The Stochastic Pseudo-Star Degree Centrality Problem

Mustafa C. Camur [1] Thomas C. Sharkey [2] Chrysafis Vogiatzis [3]

Abstract

We introduce the stochastic pseudo-star degree centrality problem, which focuses on a novel probabilistic group-based centrality metric. The goal is to identify a feasible induced pseudo-star, which is defined as a collection of nodes forming a star network with a certain probability, such that it maximizes the sum of the individual probabilities of unique assignments between the star and its open neighborhood. The feasibility is measured as the product of the existence probabilities of edges between the center node and leaf nodes and the product of one minus the existence probabilities of edges among the leaf nodes. First, the problem is shown to be $\mathcal{NP}$-complete. We then propose a non-linear binary optimization model subsequently linearized via McCormick inequalities. We test both classical and modern Benders Decomposition algorithms together with both two- and three-phase decomposition frameworks. Logic-based-Benders cuts are examined as alternative feasibility cuts when needed. The performance of our implementations is tested on small-world (SW) graphs and a real-world protein-protein interaction network. The SW networks resemble large-scale protein-protein interaction networks for which the deterministic star degree centrality has been shown to be an efficient centrality metric to detect essential proteins. Our computational results indicate that Benders implementations outperforms solving the model directly via a commercial solver in terms of both the solution time and the solution quality in every test instance. More importantly, we show that this new centrality metric plays an important role in the identification of essential proteins in real-world networks.

Keywords: Network analysis; Benders decomposition; Integer programming; Probabilistic group-based centrality

1 Introduction

A star graph can be defined as a tree graph with a maximum diameter of two, where the diameter is defined as the maximum distance between any two nodes. Different variations of star graphs have been attracting researchers' attention since the late 1980s. Akers and Krishnamurthy (1989) are the first who introduce the notion of a star graph as a new class of networks. Day and Tripathi (1992) expand this idea to generalized (n, k)-star graphs where n and k are user-defined values tuning the number of nodes and the degree / diameter trade-off. The idea is then used by Akers et al. (1994) who propose star graphs as an alternative to hyperbolic structures. Afterwards, Chou et al. (1996) propose bubble-sort star graphs as a new interconnection network structure. Past and recent studies heavily focus on the topological and functional analysis of star graphs (Chiang and Chen, 1998; Lin et al., 2020; Li et al., 2020). Further, star graphs are

[1] General Electric Research Center, Niskayuna, NY 12309.

[2] Industrial Engineering Department, Clemson University, Clemson, SC 29634.

[3] Industrial and Enterprise Systems Engineering Department, University of Illinois at Urbana-Champaign, Champaign, IL 61820.

utilized to identify facet defining inequalities during polyhedral analysis in several optimization studies as well (Mélot, 2008; Fragoso et al., 2021; Yan and Ryoo, 2022).

Centrality, on the other hand, is a well-known graph theory metric that indicates the importance of a node or a group of nodes in a given network. Degree centrality (Borgatti, 1995; Simon de Blas et al., 2018), closeness centrality (Brandes et al., 2016; Veremyev et al., 2019), betweenness centrality (Rysz et al., 2018; Bentert et al., 2020), and eigenvector centrality (Bonacich, 2007; Su et al., 2020) are among the most common node-based centrality metrics. In addition, Everett and Borgatti (1999) introduce the concept of group-based centrality metrics that aims to capture the importance of a set of nodes rather than individual nodes. Referencing the notion of group centrality, Vogiatzis and Camur (2019) use the star graph terminology (i.e., to be precise; the “induced star") as a centrality metric and introduce the *star degree centrality* (SDC) problem whose goal is to identify an induced star with the largest open neighborhood in a given network. They demonstrate that the SDC metric can help to identify essential proteins in protein-protein interaction networks (PPINs). Recently, the most-closeness central cliques, the most central representative sets, and the most-degree central cliques with regard to the deterministic group-based centrality metrics are studied by Nasirian et al. (2020), Rasti and Vogiatzis (2021), and Zhong et al. (2021), respectively. However, none of these studies consider a probabilistic setting during the detection of central sub-groups (e.g., clique, wheel). To the best of our knowledge, this paper is one of the first to propose a probabilistic group based centrality metric and use it in an optimization model.

PPINs are networks where each node and each edge represent a protein and the interaction between two proteins, respectively. We refer the reader to Szklarczyk et al. (2015) and Rasti and Vogiatzis (2019) for detailed information on PPINs. Vogiatzis and Camur (2019) show that the SDC can detect essential proteins (i.e., proteins that are vital for the existence of a cell or organism) in a PPIN better than node-based centrality metrics. The authors first show that the problem is $\mathcal{NP}$-complete, and then propose an integer programming (IP) formulation. They also introduce two approximation algorithms that work well in practice. Camur et al. (2022) examine the SDC problem on different classes of networks and widen the complexity discussions where the problem is shown to be $\mathcal{NP}$-complete on bipartite graphs and polynomially solvable on both tree and windmill graphs. More importantly, the authors design a decomposition approach as an exact solution method that significantly outperforms solving the IP models directly via a black-box solver.

In this paper, we introduce the stochastic pseudo-star degree centrality (SPSDC) problem where the goal is to detect an induced pseudo-star, that is truly a star with a high probability. With the term ‘high probability’ we mean that there is a high probability that a) the center has an edge to each leaf node, and b) there are no edges between leaf nodes. The objective is to maximize the connection probability of each neighbor node to the pseudo-star. From an application perspective, the SPSDC metric may help to identify new proteins that should be investigated to determine their essentiality (see Section 2). It may also help to confirm that essential proteins identified through the SDC metric are important.

We will be referring to the SDC as the deterministic SDC (DSDC). Our work is outlined as follows. We first formally define the DSDC and SPSDC problems and present two illustrative examples in Section 2. In Section 3, we show that the SPSDC problem and a related problem are $\mathcal{NP}$-complete on general graphs and tree networks, respectively. We then introduce a binary optimization model for the SPSDC problem in Section 4. Section 5 discusses the solution methodology that we adapt (i.e., Benders Decomposition) and we present algorithmic enhancements in Section 6. We discuss the data generation procedure and provide a wide range of computational experiments in Section 7. Lastly, we summarize our contributions and share our insights for future research directions in Section 8.

2 Problem Statements

In this section, we provide the formal definitions of the SPSDC and DSDC problems. Given an undirected network $G = (V, E)$ where V and E are the set of nodes and the set of edges, respectively, we define the open neighborhood of node $i \in V$ as the set of adjacent nodes to i (i.e., $N(i) = \{j \in V : (i, j) \in E\}$). The open neighborhood of a group of nodes $\mathcal{L}$ can be defined as $N(\mathcal{L}) = \{j \in V : j \notin \mathcal{L}, \exists\, i \in \mathcal{L}$ with $(i, j) \in E\}$. We then define the close neighborhood of a node i as $N[i] = \{i\} \cup \{j \in V : (i, j) \in E\}$. We let the k-neighborhood of node i, represented by $\bar{N}^k(i)$, be the set of nodes to which the shortest path length is exactly k from node i. Note that $\bar{N}^k(i) \cap \bar{N}^{k+1}(i) = \emptyset, \forall k \leq K$ where $k \in \mathbb{Z}^+$ and K is the length of the longest shortest path from node i to all other nodes in the network. Finally, we let p_{ij} represent the probability of existence of edge $(i, j) \in E$.

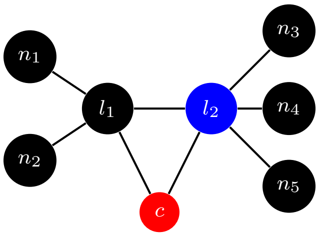
Definition 1. The *deterministic star degree centrality* of node i, represented by D_i, is a centrality metric which aims to form an induced star $\mathcal{S}_i$ centered at i with the largest size open neighborhood, where $D_i = \max\{|N(\mathcal{S}_i)| : \mathcal{S}_i$ is an induced star$\}$.

Definition 2. *The deterministic star degree centrality problem* aims to identify the node which has the largest star degree centrality in a given network.

In the deterministic setting, we are not concerned with the existence probabilities of the edges, i.e., we assume all edges in E exist in the network.

Example 1. Below we present a small example showing how to identify the DSDC of a given node (see Fig. 1). We select node c as the candidate center. Note that $N(c) = \{l_1, l_2\}$ represents the set of candidate leaf nodes. In a deterministic induced star, no two leaf nodes can be connected,. Therefore, l_1 and l_2 cannot be elements of the same star. Since the objective is to maximize the open neighborhood of the induced star, node l_2 is preferable over node l_1 as it gives access to more nodes. Thus, we obtain $\mathcal{S}_c = \{c, l_2\}$ and $N(\mathcal{S}_c) = \{l_1, n_3, n_4, n_5\}$.

Figure 1: Determining the deterministic star degree centrality of a given node where the center and leaf nodes are shown in red and blue, respectively.

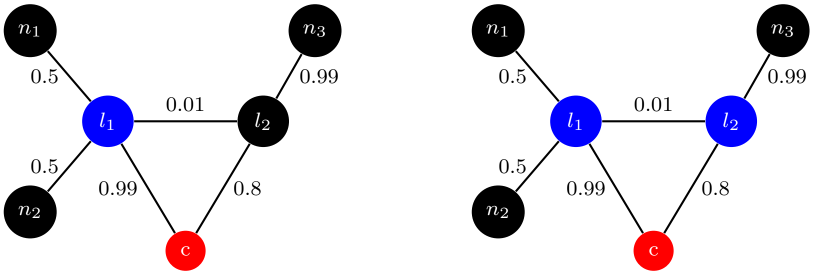


▶

In PPINs, there exist interaction scores that represent the strength of the interactions between two proteins. In fact, one can normalize the interaction scores and treat them as probability values that would indicate the likelihood of two proteins interacting. In this study, we examine the SPSDC problem where we seek to ensure that i) the probability values between the center node and each leaf node are high, and ii) the probability values between leaf nodes are low in order to ensure the feasibility of the star. It is crucial to point out that we now allow leaf nodes to connect as long as the induced star satisfies the *feasibility condition*, which will be introduced shortly. Therefore, we use the term of *pseudo-star* rather than star.

We first address why this probabilistic metric makes sense in motivating application of examining PPINs.The existence of an edge between two nodes is now a random variable and, therefore, identifying a specific structure and calculating its centrality metric becomes harder. As an example, consider Fig. 2. A feasibility condition here for an induced star centered at c to be valid would be that edges (c, l_1) and (c, l_2) exist, while the connection between l_1 and l_2 disappears with a high enough probability

Figure 2: An example of determining the stochastic pseudo-star degree centrality of a given node where the center and leaf nodes are shown in red and blue, respectively. All values on the edges are probability values. For example, the edge between l_1 and n_1 exists with probability 0.5.



We now discuss when the SPSDC problem may be more appropriate than the DSDC problem. First, in the DSDC problem, we must resort to employing thresholds (to remove edges with lower interaction probabilities) or simulation (to generate a random network where each probability appears independently as a Bernoulli random variable) to create the resulting input network to the problem. This affects the applicability in networks such as PPINs where edges may appear or disappear depending on the context, hence motivating the study of a problem with stochastic edge probabilities

This opens up an issue for the independence of the edge existence probabilities. It is true that in many practical applications the assumption of independence is strong. However, it definitely is a step towards adding realism to the model (by incorporating stochasticity), while still being able to devise efficient algorithms to solve the underlying problem. Hence, for the remainder of this work, we will make the assumption that the edges and their probabilities are independent Bernoulli random variables. This is an assumption that has been made before in the context of PPINs. Two proteins are said to interact if and only if at least a pair of their domains interact; and domain interactions have been generally assumed to be independent (see, e.g., Deng et al. (2002)).

We now proceed to the specifics of the SPSDC metric. For a given pseudo-star S_k centered at node k, let L be the set of leaf nodes. Also, let $\alpha \in [0, 1]$ be a user-defined value (please see Prékopa (2013) for details on chance constraints).

Definition 3. Given a pseudo-star S_k, the feasibility condition is defined as

$$\prod_{j \in L} p_{kj} \prod_{i,j \in L:(i,j) \in E} (1 - p_{ij}) \geq 1 - \alpha \tag{1}$$

where the first product term focuses on the probability that edges exist between the center and the leaf nodes and the second product term focuses on the probability of edges existing between two leaf nodes. We can use the log transformation (i.e., a data transformation where each data point is inserted into the logarithm function) to get rid of the multiplication operation in Ineq. (1) and obtain an equivalent expression:

$$\sum_{j \in L} \log(p_{kj}) + \sum_{i,j \in L:(i,j) \in E} \log(1 - p_{ij}) \geq \log(1 - \alpha) \tag{2}$$

In the SPSDC problem, the main objective is to assign each neighbor node to a single pseudo-star element (i.e., either the center or a leaf) which yields the largest probability value. In other words, our goal is to maximize the maximum probability value of the connection between a neighborhood node and the pseudo-star. This offers one potential way to evaluate the centrality of the pseudo-star; different metrics could be applied in the future. We can define the SPSDC problem as follows.

Definition 4. The *stochastic pseudo-star degree centrality* of node i, represented by $\mathcal{D}_i$, is a centrality metric which aims to form an induced pseudo-star S_i centered at i that maximizes the maximum probability value of each neighbor's connection to the pseudo-star, where $\mathcal{D}_i = \max\{\sum_{j \in N(S_i)} \max_{k \in S_i} p_{kj} : S_i$ is an induced pseudo-star satisfying the feasibility condition (1)$\}$.

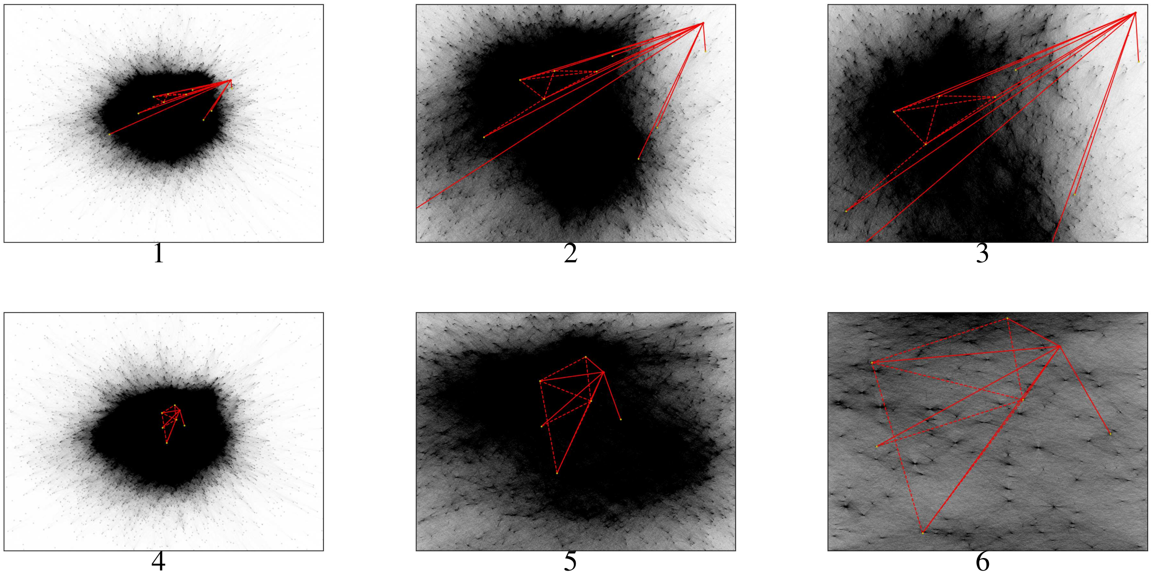
Definition 5. *The stochastic pseudo-star degree centrality problem* aims to identify the node which has the largest stochastic pseudo-star degree centrality in a given network.

Example 2. In Fig. 2, we provide an example to see how the SPSDC metric works, where the probability values are shown on the edges and α is equal to 0.2. Considering node c as the center, we can first create a candidate pseudo-star where node l_1 is the only leaf node (see the figure on the left). In this scenario, the feasibility condition is satisfied since $0.99 \geq 1 - \alpha$ and we obtain an objective of $0.5 + 0.5 + 0.8 = 1.8$. Note that node c is assigned to node l_2 since c provides a stronger connection compared to node l_1 (i.e., 0.8 vs. 0.01).

However, the probability values associated with the edges between the center node and nodes l_1 and l_2 are relatively large. Also, even though nodes l_1 and l_2 share an edge, the corresponding probability value between those two nodes shows that they are highly likely not to interact. Therefore, we can create an alternative induced pseudo-star centered at c where leaf nodes are selected as l_1 and l_2 (see the figure on the right). Such a pseudo-star would still satisfy the feasibility condition (i.e., $0.99 * 0.8 * (1 - 0.01) = 0.8821 > 1 - \alpha$). In addition, it yields a better objective, which is calculated as $0.99 + 0.5 + 0.5 = 1.99$. ▸

It is important to mention that, there cannot be any guarantee that a pseudo-star gives a better deterministic objective (i.e., the largest size of open neighborhood) than the deterministic induced star since the threshold used in the feasibility condition impacts the size of the pseudo-star. Therefore, a fair comparison cannot be made between the DSDC and the SPSDC even if they are associated with the same objective function. However, the goal of each of these problems in our motivating application is to identify essential proteins and, therefore, it may be that each of their solutions helps to diversify the set of proteins that should be investigated to determine their essentiality or confirm the likeliness of certain proteins being essential (i.e., if they appear in both the DSDC and SPSDC problems).

Figure 3: In this example, we show the pseudo-stars with maximum objective function value obtained for a non-essential and an essential proteins in a real-world PPIN, *Saccharomyces cerevisiae*. The network consists of a giant connected component that includes 6,416 nodes out of the 6,418 proteins documented in STRING-DB (Szklarczyk et al., 2015), and 939,997 edges of varying reliability (probability of existence). The networks presented are the full connected component that contains the two proteins (left), a zoomed in perspective (middle) and an even more zoomed in perspective showing the pseudo-star centered at each protein (right). The pseudo-stars are obtained with $\alpha = 0.99$; in other words, they form induced stars with probability 1%. To show the two pseudo-stars, we show in red the center and in yellow the leaves; edges from the center to the leaves are solid, whereas edges connecting two leaves are dashed. The pseudo-star obtained for the essential protein (see 4-6) leads to a higher objective function value (equal to 1293.72) than the value obtained for the pseudo-star centered at the non-essential protein (see 1-3) which is equal to 1002.57.



The advantages of the SPSDC problem compared to the DSDC problem are twofold. First and foremost, it allows us to solve the problem in a PPIN without the need to trim edges based on their probability of existence. In the DSDC problem, a threshold is employed to remove edges below a certain probability of existence. This can be problematic, as certain edges may interact with high enough probabilities just below the threshold and hence get removed; on the other hand, other edges that are just above the threshold are considered as present. As an example to showcase the success of the SPSDC in PPINs, we examine Figure 3. There we show two pseudo-stars obtained for a non-esssential protein (i.e., YBL072C or RPS8A) and for an essential protein (i.e., YAL001C or TFC3) in *Saccharomyces cerevisiae*, which is a species of yeast, with different zooming perspectives in the first three and last three images, respectively. The pseudo-star obtained with the essential protein as the center leads to higher overall objective function than the pseudo-star obtained for the non-essential protein. On the other hand, had we employed a threshold of 60% (i.e., removing all edges with likelihood smaller than 60%), the objective functions of the two stars would be reversed, leading to the non-essential one possessing a higher value.

3 Complexity Discussions

We now discuss the computational complexity of the problem of detecting the node as the center of the stochastic pseudo-star with the maximum-connection probability. Below, we present the decision versions of the SPSDC and DSDC problems where (DEC-) is used to indicate the decision problem (Camur, 2021).

Definition 6. (DEC-Stochastic Pseudo-Star Degree Centrality) Given an undirected graph $G = (V, E)$, probability vector $\overrightarrow{p}$, user-defined value α and a positive real-number ℓ, does there exists an induced pseudo-star S_k centered at any node k such that stochastic pseudo-star degree centrality is at least ℓ?

Definition 7. (DEC-Deterministic Star Degree Centrality) Given an undirected graph $G = (V, E)$ and a positive integer $\mathcal{T}$, does there exist a induced deterministic star $\mathcal{S}_k$ centered at any node k such that the cardinality of the open neighborhood is at least $\mathcal{T}$?

We show that SPSDC problem is $\mathcal{NP}$-complete.

Theorem 1. *DEC-SPSDC is $\mathcal{NP}$-complete.*

Proof. Given a set of nodes, we can verify in polynomial time whether the nodes form an induced pseudo-star satisfying the feasibility condition and the total assignment probability is greater than or equal to ℓ. Thus, the problem is clearly in $\mathcal{NP}$.

Given an instance of DEC-DSDC $< G, \mathcal{T} >$, let us generate an instance of DEC-SPSDC as $< \bar{G}, \ell, \overrightarrow{p}, \alpha >$ where $\bar{G} = G, \ell = \mathcal{T}, \overrightarrow{p} = \overrightarrow{1}, \alpha = 0$. With this formation, one can realize that the DEC-SPSDC solves the DEC-DSDC due to the fact that (i) no leaf can share an edge according to Ineq. (1), (ii) the objective function becomes the maximization of the number of nodes in the open neighborhood. Hence, the proof is relatively straightforward and we can conclude that the problem at hand is $\mathcal{NP}$-complete. □

Camur et al. (2022) discuss that the DSDC problem is solvable in polynomial time on trees by proposing an algorithm running in $O(|E|)$. In the Online Appendix, we provide a proof that a problem *related* to the DEC-SPSDC is $\mathcal{NP}$-complete on trees. In this problem, the *input parameters* for each arc (i, j) are provided as $0 \leq r_{ij} \leq 1$ and then, as part of the problem, are converted into probabilities according to the function $e^{-r_{ij}}$. Further, each arc is given a weight, w_{ij}, and the objective function is based on the maximimum arc weight that connects a node in the neighorhood to the star (note that the weighted version of the DSDC problem can be solved with a slight modification of the algorithm of Camur et al. (2022)). This allows for a reduction from the knapsack problem to this problem related to the DEC-SPSDC, similar to a reduction of Ahmed and Atamtürk (2011) where the problem they are studying has inputs raised to an exponential function.

4 Mathematical Programming Formulation

In this section, we propose an optimization model to solve the SPSDC problem that extends the improved formulation proposed by Camur et al. (2022) for the DSDC problem. The model contains three sets of binary variables: i) x_i is 1 if node i is selected as the center; 0 otherwise, ii) y_i is 1 if node i is selected as a leaf node; 0 otherwise, and iii) z_{ij} is 1 if pseudo-star element i covers node j in the pseudo-stars open neighborhood; 0 otherwise. The formulation is:

$$\textbf{IP:}\ \max \sum_{(i,j)\in E} p_{ij} z_{ij} \tag{3a}$$

$$s.t.\ x_i + y_i + \sum_{j\in N(i)} z_{ji} \leq 1 \qquad \forall i \in V \tag{3b}$$

$$z_{ij} \leq x_i + y_i \qquad \forall (i,j) \in E \tag{3c}$$

$$y_i \leq \sum_{j\in N(i)} x_j \qquad \forall i \in V \tag{3d}$$

$$\sum_{i\in V} x_i = 1 \tag{3e}$$

$$\sum_{(i,j)\in E} \log(p_{ij}) x_i y_j + \sum_{i<j:(i,j)\in E} \log(1-p_{ij}) y_i y_j \geq \log(1-\alpha) \tag{3f}$$

$$x_i, y_i \in \{0,1\} \qquad \forall i \in V \tag{3g}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in E \tag{3h}$$

The objective function (3a) maximizes the total probability of neighborhood assignments. Constraints (3b) indicate that a node i can be either i) the center, ii) a leaf or iii) selected in the open neighborhood and assigned to a node j that has an edge into i. For case (iii) to hold, node j has to be connected to the center or a leaf, which is guaranteed by Constraints (3c). Note that case (iii) ensures the unique assignment of a neighbor node to the pseudo-star. In fact, when a node j belongs to the open neighborhood of the pseudo-star, the objective function ensures the selection of the edge between it and an element of the pseudo-star with the highest probability, thereby computing the centrality measure of interest. Constraints (3d) make sure that each leaf node is connected to the center node. Constraint (3e) enforces the model to select a single center for the pseudo-star. Constraint (3f) states that the pseudo-star selected satisfies the feasibility condition. Lastly, Constraints (3g)-(3h) enforce the binary conditions on the variables.

The model proposed is a non-linear binary optimization problem where the numbers of variables and constraints are both $O(|E|)$. Thus, it remains as a challenging problem to solve even if the given graph is small and sparse. However, we can linerailize Constraints (3f) with the well-known McCormick inequalities. We will introduce variables to represent the product of binary variables as follows: $n_{ij} = x_i y_j, \forall (i,j) \in E$ and $m_{ij} = y_i y_j, \forall i < j : (i,j) \in E$. We then obtain the following linear model, which is equivalent to IP.

$$\textbf{LIP:}\ \max \text{(3a)} \tag{4a}$$

$$s.t.\ \text{(3b)} - \text{(3c)} - \text{(3d)} - \text{(3e)} - \text{(3g)} - \text{(3h)}$$

$$\sum_{(i,j)\in E} \log(p_{ij}) n_{ij} + \sum_{i<j:(i,j)\in E} \log(1-p_{ij}) m_{ij} \geq \log(1-\alpha) \tag{4b}$$

$$n_{ij} \geq x_i + y_j - 1 \qquad \forall (i,j) \in E \tag{4c}$$

$$m_{ij} \geq y_i + y_j - 1 \qquad \forall (i,j) \in E \tag{4d}$$

$$n_{ij} \in \{0,1\} \qquad \forall (i,j) \in E \tag{4e}$$

$$m_{ij} \in \{0,1\} \qquad i < j : \forall (i,j) \in E \tag{4f}$$

We first note that since $0 < p_{ij} \leq 1$, each $\log(p_{ij})$ value is non-positive. This implies that whenever m_{ij} and / or n_{ij} variables takes a positive value, then the left hand side (LHS) of Constraint (4b) decreases. As a result, assigning a positive value for either variable when it is not 'necessary' would only strain the feasibility condition and does not impact the objective function. Therefore, the McCormick upper bound (UB) constraints (e.g., $m_{ij} \leq x_i, \forall i \in V$ and $m_{ij} \leq y_j, \forall j \in V$) are not required during the linear transformation and we omit those constraints. Also, we will be using the same n_{ij} and m_{ij} variables whenever McCormick inequalities are introduced for the same transformations. Although we end up with a linear model, the number of variables and constraints are still bounded by $O(|E|)$. We propose a decomposition algorithm to solve the model at a scale, which is discussed in the next section.

5 Solution Methodology

We will be using Benders Decomposition (BD) as our solution method. BD is a popular solution technique that is helpful to solve large-scale optimization problems carrying certain block structures (Benders, 1962). In BD literature, there are two different ways to implement the algorithm: i) classical Benders where the cuts are added to the master problem (MP) in an iterative manner and the branch$\&$bound (BB) tree restarts at each iteration, and ii) modern Benders where Benders cuts are added on-the-fly and there exists a single BB tree. We will be testing both Benders implementations during our computational experiments and the reader can access more information on modern Benders (i.e., branch-and-Benders) in Fischetti et al. (2016).

In addition, we will be examining two different decomposition frameworks. Our first framework, named three-phase decomposition, will remove constraints (3b)-(3c)-(3h)-(4b)-(4c)-(4d)-(4e) and (4f) to design an MP whose aim is to identify a candidate pseudo-star. We then obtain two different sub problems (SP) where we focus on the feasibility (i.e., Constraints (4b)-(4c)-(4d)-(4e) and (4f)) and the open neighborhood (i.e., Constraints (3b)-(3c) and (3h)) components of the problem separately in that order. This is because the feasibility component has no impact on the objective.

At every candidate solution, we first check the feasibility condition. If the condition does not hold, then we eliminate the current solution via either Benders feasibility cuts or logic-based Benders cuts (LBBCs). If the pseudo-star is feasible, then we proceed to the next SP to check whether an optimality cut that aims to approximate the objective value (i.e., the open neighborhood with the maximum total probability assignment) can be generated.

Below we present the MP for the three-phase framework (i.e., TMP) without the Benders cuts, which will be presented shortly and be incorporated into the MP at every iteration as needed, where θ represents the estimation of the true objective. Note that variable θ is associated with the optimality cuts whose SP provides a non-increasing UB on its value.

$$\textbf{TMP} = \max_{\theta} \{\theta : \text{(3d)} - \text{(3e)} - \text{(3g)}, \theta \leq UB, OptimalityCuts(\theta), FeasibilityCuts\}$$

Our next decomposition framework, called two-phase decomposition, will remove only Constraints (3b)-(3c) and (3h) and keep all the feasibility-related constraints in MP. Even though such a design would increase the complexity of MP due to the high number of variables, constraints and non-zero coefficients, we expect to decrease the number of Benders cut added considerably, thereby expecting a faster convergence to optimal. The corresponding MP for the two-phase decomposition can be presented as:

$$\textbf{MP} = \max_{\theta} \{\theta : \text{(3d)} - \text{(3e)} - \text{(3g)} - \text{(4b)} - \text{(4c)} - \text{(4d)} - \text{(4e)} - \text{(4f)}, \theta \leq UB, OptimalityCuts(\theta)\}$$

The main difference between these decomposition frameworks is that while we only generate optimality cuts in the two-phase version, both optimality and feasibility cuts must be generated in the three-phase decomposition framework. Whenever we discuss adding optimality cuts to the problem, we will be referring to adding them to the MP of the framework being discussed (e.g., the problem that will be iteratively solved).

Lastly, we will also be using the *automatic Benders* (AB), which is the implementation of the BD algorithm in CPLEX (IBM, 2020). AB supports multiple SP generation, has its own acceleration techniques, as well as combining both classical and modern Benders implementations (Bonami et al., 2020). Hence, it stands as an important state-of-the art algorithm. We will now be introducing the different types of feasibility cuts that can be used in the three-phase decomposition framework (see Sections 5.1 and 5.2).

5.1 Benders Feasibility Cuts

An important observation is that the McCormick variables (i.e., n_{ij} and m_{ij}) used to linearize IP can be relaxed and both take binary values, with a similar logic as to why it is not necessary to include the UB constraints.

Proposition 1. *Variables n_{ij} and m_{ij} take binary values when they are relaxed in model LIP.*

Proof. Without loss of generality (WLOG), let us examine n_{ij}. If both x_i and y_j take the value of one, then $n_{ij} = 1$ by Constraint (4c). If (a) they are both zero or (b) either of them is zero, then n_{ij} becomes free. However, since increasing n_{ij} would only decrease the LHS of Constraint (4b), the model would not prefer to assign a non-negative value for n_{ij}. Even if there could be a degenerate case where Constraint (4b) is satisfied by assigning n_{ij} a positive value, we can set $n_{ij} = 0$ and obtain the same objective value. As a result, we obtain a binary optimal solution when variable n_{ij} is relaxed. □

Proposition (1) enables us to generate traditional Benders feasibility cuts. The second important observation is that, the feasibility condition has two components where we look at the connections between the center and the leaf nodes (i.e., $x_i y_j$), and among the leaf nodes (i.e., $y_i y_j$). Hence, one can generate two different cuts as i) a *local* feasibility cut where infeasibility occurs across both the center and leaf nodes (i.e., $\prod_{j \in L} p_{kj} \prod_{i,j \in L:(i,j) \in E} (1 - p_{ij}) \geq 1 - \alpha$), or ii) a *global* feasibility cut where infeasibility occurs directly within the set of leaf nodes (i.e., $\prod_{i,j \in L:(i,j) \in E} (1 - p_{ij}) \geq 1 - \alpha$). The reason we refer to the latter as *global* is that it is applicable to any potential center that could be connected to that set of leaf nodes.

We first show the traditional Benders local feasibility problem. Let δ, ν_{ij} and μ_{ij} be the penalty variables defined for Constraints (4b), (4c), and (4d), respectively. They aim to approximate how much we should perturb the current fixed solution ensure it satisfies the SP constraints.

BLF:

$$\min \delta + \sum_{(i,j)\in E} \nu_{ij} + \sum_{i<j:(i,j)\in E} \mu_{ij} \tag{5a}$$

$$s.t. \sum_{(i,j)\in E} \log(p_{ij}) n_{ij} + \sum_{i<j:(i,j)\in E} \log(1-p_{ij}) m_{ij} + \delta \geq \log(1-\alpha) \tag{5b}$$

$$n_{ij} + \nu_{ij} \geq \overline{x}_i + \overline{y}_j - 1 \qquad \forall (i,j) \in E \tag{5c}$$

$$m_{ij} + \mu_{ij} \geq \overline{y}_i + \overline{y}_j - 1 \qquad \forall i < j : (i,j) \in E \tag{5d}$$

$$n_{ij}, \nu_{ij} \in \mathbb{R}_+ \qquad \forall (i,j) \in E \tag{5e}$$

$$m_{ij}, \mu_{ij} \in \mathbb{R}_+ \qquad \forall i < j : (i,j) \in E \tag{5f}$$

$$\delta \in \mathbb{R}_+ \tag{5g}$$

We take the dual of the model, where dual variables ρ, υ_{ij}, and Υ_{ij} correspond to Constraints (5b), (5c), and (5d), respectively.

DBLF:

$$\max \ \log(1-\alpha)\rho + \sum_{(i,j)\in E} (\overline{x}_i + \overline{y}_j - 1)\upsilon_{ij} + \sum_{i<j:(i,j)\in E} (\overline{y}_i + \overline{y}_j - 1)\Upsilon_{ij} \tag{6a}$$

$$s.t. \ \log(p_{ij})\rho + \upsilon_{ij} \leq 0 \qquad \forall (i,j) \in E \tag{6b}$$

$$\log(1-p_{ij})\rho + \Upsilon_{ij} \leq 0 \qquad \forall i < j : (i,j) \in E \tag{6c}$$

$$0 \leq \rho \leq 1 \tag{6d}$$

$$0 \leq \upsilon_{ij} \leq 1 \qquad \forall (i,j) \in E \tag{6e}$$

$$0 \leq \Upsilon_{ij} \leq 1 \qquad \forall i < j : (i,j) \in E \tag{6f}$$

The following is what we call a local Benders feasibility cut that can be added into MP to eliminate infeasible candidate solution.

$$\log(1-\alpha)\rho + \sum_{(i,j)\in E} \upsilon_{ij}(x_i + y_j - 1) + \sum_{i<j:(i,j)\in E} \Upsilon_{ij}(y_i + y_j - 1) \leq 0 \tag{7}$$

However, if the infeasibility takes place because of the leaf nodes selected even without taking the center node into consideration, then we solve a smaller size LP to obtain a global feasibility cut. WLOG, let us use the same penalty variables (i.e., δ and ν_{ij}) and define the following feasibility problem.

BGF:

$$\min \delta + \sum_{i<j:(i,j)\in E} \mu_{ij} \tag{8a}$$

$$s.t. \quad \sum_{i<j:(i,j)\in E} \log(1-p_{ij}) m_{ij} + \delta \geq \log(1-\alpha) \tag{8b}$$

$$m_{ij} + \mu_{ij} \geq \overline{y}_i + \overline{y}_j - 1 \qquad \forall i < j : (i,j) \in E \tag{8c}$$

$$m_{ij}, \mu_{ij} \in \mathbb{R}_+ \qquad \forall i < j : (i,j) \in E \tag{8d}$$

$$\delta \in \mathbb{R}_+ \tag{8e}$$

WLOG, let variables ρ and Υ be the dual variables corresponding to Constraints (8b) and (8c). The dual of BGF can be presented as follows.

DBGF:

$$\max \ \log(1-\alpha)\rho + \sum_{i<j:(i,j)\in E} (\overline{y}_i + \overline{y}_j - 1)\Upsilon_{ij} \tag{9a}$$

$$s.t. \ \log(1-p_{ij})\rho + \Upsilon_{ij} \leq 0 \qquad \forall i < j : (i,j) \in E \tag{9b}$$

$$0 \leq \rho \leq 1 \tag{9c}$$

$$0 \leq \Upsilon_{ij} \leq 1 \qquad \forall i < j : (i,j) \in E \tag{9d}$$

In the case of a global feasibility cut, we obtain a tighter feasibility cut than Ineq. (7) since it is not tied to any center node. The constraint is:

$$\log(1-\alpha)\rho + \sum_{i<j:(i,j)\in E} (y_i + y_j - 1)\Upsilon_{ij} \leq 0 \tag{10}$$

Note that both Benders feasibility cuts introduced are associated with dual solutions, which are mostly fractional values. Our preliminary results indicate that such feasibility cuts are not able to yield a quick convergence even in small scale instances. Hence, in the following section, we examine LBBCs. In Section 7.3, we test both sets of cuts and discuss their impacts on the solution time.

5.2 Logic-Based Benders Cuts

Logic-based Benders Decomposition (LBBD) is formally introduced by Hooker and Ottosson (2003). It distinguishes itself from the traditional BD by using the *inference* dual rather than LP duality to generate the Benders cuts so as to eliminate the infeasible MP solutions. Although it has been predominantly used in scheduling problems (Roshanaei et al., 2017; Guo et al., 2021; Zhang et al., 2022), it is recently adapted to route planning (Kloimüllner and Raidl, 2017), network interdiction (Enayaty-Ahangar et al., 2019), network design (Naderi et al., 2020), and assembly line balancing (Zohali et al., 2021) problems as well.

For a fixed pseudo-star S_k centered at node k with a set of leaf nodes denoted by L, if the feasibility condition is not satisfied, then we can generate a generic *no-good* cut that aims to change the current solution by removing a single leaf node from S_k. Note that the cut should not take into consideration adding another leaf node since adding a new leaf node would only decrease the LHS of (2). Hence, we define the following LBBC:

$$\sum_{j\in L} y_j \leq (|L|-1)x_k + |L|(1-x_k) \tag{11}$$

Theorem 2. *The LBB feasibility cut* (11) *is valid.*

Proof. To prove that a LBBC is valid, we show that (i) the constraint cuts off the current master solution since it is infeasible, and (ii) it does not eliminate a global feasible solution. We use the same methodology to prove the similar theorems presented in the rest of this paper.

Note that if node k is selected as the center node (i.e., $x_k = 1$), then the right hand side (RHS) implies that at least one of the leaf nodes in L of S_k must be turned off, thereby eliminating the current solution. Otherwise, the center node alternates without enforcing any restriction on the nodes in L, thus, the infeasible solution is eliminated. As a result, pseudo-star S_k is guaranteed to be removed from consideration.

In the following iterations, when we obtain a new candidate pseudo-star (feasible or not), if it is centered at a node different than k, then it becomes clear that the cut does not eliminate a feasible solution since the RHS becomes the aggregating of the binary restrictions for the leaf nodes, in other words, a trivial constraint. If k is the candidate center node in an alternative $S_k^{'}$, since the RHS changes at most one leaf node, it guarantees that $S_k^{'} \neq S_k$. It also makes sure that the only solution removed is S_k, hence, no global feasible solution is removed. □

Note that cut (11) does not aggressively change the current solution and is not effective in general due to the fact that it only targets to eliminate S_k rather than understanding the subset of nodes at the 'root' of its infeasibility. Thus, we can design an integer SP with a fixed center k to check if we can eliminate more leaf nodes for S_k to be feasible.

$$\textbf{WF} := \max_{\vec{y} \in \{0,1\}} \left\{ \sum_{j \in L} y_j : \sum_{j \in L} p_{kj} \bar{x}_k y_j + \sum_{i<j: i,j \in L} (1 - p_{ij}) y_i y_j \geq \log(1 - \alpha) \right\}$$

The Model WF aims to identify the maximum number of leaf nodes that could be selected in L to obtain a feasible pseudo-star structure via a knapsack-type constraint. Since its nonlinearities come from the product of two binary variables, we can use the McCormick inequalities. We then obtain an equivalent linear formulation:

$$\textbf{LWF} := \max_{\vec{y}, \vec{m} \in \{0,1\}} \left\{ \sum_{j \in L} y_j : \sum_{j \in L} p_{kj} \bar{x}_k y_j + \sum_{i<j: i,j \in L} (1 - p_{ij}) m_{ij} \geq \log(1 - \alpha), m_{ij} \geq y_i + y_j - 1, \right.$$
$$\left. \forall i < j : i, j \in L \right\}$$

Let δ^* be the optimal solution to LWF. Then we define a new LBBC.

$$\sum_{i \in L} y_j \leq \delta^* x_k + |L|(1 - x_k) \qquad (12)$$

Theorem 3. *The LBB feasibility cut* (12) *is valid.*

Proof. Similar to Theorem 2, the second component of the RHS (i.e., $|L|(1 - x_k)$) guarantees that the current infeasible solution is cut off and no global feasible solution is eliminated. Therefore, we examine the non-trivial scenario when $x_k = 1$.

The objective aims to select as many leaf node as possible while ensuring the feasibility by the constraint defined in WF. First, it makes sure that at least one leaf node is removed from the candidate pseudo-star, hence, $\delta^* \leq |L| - 1$. As a result, the cut removes the current infeasible solution. Second, it implies that any alternative candidate pseudo-star $S_k^{'}$ that has more than δ^* leaf nodes is infeasible. Since the cut prevents $S_k^{'}$ from having more δ^* leaf nodes and all solutions having less than or equal to δ^* leaf nodes are still feasible, it does not cut off any global feasible solution. □

This cut is stronger than cut (11) since we have that $\delta^* \leq |L| - 1$. However, it still depends on the selection of x_k as the center meaning that once the center node is changed, the cut does not help. Therefore, we name (12) as a local LBBC. The $|L|$ term plays the role of a big-M, hence, the question becomes if we can further improve cut (12).

Based on the same argument provided in the previous section, if the infeasibility occurs directly due to the connections between the leaf nodes selected (i.e., $\prod_{i,j \in L}(1 - p_{ij}) < 1 - \alpha$), then we can focus on a smaller size IP model to obtain a better cut. This leads to a different integer SP, as well as a more general and stronger cut.

$$\textbf{SF} := \max_{\vec{y} \in \{0,1\}} \left\{ \sum_{j \in L} y_j : \sum_{i<j: i,j \in L} (1 - p_{ij}) y_i y_j \geq \log(1 - \alpha) \right\}$$

Similar to model WF, we have non-linear terms and should use the McCormick inequalities.

$$\textbf{LSF} := \max_{\vec{y}, \vec{m} \in \{0,1\}} \left\{ \sum_{j \in L} y_j : \sum_{i<j: i,j \in L} (1 - p_{ij}) m_{ij} \geq \log(1 - \alpha), m_{ij} \geq y_i + y_j - 1, \forall i < j : i, j \in L \right\}$$

Let Δ^* be the optimal objective of LSF. Then, we define the following LBBC, which does not depend on variable x and is called global LBBC.

$$\sum_{j \in L} y_j \leq \Delta^* \tag{13}$$

Theorem 4. *The LBB feasibility cut* (13) *is valid.*

Proof. LSF guarantees that $\Delta^* < |L|$ as a result of which the current infeasible solution is eliminated. The cut also carries the information of how many nodes in L can be selected by any pseudo-star. Note that it does not necessarily guarantee feasibility since according to the selection of the center node, we can still face a feasibility problem. Yet, it makes sure that no global feasible solution is removed since in any scenario having more than Δ^* leaf nodes from L is directly infeasible regardless of which node is selected as the center. □

One can realize that all LBB feasibility cuts (13) can be pre-populated and added into LIP. Yet, since the number of those cuts is bounded by $O(|V| 2^{|E|})$, it is not practical to incorporate all such cuts in advance and we generate them on the fly in our solution method.

5.3 Optimality Cuts

Once the pseudo-star fixed satisfies the feasibility condition, we proceed to solve an SP to generate an optimality cut. Given a fixed solution $(\vec{x}, \vec{y})$, we define the following primal problem.

$$
\begin{aligned}
\phi(\bar{x},\bar{y}): \quad \max \quad & \sum_{(i,j)\in E} p_{ij} z_{ij} && && \text{(14a)}\\
s.t. \quad & \sum_{j\in N(i)} z_{ji} \le 1 - \bar{x}_i - \bar{y}_i, && \forall i \in V && \text{(14b)}\\
& z_{ij} \le \bar{x}_i + \bar{y}_i, && \forall (i,j) \in E && \text{(14c)}\\
& z_{ij} \in \{0,1\}, && \forall (i,j) \in E && \text{(14d)}
\end{aligned}
$$

Here we can use the LP duality to generate the dual formulation by relaxing variables z_{ij}. The relaxation of variable z_{ij} produces binary solutions when passing an incumbent solution to $\phi(\bar{x},\bar{y})$ (see Proposition 2).

Proposition 2. *Variables z_{ij} take binary values when they are relaxed.*

Proof. Given node i, either y_i or x_i can take the value of one, hence, each z_{ij} is upper bounded by one via Constraints (14c). In addition, we ensure the unique assignment by Constraint (14b). Due to the fact that $p_{ij} \ge 0, \forall (i,j) \in E$ and the objective is maximization, each z_{ij} is guaranteed to set as its upper bound (i.e., 1) if at least one of $j \in N(i)$ becomes the member of the pseudo-star; zero otherwise. Hence, we can conclude that the relaxation of the model returns binary solution. □

Let β_i and γ_{ij} be the dual variables corresponding to Constraints (14b) and (14c), respectively. The dual of $\phi(\bar{x},\bar{y})$ is presented as follows.

$$
\Phi(\bar{x},\bar{y}) := \min_{\beta\ge 0,\gamma\ge 0} \left\{ \sum_{i\in V} (1-\bar{x}_i-\bar{y}_i)\beta_i + \sum_{(i,j)\in E} (\bar{x}_i+\bar{y}_i)\gamma_{ij} : \beta_i + \gamma_{ji} \ge p_{ji}, \forall (j,i) \in E \right\}
$$

The constraint set of the dual formulation $\Phi(\bar{x},\bar{y})$ does not depend on the fixed MP solution. Also, the constraint set is always closed and bounded, which implies that we are not concerned about the feasibility of the problem. Whenever a violated solution is identified, we generate the following optimality cut and add into MP.

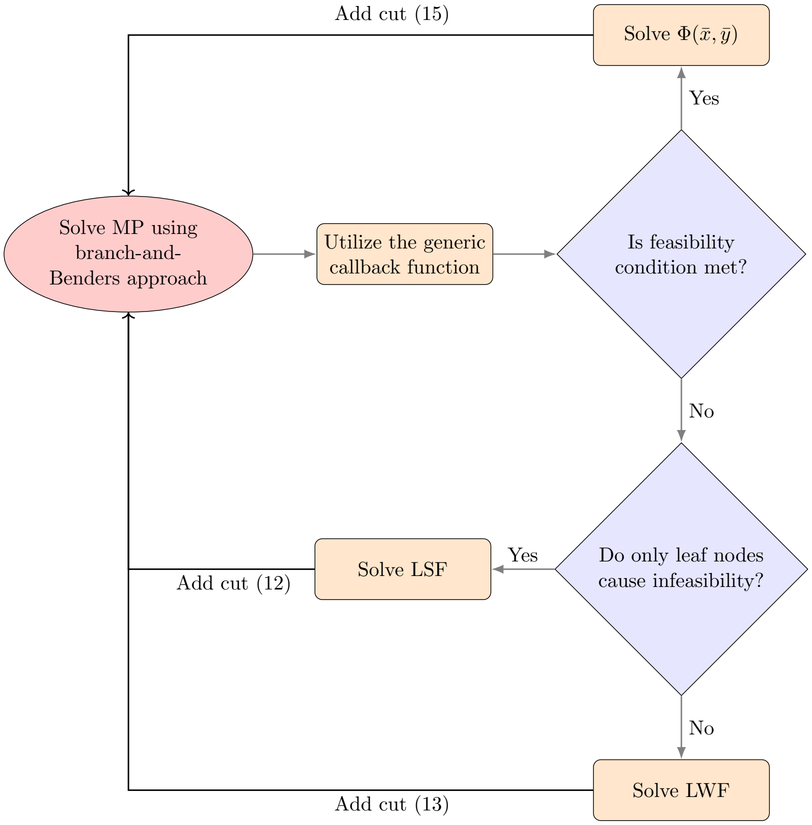
$$
\theta \le \sum_{i\in V} \bar{\beta}_i (1 - x_i - y_i) + \sum_{(i,j)\in E} \bar{\gamma}_{ij} (x_i + y_i) \tag{15}
$$

We refer the interested reader to Fig. 5 for details on three-phase modern Benders implementation . Note that here we show LBBCs (i.e., constraints (12) and (13)) as feasibility cuts. The only change to focus on Benders Feasibility Cuts is the type of SP solved and cut generated in the lower portion of the figure.

6 Algorithmic Enhancements

In this section, we present the acceleration techniques that we adapt to speed up our Benders implementations. We note that any technique applicable to the full LIP is directly adopted to make a fair comparison on our computational testing.

Figure 5: The illustration of the Benders Decomposition algorithm including Logic-based Benders cuts.


Add cut (15)
Solve $\Phi(\bar{x}, \bar{y})$
Yes
Solve MP using branch-and-Benders approach
Utilize the generic callback function
Is feasibility condition met?
No
Yes
Solve LSF
Do only leaf nodes cause infeasibility?
Add cut (12)
No
Solve LWF
Add cut (13)

6.1 Algorithmic Approach for Optimality Cuts

We observe that $\Phi(\bar{x}, \bar{y})$ can be solved by a direct algorithm for it rather than utilizing a commercial solver. More importantly, we can separate the problem over each node i thereby enabling ourselves to generate multiple cuts at every iteration. We first show how to divide $\Phi(\bar{x}, \bar{y})$ over each node as $\Phi_i(\bar{x}, \bar{y})$ and then propose an algorithm that identifies the optimal solution for $\Phi_i(\bar{x}, \bar{y})$ for a given i. This algorithm works for both incumbent and fractional solutions, i.e., it follows Modern BD. The problem over node i is:

$$\Phi_i(\bar{x}, \bar{y}) := \min_{\beta_i, \gamma \geq 0} \left\{ (1 - \bar{x}_i - \bar{y}_i)\beta_i + \sum_{j \in N(i)} (\bar{x}_j + \bar{y}_j)\gamma_{ji} : \beta_i + \gamma_{ji} \geq p_{ji}, \forall j \in N(i) \right\}$$

We first restate the objective function of the MP as $\sum_{i \in V} \theta_i$ and separate cut (15) over each node as shown below.

$$\theta_i \leq \bar{\beta}_i(1 - x_i - y_i) + \sum_{j \in N(i)} \bar{\gamma}_{ji}(x_j + y_j), \forall i \in V \tag{16}$$

In order to solve each $\Phi_i(\bar{x}, \bar{y})$, we follow the following procedure. First, for the sake of simplicity, let us assume that for a fixed node i, every node in $N(i)$ is indexed from 1 to l where $p_{1i} \geq p_{2i} \geq \cdots \geq p_{li}$. If there exists an index j such that $\sum_{k=1}^{j}(x_k + y_k) \geq (1 - x_i - y_i)$, then we identify the minimum j, denoted as j^*, satisfying the inequality and set $\beta_i = p_{j^*i}$; otherwise, we set $\beta_i = 0$. Then, we assign γ_{ji} as $\max\{p_{ji} - \beta_i, 0\}, \forall j \in N(i)$.

Proposition 3. *The procedure proposed returns the optimal solution for $\Phi_i(\bar{x}, \bar{y})$.*

Proof. Suppose no j^* exists; in other words, $(1 - x_i - y_i) > \sum_{k=1}^{l}(x_k + y_k)$, then setting $\beta_i = 0$ and $\gamma_{ji} = p_{ji}$ makes each constraint tight (i.e., a feasible solution) and the objective becomes $\sum_{k=1}^{l} \gamma_{ki}(x_k + y_k)$. Two cases: i) increasing both β_i and γ_{ji}, and ii) increasing γ_{ji} are considered trivial since they only increase the objective value. It is also clear that decreasing both variables together or individually causes infeasibility.

WLOG, the non-trivial case is that we increase β_i and decrease each γ_{ji} by $\epsilon : 0 < \epsilon \leq \min_{j \in N(i)} \gamma_{ji}$. After this change, the feasibility is preserved since the LHS of each constraint remains the same. However, the objective value increases due to the fact that:

$$\epsilon(1 - x_i - y_i) > \epsilon \sum_{k=1}^{l}(x_k + y_k) \Rightarrow \epsilon(1 - x_i - y_i) + \sum_{k=1}^{l}(p_{ji} - \epsilon)(x_k + y_k) > \sum_{k=1}^{l}(p_{ki})(x_k + y_k)$$

Hence, the initial setting guarantees to produce the optimal solution if we cannot identify an index j^*.

Now, suppose there exists a j^* and we assign $\beta_i = p_{j^*i}$, $\gamma_{ji} = \max\{p_{ji} - \beta_i, 0\}$. Since we assume that the probability values are in descending order, each constraint $\beta_i + \gamma_{ki} \leq p_{ki}$ where $k > j$ is strictly satisfied and the rest of the constraints become tight. Thus, the solution proposed is feasible. The objective value, on the other hand, is $\beta_i(1 - x_i - y_i) + \sum_{k=1}^{j^*-1} \gamma_{ki}(x_k + y_k)$. The trivial cases presented above stay valid except being able to decrease the value of β_i.

We have two alternatives. First, we increase β_i and decrease $\gamma_{ki}, \forall k < j^*$ by $\epsilon : 0 < \epsilon \leq p_{(j^*-1)i} - p_{j^*i}$. In this scenario, the LHSs of the loose and tight constraints increase and stay the same, respectively. Therefore, the solution is still feasible. Since $(1 - x_i - y_i) > \sum_{k=1}^{j^*-1}(x_k + y_k)$, the total objective increases for the same reason explained above. Second, we identity ϵ such that $0 < \epsilon \leq p_{(j^*+1)i}$ to decrease β_i and increase $\gamma_{ki}, \forall k < j$. However, now γ_{j^*i} has to be non-negative and takes the value of ϵ to ensure the feasibility. Since $\sum_{k=1}^{j^*}(x_k + y_k) \geq (1 - x_i - y_i)$, we cannot improve the objective as a result of which we conclude that the solution proposed is optimal. □

6.2 Greedy Heuristic and Warm-Start

Our preliminary experiments in the SPSDC problem indicates that our selected commercial solver, CPLEX, has difficulty in determining an initial feasible solution, as well as improving the optimality gaps in both solving the LIP directly and solving the problem via BD. Therefore, we design a greedy heuristic that produces an induced pseudo-star for every node in order to test the impact of warm-start (see Alg. 1).

Given a node i and pseudo-star S_i centered at i, let uncovered (i.e., nodes that are not yet covered by an element in S_i) first and second neighborhood nodes of i be represented by $\mathcal{R} \subseteq \bar{N}^2(i) \cup N(i) \setminus \cup_{j \in S_i} N(j)$. We then define $\overline{\mathcal{R}}$ as the complement of $\mathcal{R}$. We let h_j be the index of the element in S_i that is assigned to a neighbor node j in $\mathcal{R}$. For a node $j \in N(i)$, we define u_j as the contribution of node j as the total increase in the objective in case it is selected as a leaf node where $u_j := \sum_{k \in \mathcal{R} \cap N(j)} p_{jk} + \sum_{k \in \overline{\mathcal{R}} \cap N(j)} \max\{p_{jk} - p_{h_j j}, 0\}$. Finally, we let ζ_j be the total probability value that node j contributes into S_i if j is selected as a leaf node (i.e., $\zeta_j = p_{ij} \prod_{k \in S_i \setminus \{i\}}(1 - p_{kj})$).

Given a node i, the heuristic identifies the candidate leaf node that has the highest weighted function $\big($i.e., $w_j = u_j \zeta_j, j \in C\big)$ in candidate leaf nodes and the node is added into S_i as long as it does not violate the feasibility condition. If it is violated, then the node is removed from C. Once we obtain one

Algorithm 1: Greedy Heuristic

```
Input: $i \in V$
$S_i \leftarrow \{i\}$;
$C \leftarrow N(i)$;    #candidate leaf nodes
$\pi = 1$;    # total probability of $S_i$
$z_{ij} = 1, \forall j \in N(i)$;    # initially assign the center to every node in $N(i)$
$\mathcal{R} = \bar{N}^2(i)$
while $C \neq \emptyset$ do
    $j^* = \underset{j \in C}{\operatorname{argmax}}\, u_j \zeta_j$;
    if $\zeta_j \pi < 1 - \alpha$ then
        $C \leftarrow C \setminus j^*$;
    else
        $S_i \leftarrow S_i \cup \{j^*\}$;
        $z_{ij^*} = 0$;
        $C \leftarrow C \setminus j^*$;
        $\mathcal{R} \leftarrow \bar{N}^2(i) \setminus N(j^*)$;
        $\pi = \zeta_{j^*} \pi$;
        for $k \in N(j^*)$ do
            if $\exists h_k \in S_i : z_{h_k k} = 1$ then
                if $p_{h_k k} < p_{j^* k}$ then
                    $z_{h_k k} = 0$;
                    $z_{j^* k} = 1$;
                else
                    continue;
            else
                $z_{j^* k} = 1$;
return $S_i, \vec{z}$
```

candidate pseudo-star centered at each node, we evaluate the objective value of each (i.e., $\sum_{(i,j) \in E} p_{ij} z_{ij}$) and warm-start (WS) both the LIP and the MP via the best solution. It is important to mention that, when using the classical Benders implementation, we start the WS process with the greedy solution proposed, and then in the following iterations, the MP is warm-started via the solution produced in the previous iteration.

We note that in future studies of the SPSDC problem, it may be of interest to conduct worst-case scenario analysis to determine approximation guarantees for the proposed greedy heuristic (Sun and Sharkey, 2017; Vogiatzis and Camur, 2019). In addition, employing iterative heuristics (Lozano et al., 2011; Absi et al., 2015), optimization-based heuristics (Camur et al., 2021; Averbakh and Pereira, 2021), and meta-heuristics (Pellerin et al., 2020; Karimi-Mamaghan et al., 2022) could be alternative ways to improve the initial solutions fed into the model. Lastly, it might be of interest to perform nonparametric statistical tests for potential evolutionary algorithms (García et al., 2009; Derrac et al., 2011)

6.3 Valid Inequalities

While we aim to help the solver with improving the primal bounds via warm-start, it is also important to use valid inequalities to help with the dual bounds. With this purpose, we use the heuristic algorithm proposed by Camur et al. (2022) and adapt it to our problem. While more details and pseudocode can be found in Camur et al. (2022), here we informally explain the heuristic and our slight modification.

First, we note that the heuristic remains as a valid UB even if we are concerned with a deterministic objective in the SPSDC problem. For a given node i and a candidate induced star S_k, let δ_{S_k} be the UB. We initially set $\delta_{S_k} = |N(i) \cup \bar{N}^2(i)|$. The heuristic identifies each node j in $N(i)$, which creates a unique path to a node in $\bar{N}^2(i)$ and decreases δ_{S_k} for each j identified.

Once the bound is obtained, we sum up the δ_{S_i} largest probability values in the set $\mathcal{V}_i = \{p_{ij} : j \in N(i)\} \cup \{p_{jk} : j \in N(i), k \in N(i) \cup \bar{N}^2(i)\}$ and represent the summation by τ_i. Then, the following is a valid inequality which can be placed in MP.

$$\sum_{i \in V} \theta_i \leq \tau_i x_i \tag{17}$$

Note that we use the same bound for the objective function (4a) in LIP. In addition, since we are looking for a unique assignment between a neighbor node and a pseudo-star element, the contribution of each node to the objective is bounded above by the largest probability connection. The following is a valid inequality that can be only used in MP:

$$\theta_i \leq \max_{j \in N(i)} p_{ji}, \forall i \in V \tag{18}$$

6.4 Separation of Fractional Solutions

One of the benefits in using modern BD is that fractional solutions may be separated during the BB process thereby generating what is called a user cut. Our preliminary experiments indicate that the initial MP quickly ends up being overloaded with feasibility cuts, thus limiting its ability to solve the problem. In addition, having fractional values for the center variable increases the difficulty of the feasibility separation problem. Therefore, in our implementation, fractional solutions are only separated when all variables x_i are binary and the leaf variables are fractional. This implies that we generate a user cut solely when $\exists! i \in V : x_i = 1$ and $\exists j \in N(i) : 0 < y_j < 1$. Otherwise, we let the solver continue its branching process.

When it comes to separating fractional y solutions, we adapt two different strategies. First, we treat each y_i having a fractional value as a leaf node and conduct the feasibility test accordingly. In other words, we apply a rounding heuristic to turn the fractional solution into an integer solution. If the current solution is not feasible, then we proceed to solve a feasibility problem. If the feasibility condition is met, then we focus on the dual problem with the original fractional solutions. As a second approach, we follow the standard procedure and perform the feasibility test with the original fractional values implying that no rounding strategy is applied. Employing the latter strategy turns out to be the most effective since the solver generates fewer user-defined cuts, as well as branching to a fewer number of nodes to reach the optimal in most of the instances.

7 Experimental Results

We first apply the Benders implementations on the randomly generated networks using the Java API and CPLEX solver 12.8.1 on a laptop having a 1.4 GHz Quad-Core Intel Core i5 and 16 GB of RAM. For the real-world PPIN, experiments were ran using CPLEX 20.1 on a Macbook with 2.3 GHz Quad-core, Intel Core i7 and 16GB of RAM (1600 MHz DDR3). We change the default CPLEX settings during the decomposition implementation. We switch the MIP emphasis to optimality over feasibility and set the heuristic frequency to be 1,000 (i.e., RINSHeur = 1,000). Furthermore, we set the number of threads as the number of cores on the laptop both when solving the IP directly and solving the model via BD. All data sets and code sources used in our study will be available online at https://github.com/mcamur/Stochastic-Pseudo-Star-Degree-Centrality.

7.1 Networks Based on the Watts-Strogatz Model

Initially, we randomly generate network instances for testing purposes. The instances are created based on the the Watts-Strogatz (WaSt) model (Watts and Strogatz, 1998), which is also called the small-world model. In such networks, we observe local clusters and small average path length that is tuned by the rewiring probability. The reason for selecting the small-world network as our choice is that one can observe a large number of local clusters in PPINs. Second, the diameter of PPINs is relatively small. For instance, the PPIN of the organism *Helicobacter Pylori* (HP) has 1,570 nodes and diameter 6 while the PPIN of *Staphylococcus Aureus* (SA) has 2,853 nodes as and a diamter of six (Szklarczyk et al., 2015).

In WS models, one can tune the neighborhood parameter (nei) and rewiring probability (rp) to generate different network instances. We consider instances with $|V| \in \{500, 750, 1000, 1250\}$, $nei \in \{14, 16, 18\}$, and $p \in \{0.3, 0.5, 0.7\}$, for a total of 36 instances. Note that increasing rp in a WS network does not change the density of the graph where density is defined as $\psi = \frac{|E|}{|V|(|V|-1)}$. On the contrary, the rewiring probability controls the density of the edges. As rp approaches one, the network turns into an Erdös–Rényi graph (Watts and Strogatz, 1998).

7.2 Calculation of Probability Values

In this section, we present the methodology that we use to identify the probability values associated with the edges. In PPINs, there exists interaction scores in (1,1000) where the higher score implies a stronger interaction between two proteins. We normalize the interaction scores and plot the distribution of the normalized scores (see Figures 6 and 7 for HP and SA, respectively).

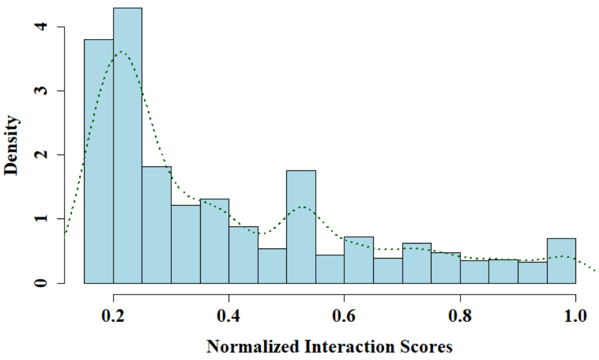


Figure 6: Distribution of interaction scores in HP

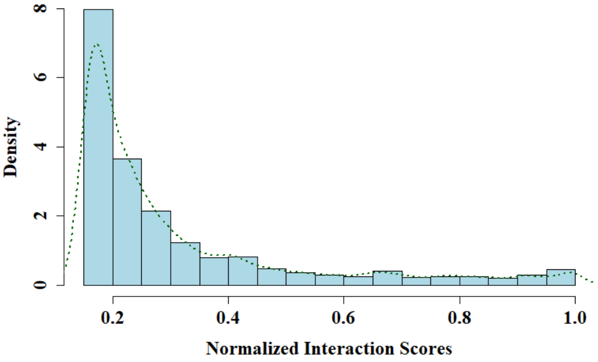


Figure 7: Distribution of interaction scores in SA

We observe that the normalized scores show a right-skewed distribution, which resembles both the gamma distribution with a shape parameter less than or equal to one and the exponential distribution with a rate parameter around 1.5. Therefore, generating probability values according to either distribution can be acceptable. We will be using the exponential distribution with rate parameter 1.5 to generate the probability values associated with edges. It is important to mention that one can also use Monte-Carlo sampling on the real-data sets in order to both generate network samples and probability values. However, sampling from a PPIN would favor that specific network. Hence, we prefer to use our proposed random generation process over Monte-Carlo sampling in order to demonstrate its wider applicability.

7.3 Computational Experiments

We set a time limit of 7,200 seconds and α as 0.99. Let us first share our findings from the preliminary experiments. First of all, we observe that WS plays a crucial role in every method; hence, we do not find it necessary to make a comparison without WS. Second, using LBB cuts both practically and theoretically dominates the traditional Benders cuts presented in Section 5.1. Thus, we use only LBB cuts when implementing the three-phase decomposition. Lastly, implementation of the three-phase decomposition framework with classical Benders performs quite poorly and cannot return optimal solutions within the time limit even for small instances.

Before getting into a detailed analysis, we share a macro-table which summarizes the experimental results. In Table 1, we compare five methods including: i) solving LIP directly, ii) the automatic BD (ABD) provided by CPLEX, iii) three-phase modern BD with the LBB cuts (LBBD) used as feasibility cuts, iv) two-phase classical BD (CBD), v) two-phase modern BD (MBD). For each method, we report the number of instances solved to optimality, the ratio between the number of instances where the optimal solution was found and the total number of instances, the average optimality gap calculated over all the instances, as well as the number of times the method achieved the best performance across all five methods. The best performance is first evaluated according to the optimality gaps. If more than one method returns the optimal solution for the same test instance, then we examine the time to reach the optimal. In addition, we use the bold font to indicate the best method for each criteria.

Table 1: Summary of results (36 Instances)

	LIP	**ABD**	**LBBD**	**CBD**	**MBD**
Optimal	3	2	14	18	**34**
Percentage (%)	0.08	0.06	0.39	0.5	**0.94**
Average Gap (%)	351.83	358.67	255.37	**3.72**	20.74
Best Performance	0	0	2	5	**29**

One can clearly observe that the MBD method shows an outstanding performance where over 90% of the instances are solved to optimality. On the other hand, neither LIP nor ABD had the best performance in any of the instances, which expresses the necessity of customized solution approaches for our problem. Although both LBBD and CBD reach the optimal solution in similar number of instances, CBD produces more stable optimality gaps; thus, we can reach a solid conclusion that the two-phase framework outperforms the three-phase one.

We now move into a detailed analysis and share the computational results obtained through all five methods for each instance. In Table 2, we report the following outputs: i) time spent to reach the solution in seconds, ii) the final optimality gap in percentage, and iii) the number of B&B nodes visited by the solver. For CBD, we share the number of iterations (i.e., the number of Benders cuts added) instead. Note that if the optimal solution is not reached within the time limit (TL), then we use TL as an abbreviation in the table.

Table 2: The computational results with $\alpha = 0.99$

					LIP			ABD			LBBD			CBD			MBD		
$\|\mathbf{V}\|$	$\|\mathbf{E}\|$	ψ	**nei**	**rp**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**Iteration Number**	**Time (sec)**	**Gap (%)**	**BB Nodes**
500	7000	0.0561	14	0.3	797.54	0	1187	TL	251.29	57292	596.48	0	55942	4775.51	0	15	**220.80**	**0**	**1926**
500	7000	0.0561	14	0.5	TL	241.60	775	TL	264.75	55361	675.91	0	36051	1084.66	0	13	**115.85**	**0**	**2260**
500	7000	0.0561	14	0.7	TL	292.25	869	TL	298.96	59080	2193.75	0	41548	TL	0.50	20	**93.27**	**0**	**2483**
500	8000	0.0641	16	0.3	TL	251.30	857	TL	256.24	52700	TL	394.57	45390	3871.72	0	13	**236.57**	**0**	**2102**
500	8000	0.0641	16	0.5	TL	248.64	545	TL	260.18	28495	TL	22.35	56094	3371.39	0	12	**2010.77**	**0**	**1931**
500	8000	0.0641	16	0.7	TL	262.09	854	TL	305.35	27313	TL	255.30	41996	5677.57	0	14	**1947.36**	**0**	**2186**
500	9000	0.0721	18	0.3	TL	291.82	661	TL	302.70	34856	TL	374.77	0	TL	9.73	7	**1275.57**	**0**	**2136**
500	9000	0.0721	18	0.5	5807.68	0	1126	TL	302.24	82690	4315.79	0	149619	TL	17.41	5	**2034.07**	**0**	**3545**
500	9000	0.0721	18	0.7	TL	260.45	779	TL	291.82	35604	TL	308.43	36122	3897.40	0	12	**1981.32**	**0**	**1834**
750	10500	0.0374	14	0.3	TL	297.55	908	TL	301.19	31610	6046.44	0	43978	2938.68	0	11	**372.17**	**0**	**2238**
750	10500	0.0374	14	0.5	TL	379.59	270	TL	356.32	31399	5580.21	0	88719	3593.38	0	17	**199.83**	**0**	**2122**
750	10500	0.0374	14	0.7	TL	317.31	609	TL	320.07	32719	5077.85	0	79687	4956.98	0	15	**178.19**	**0**	**2290**
750	12000	0.0427	16	0.3	TL	406.24	346	TL	372.68	32046	TL	460.43	50549	TL	8.32	10	**3519.90**	**0**	**3784**
750	12000	0.0427	16	0.5	TL	445.23	753	TL	459.38	31762	5375.51	0	36465	6466.95	0	11	**2109.01**	**0**	**3591**
750	12000	0.0427	16	0.7	TL	464.60	209	TL	437.89	21775	TL	383.31	57189	TL	2.44	22	**2041.05**	**0**	**2974**
750	13500	0.0481	18	0.3	TL	422.46	127	TL	415.85	35130	TL	547.05	31686	6799.50	0	7	**5951.80**	**0**	**2566**
750	13500	0.0481	18	0.5	TL	402.93	390	TL	450.34	17978	TL	395.60	43769	TL	5.08	9	**5895.60**	**0**	**2967**
750	13500	0.0481	18	0.7	TL	458.21	386	TL	471.73	22069	TL	432.06	55451	TL	7.10	8	**1886.65**	**0**	**3122**
1000	14000	0.0280	14	0.3	TL	302.32	985	TL	287.96	47102	TL	398.02	31949	5891.95	0	13	**2177.57**	**0**	**3373**
1000	14000	0.0280	14	0.5	TL	308.69	391	TL	294.42	24325	TL	280.40	41598	2938.88	0	7	**2121.30**	**0**	**3414**
1000	14000	0.0280	14	0.7	TL	361.15	280	4050.08	0	13243	630.97	0	28903	**512.41**	**0**	**5**	3001.01	0	3244
1000	16000	0.0320	16	0.3	TL	443.02	122	TL	425.71	18004	**1989.26**	**0**	**64895**	TL	4.58	6	4467.90	0	4731
1000	16000	0.0320	16	0.5	TL	504.19	19	TL	435.30	19137	**2666.17**	**0**	**36171**	TL	0.88	10	3975.74	0	3804
1000	16000	0.0320	16	0.7	TL	465.02	212	TL	448.54	26217	TL	383.00	44790	TL	8.35	12	**2142.86**	**0**	**3802**
1000	18000	0.0360	18	0.3	TL	440.06	220	TL	442.63	9163	TL	619.53	26795	TL	4.01	6	**1286.45**	**0**	**2673**
1000	18000	0.0360	18	0.5	TL	395.97	33	TL	372.67	17720	TL	487.39	21019	4756.36	0	7	**810.59**	**0**	**3523**
1000	18000	0.0360	18	0.7	TL	466.66	92	TL	448.32	20451	4913.15	0	51550	TL	12.17	5	**744.75**	**0**	**3099**
1250	17500	0.0224	14	0.3	TL	326.45	1440	TL	450.05	12173	TL	361.75	34661	5365.21	0	11	**554.04**	**0**	**3161**
1250	17500	0.0224	14	0.5	6331.25	0	2583	TL	378.32	32260	TL	319.11	42463	TL	0.75	20	**326.84**	**0**	**3658**
1250	17500	0.0224	14	0.7	TL	278.50	583	6240.86	0	8597	3574.57	0	23784	**566.13**	**0**	**6**	2126.32	0	3819
1250	20000	0.0256	16	0.3	TL	479.67	217	TL	421.47	53637	TL	536.34	33257	**TL**	**12.64**	**7**	TL	347.79	3425
1250	20000	0.0256	16	0.5	TL	484.24	84	TL	452.10	26271	TL	526.02	27661	TL	9.43	7	**2160.19**	**0**	**5197**
1250	20000	0.0256	16	0.7	TL	439.96	118	TL	395.53	19385	6637.18	0	33576	**3183.06**	**0**	**3**	3961.91	0	3581
1250	22500	0.0288	18	0.3	TL	509.46	108	TL	489.96	19168	TL	590.91	27659	**TL**	**14.07**	**5**	TL	398.85	2540
1250	22500	0.0288	18	0.5	TL	495.61	40	TL	499.84	21576	TL	495.01	28624	TL	10.39	5	**6581.45**	**0**	**4335**
1250	22500	0.0288	18	0.7	TL	522.57	61	TL	550.32	8545	TL	621.81	27870	TL	6.01	7	**6049.89**	**0**	**5176**

Also, similar to Table 1, we use the bold font to indicate which method performs the best in each network instance.

Solving the LIP directly shows a poor performance over all the instances, where only three instances are solved to optimality. The reason behind this could be two-fold. First, the number of BB nodes pruned by the solver is relatively small (i.e., the average is 534) which indicates that the size of the model becomes an issue for the solver to detect new branches. Second, by looking at the engine logs we observe that the number of feasible solutions identified by the solver within the time limit is quite small. Thus, the solver has a hard time in both reaching the optimal solution and determining a feasible solution. Despite the fact that ABD combines the classical and modern Benders implementation, overall, it shows the worst performance. Looking at the number of BB nodes, our intuition is that CPLEX favors separating the feasibility constraints from the MP as a result for which more nodes are pruned. We believe that our decomposition framework where we use an algorithmic approach to generate multi-cuts, separate the fractional solutions as explained in Section 6.4, and the valid inequalities greatly enhances the ability of MBD to solve the problem.

While LBBD reaches the optimal in roughly 40% of the instances, it returns an average optimality gap of 417.87% for the other instances, which are the denser networks (see Table 2). Thus, we believe that LBBD might be an alternative solution approach when dealing with sparser networks. Further, we find the performance of CBD quite successful where it takes roughly 11 iterations to converge to the optimal. As for the instances of which only a feasible solution is obtained, the final optimality gaps are quite good (on average 7.44%). The only reason why MBD has a higher average optimality gap compared to CBD (see Table 1) is that two instances (i.e., (1250-16-0.3 and 1250-18-0.3) which are not solved to optimality have high optimality gaps. Yet, we do not find this highly problematic. The reason is that looking at the engine logs, we observe that the solver has a hard time in improving the dual bounds over the primal bounds. In fact, while CBD produces an objective of 20.038 and 24.705, MBD returns an objective of 18.04 and 22.709 for the same instances.

7.4 PPIN Analysis

In this section, we use the SPSDC metric to calculate the centrality score of each individual protein in HP and make a comparison with other nodal centrality metrics, including degree, closeness, betweenness, and eigenvector, and a group based centrality metric DSDC. It is important to observe that given a node $i \in V$, the objective and constraints in our model are not impacted by any node in $\bar{N}^k(i)$ where $k \geq 3$; thus, we can generate an induced subgraph containing only nodes $i \cup N(i) \cup \bar{N}^2(i)$ to solve the problem for i. Even with this observation, the performance trends from Section 7.3 continue even when we focus on calculating the SPSDC metric for a fixed center (protein) - solving the LIP directly or applying AB to a fixed center does not perform well.

Therefore, we will calculate the SPSDC metric of each protein (node) through our Benders implementation which once again shows the success and importance of our decomposition framework. Our focus here is on demonstrating the ability of the SPSDC to identify essential proteins. We will not present computational run-times; however, our Benders implementation solved all required instances in only a few minutes each for every α value where $\alpha \in \{0.99, 0.9, 0.8, 0.7, 0.6, 0.5\}$.

We rank proteins based on their centrality scores, where a higher centrality score implies a higher rank, in descending and ascending orders to obtain the top and bottom ones, respectively. In Figs. 8 and 9, we

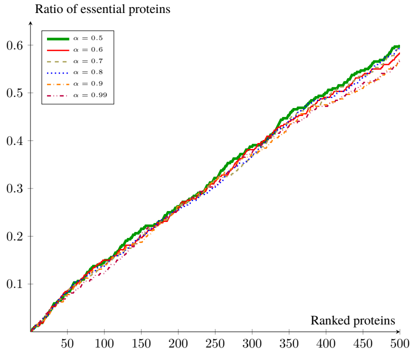


Figure 8: A comparison of different α values when looking at the top ranked proteins

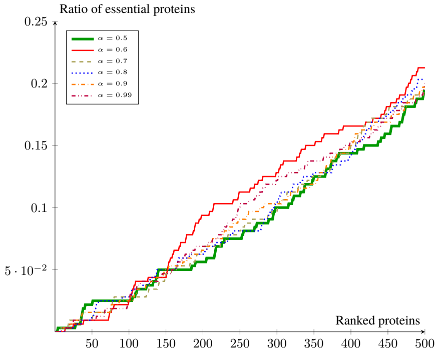


Figure 9: A comparison of different α values when looking at the bottom ranked proteins

analyze first 500 top and 500 bottom proteins (i.e, x-axis) and illustrate the success ratios in the detection of essential proteins (i.e, y-axis). Note that we would want the ratio to be as high as possible (and close to 1) in the top 500 proteins and as low as possible (and close to 0) in the bottom 500 proteins. In the top ranked proteins, the experiment where $\alpha = 0.5$ is able to detect roughly 60% of the essential proteins successfully which is slightly better than the other α values (see Fig. 8); in other words, altering the value of α results in minor changes. Looking at Fig. 9, we can observe that $\alpha = 0.5$ shows the best performance where it wrongly identifies the essential proteins with a lower ratio compared to the other α values.

Now, we move on the general comparison and perform a similar analysis among all the metrics where SPSDC is used with $\alpha = 0.5$. In the top 500 proteins, SPSDC is able to identify more essential proteins than DSDC, which in turn outperforms all nodal centrality metrics as depicted in Fig. 10. A similar pattern is observed in the bottom 500 ranked proteins; that said, DSDC and SPSDC both perform well in identifying proteins with very low essentiality scores. (see Fig. 11).

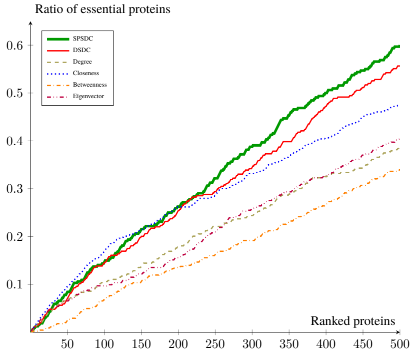


Figure 10: The ratio of essential proteins detected in the top ranked proteins according to each metric for the Helicobacter pylori organism.

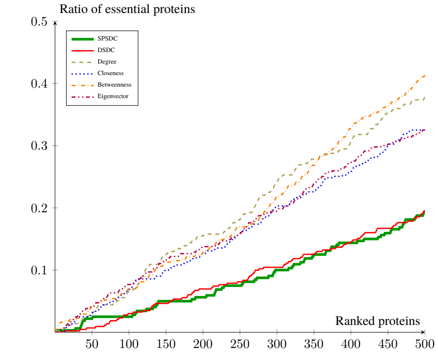


Figure 11: The ratio of essential proteins detected in the bottom ranked proteins according to each metric for the Helicobacter pylori organism.

Overall, we can conclude that SPSDC outperforms other metrics in the detection of essential proteins and may be considered an alternative group-based centrality metric in the literature. It is important to realize that as α decreases, the feasibility condition gets tighter and the size of the pseudo-star is expected to be smaller. In other words, the model would potentially be more selective in adding leaf nodes and favor the ones that do not share too many connections with other leaf nodes. Hence, while resembling the deterministic star stands as important, having freedom in adding leaf nodes that are connected helps the probabilistic version in its task to identify essential proteins. This might explain why SPSDC shows a better performance than the other metrics. Further, our Benders implementation was critical to effectively determine the SPSDC metric for the proteins, thus allowing us to analyze the value of this metric.

8 Conclusion

In this work, we introduce the very first probabilistic group based centrality metric called the stochastic pseudo-star degree centrality (SPSDC) for which we propose a non-linear binary optimization model. We study the complexity of the problem and show that it is $\mathcal{NP}$-complete on general graphs together with trees, and windmill graphs. We implement both classical and modern Benders decomposition techniques with two different frameworks including two- and three-phase decompositions together with several acceleration techniques (e.g., valid inequalities and generation of multi-cuts). In addition, we analyze the impact of logic-based Benders cuts compared to the traditional Benders feasibility cuts. Our two-phase decomposition method implemented through the branch-and-Benders setting shows the best performance. Further, it outperforms solving the model via a commercial solver to a great extent in terms of both solution time and quality.

Our test cases are generated according to the small-world networks which resemble the real-world protein-protein interaction networks (PPINs). The deterministic star degree centrality concept was shown to be an effective centrality metric in order to detect essential proteins in PPINs and our proposed centrality metric can add to the set of proteins to explore for essentiality. In fact, our analysis conducted on a real-world PPIN indicate that SPSDC performs better than its deterministic counterpart; thus, it may be considered an alternative centrality metric in other studies. Lastly, the Benders implementation was critical in effectively determining the SPSDC metric across the proteins in PPINs.

In a future study, it might be interesting to identify a new application area where the SPSDC can be utilized. One good example might be to investigate the network resilience in financial networks in order to detect the most important financial entities in a market.

Acknowledgement

We note that a major part of this work was conducted when M. Can Camur was a Ph.D. candidate at Clemson University in Industrial Engineering Department. First, we thank Dr. Yongjia Song of Clemson University for his worthwhile comments during the initial phase of this project. We also thank Dr. Tyler Perini of Rice University for his kindness in letting us use his personal computer during a part of our computational experiments. Lastly, we appreciate the constructive suggestions of the associate editor and three anonymous reviewers that helped improve the quality of this paper.

References

Absi, N., Archetti, C., Dauzère-Pérès, S., Feillet, D., 2015. A two-phase iterative heuristic approach for the production routing problem. Transportation Science 49, 784–795.

Ahmed, S., Atamtürk, A., 2011. Maximizing a class of submodular utility functions. Mathematical programming 128, 149–169.

Akers, S.B., Harel, D., Krishnamurthy, B., 1994. The star graph: An attractive alternative to the n-cube. Proceedings of the International Conference on Parallel Processing , 393–400.

Akers, S.B., Krishnamurthy, B., 1989. A group-theoretic model for symmetric interconnection networks. IEEE Transactions on Computers 38, 555–566.

Averbakh, I., Pereira, J., 2021. Tree optimization based heuristics and metaheuristics in network construction problems. Computers & Operations Research 128, 105190.

Benders, J.F., 1962. Partitioning procedures for solving mixed–variables programming problems. Numerische Mathematik 4, 238–252.

Bentert, M., Dittmann, A., Kellerhals, L., Nichterlein, A., Niedermeier, R., 2020. An adaptive version of brandes' algorithm for betweenness centrality. Journal of Graph Algorithms and Applications 24, 483–522.

Bonacich, P., 2007. Some unique properties of eigenvector centrality. Social Networks 29, 555–564.

Bonami, P., Salvagnin, D., Tramontani, A., 2020. Implementing automatic benders decomposition in a modern mip solver., in: IPCO, pp. 78–90.

Borgatti, S.P., 1995. Centrality and aids. Connections 18, 112–114.

Brandes, U., Borgatti, S.P., Freeman, L.C., 2016. Maintaining the duality of closeness and betweenness centrality. Social Networks 44, 153–159.

Camur, M.C., 2021. Large-Scale Optimization Models with Applications in Biological and Emergency Response Networks. Ph.D. thesis. Clemson University.

Camur, M.C., Sharkey, T., Vogiatzis, C., 2022. The star degree centrality problem: A decomposition approach. INFORMS Journal on Computing 34, 93–112.

Camur, M.C., Sharkey, T.C., Dorsey, C., Grabowski, M.R., Wallace, W.A., 2021. Optimizing the response for Arctic mass rescue events. Transportation Research Part E: Logistics and Transportation Review 152, 102368.

Chiang, W.K., Chen, R.J., 1998. Topological properties of the (n, k)-star graph. International Journal of Foundations of Computer Science 9, 235–248.

Chou, Z.T., Hsu, C.C., Sheu, J.P., 1996. Bubblesort star graphs: A new interconnection network, in: Proceedings of 1996 International Conference on Parallel and Distributed Systems, IEEE. pp. 41–48.

Day, K., Tripathi, A., 1992. Arrangement graphs: A class of generalized star graphs. Information Processing Letters 42, 235–241.

Deng, M., Mehta, S., Sun, F., Chen, T., 2002. Inferring domain-domain interactions from protein-protein interactions, in: Proceedings of the sixth annual international conference on Computational biology, pp. 117–126.

Derrac, J., García, S., Molina, D., Herrera, F., 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation 1, 3–18.

Enayaty-Ahangar, F., Rainwater, C.E., Sharkey, T.C., 2019. A logic-based decomposition approach for multi-period network interdiction models. Omega 87, 71–85.

Everett, M.G., Borgatti, S.P., 1999. The centrality of groups and classes. The Journal of Mathematical Sociology 23, 181–201.

Fischetti, M., Ljubić, I., Sinnl, M., 2016. Benders decomposition without separability: A computational study for capacitated facility location problems. European Journal of Operational Research 253, 557–569.

Fragoso, F.C., de Sousa Filho, G.F., Protti, F., 2021. Declawing a graph: Polyhedra and branch-and-cut algorithms. Journal of Combinatorial Optimization , 1–40.

García, S., Molina, D., Lozano, M., Herrera, F., 2009. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the cec'2005 special session on real parameter optimization. Journal of Heuristics 15, 617–644.

Guo, C., Bodur, M., Aleman, D.M., Urbach, D.R., 2021. Logic-based Benders decomposition and binary decision diagram based approaches for stochastic distributed operating room scheduling. INFORMS Journal on Computing URL: `https://doi.org/10.1287/ijoc.2020.1036`.

Hooker, J.N., Ottosson, G., 2003. Logic-based Benders decomposition. Mathematical Programming 96, 33–60.

IBM, 2020. IBM ILOG CPLEX Optimization Studio 20.1.0 - IBM Documentation. `https://www.ibm.com/docs/en/icos/20.1.0`. (Accessed on 01/23/2022).

Karimi-Mamaghan, M., Mohammadi, M., Meyer, P., Karimi-Mamaghan, A.M., Talbi, E.G., 2022. Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art. European Journal of Operational Research 296, 393–422.

Kloimüllner, C., Raidl, G.R., 2017. Full-load route planning for balancing bike sharing systems by logic-based Benders decomposition. Networks 69, 270–289.

Li, C., Lin, S., Li, S., 2020. Structure connectivity and substructure connectivity of star graphs. Discrete Applied Mathematics 284, 472–480.

Lin, L., Huang, Y., Hsieh, S.Y., Xu, L., 2020. Strong reliability of star graphs interconnection networks. IEEE Transactions on Reliability , 1–14 URL: `https://doi.org/10.1109/TR.2020.3013158`.

Lozano, M., Molina, D., Garcı, C., et al., 2011. Iterated greedy for the maximum diversity problem. European Journal of Operational Research 214, 31–38.

Mélot, H., 2008. Facet defining inequalities among graph invariants: The system GraPHedron. Discrete Applied Mathematics 156, 1875–1891.

Naderi, B., Govindan, K., Soleimani, H., 2020. A Benders decomposition approach for a real case supply chain network design with capacity acquisition and transporter planning: Wheat distribution network. Annals of Operations Research 291, 685–705.

Nasirian, F., Pajouh, F.M., Balasundaram, B., 2020. Detecting a most closeness-central clique in complex networks. European Journal of Operational Research 283, 461–475.

Pellerin, R., Perrier, N., Berthaut, F., 2020. A survey of hybrid metaheuristics for the resource-constrained project scheduling problem. European Journal of Operational Research 280, 395–416.

Prékopa, A., 2013. Stochastic programming. volume 324. Springer Science & Business Media.

Rasti, S., Vogiatzis, C., 2019. A survey of computational methods in protein–protein interaction networks. Annals of Operations Research 276, 35–87.

Rasti, S., Vogiatzis, C., 2021. Novel centrality metrics for studying essentiality in protein–protein interaction networks based on group structures. Networks .

Roshanaei, V., Luong, C., Aleman, D.M., Urbach, D., 2017. Propagating logic-based Benders' decomposition approaches for distributed operating room scheduling. European Journal of Operational Research 257, 439–455.

Rysz, M., Pajouh, F.M., Pasiliao, E.L., 2018. Finding clique clusters with the highest betweenness centrality. European Journal of Operational Research 271, 155–164.

Simon de Blas, C., Simon Martin, J., Gomez Gonzalez, D., 2018. Combined social networks and data envelopment analysis for ranking. European Journal of Operational Research 266, 990–999.

Su, H.C., Kao, T.W.D., Linderman, K., 2020. Where in the supply chain network does ISO 9001 improve firm productivity? European Journal of Operational Research 283, 530–540.

Sun, H., Sharkey, T.C., 2017. Approximation guarantees of algorithms for fractional optimization problems arising in dispatching rules for INDS problems. Journal of Global Optimization 68, 623–640.

Szklarczyk, D., Franceschini, A., Wyder, S., Forslund, K., Heller, D., Huerta-Cepas, J., Simonovic, M., Roth, A., Santos, A., Tsafou, K.P., et al., 2015. STRING v10: Protein–protein interaction networks, integrated over the tree of life. Nucleic Acids Research 43, D447–D452.

Veremyev, A., Prokopyev, O.A., Pasiliao, E.L., 2019. Finding critical links for closeness centrality. INFORMS Journal on Computing 31, 367–389.

Vogiatzis, C., Camur, M.C., 2019. Identification of essential proteins using induced stars in protein–protein interaction networks. INFORMS Journal on Computing 31, 703–718.

Watts, D.J., Strogatz, S.H., 1998. Collective dynamics of 'small-world'networks. Nature 393, 440–442.

Yan, K., Ryoo, H.S., 2022. Graph, clique and facet of boolean logical polytope. Journal of Global Optimization , 1–38.

Zhang, Z., Song, X., Huang, H., Zhou, X., Yin, Y., 2022. Logic-based Benders decomposition method for the seru scheduling problem with sequence-dependent setup time and DeJong's learning effect. European Journal of Operational Research 297, 866–877.

Zhong, H., Pajouh, F.M., Prokopyev, O.A., 2021. Finding influential groups in networked systems: The most degree-central clique problem. Omega 101, 102262.

Zohali, H., Naderi, B., Roshanaei, V., 2021. Solving the type-2 assembly line balancing with setups using logic-based Benders decomposition. INFORMS Journal on Computing .