)\left(\sum_{i \in O} c_i\right)}{\left(\sum_{i \in O} w_i\right)\left(\sum_{i \in O} r_{ji}\right)}.$$

Note that computing this score is simple and does not require any optimization software.

When comparing the heuristic to a lower bound, as in Section 5.4, the median gap (resp. IQR) is 19% (resp. 13% – 28%). In comparison, when paths are chosen randomly (but the choice of nodes within paths is still optimized based on w_i), the gap is 44% (resp. 25% – 76%).[14] The heuristic

[14] The median gap is 94% (resp. 74% – 137%) when nodes within paths are also chosen at random. When supply chain effects are ignored, and random selection is by node instead of by path, the gap is 135% (resp. 95% – 245%).

even outperforms the upper bound from Section 5.4 with $\kappa = 0$. However, it performs worse than the upper bound when $\kappa = 1$ and, naturally, even worse as κ grows further. Hence, the heuristic is a starting point for managers who desire a quick estimate of the cost of seeding or the companies to seed. When more computation time and an optimization solver are available, the principled heuristics providing upper and lower bounds from Section 5.4 are preferable.

7. Conclusion

Modern traceability technologies can alleviate supply chain risks, improve sustainability, reduce transaction costs, and enhance demand. However, the benefits of such technologies are only unlocked when subsets of firms in a supply chain adopt the technologies, a phenomenon we refer to as the supply chain effect. This effect has profound implications for companies leading traceability initiatives that often struggle to design a dissemination strategy for their technology (World Economic Forum 2021). Successful traceability technology dissemination strategies must address a few key questions:*(i) What is the lowest-cost seed set that ensures the whole network eventually adopts the technology? (ii) How does the size of the seed set depend on the network structure?* and *(iii) What are the different roles that seed set firms play in the diffusion process?* While the technology diffusion literature offers several network models and optimization frameworks to address these questions, they cannot be readily applied to supply chain networks due to the supply chain effect.

To address the questions above, we contribute to the technology diffusion literature by introducing the Supply Chain Traceability Model as a new framework incorporating the supply chain effect. We then define $MIN\text{-}SCTM$, the problem of finding the minimum cost seed set of nodes that guarantees diffusion throughout the network, which can be viewed as the optimization formulation of question *(i)*. We prove that $MIN\text{-}SCTM$ is not just NP-hard; it is inapproximable in polynomial time, even when all supply chains in the network only contain three firms and when the cost-benefit analysis is trivial. This result indicates that the supply chain effect and the network's intricate structure are the two drivers of $MIN\text{-}SCTM$'s complexity. Thus, any effective procedure for solving $MIN\text{-}SCTM$ must take advantage of particular network structures.

Therefore, we design an LP-based FPT algorithm for $MIN\text{-}SCTM$ with parameter treewidth of the supply chain network. The use of treewidth is practice-driven: the treewidth of publicly available supply chain networks is often an order of magnitude smaller than the network size. The approach we use to design our FPT algorithm is based on recent integer programming techniques that are new to the technology diffusion literature. Specifically, we show how to bound the treewidth of the intersection graph of an integer programming formulation of $MIN\text{-}SCTM$ by the treewidth of the network where technology diffusion occurs. The resulting FPT algorithm is an explicit LP formulation of this integer program that can be solved using existing optimization solvers. Our

procedure also outputs a hierarchy of approximations of $MIN\text{-}SCTM$ with an explicit tradeoff between the accuracy of the solution and the time taken to compute it. In short, our optimization framework answers question *(i)* by introducing algorithms that solve large instances of $MIN\text{-}SCTM$ to near-optimality within reasonable computational time and that are easy to implement.

We further employ our optimization framework to address questions *(ii)* and *(iii)* and obtain several new managerial insights. For the relationship between the network structure and the seed set size, we observe that a supply chain network with high Jaccard clustering tends to have a large optimal seed set. This result contrasts with existing and conflicting results in the network diffusion literature. While higher clustering might facilitate diffusion due to neighborhood overlaps (leading to smaller seed sets), it also makes it more difficult for the diffusion process to enter tightly-knit "clusters" of firms (leading to larger seed sets). In supply chain networks, the latter phenomenon dominates due to the supply chain effect: If firms tend to be in the same supply chains, more firms are needed in the seed set. As for the different roles played by seed set firms, we observe that networks with an intermediate degree of modularity require "helper" firms, i.e., firms that help transfer diffusion between different parts of the network. These firms are part of supply chains that only activate in later stages of the diffusion process and can thus be targeted later. Collectively, our insights can help managers shape their technology diffusion strategy, for example, by indicating which product categories (and corresponding supply chain networks) tend to require smaller seed sets and which networks are more likely to need helper firms to promote diffusion.

A promising future research direction is to extend $SCTM$ to address the limitations discussed in Section 3.2. In particular, assuming that firms engage in strategic games or adjust sourcing decisions based on their technology adoption could lead to interesting new results. Another exciting research direction is to explore the connections between integer programming techniques and network diffusion problems. For instance, examining LP formulations for seed set selection that depend on other graph parameters (beyond treewidth) might produce new approaches and insights.

More broadly, the optimization-based tool set and analysis we develop might be helpful for other problems requiring the coordination of multiple firms in supply chains, such as adopting sustainable supply chain practices. Consider, for example, circular economy initiatives. A supply chain can only really be "circular" if all companies in the supply chain adopt a consistent set of practices and technologies. We speculate that the tools and approaches introduced in this paper can assist with designing new sustainable supply chain strategies.

Acknowledgments

We would like to thank the Department Editor, the Associate Editor, and two anonymous reviewers for their constructive comments and suggestions throughout the revision process. We would also like to express

our gratitude to IBM Research and Ashish Jagmohan, for stimulating the initial research on this project, and to Manpreet Hoora, Dan Iancu, Mihalis Markakis, Gonzalo Muñoz, Karthik Ramachandran, and Beril Toktay who provided invaluable reflections at different stages of the revision process. Finally, we would like to acknowledge the research cyberinfratsructure resources and services provided by the Partnership for an Advanced Computing Envrionment (PACE) at Georgia Institute of Technology.

References

Acemoglu D, Ozdaglar A, Yildiz E (2011) Diffusion of innovations in social networks. *50th IEEE Conference on Decision and Control and European Control Conference*, 2329–2334 (IEEE).

Ackerman E, Ben-Zwi O, Wolfovitz G (2010) Combinatorial model and bounds for target set selection. *Theor. Comput. Sci.* 411(44-46):4017–4022.

Ahmed WA, MacCarthy BL (2021) Blockchain-enabled supply chain traceability in the textile and apparel supply chain: A case study of the fiber producer, Lenzing. *Sustainability* 13(19):10496.

Babich V, Hilary G (2020) OM Forum—Distributed ledgers and operations: What operations management researchers should know about blockchain technology. *Manuf. Serv. Op.* 22(2):223–240.

Behnke K, Janssen M (2020) Boundary conditions for traceability in food supply chains using blockchain technology. *Int. J. Inform. Manage.* 52:101969.

Ben-Zwi O, Hermelin D, Lokshtanov D, Newman I (2011) Treewidth governs the complexity of target set selection. *Discrete Optim.* 8(1):87–96.

Bhandalkar S, Das D (2019) Food traceability market outlook—2025. https://tinyurl.com/yyqrobql (Accessed May 1, 2023).

Bienstock D, Muñoz G (2018) LP formulations for polynomial optimization problems. *SIAM J. Optim.* 28(2):1121–1150.

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

Centeno CC, Dourado MC, Penso LD, Rautenbach D, Szwarcfiter JL (2011) Irreversible conversion of graphs. *Theor. Comput. Sci.* 412(29):3693–3700.

Centola D (2010) The spread of behavior in an online social network experiment. *Science* 329(5996):1194–1197.

Centola D, Eguíluz VM, Macy MW (2007) Cascade dynamics of complex propagation. *Physica A* 374(1):449.

Chen N (2009) On the approximability of influence in social networks. *SIAM J. Disc. Math.* 23(3):1400–1415.

Chod J, Trichakis N, Tsoukalas G, Aspegren H, Weber M (2020) On the financing benefits of supply chain transparency and blockchain adoption. *Manage. Sci.* 66(10):4378–4396.

Chopin M, Nichterlein A, Niedermeier R, Weller M (2014) Constant thresholds can make target set selection tractable. *Theor. Comput. Syst.* 55(1):61–83.

Cui Y, Hu M, Liu J (2023) Values of traceability in supply chains. *Manuf. Serv. Op.* Articles in Advance.

Dell H, Komusiewicz C, Talmon N, Weller M (2018) The PACE 2017 parameterized algorithms and computational experiments challenge. *12th Int. Symp. on Parameterized and Exact Computation.*

Downey RG, Fellows MR (2012) *Parameterized complexity* (Springer Science & Business Media).

Dutta A, Lee HL, Whang S (2007) RFID and operations management: Technology, value, and incentives. *Prod. Oper. Manage.* 16(5):646–655.

Granovetter M (1978) Threshold models of collective behavior. *Am. J. Sociol.* 83(6):1420–1443.

Graves SC, Willems SP (2000) Optimizing strategic safety stock placement in supply chains. *Manuf. Serv. Op.* 2(1):68–83.

Gurobi (2022) Gurobi optimizer manual. https://tinyurl.com/4s2c532j (Accessed May 1, 2023).

Haig S (2020) Walmart joins Hyperledger alongside 7 other companies. https://tinyurl.com/s796l3k (Accessed May 1, 2023).

Heese HS (2007) Inventory record inaccuracy, double marginalization, and RFID adoption. *Prod. Oper. Manage.* 16(5):542–553.

Hiba J (2023) Argentina's soy producers push for traceability to combat deforestation. https://tinyurl.com/29z5zrxw (Accessed May 1, 2023).

ISO (2005) ISO 9000:2005 Quality management systems—Fundamentals and vocabulary. https://tinyurl.com/y6dudvml (Accessed May 1, 2023).

Jaccard P (1912) The distribution of the flora in the alpine zone. *New Phytol.* 11(2):37–50.

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

Kempe D, Kleinberg J, Tardos É (2003) Maximizing the spread of influence through a social network. *Proceedings of the Ninth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 137–146.

Klamt S, Haus UU, Theis F (2009) Hypergraphs and cellular networks. *PLoS Comput. Biol.* 5(5):e1000385.

Kloks T (1994) *Treewidth: Computations and Approximations* (Springer).

Latapy M, Magnien C, Del Vecchio N (2008) Basic notions for the analysis of large two-mode networks. *Soc. Networks* 30(1):31–48.

Laurent M (2003) A comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre relaxations for 0–1 programming. *Math. Oper. Res.* 28(3):470–496.

Laurent M (2009) Sums of squares, moment matrices and optimization over polynomials. *Emerging Applications of Algebraic Geometry*, 157–270 (Springer).

Lee B (2022) Kinder Easter chocolate recall after salmonella outbreak leaves 150 ill, mostly young children. https://tinyurl.com/yc778xjv (Accessed May 1, 2023).

Lock H (2019) Fight the fakes: How to beat the $200bn medicine counterfeiters. https://tinyurl.com/y4ukkpeo (Accessed May 1, 2023).

Lokshtanov D, Marx D, Saurabh S (2011) Known algorithms on graphs of bounded treewidth are probably optimal. *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, 777–789.

Naidu R, Irrera A (2017) Nestle, Unilever, Tyson and others team with IBM on blockchain. https://tinyurl.com/y4pxw2w9 (Accessed May 1, 2023).

Nash KS (2018) Walmart requires lettuce, spinach suppliers to join blockchain. https://tinyurl.com/yckjtm5n (Accessed May 1, 2023).

Nematzadeh A, Ferrara E, Flammini A, Ahn YY (2014) Optimal network modularity for information diffusion. *Phys. Rev. Lett.* 113(8):088701.

Newman ME (2006) Modularity and community structure in networks. *PNAS* 103(23):8577–8582.

Nichterlein A, Niedermeier R, Uhlmann J, Weller M (2013) On tractable cases of target set selection. *Soc. Network Anal. Mining* 3(2):233–256.

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

Retail Leader (2016) Manufacturers accountable for supply chain transparency, consumer trust. https://tinyurl.com/y3uz2kr4 (Accessed May 1, 2023).

Rogers EM (2010) *Diffusion of innovations* (Simon and Schuster).

Saenz H, Hinkel J (2022) Supply chain traceability is key to sustainability—and improved performance. *Supply Chain Manag. Rev.* July/August 2022.

Sandoval E, Morris A, Ngo M (2022) A baby formula shortage leaves desperate parents searching for food. https://tinyurl.com/bdda87nr (Accessed May 1, 2023).

Shakarian P, Paulo D (2012) Large social networks can be targeted for viral marketing with small seed sets. *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 1–8.

Sternberg HS, Hofmann E, Roeck D (2021) The struggle is real: Insights from a supply chain blockchain case. *J. Bus. Logist.* 42(1):71–87.

The White House (2022) Economic report of the president. https://tinyurl.com/5yvzcmpp (Accessed May 1, 2023).

Watters LJ (1967) Reduction of integer polynomial programming problems to zero-one linear programming problems. *Oper. Res.* 15(6):1171–1174.

Whang S (2010) Timing of RFID adoption in a supply chain. *Manage. Sci.* 56(2):343–355.

Willems SP (2008) Real-world multiechelon supply chains used for inventory optimization. *Manuf. Serv. Op.* 10(1):19–23.

Wilson T (2014) Understanding the origin of products is key to ending supply chain scandals. https://tinyurl.com/yykeao9q (Accessed May 1, 2023).

World Economic Forum (2021) Digital traceability: A framework for more sustainable and resilient value chains. https://bit.ly/42kT2tU (Accessed May 1, 2023).

Wowak KD, Craighead CW, Ketchen Jr DJ (2016) Tracing bad products in supply chains: The roles of temporality, supply chain permeation, and product info. ambiguity. *J. Bus. Logist.* 37(2):132–151.

Youngdahl WE, Hunsaker BT (2018) Coda coffee and bext360 supply chain: Machine vision, AI, IoT, and blockchain. https://tinyurl.com/ycs5bo88 (Accessed May 1, 2023).

Appendix A: Results Linked to the Model Definition

This section contains the proofs of all results contained in Section 3.

Proof of Proposition 1 As $S'_0 \subseteq N_F$ and the graph is bipartite, $S'_{2t+1} \backslash S'_{2t} \subseteq N_{SC}$ and $S'_{2t+2} \backslash S'_{2t+1} \subseteq N_F$. In other words, SC-nodes activate in odd time periods and firm-nodes activate in even time periods. In particular, at time $2t+1$, the nodes added are

$$\left\{ j \in N_{SC} \backslash S_{2t} \ : \ |\{(i,j) \in E' \ : \ i \text{ is active}\}| \geq \theta_j - 1 \right\} = \bigcup_{i \in S_{2t} \cap N_F} \mathcal{B}_i(S'_{2t} \cap N_F).$$

In light of this, at time $2t+2$, the nodes added are

$$\left\{ i \in N_F \backslash S_{2t+1} \ : \ \sum_{(j,i) \in E' \ : \ j \text{ is active}} r_{ji} \geq c_i \right\} = \left\{ i \in N_F \backslash S_{2t+1} \ : \ \sum_{j \in \mathcal{B}_i(S'_{2t} \cap N_F)} r_{ji} \geq c_i \right\}.$$

From this and the fact that no firm nodes are added at time $2t+1$, it follows that

$$S'_{2t+2} \cap N_F = (S'_{2t} \cap N_F) \cup \left\{ i \in N_F \backslash S_{2t} \ : \ \sum_{j \in \mathcal{B}_i(S'_{2t} \cap N_F)} r_{ji} \geq c_i \right\}.$$

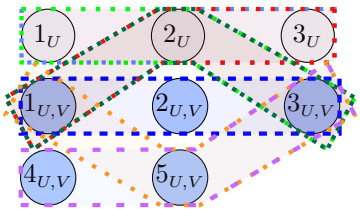
As $S_0 = S'_0$, we get the state equations of the SCTM activation process, so $S'_{2t} \cap N_F = S_t$, $\forall t$. □

Proof of Corollary 1 This is an immediate consequence of Proposition 1 and the fact that if $S_\infty = N_F$, then it must be the case that $S'_\infty = N'$, by virtue of $|\{i \in N_F \ : \ i \in e_j\}| = k_j \geq \theta_j$ for all $j \in N_{SC}$. □

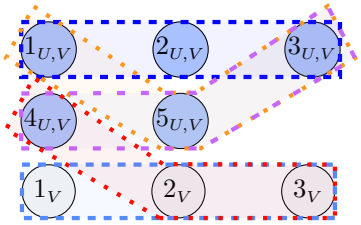
Appendix B: Computational Complexity Results

B.1. Definition of Building Blocks Used in the Proofs

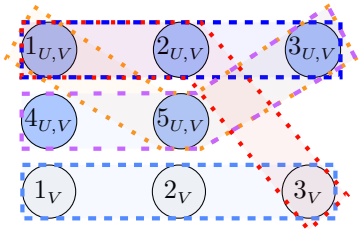
The proofs in this section require similar hypergraph structures with edges of size $k_j = 3$ and thresholds $\theta_j = 3$. These "building blocks" B_U (resp. $C_{U,V}$) consist of 3 (resp. 5) nodes and are defined in Figure 11 in gray (resp. blue). $C_{U,V} \xrightarrow{L,R} B_V$ denotes that links $C_{U,V} \xrightarrow{L} B_V$ and $C_{U,V} \xrightarrow{R} B_V$ exist.



(a) $B_U \leftrightarrow C_{U,V}$ with 7 hyperedges (B_U active $\Leftrightarrow$ $C_{U,V}$ active).



(b) $C_{U,V} \xrightarrow{L} B_V$ with 5 hyperedges ($C_{U,V}$ active $\Rightarrow$ B_V requires 1_V or 3_V to activate).



(c) $C_{U,V} \xrightarrow{R} B_V$ with 5 hyperedges ($C_{U,V}$ active $\Rightarrow$ B_V requires 1_V or 2_V to activate).

Figure 11 Construction of building blocks.

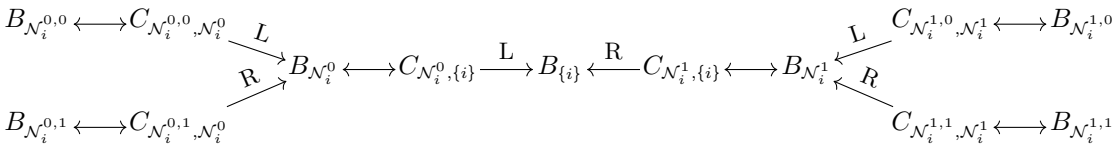


Figure 12 Linkage of building blocks for incoming edges.

B.2. NP-Hardness of *MIN-SCTM* (Theorem 1)

LEMMA 1. *DEC-SCTM is NP-hard when* $k = 3$ *and* $\theta = 3$.

Proof. Let *DEC-LTM* be the following decision problem:
INPUT: A directed graph $G' = (N', E')$ with weights w'_{ij} $\forall (i,j) \in E'$ and thresholds c'_i $\forall i \in N'$, and $h' \in \mathbb{N}$.
QUESTION: Is there a seed set of size less than or equal h' leading to full activation of G in the LTM sense?[15]
DEC-LTM is NP-hard, even if $w'_{ij} = 1$ $\forall (i,j) \in E'$ and $c'_i = |\mathcal{N}_i|$ $\forall i \in N'$ (see Kempe et al. 2003, Proof of Theorem 2.7), which we assume. Wlog, we also assume that $|\mathcal{N}_i| > 0$ for $i \in N'$ (if a node has no incoming edges, its benefit is equal to 0, as is its threshold. It will thus always activate by itself and would never be part of a minimal seed set). We construct a reduction from *DEC-LTM* to *DEC-SCTM*.

Construction of the reduction. We use the building blocks from Appendix B.1 to build G from G', indexing blocks B_U and $C_{U,V}$ using sets of nodes in N', i.e., $U, V \subseteq N'$. For any node i in N', add a block B_i to G. Recursively split its set of incoming neighbors $\mathcal{N}_i$ into two sets of equal size (if $|\mathcal{N}_i|$ is even), or into two sets of size differing by one (if $|\mathcal{N}_i|$ is odd). In other words, $\mathcal{N}_i = \mathcal{N}_i^0 \cup \mathcal{N}_i^1 = \{\mathcal{N}_i^{0,0} \cup \mathcal{N}_i^{0,1}\} \cup \{\mathcal{N}_i^{1,0} \cup \mathcal{N}_i^{1,1}\} = \ldots$, stopping when the sets in the decomposition contain one element. Add corresponding blocks $B_{\mathcal{N}_i^0}$, $B_{\mathcal{N}_i^1}$, $B_{\mathcal{N}_i^{0,0}}$, $B_{\mathcal{N}_i^{0,1}}$, $B_{\mathcal{N}_i^{1,0}}$, $B_{\mathcal{N}_i^{1,1}}, \ldots$ to G, except when $|\mathcal{N}_i^x| = 1$. Further, add blocks $C_{\mathcal{N}_i^0,\{i\}}$, $C_{\mathcal{N}_i^1,\{i\}}$, $C_{\mathcal{N}_i^{0,0},\mathcal{N}_i^0}$, $C_{\mathcal{N}_i^{0,1},\mathcal{N}_i^0}$, $C_{\mathcal{N}_i^{1,0},\mathcal{N}_i^1}$, $C_{\mathcal{N}_i^{1,1},\mathcal{N}_i^1} \ldots$ to G and link the blocks as follows: $C_{\mathcal{N}_i^0,\{i\}} \xrightarrow{L} B_{\{i\}}$, $C_{\mathcal{N}_i^1,\{i\}} \xrightarrow{R} B_{\{i\}}$, $B_{\mathcal{N}_i^0} \leftrightarrow C_{\mathcal{N}_i^0,\{i\}}$, $B_{\mathcal{N}_i^1} \leftrightarrow C_{\mathcal{N}_i^1,\{i\}}, \ldots$. This is illustrated in Figure 12. If a block already exists (e.g., if incoming neighbors overlap), use the existing one. Finally, add a block B_0 and blocks $C_{\{i\},0}$ $\forall i \in N$ with $B_{\{i\}} \leftrightarrow C_{\{i\},0}$ and $C_{\{i\},0} \xrightarrow{L,R} B_0$. As constructed, G is a hypergraph with edges of size $k = 3$. Furthermore, we assume that $r_{ji} = 1$, $c_i = 1$, $\theta_j = 3$, and $w_i = 1$ for $i = 1, \ldots, n$ and $j = 1, \ldots, m$, and we let $h = h' + 1$.

This construction is polynomial in $|N'|$ as the number of blocks is polynomial in $|N'|$. Indeed, as $|\mathcal{N}_i| \leq |N'| - 1$ $\forall i \in N$, there are at most $\lceil \log_2(|N'| - 1) \rceil$ recursive splits on $\mathcal{N}_i$, which implies that the number of sets $\mathcal{N}_i^0, \mathcal{N}_i^1, \ldots$ generated for node i is at most equal to $2^1 + 2^2 + \ldots + 2^{\lceil \log_2(|N'|-1) \rceil} \leq 8(|N'| - 1)$.

Moreover, the LTM activation process in G' can be replicated via SCTM activation in G by equating active nodes i in G' with active blocks $B_{\{i\}}$ in G and assuming that no other nodes are initially active in G. We can then show that a node i_a activates in G' if and only if block $B_{\{i_a\}}$ activates in its entirety in G. To see this, note that each block $B_{\{i\}}$ only appears in two places in G: exactly once at the root of a tree such as the one given in Figure 12 and possibly many times as a leaf of trees of this type, associated to other blocks $B_{\{j\}}$. When $B_{\{i\}}$ is at the root of a tree, it activates if all of the leaves of the tree (corresponding

[15] Recall that a node activates in the LTM sense if the sum of its incoming edge weights from active nodes exceeds its threshold; see Section 3.3.

to blocks $B_{\{j\}}, j \in \mathcal{N}_i$) activate. If it is a leaf, it will not activate through this particular tree if the root of the tree activates, nor if any other leaves activate. From this, we prove the statement: if node i_a is active in G', then all incoming neighbors $j \in \mathcal{N}_{i_a}$ of i_a are active. This implies that blocks $B_{\{j\}}, j \in \mathcal{N}_{i_a}$ are fully active, which leads from our previous discussion, to activation of $B_{\{i_a\}}$. Conversely, if $B_{\{i_a\}}$ activates with only blocks $B_{\{i\}}, i \in N'$ having initially been activated, then it must be the case that all blocks in the tree rooted at $B_{\{i_a\}}$ became active at some point, which can only happen if all of its leaves $B_{\{j\}}, j \in \mathcal{N}_{i_a}$ were fully activated. By equivalence, this means that in G', nodes $j \in \mathcal{N}_{i_a}$ are active, and so i_a would activate.

DEC-LTM with (G', h') answers YES if and only if DEC-SCTM with (G, h) answers YES. "Only if": Take a seed set S_0' in the LTM of size h' leading to full activation. For each $i \in S_0'$, add the corresponding node $5_{\{i\},0}$ to the seed set S_0 of the SCTM. Finally, take any one of the nodes in S_0', say j, and add the node $1_{\{j\},0}$ to S_0, such that $|S_0| = h + 1$. Clearly, the block $C_{\{j\},0}$ fully activates, then, block B_0, then each block $C_{\{i\},0}$ with $i \in S_0$. Next, $B_{\{i\}}$ activates for all $i \in S_0$. Each other block $B_{\{j\}}$ with $j \notin S_0$ only activates if all incoming nodes are fully activated. Hence, activation proceeds exactly as under the LTM and full activation eventually occurs under the SCTM.

"If": Assume NO for *DEC-LTM* and YES for *DEC-SCTM*. Let S_0 be an activating seed set for G of size h. If there is a node $j \in S_0 \cap \left(C_{\mathcal{N}_i^x, \mathcal{N}_i^y} \cup B_{\mathcal{N}_i^x}\right)$ for some $\mathcal{N}_i^x$ that is not a singleton, then there is another activating seed set $\tilde{S}_0$, $|\tilde{S}_0| \leq |S_0|$, without j, but with $2_{\{i\}}$ or $3_{\{i\}}$. Consider a block associated with $i \in N'$ that contains a node $j \in \tilde{S}_0$. Once it is active, the block B_0 fully activates, which in turn implies that any other such block can be activated with only the node $5_{\{i\},0}$. The first block needs to have two nodes for any activation to occur and the seed set to be minimal. Assume this block is associated with node $j \in N'$. We can replace the seed nodes in this block with the nodes $4_{\{j\},0}$ and $5_{\{j\},0}$ wlog. We then know from before that a set $S_0' = \{i | 5_{\{i\},0} \in \tilde{S}_0\}$ (of size $\leq h - 1 = h'$) leads to full activation in the LTM, giving a contradiction. □

Proof of Theorem 1. Lemma 1 shows that *DEC-SCTM* is hard when $(k = 3, \theta = 3)$. We first show that the cases $\theta = 1$ and $\theta = 2$ are in P. If $\theta = 1$, then it is easy to see that all firms will adopt at the first time step so the seed set can be taken to be empty. If $\theta = 2$, then one need only seed the node of lowest cost. This will lead to all supply chains it belongs to activating, which will in turn lead to all supply chains connected to those supply chains to activate. Connectivity of G thus ensures full activation.

We now show that if *DEC-SCTM* is hard for fixed k and $\theta = k$ and $\theta \leq k$, then *DEC-SCTM* remains hard for $k + 1$ and $\theta + 1 = k + 1$. To do this, let G be the graph over which *DEC-SCTM* is hard with fixed k and fixed h. We let $\tilde{G}$ be G, except that we add the same additional node ι, to each hyperedge $e_j \in G$. Note that the size of each hyperedge is now $k + 1$. We further let the cost to seed of the additional node be $w_\iota = 1$ for all $i \in N$, let the edge threshold be $k + 1$, and take $\tilde{h} = h + 1$. One can check that $S_0 \cup \{\iota\}$ is a seed set of size less than or equal to $\tilde{h}$ leading to full activation of $\tilde{G}$ if and only if S_0 is a seed set of size less than equal to h leading to full activation of G, which implies the result. Similarly, we can show that if *DEC-SCTM* is hard for fixed k and θ, then *DEC-SCTM* remains hard for $k + 1$ and θ. This involves building the same graph as $\tilde{G}$ above, except that we do not add $\{\iota\}$ to the seed set and we let $\tilde{h} = h$. Thus, for any (θ, k) such that $\theta \leq k$, one can proceed from the case $(k = 3, \theta = 3)$ to (k, θ) by applying successively the operations $(k, \theta = k) \mapsto (k + 1, \theta + 1 = k + 1)$ and $(k, \theta) \mapsto (k + 1, \theta)$ as described above. □

B.3. Hardness of Approximation of $MIN\text{-}SCTM$ (Proposition 2)

Proof of Proposition 2. We proceed as before using a reduction from the LTM on a graph G' where all nodes have a threshold $c_i' \le 2$ and all edges have weight $w_{ij}' = 1$. Below, we will show that for any G' with $|N'| = n'$ nodes and LTM activation, we can create, in polynomial time, a hypergraph G with $r_{ji} = 1, c_i = 1$, $k = 3, \theta = 3$, activating via the $SCTM$, with the following properties: (i) the number of nodes is $n \le (n')^\beta$ for a constant $1 \le \beta < \infty$; (ii) if OPT_{LTM} is the size of the minimum seed set for G', and S_0^* is a minimum size seed set of G with $OPT = |S_0^*|$, then $OPT = OPT_{LTM} + 1$.

Given the construction, assume that for all $\alpha > 1$ there is an algorithm approximating $MIN\text{-}SCTM$ with result OPT' such that $OPT' = OPT \cdot O(\alpha^{\log^{1-\xi} n})$ for some $\xi > 0$. Clearly, OPT' is an upper-bound on OPT_{LTM}. Moreover, $OPT' = (OPT_{LTM} + 1) \cdot O(\alpha^{\log^{1-\xi} n}) < OPT_{LTM} \cdot O(\alpha^{\log^{1-\xi} n}) < OPT_{LTM} \cdot O(\alpha^{\beta^{1-\xi} \log^{1-\xi} n'}) = OPT_{LTM} \cdot O\left(\left(\alpha^{\beta^{1-\xi}}\right)^{\log^{1-\xi} n'}\right)$. As $\alpha > 1$, let $\alpha = 2^{\frac{1}{\beta^{1-\xi}}}$. We then have a direct contradiction to the result that there is no polynomial-time approximation algorithm with output OPT' and $OPT' < OPT_{LTM} \cdot O(2^{\log^{1-\xi} n'})$ for any $\xi > 0$ (cf. Corollary 4.1 in Chen 2009). The case where $k \ge \theta \ge 3$ can be obtained in a similar fashion using the extension operations given in the proof of Theorem 1.

A note on the complexity class. Assume instead that $OPT' = OPT \cdot O(2^{\log^{1-\xi} n})$. Then, $OPT' < OPT_{LTM} \cdot O\left(2^{\beta^{1-\xi} \log^{1-\xi} n'}\right)$. To establish a contradiction, we require that there is a constant $M > 0$ such that $M \cdot 2^{\beta^{1-\xi} \log^{1-\xi} n'} \le 2^{\log^{1-\xi} n'}$. This is equivalent to $M \le 2^{\log^{1-\xi} n' \left(1-\beta^{1-\xi}\right)}$. However, unless $\beta = 1$, $1 - \beta^{1-\xi} < 0$, and the right-hand side tends to zero as n goes to infinity. But because $M > 0$, this does not hold. The same issue arises in the sequence of proofs leading up to Corollary 4.1 in Chen (2009), where a reduction is created to a graph with a higher number of nodes at each step (that is, $\beta > 1$). This is of no consequence for the complexity class that we apply here, however, as we can replace 2 by some $\alpha \in (1, 2)$ in each step.

Construction of the reduction. Let $G' = (N', E')$, $|N'| = n'$ be a directed graph, activated via the LTM. Assume that all nodes have a threshold $c_i' \in \{1, 2\}$ and all edges have weight $w_{ij}' = 1$. We proceed with a similar construction of a hypergraph $G = (N, E)$ with $k = 3$, as in the proof of Theorem 1. For each $i \in N'$, define block $B_{\{i\}}$. Either (i) node i has a threshold of 1, then for each (directed) edge $(j, i) \in E'$, construct a block $C_{\{j\},\{i\}}$, as well as the linkages $B_{\{j\}} \leftrightarrow C_{\{j\},\{i\}} \xrightarrow{L,R} B_{\{i\}}$. Alternatively, (ii) it has a threshold of 2. Say the set of incoming neighbors is $\mathcal{N}_i$. Then, construct block $B_{\{j,j'\}}$ for any pair $(j, j') \in \mathcal{N}_i^2$. There are $\binom{|\mathcal{N}_i|}{2} \le \binom{n-1}{2} \le n^2$ such pairs. For each pair, construct blocks $C_{\{j\},\{j,j'\}}$, $C_{\{j'\},\{j,j'\}}$, and $C_{\{j,j'\},\{i\}}$, as well as the following linkages: $B_{\{j\}} \leftrightarrow C_{\{j\},\{j,j'\}} \xrightarrow{L} B_{\{j,j'\}}$, $B_{\{j'\}} \leftrightarrow C_{\{j\},\{j,j'\}} \xrightarrow{R} B_{\{j,j'\}}$, and $B_{\{j,j'\}} \leftrightarrow C_{\{j,j'\},\{i\}} \xrightarrow{L,R} B_{\{i\}}$. As before, we add a block B_0 and the corresponding blocks $C_{\{i\},0}$ for all $i \in N$ with connections $B_{\{i\}} \leftrightarrow C_{\{i\},0} \xrightarrow{L,R} B_0$. The construction is, again, polynomial in n'. Moreover, following the same steps as before, one can show that the solution to $MIN\text{-}LTM$ on graph G' is OPT_{LTM} if and only if the solution to $MIN\text{-}SCTM$ on graph G is $OPT = OPT_{LTM} + 1$. Note that $n \le (n')^\beta$ for a constant $1 \le \beta < \infty$. □

Appendix C: Results Regarding the Linear Programming-Based FPT Algorithm

C.1. Proof of Propositions 3, 5, and 6

Proof of Proposition 3 This follows from Courcelle (2015, Lemma 14), the definition of G', noting that $(i, j) \in E'$ if and only if $(j, i) \in E'$, and considering the undirected version of G'. □

The proof of Proposition 5 follows that given in Ackerman et al. (2010) with some small modifications linked to the specificities of G' (e.g., its bipartiteness, the fact that its edges are weighted, etc.).

Proof of Proposition 5 Due to Corollary 1, we show that any feasible solution for (2) is feasible for (4) and vice-versa, and that the objective functions are equivalent.

Let $S'_0 \cup N_F$ be a feasible solution to (2). We construct the following solution to (4): we let $s_i = 1$ if $i \in S'_0$ and $s_i = 0$ if not, and we let $\ell_{ij} = 1$ (resp. $\ell_{ji} = 1$) if $i \in N_F$ precedes $j \in N_{SC}$ (resp. $j \in N_{SC}$ precedes $i \in N_F$) in terms of activation. We show that this solution is feasible, constraint-by-constraint. Constraint (4a) trivially holds if $i \in S_0$. If $i \notin S_0$, then $s_i = 0$. As S_0 leads to full activation of G', there must be some t such that $\sum_{\{(j,i)\in E_{SC,F} \text{ s.t. } j\in S_t\}} r_{i,j} \geq c_i$. By construction, the relevant variables ℓ_{ji} must be set to 1. Hence, $\sum_{\{(j,i)\in E_{SC,F}\}} r_{i,j}\ell_{ji} \geq c_i$ and (4a) holds. A similar argument can be used to show that (4b) holds. Constraint (4c) also holds as i cannot simultaneously activate j and j activate i. Likewise, if (4d) were violated, then $\ell_{i_1j_1} + \ell_{j_1i_2} + \ell_{i_2j_2} + \ell_{j_2i_1} = 4$, which would imply that i_1 activates j_1, which activates i_2, which activates j_2, which activates i_1, once again. This is not possible as i_1 is already activated. Thus (4d) holds. Finally, the objective function of (4) is equal to the weight of the seed set, which is identical to that of (2).

We now consider a feasible solution to (4). To set $\{\ell_{ij}, \ell_{ji}\}$, we can associate a directed acyclic graph on N' with an edge from $i \in N_F$ (resp. $j \in N_{SC}$) to $j \in N_{SC}$ (resp. $i \in N_F$) if $\ell_{ij} = 1$ (resp. $\ell_{ji} = 1$). The graph is directed (constraint (4c)) and acyclic (constraint (4d)), as G' is bipartite. Thus, we are able to define a topological ordering on the nodes in N'. We let $S'_0 = \{i \in N_F \mid s_i = 1\}$. Consider $t \geq 1$. We define:

$$S'_{2t} = S'_{2t-1} \cup \{i \in N_F \backslash S'_{2t-1} \mid \ell_{ji} = 1 \Rightarrow j \in S'_{2t-1}, \forall j \in \{j \mid (j,i) \in N_{SC,F}\}\}$$

and

$$S'_{2t+1} = S'_{2t} \cup \{j \in N_{SC} \backslash S'_{2t} \mid \ell_{ij} = 1 \Rightarrow i \in S'_{2t}, \forall i \in \{i \mid (i,j) \in N_{F,SC}\}\}.$$

As $\{\ell_{ij}, \ell_{ji}\}$ define a topological ordering on N', we have $S'_\infty = N'$. Furthermore, for $i \in S'_{2t} \backslash S'_{2t-1}$, we have:

$$c_i \leq \sum_{\{j|(j,i)\in E_{SC,F}\}} r_{ji}\ell_{ji} = \sum_{\{j|(j,i)\in E_{SC,F}\cap \ell_{ji=1}\}} r_{ji}\ell_{ji} = \sum_{\{j|(j,i)\in E_{SC,F}\cap S'_{2t-1}\}} r_{ji} = \sum_{\{j \mid (j,i)\in E_{SC,F}\}} r_{ji}x_{j(2t)},$$

where the first inequality is due to (4a) and the second equality is due to the definition of S'_{2t}. A similar set of inequalities can be derived for $j \in S'_{2t+1} \backslash S'_{2t}$, and thus the activation process defined in this way is exactly that given in (2). As the objectives of (2) and (4) are equivalent, this concludes the proof. □

Proof of Proposition 6 Constraint (4d) enforces that $\ell_{i_1j_1}$ and $\ell_{i_2j_2}$ are connected in the intersection graph of (4) for any $i_1, i_2 \in N_F$ and $j_1, j_2 \in N_{SC}$. There are $n \times m$ pairs $(i,j) \in N_F \times N_{SC}$ and all of the corresponding nm variables ℓ_{ij} are connected: this creates a clique in the intersection graph of (4) of size nm. As the treewidth of a clique is $nm - 1$ and the treewidth of any subgraph of the intersection graph lower-bounds the treewidth of the intersection graph, the result follows. □

C.2. Formulation of the Final BiLP and Proofs of Propositions 7–9 and Theorem 2

Proof of Proposition 7. The two objective functions are identical, so it suffices to show that (4) and (5) are equivalent. First, let $\left(\{\tilde{\ell}_{ij}, \tilde{\ell}_{ji}\}_{i\in N_F, j\in N_{SC}}, \{\tilde{s}_i\}_{i\in N_F}\right)$ be a feasible solution to (4). Define a solution to (5): set $s_i = \tilde{s}_i$ for $i = 1, \ldots, N_F$ and $\ell_{ij} = \tilde{\ell}_{ij}$ (resp. $\ell_{ji} = \tilde{\ell}_{ji}$) for any ℓ_{ij} (resp. ℓ_{ji}) present in (5) with $i \in N_F$ and

$j \in N_{SC}$. Recall that the variables $\{\tilde{\ell}_{ij}\}_{ij}$ from (4) define a DAG and that this DAG is connected. For any pair $(i,j) \in X'_z$ for some z, with $i,j \in N_F$ or $i,j \in N_{SC}$, we set $\ell_{ij} = 1$ (resp. $\ell_{ij} = 0$) if there exists a directed path from i to j (resp. from j to i) in the DAG. If neither exists, we assign either 0 or 1 to ℓ_{ij} only ensuring that $\ell_{ij} + \ell_{ji} = 1$. It is easy to see that a solution to (5) thus defined satisfies constraints (5a) through (5c). Now consider constraint (5d) and suppose that $i \in N_F, j \in N_F$, and $k \in N_{SC}$ wlog (other cases can be treated in the same way). If $\ell_{jk} = \ell_{ki} = 1$, then this means that $\tilde{\ell}_{jk} = \tilde{\ell_{ki}} = 1$ and the DAG defined by $(\tilde{\ell}_{ij})_{ij}$ contains a directed path from j to i. This implies that $\ell_{ji} = 1$ and $\ell_{ij} = 0$ and, thus, constraint (5d) holds. If either ℓ_{jk} or $\ell_{ki} \neq 1$, then the constraint trivially holds. This shows the implication.

Now, suppose we have a feasible solution to (5). For each bag X'_z, we can draw a directed graph with nodes in X'_z and an edge from node i to node j in X'_z if $\ell_{ij} = 1$. As the constraint (5d) precludes cycles from forming, these graphs are all DAGs. Thus, each bag gives rise to a partial ordering of the nodes it contains. One can then consider a partial ordering on all nodes (as each node appears in at least one bag) obtained by taking the union of the partial orders across bags. Indeed, if two nodes are ordered in a specific way in one bag, the presence of the $\{\ell_{ij}\}$ will enforce the same ordering in any other bag where they both appear. Then, from the order-extension principle, one can define a total order on all nodes in $N_F \cup N_{SC}$ which is consistent with this partial order. That is, one can extend $\{\ell_{i,j}\}_{i,j \in N' \cup X'_z, z \in T'}$ to a sequence $\{\ell_{ij}\}_{i,j \in N'}$ in such a way that this latter set represents a DAG. By taking as our solution to (4) the appropriate subset $\{\ell_{ij}, \ell_{ji}\}_{i \in N_F, j \in N_{SC}}$ of the aforementioned sequence combined with the $\{s_i\}$ given by (5), we obtain a feasible solution to (4) as the no-cycle constraint of (4d) is guaranteed to hold by acyclicity of the DAG. □

Formulation of the Final BiLP.

$$
\begin{aligned}
\min \quad & \sum_{i \in N_F} w_i s_i \\
\text{s.t.} \quad & \forall i \in N_F: \ \sum_{j \in J_z^i} r_{ji} \ell_{ji} \geq u_{iz}, \ \forall z \in T_i^G, \quad 0 \geq u_{iz}, \ \forall z \in T_i^{\tilde{G}} && \text{(10a)} \\
& u_{iz} + \tilde{u}_{ic_1(z)} + \tilde{u}_{ic_2(z)} \geq \tilde{u}_{iz}, \ \forall z \neq z_0 \in T'_i, && \text{(10b)} \\
& \tilde{u}_{iz_0^i} + \tilde{u}_{ic_1(z_0^i)} + \tilde{u}_{ic_2(z_0^i)} \geq c_i(1 - s_i), && \text{(10c)} \\
& u_{iz} = \sum_{\tau=0}^{n_u} 2^\tau u_{iz}^\tau \text{ and } \tilde{u}_{iz} = \sum_{\tau=0}^{n_u} 2^\tau \tilde{u}_{iz}^\tau, \ \forall z \in T'_i, && \text{(10d)} \\
& \forall j \in N_{SC}: \ \sum_{i \in I_z^j} \ell_{ij} \geq v_{jz}, \ \forall z \in T_j^G, \quad 0 \geq v_{jz}, \ \forall z \in T_j^{\tilde{G}}, && \text{(10e)} \\
& v_{jz} + \tilde{v}_{jc_1(z)} + \tilde{v}_{jc_2(z)} \geq \tilde{v}_{jz}, \ \forall z \neq z_0 \in T'_j, && \text{(10f)} \\
& \tilde{v}_{jz_0^j} + \tilde{v}_{jc_1(z_0^j)} + \tilde{v}_{jc_2(z_0^j)} \geq \theta_j - 1, && \text{(10g)} \\
& v_{jz} = \sum_{\tau=0}^{n_v} 2^\tau v_{jz}^\tau \text{ and } \tilde{v}_{jz} = \sum_{\tau=0}^{n_v} 2^\tau \tilde{v}_{jz}^\tau, \ \forall z \in T'_j, && \text{(10h)} \\
& \forall z \in T': \ \ell_{ij} + \ell_{ji} = 1, \ \forall i,j \in X'_z \cap N', && \text{(10i)} \\
& \ell_{ij} + \ell_{jk} + \ell_{ki} \leq 2, \ \forall i,j,k \in X'_z \cap N', && \text{(10j)} \\
& s_i \in \{0,1\}, \ \forall i \in N_F, \quad \ell_{ij} \in \{0,1\}, \ \forall i,j \in N' \cap X'_z, \forall z \in T',
\end{aligned}
$$

$$u^\tau_{iz} \text{ (resp. } \tilde{u}^\tau_{iz}) \in \{0,1\},\ \forall \tau, \forall i \in N_F, \forall z \in T' \text{ (resp. } \forall z \in T' \setminus \bigcup_{i \in N_F} z^0_i),$$
$$v^\tau_{jz} \text{ (resp. } \tilde{v}^\tau_{jz}) \in \{0,1\},\ \forall \tau, \forall j \in N_{SC}, \forall z \in T' \text{ (resp. } \forall z \in T' \setminus \bigcup_{j \in N_{SC}} z^0_j).$$

Proof of Proposition 8. First, note that (10) is equivalent to the following integer linear program (ILP):

$$\begin{aligned}
\min \quad & \sum_{i \in N_F} w_i s_i \\
\text{s.t.} \quad & \forall i \in N_F : \sum_{j \in J^i_z} r_{ji}\ell_{ji} \geq u_{iz},\ \forall z \in T^G_i, \quad 0 \geq u_{iz},\ \forall z \in T^{\tilde{G}}_i && \text{(11a)} \\
& u_{iz} + \tilde{u}_{ic_1(z)} + \tilde{u}_{ic_2(z)} \geq \tilde{u}_{iz},\ \forall z \neq z_0 \in T'_i, \quad \tilde{u}_{iz^i_0} + \tilde{u}_{ic_1(z^i_0)} + \tilde{u}_{ic_2(z^i_0)} \geq c_i(1-s_i), && \text{(11b)} \\
& \forall j \in N_{SC} : \sum_{i \in I^j_z} \ell_{ij} \geq v_{jz},\ \forall z \in T^G_j, \quad 0 \geq v_{jz},\ \forall z \in T^{\tilde{G}}_j, && \text{(11c)} \\
& v_{jz} + \tilde{v}_{jc_1(z)} + \tilde{v}_{jc_2(z)} \geq \tilde{v}_{jz},\ \forall z \neq z_0 \in T'_j, \quad \tilde{v}_{jz^j_0} + \tilde{v}_{jc_1(z^j_0)} + \tilde{v}_{jc_2(z^j_0)} \geq \theta_j - 1, && \text{(11d)} \\
& \forall z \in T' : \ \ell_{ij} + \ell_{ji} = 1,\ \forall i,j \in X'_z \cap N', && \text{(11e)} \\
& \ell_{ij} + \ell_{jk} + \ell_{ki} \leq 2,\ \forall i,j,k \in X'_z \cap N', && \text{(11f)} \\
& s_i \in \{0,1\},\ \forall i \in N_F, \quad \ell_{ij} \in \{0,1\},\ \forall i,j \in N' \cap X'_z, \forall z \in T', \\
& u_{iz} \text{ (resp. } \tilde{u}_{iz}) \in \{0,\dots,c_{\max}\}\ \forall i \in N_F,\ \forall z \in T' \text{ (resp. } \forall z \in T' \setminus \cup_{i \in N_F} z_{i_0}) \\
& v_{jz} \text{ (resp. } \tilde{v}_{jz}) \in \{0,\dots,\theta_{\max}\}\ \forall j \in N_{SC},\ \forall z \in T' \text{ (resp. } \forall z \in T' \setminus \cup_{j \in N_{SC}} z_{j_0}),
\end{aligned}$$

where $c_{\max} = \max\{c_1,\dots,c_n\}$ and $\theta_{\max} = \max\{\theta_1,\dots,\theta_m\} - 1$. Simply note that $\sum_{\tau=0}^{n_u} u^\tau_{iz}$ (resp. $\tilde{u}^\tau_{iz}$) is the binary formulation of u_{iz} (resp. $\tilde{u}_{iz}$), and $\sum_{\tau=0}^{n_v} v^\tau_{jz}$ (resp. $\tilde{v}^\tau_{jz}$) is the binary formulation of v_{jz} (resp. $\tilde{v}_{jz}$).

We now show that the ILP (11) is equivalent to (5). By virtue of Proposition 7 and the above, the result follows. Assume that constraints (11a) and (11b) hold. By iteratively using constraint (11b) as we go up the tree T'_i from the leaves to the roots, we obtain that $\sum_{z \in T'_i} u_{iz} \geq c_i(1-s_i)$. Now, using (11a), it follows that $\sum_{z \in T^G_i} \sum_{j \in J^i_z} r_{ji}\ell_{ji} \geq c_i(1-s_i)$. As the sets $\{J^i_z\}_{z \in T^G_i}$ partition $J_i = \{j \mid (j,i) \in E_{SC,F}\}$, we obtain (5a). Conversely, if (5a) holds, then we can simply set $u_{iz} = \min\{c_{\max}, \sum_{j \in J^i_z} r_{ji}\ell_{ji}\}$ if $z \in T^G_i$, or $u_{iz} = 0$ if $z \in T^{\tilde{G}}_i$, and $\tilde{u}_{iz} = \min\{c_{\max}, u_{iz} + \tilde{u}_{ic_1(z)} + \tilde{u}_{ic_2(z)}\}$. Then, (11a) and (11b) hold.

Likewise, assume that constraints (11c) and (11d) hold. By iteratively using constraint (11d) as we go up the tree T'_j from the leaves to the roots, we obtain that $\sum_{z \in T'_j} v_{jz} \geq \theta_j - 1$. Now, using (11c), it follows that $\sum_{z \in T^G_j} \sum_{i \in I^j_z} \ell_{ij} \geq \theta_j - 1$. As the sets $\{I^j_z\}_{z \in T^G_j}$ partition $I_j = \{i \mid (j,i) \in E_{SC,F}\}$, we obtain (5b). Conversely, if (5b) holds, then we can simply set $v_{jz} = \min\{\theta_{\max}, \sum_{i \in I^j_z} \ell_{ij}\}$ if $z \in T^G_j$, or $v_{jz} = 0$ if $z \in T^{\tilde{G}}_j$, and $\tilde{v}_{jz} = \min\{\theta_{\max}, v_{jz} + \tilde{v}_{jc_1(z)} + \tilde{v}_{jc_2(z)}\}$. Then, (11c) and (11d) hold. As the remaining constraints and the objectives are the same, (11) and (5) are equivalent. □

Proof of Proposition 9. Recall that $\mathcal{T}' = (T', \{X'_z\}_{z \in T'})$ is the tree decomposition of G', where we assume that each node z has no more than two children. We build a tree decomposition $\mathcal{S} = (S, \{W_z\}_{z \in S})$ from $\mathcal{T}'$, where $S = T'$ and each bag W_z contains variables from (10) instead of nodes. We then show that such a tree decomposition is in fact a valid tree decomposition of the intersection graph of (10) and that its width is upper-bounded by $O(\omega'^2 + \omega' \log_2(\vartheta_{\max}))$. This proves the result. We now specify the bags $\{W_z\}_{z \in S}$:

1. Let $z \in S$ and let X'_z be the corresponding bag of nodes in T'. Then, $W_z = \bigcup_{i,j \in N' \cap X'_z} \{\ell_{ji}\}$.

2. For each $i \in N_F$, consider T_i' in T' with an arbitrary root node. For each node z in T_i':
   (a) If z is the root node, add variables $\{u_{iz}^\tau\}_{\tau=0,\dots,n_u}$ and s_i, as well as the variables also present in its children's nodes, $\{\tilde{u}_{ic_1(z)}^\tau\}_{\tau=0,\dots,n_u}$ and $\{\tilde{u}_{ic_2(z)}^\tau\}_{\tau=0,\dots,n_u}$, to W_z.
   (b) If z is not a leaf and not the root, add variables $\{u_{iz}^\tau\}_{\tau=0,\dots,n_u}$ and $\{\tilde{u}_{iz}^\tau\}_{\tau=0,\dots,n_u}$, as well as the variables also present in its children's nodes, $\{\tilde{u}_{ic_1(z)}^\tau\}_{\tau=0,\dots,n_u}$ and $\{\tilde{u}_{ic_2(z)}^\tau\}_{\tau=0,\dots,n_u}$, to W_z.
   (c) If z is a leaf, add variables $\{u_{iz}^\tau\}_{\tau=0,\dots,n_u}$ and $\{\tilde{u}_{iz}^\tau\}_{\tau=0,\dots,n_u}$ to W_z.
3. Likewise, for each $j \in N_{SC}$, consider T_j'. For each node z in T_j':
   (a) If z is the root node, add variables $\{v_{jz}^\tau\}_{\tau=0,\dots,n_v}$, as well as the variables also present in its children's nodes, $\{\tilde{v}_{jc_1(z)}^\tau\}_{\tau=0,\dots,n_v}$ and $\{\tilde{v}_{jc_2(z)}^\tau\}_{\tau=0,\dots,n_v}$, to W_z.
   (b) If z is not a leaf and not the root, add variables $\{v_{jz}\}_{\tau=0,\dots,n_v}$ and $\{\tilde{v}_{jz}\}_{\tau=0,\dots,n_v}$, as well as the variables also present in its children's nodes, $\{\tilde{v}_{jc_1(z)}^\tau\}_{\tau=0,\dots,n_v}$ and $\{\tilde{v}_{jc_2(z)}^\tau\}_{\tau=0,\dots,n_v}$, to W_z.
   (c) If z is a leaf, add variables $\{v_{jz}^\tau\}_{\tau=0,\dots,n_v}$ and $\{\tilde{v}_{jz}^\tau\}_{\tau=0,\dots,n_v}$ to W_z.

We now show that $\mathcal{S}$ as constructed is a valid tree decomposition for the intersection graph of (10). To do this, we need to prove three points. First, all variables involved in the optimization problem appear in at least one of the bags $\{W_z\}_{z\in S}$. This is straightforward to check. Second, if a variable appears in two distinct bags, then it appears in all bags in-between. We proceed by groups of variables. The variables $s_i, \{u_{iz}^\tau\}_\tau, \{v_{jz}^\tau\}_\tau$ each only appear in one bag, thus this trivially holds for them. The variables $\{\tilde{u}_{iz}^\tau\}_\tau$ and $\{\tilde{v}_{jz}^\tau\}_\tau$ appear in two bags, however, these are parent/children combinations, so the property holds. This leaves variables ℓ_{ji}. Suppose that ℓ_{ji} appears in bag W_{z_1} and W_{z_2} and that there is at least one bag between W_{z_1} and W_{z_2}. The assumption implies that $j, i \in X_{z_1}'$ and $j, i \in X_{z_2}'$. As $\mathcal{T}$ is a tree decomposition, it follows that j and i appear in all bags between X_{z_1}' and X_{z_2}', thus ℓ_{ji} also appears in all bags between W_{z_1} and W_{z_2}. Third, if a group of variables appears in a constraint, then this group appears in at least one bag of $\mathcal{S}$, because the group forms a clique in the intersection graph. We proceed constraint by constraint. For constraint (10a), by construction, $j \in J_z^i \subseteq X_z'$ for $z \in T_i^G$, thus $i, j \in X_z'$, and $\ell_{ji} \in W_z$. Furthermore, from Step 2b in the construction of $\mathcal{S}$, $\{u_{iz}^\tau\}_\tau \in W_z$. For constraints (10b) and (10c), this is straightforward from steps 2a and 2b. A similar reasoning applies to constraints (10e), (10f), and (10g). For constraints (10j), this follows from step 1. Thus, $\mathcal{S}$ is a valid tree decomposition for the intersection graph of (10).

We now upper-bound the treewidth by looking at the size of each one of the bags $\{W_z\}_{z\in S}$. Consider the algorithm to build $\mathcal{S}$ and recall that ω' is the treewidth of G', and thus the maximum size of X_z' for all $z \in T$. Step 1 only occurs once and at the end of it, W_z contains at most ω'^2 nodes. Then, for each node $z \in T'$, steps 2-3 happen at most a combined ω' times as X_z' contains at most ω' nodes. Thus, z appears in at most ω' trees T_i' or T_j'. During Step 2 (resp. Step 3), a maximum of 4 sets of variables are added to the node, with each set having size at most $\max\{n_u, n_v\}$, where n_u, n_v are as defined in (6). Thus, at the end of the construction of $\mathcal{S}$, W_z contains at most $\omega'^2 + 4\max\{n_u, n_v\} \cdot \omega' = O(\omega'^2 + \omega' \log_2(\vartheta_{\max}))$ nodes. □

Proof of Theorem 2. From Proposition 8, we have that (10) solves *MIN-SCTM*. From Proposition 9, the intersection graph of (10) has treewidth at most $O(\omega'^2 + \omega' \log_2(\vartheta_{\max}))$. We now count the variables appearing in (10). Recall that T' has at most $4(n+m)$ bags. We have that (10) has n variables s_i, at most $4(n+m) \cdot \omega'^3$ variables ℓ_{ij}, at most $3 \cdot n_u \cdot 4(n+m)$ variables $\{u_{iz}^\tau\}$ and $\{\tilde{u}_{iz}^\tau\}$, and at most $3 \cdot n_v \cdot 4(n+m)$ variables $\{v_{jz}^\tau\}$ and $\{\tilde{v}_{jz}^\tau\}$. From Propositions 3 and 4, we obtain the result. □

C.3. Proof of Proposition 10

Proof of Proposition 10. When $\kappa = 0$, (8) reads:

$$\begin{aligned} \min_{Y_{\mathbb{S}}} \quad & \sum_i w_i Y_{\{s_i\}} \\ \text{s.t.} \quad & Y_{\varnothing} = 1 \\ \forall z \in S: \; & Y_{\varnothing} - Y_{\{x_i^z\}} \geq 0, Y_{\{x_i^z\}} \geq 0, \;\; \forall i = 1, \ldots, \omega_z \\ & g^l_{\varnothing} Y_{\varnothing} + \sum_{i=1}^{\omega_z} g^l_{\{x_i^z\}} \cdot Y_{\{x_i^z\}} \geq 0, \;\; \forall l = 1, \ldots, l_z, \end{aligned}$$

By definition of an intersection graph, all constraints of (10) are associated to at least one node $z \in S$. The result follows. □

Appendix D: A Dynamic Programming-Based FPT Algorithm

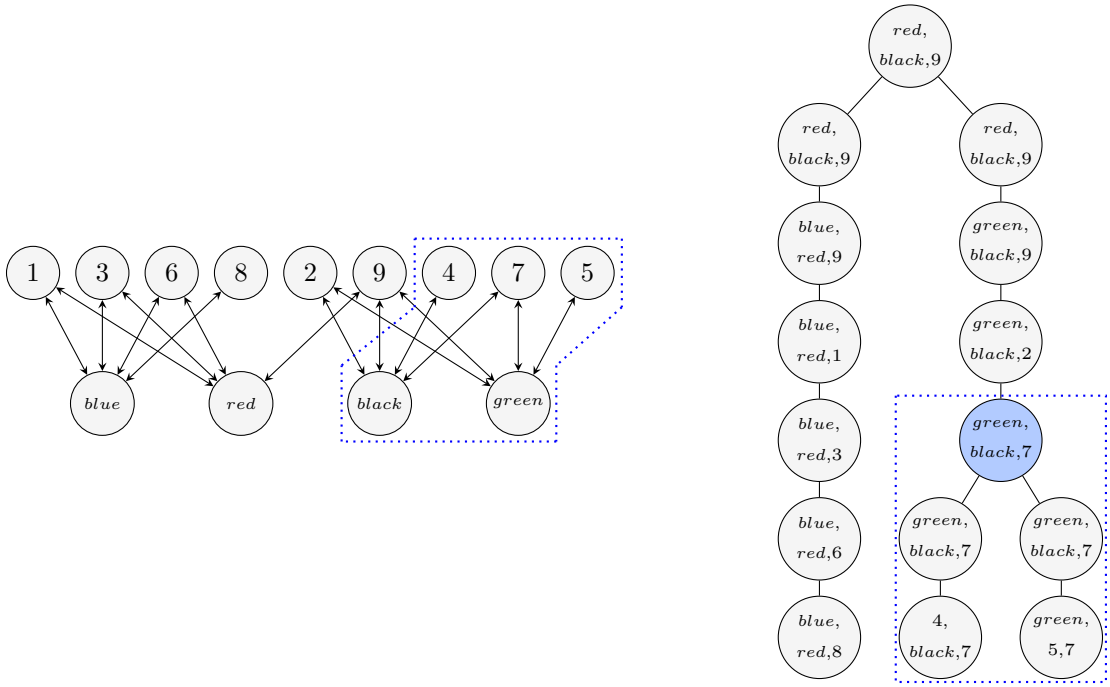
Corollary 1 shows that if we can solve (2) for G', we can solve *MIN-SCTM* for G. The algorithm introduced by Ben-Zwi et al. (2011) is an FPT algorithm (with parameter ω') for target set selection under the LTM, assuming thresholds are bounded. Thus, we show how to generalize the algorithm to our setting, taking into consideration the characteristics of the auxiliary graph G': (i) edges are weighted, (ii) only certain types of nodes (the firm-nodes) can be part of the seed set, (iii) the objective is minimizing costs rather than number of nodes in the seed set, and (iii) we are interested in structural assumptions on G more so than G'.

The algorithm first computes a tree decomposition of G' with treewidth ω' and of a specific type:

Definition 5 (Nice tree decomposition). Let $G' = (N', E')$. A tree decomposition $\mathcal{T}' = (T', (X'_z)_{z \in Z})$ of G' with treewidth ω' is nice if and only if T' is rooted at some node $\tilde{z}$, $|X'_z| = \omega' + 1 \;\; \forall z \in T'$, and all nodes are of exactly one of the following types: (a) leaf nodes, (b) replace nodes (z has exactly one child, z_0, and there are $u, v \in N'$, $u \neq v$, such that $X'_z \setminus X'_{z_0} = \{u\}$ and $X'_{z_0} \setminus X'_z = \{v\}$), and (c) join nodes ($z$ has exactly two children, z_0 and z_1, and $X'_z = X'_{z_0} = X'_{z_1}$).

An example of a nice tree decomposition is given in Figure 13.

Before any formal results, we provide high-level intuition as to how the algorithm works. Assume a nice tree decomposition $\mathcal{T}' = (T', (X'_z)_{z \in Z})$ of G' with treewidth ω'. Possibly overlapping subgraphs of G' are created from the leafs of T', each containing at most ω' nodes. Moving up one level in the tree corresponds to growing the subgraphs constructed at the previous level, either by adding nodes or merging two subgraphs. When we get to the root node, the subgraph considered is the complete graph G'. The key property of the subgraphs constructed via this process is that each one only interacts with its complement in G' via a set of *boundary nodes*, which is of size at most ω'. In other words, if there exists an edge between node i in the subgraph and a node outside of the subgraph, then it must be that i belongs to the boundary nodes, and there cannot be more than ω such nodes. The minimum cost seed set for G' is then obtained recursively using these subgraphs. First, the minimum cost seed sets for the subgraphs corresponding to the leafs are found through enumeration (which is exponential in ω' but not necessarily in n). Then, as we go up the tree, the algorithm uses the previously established minimum cost seed sets for smaller subgraphs to obtain that of the larger subgraph. The computational effort is limited because nodes in the smaller subgraph that are not part of the boundary are not affected by any nodes that may enter the subgraph. Thus, we can focus on



(a) Rearranged auxiliary graph G' from Figure 4b. (b) Nice tree decomposition T of G'.

Figure 13 Example of a nice tree decomposition. Let $r_{ji} = c_i = \theta_j = 1$ for all nodes i and hyperedges e_j.

nodes in the boundary. As we reach the root node, the relevant subgraph is all of G', allowing us to derive a minimum cost seed set for the full auxiliary graph and, thus, the original graph G.

LEMMA 2. *Assume an auxiliary graph G' as defined in Section 3.3, as well as the LTM activation process. Algorithm 2 applied to G' returns a minimum cost seed set.*

Proof. We introduce some notation. Consider a subtree in T' rooted at a node z_0. Define with $G^{z_0} = (N^{z_0}, E^{z_0})$ the subgraph of G' induced by $\bigcup_{z \in subtree(z_0)} X'_z$. Nodes in X'_{z_0} may be connected to nodes in G' outside of G^{z_0}, but other nodes in G^{z_0} cannot be connected to those outside. Thus, denote X'_{z_0} as the *boundary* of z_0 (it may include unconnected nodes, due to the requirement that $|X'_z| = \omega' + 1$). For example, in Figures 13b (resp. Figure 13a), the dashed line indicates the subtree rooted at $\{green, black, 7\}$ (resp. the associated subgraph). Nodes *black* and *green*, which are in the boundary, connect to nodes 2 and 9 outside of the dashed line. The boundary may include additional nodes such as 7 due to $|X'_z| = \omega' + 1$. Node 4, on the other hand, is not in the boundary and cannot connect to nodes outside of the dashed line. Moving upwards in the tree from z_0 to a replace node z_R, a node $v \in X'_{z_0}$ is replaced by a node u to arrive at the new boundary X'_{z_R}. By construction, node v cannot share edges with any of the nodes outside G^{z_0}. Say, we move from $\{green, black, 7\}$ to $\{green, black, 2\}$. In the boundary, Node 7 is replaced by Node 2, and Node 7 does indeed not share edges with nodes outside of the subgraph induced by *black*, *green*, 2, 4, and 5.

Next, let $\tilde{c} \in \{0, \dots, \vartheta_{\max}\}^{\omega'+1}$ and $\tilde{a} \in \{0, \dots, \omega' + 1\}^{\omega'+1}$ be threshold and activation vectors, where $\vartheta_{\max} = \max\{\max_{i=1,\dots,n}\{c_i\}, \max_{j=1,\dots,m}\{\theta_j\} - 1\}$. We arbitrarily assign a one-to-one mapping from the root node boundary to these vectors, denoting with $\tilde{c}(i')$ (resp. $\tilde{a}(i')$) the mapping from i' in the boundary. Define

Algorithm 2: Algorithmic solution to minimum cost seed set problem.

Data: Auxiliary graph G' with $tw(G') = \omega'$.
Result: Minimum cost seed set $(S_0')^*$ of G'.

Initialization: Compute a nice tree decomposition $\mathcal{T}' = \left(T', \{X_z'\}_{z \in T'}\right)$ of G', rooted at $\tilde{z}$ with width ω'; $\tilde{C} \leftarrow \{0, \dots, \vartheta_{max}\}^{\omega'+1}$; $\tilde{A} \leftarrow \{0, \dots, \omega'+1\}^{\omega'+1}$; $red \leftarrow 0^{\omega'+1}$;
for $z \in T'$ *where* z *is a leaf node* **do**
  for $\tilde{c} \in \tilde{C}$ **do**
    for $\tilde{a} \in \tilde{A}$ **do**
      $S_0^z[\tilde{c}, \tilde{a}] \leftarrow$ compute minimum cost seed set through enumeration;
while $z \in T'$ *has not been traversed, but all its child nodes have* **do**
  if z *is a replace node with child* z_0 **then**
    $G_0^z = (G_0^z, E_0^z) \leftarrow G^z$; $E_0^z \leftarrow E_0^z \setminus \{(u, i'), (i', u) : i' \in X_z'\}$;
    for $\tilde{c} \in \tilde{C}$ **do**
      $c \leftarrow \tilde{c}$; $c(v) \leftarrow c_v'$;
      for $\tilde{a} \in \tilde{A}$ **do**
        $\hat{A} \leftarrow \{a \in \tilde{A} : a(i') = \tilde{a}(i') \ \forall i' \neq v\}$; $a^* \leftarrow \arg\min_{a \in \hat{A}} \mathcal{C}\left(S_0^{z_0}[c, a]\right)$;
        $$S_0^z[\tilde{c}, \tilde{a}] \leftarrow \begin{cases} S_0^{z_0}[c, a^*] & \text{if } \tilde{c}(u) = 0 \\ S_0^{z_0}[c, a^*] \cup \{u\} & \text{if } \tilde{c}(u) \neq 0, u \text{ is a firm-node} \\ \{i' \in N' \ : \ i' \text{ is a firm-node}\} & \text{otherwise.} \end{cases}$$
    for $e_1 = (u, i') \in \{(u, i') : i' \in X_z'\}$ *and* $e_2 = (i', u)$ **do**
      $E_0^z \leftarrow E_0^z \cup \{e_1, e_2\}$; $S_0^{z'} \leftarrow S_0^z$;
      for $\tilde{c} \in \tilde{C}$ **do**
        $\tilde{c}^u \leftarrow \tilde{c}$; $\tilde{c}^{i'} \leftarrow \tilde{c}$; $\tilde{c}^u(u) \leftarrow \max\{\tilde{c}(u) - w_{i',u}', 0\}$; $\tilde{c}^{i'}(i') \leftarrow \max\{\tilde{c}(i') - w_{u,i'}', 0\}$;
        for $\tilde{a} \in \tilde{A}$ **do**
          $$S_0^z[\tilde{c}, \tilde{a}] \leftarrow \begin{cases} S_0^{z'}[\tilde{c}, \tilde{a}] & \text{if } \tilde{a}(i') = \tilde{a}(u) \\ S_0^{z'}[\tilde{c}^u, \tilde{a}] & \text{if } \tilde{a}(i') < \tilde{a}(u); \\ S_0^{z'}[\tilde{c}^{i'}, \tilde{a}] & \text{if } \tilde{a}(i') > \tilde{a}(u) \end{cases}$$
  if z *is a join node with children* z_0 *and* z_1 **then**
    for $\tilde{a} \in \tilde{A}$ **do**
      for $i' \in X_z'$ **do**
        $red(i') \leftarrow \sum_{\{j' \in X_z \ : \ (j', i') \in E' \text{ and } \tilde{a}(j') < \tilde{a}(i')\}} w_{j',i'}'$;
      for $\tilde{c} \in \tilde{C}$ **do**
        $(\tilde{c}^{z_0}, \tilde{c}^{z_1}) \leftarrow \arg\min_{\left\{\tilde{c}^{z_0'}, \tilde{c}^{z_1'} \ : \ \tilde{c}^{z_0'} + \tilde{c}^{z_1'} = \tilde{c} + red\right\}} \mathcal{C}\left(S_0^{z_0}[\tilde{c}^{z_0'}, \tilde{a}] \cup S_0^{z_1}[\tilde{c}^{z_1'}, \tilde{a}]\right)$;
        $S_0^z[\tilde{c}, \tilde{a}] \leftarrow S_0^{z_0}[\tilde{c}^{z_0}, \tilde{a}] \cup S_0^{z_1}[\tilde{c}^{z_1}, \tilde{a}]$;
$(S_0')^* \leftarrow S_0^{\tilde{z}}[c, a]$, with $c^*(i') = c_{i'}' \ \forall i' \in X_{\tilde{z}}'$ and $a = \arg\min_{\tilde{a} \in \tilde{A}} \mathcal{C}\left(S_0^{\tilde{z}}[c^*, \tilde{a}]\right)$;

mappings for other boundaries recursively: (i) if v replaces u, v is mapped to the same index and the mapping remains unchanged otherwise; and (ii) if a node has two children, boundary and mapping remain unchanged.

Finally, for any $z \in T'$, we define a matrix S_0^z with rows indexed by the vectors $\tilde{c}$ and columns indexed by the vectors $\tilde{a}$. The size of the matrix is $[(\vartheta_{\max}+1)\cdot(\omega'+2)]^{\omega'+1} = [\vartheta_{\max}\cdot\omega']^{O(\omega')}$. Each entry is a set of nodes in G' and matrices will be computed recursively bottom-up. The last matrix to be computed (corresponding to the root node $\tilde{z}$) gives us the minimum cost seed set of G', $(S_0')^*$. For any set $S_0^z[\tilde{c},\tilde{a}]$, we define its cost by $\mathcal{C}(S_0^z[\tilde{c},\tilde{a}]) = \sum_{i' \in S_0^z[\tilde{c},\tilde{a}]} w_i$, where w_i is the cost of adding node $i \in N_F$ to the seed set.

Algorithm 2 assumes a nice tree decomposition $\mathcal{T}'$ of G'. It is based on Ben-Zwi et al. (2011), but adapted to reflect the specificities of G'. For each leaf node z, each $\tilde{c}$, and each $\tilde{a}$, we compute a "minimum cost seed set" (Lines 2–5 in Algorithm 2) based on the subgraph G^z and assuming that a node $i' \in X_z'$ (i) has threshold $\tilde{c}(i')$; (ii) can activate only if all nodes $j' \in X_z'$ with $\tilde{a}(j') < \tilde{a}(i')$ are already active and if all $j' \in X_z'$ with $\tilde{a}(j') = \tilde{a}(i')$ activate at the same time; and (iii) can only be part of the seed set if it is a firm-node. For example, let $z_0 = \{4, black, 7\}$, $\tilde{c} = (1,1,0)$, and $\tilde{a} = (2,1,0)$. Node 7 has a threshold of 0. It also has the lowest value in the activation vector. Hence, Node 7 activates for any seed set. Node *black* has a threshold of 1. However, there is an edge $(7, black)$ with $w'_{7,black} = 1$ and Node *black* has a lower value in the activation vector than Node 4. Once Node 7 is active, Node *black* also activates. Thereafter, Node 4 can also activate. It follows that the minimum cost seed set is $\varnothing$. Assume, instead, that $\tilde{c} = (1,1,3)$, and $\tilde{a} = (2,1,0)$. Node 7 still has to activate first. However, its threshold is 3, so it must be part of the seed set. With Node 7 in the seed set, the remainder of the activation process is unchanged, so the minimum cost seed set is $\{7\}$. Finally, assume that $\tilde{c} = (1,1,3)$, and $\tilde{a} = (2,0,1)$. Node *black* has to activate first, but SC-nodes cannot be added to the seed set. Hence, no such minimum cost seed set exists, and we adopt the convention of equating the minimum cost seed set to the entire set of node-nodes. That is, $S_0^{z_0}[(1,1,3),(2,0,1)] = \{i' \in N' \,:\, i' \text{ is a firm-node}\}$.

Next, we recursively generate S_0^z for all other nodes $z \in T'$. If z is a replace node (Lines 7–19 in Algorithm 2) with child z_0, the subgraph G^z of G' has exactly one more node than G^{z_0}, node u. Node u replaces node v in the boundary, mapped to entry i_v in both activation and threshold vectors. For each $\tilde{a}$, define set $\hat{A} = \{a \in \{0,\ldots,\omega'+1\}^{\omega} : a(i') = \tilde{a}(i') \;\; \forall i' \neq v\}$ grouping all activation vectors that differ only in entry i_v. For example, if $z_0 = \{4, black, 7\}$, $z = \{green, black, 7\}$, then $u = green$ and $v = 4$. Moreover, if $\tilde{a} = (2,1,0)$, then $\hat{A} = \{(0,1,0),(1,1,0),(2,1,0),(3,1,0)\}$. Then, for each $\tilde{c}$, construct intermediary sets $S_0^z[\tilde{c},\tilde{a}]$ to remove dependencies on v and introducing the node u (Lines 9–13 in Algorithm 2). We define them thus: (i) if $\tilde{c}(u) = 0$, then $S_0^z[\tilde{c},\tilde{a}] = S_0^{z_0}[c,a]$ where $c(i') = \tilde{c}(i'), i' \neq v$, $c(v) = c'_v$, and $a = \arg\min_{a \in \hat{A}} \mathcal{C}(S_0^{z_0}[c,a])$, (ii) if $\tilde{c}(u) > 0$ and u is a firm-node, then $S_0^z[\tilde{c},\tilde{a}] = S_0^{z_0}[c,a] \cup \{u\}$ with a and c as before, (iii) if $\tilde{c}(u) > 0$ and u is a SC-node, then $S_0^z = \{i' \in N' \,:\, i' \text{ is a firm-node}\}$. In the previous example, if $\tilde{a} = (2,1,0)$ and $\tilde{c} = (0,1,3)$, then $c = (1,1,3)$ and $S_0^z[(2,1,0),(0,1,3)]$ is the least costly of the following seed sets: $\{4,7\}$ (corresponding to $a = (0,1,0)$), $\{4,7\}$ (corresponding to $a = (1,1,0)$), $\{7\}$ (corresponding to $a = (2,1,0)$), $\{7\}$ (corresponding to $a = (3,1,0)$). As $\tilde{c}(green) = 0$, it follows that $S_0^z[(2,1,0),(0,1,3)] = \{7\}$.

Given $\tilde{c}$ and $\tilde{a}$, $S_0^z[\tilde{c},\tilde{a}]$ reflects the minimum cost seed set to activate the subgraph G^z minus any edges between node u and other nodes in the boundary. We iteratively refine the intermediary sets while adding these edges to the subgraph (Lines 14–19 in Algorithm 2). Choose one such edge, (u, i'), and the corresponding edge (i', u) and copy the entries of matrix S_0^z to a new matrix $S_0^{z'}$. For any $\tilde{a}$ and $\tilde{c}$, there are three options: (i) if $\tilde{a}(u) < \tilde{a}(i')$, node u has to activate before node i'. Once u is active, it contributes $w'_{i',u}$ towards activation of

i'. Hence, update $S_0^z[\tilde{c},\tilde{a}] = S_0^{z'}[\tilde{c}^{i'},\tilde{a}]$, where $\tilde{c}^{i'}$ is identical to $\tilde{c}$, except in entry i': $\tilde{c}^{i'}(i') = \max\{\tilde{c}(i') - w'_{u,i'}, 0\}$. (ii) if $\tilde{a}(u) > \tilde{a}(i')$, u has to activate after i'. Once i' is active, it contributes a benefit of $w'_{i',u}$ towards activation of u. Hence, update $S_0^z[\tilde{c},\tilde{a}] = S_0^{z'}[\tilde{c}^{u},\tilde{a}]$, where $\tilde{c}^{u}$ is identical to $\tilde{c}$, except in entry u: $\tilde{c}^{u}(u) = \max\{\tilde{c}(u) - w'_{u,i'}, 0\}$. (iii) if $\tilde{a}(u) = \tilde{a}(i')$, the nodes have to activate simultaneously, so they cannot influence each other. Thus, $S_0^z[\tilde{c},\tilde{a}] = S_0^{z'}[\tilde{c},\tilde{a}]$. In the previous example, add the edges $(green, 7)$ and $(7, green)$ with $w'_{green,7} = w'_{7,green} = 1$, and assume again that $\tilde{a} = (2,1,0)$ and $\tilde{c} = (0,1,3)$. Then, $\tilde{a}(green) = 2 > 0 = \tilde{a}(7)$ and we consider $\tilde{c}^{green} = (0,1,3)$. The latter vector is unchanged because the threshold in the second entry is already 0, so $S_0^z[(0,1,3),(2,1,0)] = S_0^{z'}[(0,1,3),(2,1,0)]$. If, however, $\tilde{a} = (0,1,2)$, then we need to consider the threshold vector $\tilde{c}^{7} = (0,1,2)$ to account for the fact that 7 always activates after *green*. In this case, $S_0^z[(0,1,3),(0,1,2)] = S_0^{z'}[(0,1,2),(0,1,2)]$. Revise $S_0^{z'} = S_0^z$ and repeat for all edges.

If z is a join node (Lines 20–26 in Algorithm 2) with children z_0 and z_1, $G^z = G^{z_0} \cup G^{z_1}$ (there are no edges between nodes of the two subgraphs outside the boundary). Fix $\tilde{a}$ and $\tilde{c}$. A node $i' \in X'_z$ may benefit from activations in both subgraphs G^{z_0} and G^{z_1}. For example, consider the join node with $z = (red, black, 9)$ in Figure 13 and take $\tilde{a} = (2,0,1)$ (i.e., *black* has to activate first, followed by 9, then *red*) and $\tilde{c} = (1,0,2)$ (i.e., $\tilde{c}_{red} = 1$, $\tilde{c}_{black} = 0$, $\tilde{c}_9 = 2$). In this case, the minimum cost seed set for the subgraph in G' associated with the left-hand subtree is $\{9\}$: *black* activates due to its threshold, *red* activates as 9 is active and then 1, 3 and 6 activate, followed by *blue* and finally 8. Following a similar reasoning, a minimum cost seed set for the subgraph in G' associated to the right-hand subtree is $\{5\}$. However, while G' does activate if $\{5,9\}$ is active, this is not the smallest seed set, which is given by $\{9\}$. Thus, we construct $S_0^z[\tilde{c},\tilde{a}]$ to account for synergies. For each $i' \in X'_z$, we define the following weight reduction which avoids double-counting: $red(i') = \sum_{\{j' \in X'_z \,:\, (j',i') \in E' \text{ and } \tilde{a}(j') < \tilde{a}(i')\}} w'_{i',j'}$. We then take $S_0^z[\tilde{c},\tilde{a}] = S_0^{z_0}[\tilde{c}^{z_0},\tilde{a}] \cup S_0^{z_1}[\tilde{c}^{z_1},\tilde{a}]$, where

$$(\tilde{c}^{z_0},\tilde{c}^{z_1}) = \arg \min_{f,g \in \{0,\dots,k-1\}^\omega} \mathcal{C}\left(S_0^{z_0}[f,\tilde{a}] \cup S_0^{z_1}[g,\tilde{a}]\right)$$
$$\text{s.t. } f(i') + g(i') = \tilde{c}(i') + red(i') \text{ for all } i' \in X'_z.$$

For example, if $X'_{z_0} = X'_{z_1} = \{green, black, 7\}$, $\tilde{a} = (2,1,0)$, and $\tilde{c} = (0,1,3)$, then $red(black) = 1$, $red(green) = 1$, and $red(7) = 0$. It follows that $S_0^z[(0,1,3),(2,1,0)]$ is based on the union of seed sets corresponding to the threshold vectors $(\tilde{c}^{z_0},\tilde{c}^{z_1}) = \arg\min_{\{f,g \,:\, f+g=(1,2,3)\}} \mathcal{C}\left(S_0^{z_0}[f,(2,1,0)] \cup S_0^{z_1}[g,(2,1,0)]\right)$.

Proceed until the root note $\tilde{z}$. To obtain $(S'_0)^*$, take the threshold vector c^* corresponding to the actual thresholds, that is $c^*(i') = c'_{i'}$ $\forall i' \in X'_{\tilde{z}}$. As the optimal seed set induces an activation sequence $\tilde{a}$, $(S'_0)^* = S_0^{\tilde{z}}[c^*,a^*]$, with $a^* = \arg\min_{\tilde{a}} S_0^{\tilde{z}}[c^*,\tilde{a}]$. □

LEMMA 3. *Assume an auxiliary graph G' as defined in Section 3.3 with treewidth $tw(G') = \omega'$. Algorithm 2 applied to G' runs in $[\vartheta_{\max} \cdot \omega']^{O(\omega')}(n+m)$ time.*

Proof. The decomposition of G' into a minimal tree requires time exponential in ω' but linear time when ω' is bounded (Bodlaender 1996) and, given an arbitrary tree decomposition, one can always construct a a nice tree decomposition of the same width in linear time. (Ben-Zwi et al. 2011). Moreover, G' contains $n+m$ nodes, so a nice tree decomposition of width ω' exists with at most $(\omega'+1)\cdot(n+m)$ nodes. Hence, the number of entries of any S_0^z is bounded by $[\vartheta_{max}\omega']^{O(\omega')}$.

Table 2 Means and standard deviations of key statistics of each data set employed.

	Willems (2008)	Random	Modular
n	152.54 (178.38)	24.23 (0.77)	29.91 (0.38)
m	157.33 (337.27)	27.54 (10.27)	54.57 (19.32)
k	5.34 (1.98)	5.0 (0.0)	5.0 (0.0)
ω'	8.96 (9.02)	9.62 (2.72)	14.85 (4.25)

The number of computations for each entry of a leaf node is $2^{\omega'+1}$ (Lines 2–5 in Algorithm 2). The number of computations for each entry of a replace node is determined by comparing all $\omega' + 2$ activation options of v (Lines 8–13 in Algorithm 2) and iterating through each of at most ω' edges that u shares with other nodes in the boundary (Lines 14–19 in Algorithm 2). The number of computations for each entry of a join node is determined by comparing combinations of thresholds of the boundaries, which is upper-bounded by a constant factor of $\vartheta_{max}^{\omega'+1}$. It follows that the maximum number of computations required for Algorithm 2 is in $\left[\vartheta_{max}\omega'\right]^{O(\omega')}(n+m)$. □

We can now state and proof the main result regarding the DP algorithm:

THEOREM 3. *Let G be a hypergraph as defined in Section 3. Assume that $tw(G) = \omega$. Then, MIN-SCTM can be solved exactly via a dynamic program in $[\vartheta_{\max} \cdot \omega]^{O(\omega)}(n+m)$ time.*

Proof of Theorem 3. From Lemma 2, we know that Algorithm 2 provides a minimum cost seed set for auxiliary graph G', assuming the LTM activation process. Meanwhile, Line 13 of the algorithm ensures that the seed set contains only nodes $i \in N_F$ and the assumption on how seed set costs are computed ensures that these nodes are weighted with w_i. It follows from Proposition 1 that the seed set identified is also a solution to *MIN-SCTM*. Combined with Lemma 3, the fact that constructing G' from G requires a constant factor of $n+m$ operations, and Proposition 3 ($\omega' \leq \omega + 1$), the result follows. □

From this, we can directly derive a corollary corresponding to Corollary 2:

COROLLARY 3. *There is a dynamic programming-based FPT algorithm for solving MIN-SCTM with parameter ω and $\vartheta_{\max}$, running in time at most $2^{O(\omega \log_2(\omega))}\vartheta_{\max}^{O(\omega)}(n+m)$.*

Appendix E: Description of data and measures for numerical experiments

E.1. Description of supply chain network data sets

We generate three datasets of supply chain networks, with key statistics summarized in Table 2. We discuss the generation processes below. In all cases, we let $c_i = r_{ji} = 1$, $\theta_j = k_j$, and $w_i \sim N(1, 0.1)$ for all $i \in N_F, j \in N_{SC}$. Standardization allows us to remove (predictable) effects of costs and benefits while focusing on the effects of graph structures. At the same time, perturbing w_i enables us to obtain unique solutions even in networks with repetitive structures.

Willems (2008) networks. The original data set represents 38 acyclic networks of companies in 22 industries. While the data contains information about direct connections between nodes, it does not specify which sets of nodes belong to which supply chains (i.e., we do not have access to the hyperedges of G). To circumvent this difficulty, we randomly generate the hyperedges of the graph using the observable edges in the following way: We consider all possible paths between nodes in the first and the last tier, that is, nodes with no incoming, respectively, no outgoing edges. We then assume for each path with probability 0.05, 0.25, or 0.5 that it is a supply chain and remove nodes not part of any supply chain. We repeat this generation process ten times for each network-probability combination and remove networks with less than 15 remaining nodes, as well as those for which we cannot find a seed set guaranteed to be within 5% of the minimum cost within two hours using Gurobi. This results in a total of 657 supply chain networks.

Random networks. We fix five tiers and the following configurations of nodes per tier: $(5,5,5,5,5)$, $(2,6,9,6,2)$, and $(2,2,4,7,10)$, where the value at index l represents the number of nodes at tier l. We also fix a number of supply chains $\overline{m} \in \{20,40\}$. Each node and supply chain is randomly assigned a value drawn from the uniform distribution on $(0,1)$. Then, at each tier, we identify the $h \in \{1,2,\dots,6\}$ nodes whose assigned value is closest to that of the supply chain. Of those nodes, we randomly choose one to be part of the supply chain. We repeat the generation process if a supply chain with the same nodes already exists and remove nodes that belong to no supply chain after all have been generated. The instance is discarded if more than 10% of the generated nodes have been removed. We repeat the entire process 20 times for each parameter combination and discard networks for which we cannot find a seed set guaranteed to be within 5% of the minimum cost within two hours using Gurobi. This results in 525 supply chain networks.

Random modular networks. We fix the node-tier structure $(3,3,3,3,3)$ and generate $\overline{m} \in \{5,10,15,20,25,30\}$ distinct supply chains by randomly selecting one node of each tier to be part of the supply chain. We copy the resulting network and obtain two disconnected (but identical) hypergraphs. We then generate $\overline{m}$ "connecting" supply chains, that is, supply chains that contain one node of each tier from either of the two originally disconnected hypergraphs. We keep track of each supply chain network generated in this process (one for each additional connecting supply chain). We repeat this process four times for each initial parameter combination. Nodes that do not belong to any supply chains are removed; if more than 10% of the generated nodes have been removed, the instance is discarded. This results in 434 supply chain networks. Based on how we generate the networks, they have comparatively high treewidth, so Gurobi is frequently unable to identify a useful lower bound. Hence, we take the solutions obtained by Gurobi after three hours of run time and experiment with different improvement heuristics. As we cannot identify a single instance in which an improvement is found, we assume the solutions found using Gurobi are sufficiently close to optimal. Note that these networks are designed to obtain varying degrees of modularity (see Appendix E.3) while keeping other structural measures largely constant.

E.2. The relationship between Jaccard clustering and IP (5)

We make explicit the connection between Jaccard clustering and the integer program given in (5). Consider a firm-node i in G' and let k be a firm-node sharing a supply chain $j \in N_{SC}$ with i. As i and j (resp. j and k)

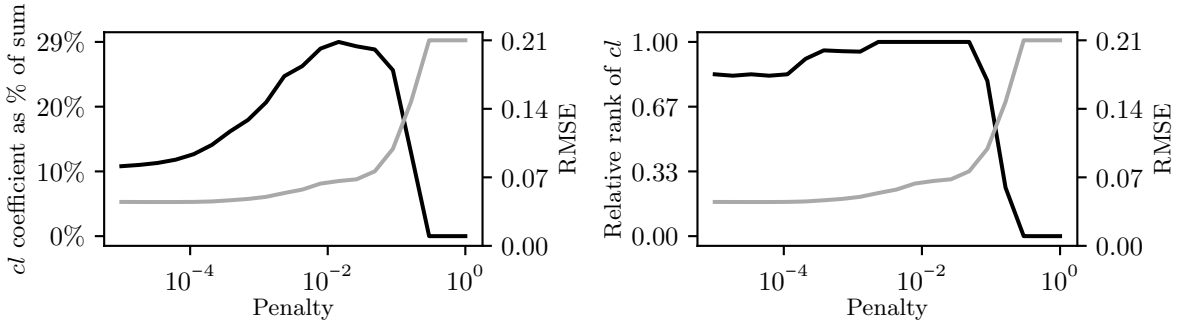
are neighbors in G', i and j (resp. j and k) necessarily appear in the same bag in the tree decomposition of G'. If i, j and j, k appear in different bags, then j must appear in all intermediate bags, by definition of a tree decomposition. Thus, in a setting where i and k share many supply chains, i.e., the numerator of $NS(i,k)$ is high, the minimal tree decomposition will likely place i and k, together with all shared supply chains j, in the same bag to minimize treewidth. If i, j, k are all in the same bag in the tree decomposition of G', then constraint (4d) now applies to the three nodes, i.e., we need $\ell_{ij} + \ell_{jk} + \ell_{ki} \leq 2$. For this constraint to hold, at least one of $\ell_{ji}, \ell_{jk}, \ell_{ki}$ needs to be equal to zero. If this is either ℓ_{ji} or ℓ_{jk}, then constraints (5a) and (5b) become harder to meet, unless we set s_i or s_k to 1, that is, we add i or k to the seed set. This phenomenon is further exacerbated in the setting where the denominator of $NS(i,k)$ is low. In this case, i and k belong to few supply chains in total, which makes constraint (5b) harder to satisfy as the sum over all supply chains that i (resp. k) belongs to only contains few terms. This, in turn, also pushes the seed set to be larger.

E.3. Description of network measures and parameters

We consider the following previously defined measures: the number of nodes n, the number of supply chains m, the maximum number of nodes per supply chain k, the treewidth of the auxiliary graph ω', and the clustering metric J. In addition, we also define the following measures:

Alternative clustering metrics. First, *projection clustering*. We consider the (non-bipartite) projection of G' onto firm-nodes and use the traditional definition of clustering for graphs. More specifically, we construct a graph G'' from the n firm-nodes, with an edge between two firm-nodes if they have at least one supply chain in common. The standard clustering coefficient is defined, for example, in Latapy et al. (2008), and we take the average over all nodes of the projection. Second, *projection clustering (weighted)* follows the same principles, but each edge in G'' is weighted by the number of supply chains the nodes of the edge shave in common. Third, *hourglass clustering*. On non-bipartite graphs, the clustering coefficient of a node can equivalently be defined as the number of triangles containing the node divided by the number of triplets containing the node with at least two edges. To extend this idea to bipartite graphs, one can divide the number of fully connected quadruplets by the number of quadruplets with at least three links (Latapy et al. 2008). We employ this extension and consider both the average ratio across firm-nodes and the total ratio throughout the graph. Finally, *repetition of partners*. This measures how many firm-nodes, on average, a given firm-node shares supply chains with.

Modularity. A commonly used definition is provided by Newman (2006): given a partition of n nodes of a graph H into z groups $\mathcal{P} = (p_1, \ldots, p_z)$, the modularity of H is $Q = \frac{1}{2\eta} \sum_{ij} \left(A_{ij} - \frac{\eta_i \eta_j}{2\eta} \right) \delta(p_i, p_j)$, where η is the sum of all edge weights in the graph, η_i is the sum of the weights of the edges attached to node i, A is the (weighted) adjacency matrix of H, and $\delta(p_i, p_j)$ is equal to 1 if $p_i = p_j$ (that is, i and j are in the same community) and 0 otherwise. This definition has no direct extension for hypergraphs and does not apply to bipartite graphs (such as our auxiliary graph). Hence, we use the weighted projection of G' onto firm-nodes, denoted by G'', as in the case of *projection clustering (weighted)* to compute modularity. The graph G'' is non-bipartite but keeps most of the relevant information about linkages between nodes. We then use the commonly applied Clauset-Newman-Moore greedy modularity maximization algorithm to find the partition leading to the largest Q (Clauset et al. 2004).



(a) Relative weight of J's coefficient

(b) Relative rank of J's coefficient

Figure 14 Elastic net regression on the Willems (2008) networks with an L1-ratio of 0.8 and varying penalty coefficient. The black (resp. gray) lines indicate the value of the left-hand (resp. right-hand) axes.

Other relevant measures from the supply chain and network literature. For our predictive models, we identify *accessibility* and *interconnectedness* as important measures for supply chain networks (Bellamy et al. 2014). Accessibility refers to the information centrality of nodes, that is, the length of paths ending at a given node. Interconnectedness, meanwhile, refers to the number of shared relationships between connections of a node. As the measures directly relate to diffusion and are defined for undirected graphs, we seek to apply them to the auxiliary graph G'. However, they are not well-defined for bipartite graphs, so we consider them on the (weighted) projection G''.

Finally, we compute all measures from Perera et al. (2017, Table 4) on a modified version of G, where directed edges connect nodes in subsequent tiers sharing a supply chain. The authors summarize key supply chain network measures from the empirical literature. We omit clustering and modularity, which are already specified. As we deal with a directed graph, we compute *assortativity* based on all in/out degree combinations. This leads to eleven measures, to which we add the *average number of supply chains of a firm*.

E.4. Predictive models of the seed set size

We first use a random forest regression to predict the inverse logit function of the percentage of seed nodes in the optimal solution on the measures introduced in Appendix E.3. In particular, we select 80% of the instances of a dataset for training and choose hyperparameters by applying 5-fold cross-validation on 100 randomly chosen combinations. We then evaluate the best model identified by computing the root mean square errors (RMSE) for the remaining 20% of instances between the actual and predicted percentage of seed nodes. RMSEs are 0.021, 0.024, and 0.016 for the three datasets. Unlike the case of linear regression, the importance of each regressor in a random forest can only be computed indirectly by calculating and weighing the Shapley values of the different decision trees for a subset of data. We use the Python package *shap* for this task (Lundberg and Lee 2017), finding that J has the highest importance among clustering metrics for all networks and either the highest or second-highest importance among all regressors.

To gain more insight into the importance of Jaccard clustering compared to other variables, we introduce an elastic net regression on the same variables, i.e., a regression combining L1-norm ("Lasso") and L2-norm ("Ridge") penalty terms, and vary the regression's regularization penalty coefficient on a logarithmic scale. Figure 14 depicts the results for the Willems (2008) networks. The results for other datasets are omitted

for brevity but show the same patterns. In particular, any model with good explanatory power (before the RMSE increases steeply) puts a high weight on J. As we increase the penalty, i.e., requiring the model to have fewer explanatory variables, J becomes more important. In Figure 14a, we consider the coefficient obtained for each variable during the regression and plot in black the coefficient corresponding to J divided by the sum of all (absolute) coefficients. As can be seen, this ratio increases to a third. In Figure 14b, we observe that the relative rank (in black) of the absolute value of the coefficient of J increases to 1. The two curves then drop suddenly, but only when we have reached a penalty value that generates a highly biased model (as seen from the "explosion" of the RMSE in gray). Our results are consistent for all relative weights of L1 and L2, as long as the L1 term is high enough to ensure convergence, i.e., in the range $\frac{L1}{L1+L2} \in [0.5, 1.0]$.

References for the Appendix

Ackerman E, Ben-Zwi O, Wolfovitz G (2010) Combinatorial model and bounds for target set selection. *Theor. Comput. Sci.* 411(44-46):4017–4022.

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

Ben-Zwi O, Hermelin D, Lokshtanov D, Newman I (2011) Treewidth governs the complexity of target set selection. *Discrete Optim.* 8(1):87–96.

Bodlaender HL (1996) A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.* 25(6):1305–1317.

Chen N (2009) On the approximability of influence in social networks. *SIAM J. Disc. Math.* 23(3):1400–1415.

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

Courcelle B (2015) Clique-width and tree-width of sparse graphs. https://tinyurl.com/2wf6emvb (Accessed May 1, 2023).

Kempe D, Kleinberg J, Tardos É (2003) Maximizing the spread of influence through a social network. *Proceedings of the Ninth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 137–146.

Latapy M, Magnien C, Del Vecchio N (2008) Basic notions for the analysis of large two-mode networks. *Soc. Networks* 30(1):31–48.

Lundberg SM, Lee SI (2017) A unified approach to interpreting model predictions. Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R, eds., *Advances in Neural Information Processing Systems 30*, 4765–4774 (Curran Associates, Inc.).

Newman ME (2006) Modularity and community structure in networks. *PNAS* 103(23):8577–8582.

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

Willems SP (2008) Real-world multiechelon supply chains used for inventory optimization. *Manuf. Serv. Op.* 10(1):19–23.