The Star Degree Centrality Problem: A Decomposition Approach

Mustafa C. Camur
Clemson University, mcamur@clemson.edu,

Thomas C. Sharkey
Clemson University, sharkt@clemson.edu,

Chrysafis Vogiatzis
University of Illinois at Urbana-Champaign, chrys@illinois.edu,

We consider the problem of identifying the induced star with the largest cardinality open neighborhood in a graph. This problem, also known as the *star degree centrality* (SDC) problem, has been shown to be $\mathcal{NP}$-complete. In this work, we first propose a new integer programming (IP) formulation, which has a smaller number of constraints and non-zero coefficients in them than the existing formulation in the literature. We present classes of networks where the problem is solvable in polynomial time, and offer a new proof of $\mathcal{NP}$-completeness that shows the problem remains $\mathcal{NP}$-complete for both bipartite and split graphs. In addition, we propose a decomposition framework which is suitable for both the existing and our formulations. We implement several acceleration techniques in this framework, motivated by techniques used in Benders decomposition. We test our approaches on networks generated based on the Barabási–Albert, Erdös–Rényi, and Watts–Strogatz models. Our decomposition approach outperforms solving the IP formulations in most of the instances in terms of both solution time and solution quality; this is especially true for larger and denser graphs. We then test the decomposition algorithm on large-scale protein-protein interaction networks, for which SDC was shown to be an important centrality metric.

Key words: Star degree centrality; Decomposition algorithm; Protein-protein interaction networks

1 Introduction

Centrality is one of the best-studied concepts in network analysis. It has been used in a variety of applications to quantify the importance of nodes or entities in a network. The main idea is that the more central a node is, the more importance it has. Expectedly, not every measure of importance is equally valid in every application. Hence, a series of simpler or more complex notions of centrality have been proposed over the years. They range from the early work by Bavelas [1948, 1950] and Leavitt [1951] on task-oriented group creation, as well as the introduction of eigenvector and bargaining centrality by Bonacich [1972, 1987], to more recent ideas about subgraph [Estrada and Rodríguez-Velázquez 2005], residual [Dangalchev 2006] or diffusion [Banerjee et al. 2013] centrality. In this work, we turn our focus to a concept referred to as group centrality [Everett and Borgatti 1999]. More specifically, we study the recently introduced measure of star degree centrality (SDC) by Vogiatzis and Camur [2019] where SDC has been shown to be a highly efficient centrality metric to identify the essential proteins in protein-protein interaction networks (PPINs). The results indicate that it performs better than the other well-known metrics (i.e., degree, closeness, betweenness, and eigenvector) in the determination of the essential proteins. The contributions of Vogiatzis and Camur [2019] are in approximation algorithms for finding nodes with high SDC whereas we contribute by providing exact solution approaches that are able to solve problems of significant size.

In a fundamental contribution, Freeman [1978] examined three distinct and recurring concepts in centrality studies, namely *degree*, *betweenness*, and *closeness*. The basic definitions involved with each of the concepts are as follows. Degree is related to the number of connections that a node has (i.e., number of nodes adjacent to a given node i, often normalized by the number of nodes in the network minus 1); betweenness can be quantified as the fraction of shortest (geodesic) paths that use a specific node i; finally, closeness is a function of the shortest (geodesic) paths that a node i has to every other node in the network. A common theme behind the above definitions is their nodal consideration.

Group extensions to centrality have recently been proposed to help address questions of importance for a group as a whole, as well as for introducing importance that can be attributed to the node versus to the group it belongs. This idea was presented by Everett and Borgatti [1999, 2005] and was immediately picked up and expanded upon by a series of researchers. Prominent extensions included the definition of clique (cohesive subgroup) centrality [Vogiatzis et al. 2015, Rysz et al. 2018, Nasirian et al. 2020]. Identifying a general group of nodes with highest betweenness centrality is also studied by Veremyev et al. [2017], where they also mention the possibility to introduce additional "cohesiveness" constraints.

Star degree centrality (also stylized as star centrality) tasks itself with identifying the induced star centered at a given node i that possesses the maximum cardinality open neighborhood. An induced star centered at i includes i and a subset of its neighbors under the condition that no two neighbors are adjacent. A node is in the open neighborhood of the star if it is not in the induced star and is adjacent to a node in the induced star. Vogiatzis and Camur [2019] study the problem in the context of a PPIN. The authors derive the computational complexity of the problem and show it is $\mathcal{NP}$-hard; additionally, they provide integer programming (IP) formulations and approximation algorithms to solve it efficiently. More importantly, they show that this is indeed a viable proxy for predicting essentiality in PPINs. Essential genes (and their essential proteins) are ones whose absence leads to lethality or the inability of an organism to properly reproduce themselves [Kamath et al. 2003]. Thus, identifying the node with the highest star degree centrality finds an important application in PPINs.

PPINs are networks where nodes represent proteins and arcs represent protein-protein interactions. These networks have been heavily studied over the last two decades: for a series of surveys on computational methods for complex detection, clustering, detecting essentiality, among others, in protein-protein interaction networks, we refer the interested reader to the recent reviews by Wang et al. [2013], Bhowmick and Seah [2015], and Rasti and Vogiatzis [2019]. Centrality has been a staple in the study of biological networks, and specifically PPINs: CentiServer [Jalili et al. 2015] is a database that has collected a large number of centrality-based approaches for biological networks at https://www.centiserver.org.

Jeong et al. [2001] proposed the "lethality-centrality" rule, in which the more central a protein is, the higher the probability it is essential. This work led to significant research interest in centrality metrics in PPINs (see the works by Joy et al. [2005] on betweenness, Estrada [2006] on subgraph centrality, Wuchty and Stadler [2003] on closeness centrality). An updated survey and comparison of 27 commonly used centrality metrics (including degree, betweenness, and closeness) is presented in the work by Ashtiani et al. [2018].

At this point, we should mention that the high computational complexity in PPINs did not allow Vogiatzis and Camur [2019] to conduct a full analysis across the entire network. That is why they used two different approaches to simplify the problem: i) setting extremely high thresholds to prune the edges in the networks and ii) utilizing a probabilistic approach to create the interactions between the proteins. In addition, the essential protein analysis is performed by selecting k (i.e., a user-defined value) top proteins for each of which an individual IP is solved assuming each as the center. On the other hand, our decomposition implementation opens the door to a full analysis of large-scale networks by being able to identify the node with the highest SDC across the entire network. Our computational results indicate that we can avoid using high thresholds to perform analysis in real-world PPINs.

Our work is outlined as follows. First, we provide a formal problem definition together with two illustrative examples detailing how the SDC is applied in Section 2. We begin the discussion in Section 3 from the previously introduced formulation by Vogiatzis and Camur [2019] and then propose a new, compact formulation. Section 4 presents classes of networks where the problem is solvable in polynomial time and offers a new proof of $\mathcal{NP}$-completeness that shows the problem remains $\mathcal{NP}$-complete even for bipartite and split graphs (thus tightening the complexity analysis of Vogiatzis and Camur [2019]). In Section 5, we provide a decomposition implementation for solving the problem on real-life, large-scale networks, such as the ones typically encountered in computational biology and specifically in PPINs. Section 6 discusses acceleration techniques, motivated by accelerating Benders decomposition methods, for both IP formulations and the decomposition approaches. All our algorithmic advancements are put to the test in Section 7 which is divided into two subsections for randomly generated instances and PPIN instances. We conclude with a summary of our findings and recommendations for future work in Section 8.

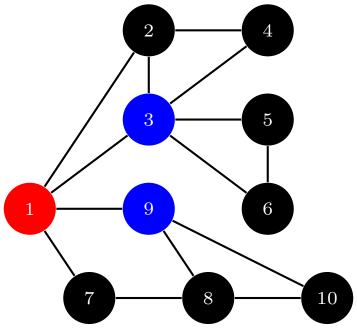
2 Problem Definition

Let $G = (V, E)$ be an undirected graph consisting of a vertex set V and an edge set E where $|V| = n$ and $|E| = m$. We define the open neighborhood of a node $i \in V$ as the set of nodes adjacent to i; in other words, $N(i) = \{j \in V : (i,j) \in E\}$. Similarly, the closed neighborhood of a node $i \in V$ is defined as $N[i] = N(i) \cup \{i\}$. For a set of nodes S, we define the open neighborhood as $N(S) = \{j \in V : i \in S, j \notin S, (i,j) \in E\}$. Additionally, we define the k-neighborhood of a node $i \in V$ as the set of nodes whose shortest path from i is exactly k edges and denote it as $\bar{N}^k(i)$. In other words, $\bar{N}^k(i)$ represents the set of nodes that are reachable from i within exactly k-edge hops.

Definition 1. The *star degree centrality* of a given node i is a centrality measure identifying the induced star S_i centered at i with the largest open neighborhood and is formally defined as $\vartheta_i = \max\{|N(S_i)| : S_i \subset V,\ (i,j) \in E\ \forall j \in S_i \backslash \{i\},\ (j', j'') \notin E\ \forall j', j'' \in S_i \backslash \{i\}\}$.

Example 1. We present how to construct a feasible induced star with the largest open neighborhood in a toy example in Fig. 1. We let node 1 be the center of the induced star. Since each leaf must be connected to the center node, there are four candidate leaf nodes (i.e., nodes 2, 3, 7, and 9). However, recall that no two leaf nodes are allowed to share an edge in a feasible star. Therefore, both nodes 2 and 3 together cannot be a part of a star centered at node 1. The one that is not in the star goes into the open neighborhood using the edge from 1. Since the contribution of node 3 to the objective (i.e., it increases the objective by three - nodes 4, 5, and 6 are selected in the open neighborhood if 3 is a leaf node) is larger than the contribution of node 2 (i.e., it increases the objective by one - only node 4 is selected in the open neighborhood), node 3 would be selected to be in the induced star.

Figure 1 **Example of an induced star in a given network where the center node and leaf nodes are shown in red and blue color, respectively.**



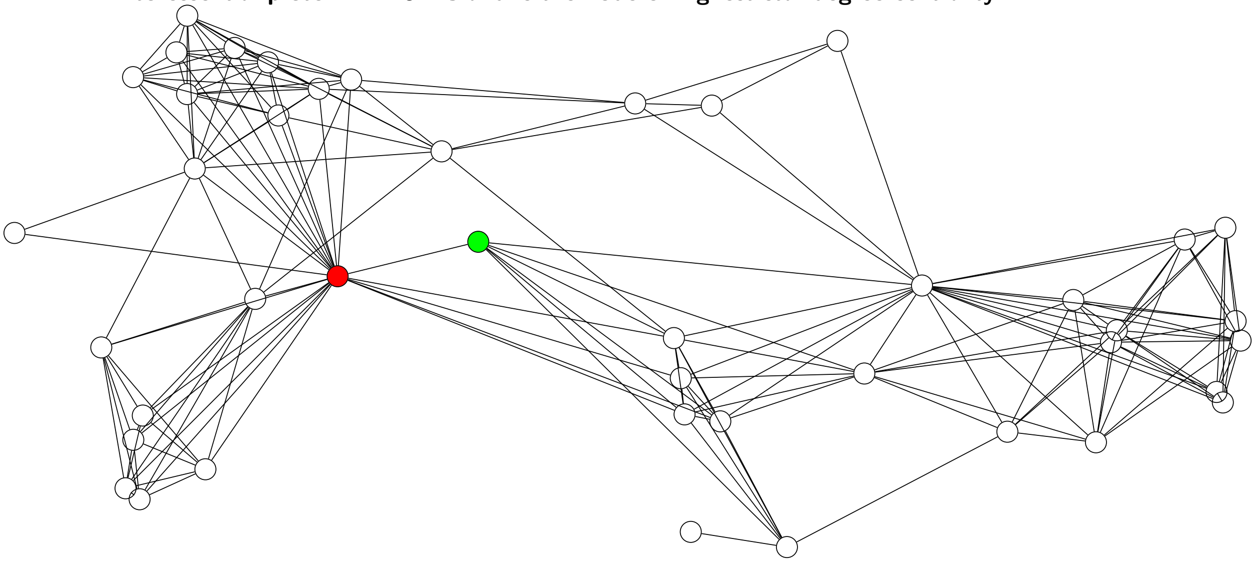
Similarly, node 9 would be preferred as a leaf node over node 7, because its contribution to the objective is higher compared to node 7 (i.e., 9 allows 8 and 10 to be in the neighborhood while 7 only allows 8, respectively). It is important to observe that although both nodes 7 and 9 can exist in a feasible induced star together, incorporating node 7 along with node 9 into a star centered at node 1 would decrease the objective by one since 7 is in the neighborhood if it is not a leaf node. This also shows that the star centrality function defined as the size of the open neighborhood of a feasible star cannot be claimed to be monotonically increasing. In other words, greedily adding leaf nodes does not guarantee to increase the objective value. Overall, the star $S_1 = \{1, 3, 9\}$ has $N(S_1) = \{2, 4, 5, 6, 7, 8, 10\}$

Example 2. In Fig. 2, we present some of the notions in this work using a real-life example from the yeast proteome (Saccharomyces Cerevisiae) keeping only interactions above a threshold of 92% (so that the induced subgraph is sparse enough for visualization purposes).

The highest degree centrality protein is also known as YMR300C (marked in red) and despite its central location and its many documented interactions, it is not essential. We observe that YMR300C is adjacent to two main protein complexes (dense subgraphs). This means that many of the connections that YMR300C has to other nodes are also shared among the nodes themselves. Hence, if we were to discard connections between neighbors (that is, we enforced a “star” constraint), its importance would be sure to decrease.

On the other hand, the highest star degree centrality protein is known as YHL011C (marked in green), an essential protein for many cell activities as it is used to synthesize

Figure 2 **An example of why a star structure helps identify essential proteins. In this figure, we present a subgraph of the PPIN of Saccharomyces Cerevisiae (yeast) using a threshold of 92%. The node in red corresponds to non-essential protein YMR300C and is the node of highest degree; the node in green corresponds to essential protein YHL011C and is the node of highest star degree centrality.**



phosphoribosyl pyrophosphate. We observe that while its degree centrality is small (the number of neighbors it has is only 7, compared to a degree centrality of 23 for YMR300C), it is adjacent to nodes that connect different protein complexes and communities.

3 Mathematical Formulations

First, we present the formulation that appears in the literature (the Vogiatzis and Camur [2019] integer programming (VCIP) formulation). Then, we introduce a new formulation, which is more compact in theory with respect to the number of constraints. In the original formulation, there are three sets of binary variables: (i) x_i is equal to 1 if and only if $i \in V$ is the center of the star, (ii) y_i is equal to 1 if node i is in the star, and (iii) z_i is equal to 1 if node i is in the open neighborhood of the star. The IP model is now provided in (1).

[VCIP]:

$$\max \sum_{i \in V} z_i \tag{1a}$$

$$s.t.\ y_i + z_i \leq 1, \qquad \forall i \in V \tag{1b}$$

$$z_i \leq \sum_{j \in N(i)} y_j, \qquad \forall i \in V \tag{1c}$$

$$y_i \leq \sum_{j \in N[i]} x_j, \qquad \forall i \in V \tag{1d}$$

$$x_i \le y_i, \qquad \forall i \in V \tag{1e}$$

$$y_i + y_j \le 1 + x_i + x_j, \qquad \forall (i,j) \in E \tag{1f}$$

$$\sum_{i \in V} x_i = 1, \tag{1g}$$

$$x_i, y_i, z_i \in \{0,1\}, \qquad \forall i \in V. \tag{1h}$$

The objective function (1a) maximizes the number of the nodes adjacent to the star. Constraints (1b) indicate that no node can be in the star and the neighborhood. Constraints (1c) ensure that for a node to be a neighbor to the star, it must be adjacent to at least one node in the star. In addition, every node in the star must be in the closed neighborhood (i.e., a neighborhood containing the node itself) of the center node by constraints (1d). We should point out that constraints (1e) ensuring that the center node is part of the star were absent in the printed version in Vogiatzis and Camur [2019]. Constraints (1f) prevent two adjacent nodes from being in the star if neither is the center. This computationally stands as the most expensive constraint due to the fact that it must appear for every edge. Constraint (1g) makes sure that the model identifies a single star by selecting one center node. Last, constraints (1h) dictate the binary requirements for each variable. Note that there is a total of $4n + m + 1$ constraints in [VCIP]. Further, we can examine the number of total non-zero coefficients across each type of constraint: (1b) has $2n$; (1c) has $n + 2m$; (1d) has $2n + 2m$ (since $i \in N[i]$); (1e) has 2n; (1f) has $4m$; and (1g) has n. These sum to a total of $8n + 8m$ non-zero coefficients.

In the former formulation [VCIP], though there is a specific variable used for the center node (i.e., x_i), variable y_i corresponds to any node in the star without making any distinction. An important observation is that leaf nodes in a star carry a unique characteristic which differentiates them from the center node. That is, while a leaf node has solely one edge connecting it to the star via the center node, the center node shares an edge with every leaf node. Hence, we remove variable y_i and introduce a new variable to represent the leaf nodes.

$$l_i = \begin{cases} 1, & \text{if node } i \in V \text{ is a leaf of the star} \\ 0, & \text{otherwise.} \end{cases}$$

After this conversion, we can remodel the problem with a new IP (NIP) formulation.

[NIP]:

$$
\begin{aligned}
\max \ & \sum_{i\in V} z_i && && \text{(2a)}\\
s.t. \ & x_i + l_i + z_i \le 1, && \forall i \in V && \text{(2b)}\\
& z_i \le \sum_{j\in N(i)} (l_j + x_j), && \forall i \in V && \text{(2c)}\\
& l_i \le \sum_{j\in N(i)} x_j, && \forall i \in V && \text{(2d)}\\
& \sum_{j\in N(i)} l_j \le |N(i)|(1-l_i), && \forall i \in V && \text{(2e)}\\
& \sum_{i\in V} x_i = 1, && && \text{(2f)}\\
& x_i, l_i, z_i \in \{0,1\}, && \forall i \in V. && \text{(2g)}
\end{aligned}
$$

First of all, constraints (2a), (2f), and (2g) correspond to constraints (1a), (1g), and (1h), respectively. Constraints (2b) guarantee that a node cannot be the center, a leaf, and a neighbor of the star at the same time, which is similar to original constraints (1b). Constraints (2c) replace (1c) and indicate that a node should be adjacent to either the center node or at least one of the leaf nodes, if it is adjacent to the star. Each leaf node is connected to the center node to form a feasible star, which is enforced by constraints (2d). With the new variable definition (i.e., l_i), we eliminate two constraints (that is, (1e) and (1f)), and no longer need to account for all edges in the graph. Constraints (2e) state that if a node is selected as a leaf, none of the nodes which are adjacent to it can also be a leaf node. Note that there is a total of $4n+1$ constraints in [NIP]. Further, we can examine the number of total non-zero coefficients across each type of constraint: (2b) has $3n$; (2c) has $n+4m$; (2d) has $n+2m$; (2e) has $n+2m$; and (2f) has n. These sum to a total of $7n+8m$ non-zero coefficients.

We now examine the tightness of the linear programming (LP) relaxations of these two formulations.

Theorem 1. *The LP relaxation of [VCIP] is stronger than the LP relaxation of [NIP].*

Proof. See the online supplement. □

Even though [VCIP] is a stronger formulation than [NIP] in terms of the LP-relaxation, we observe here that while the constraint set is bounded by $O(n+m)$ in [VCIP], the new formulation [NIP] is associated with a constraint set bounded by $O(n)$. Furthermore, the number of non-zero coefficients are slightly higher in [VCIP] (i.e., $8n + 8m$) compared to [NIP] (i.e., $7n + 8m$). It is worth mentioning that the number of non-zero coefficients can be reduced with a constraint tightening in [NIP], which is discussed in Section 6.1. All of these factors may impact the computational performance of solving these problems. This is further examined in Section 7, where we demonstrate that [NIP] is the foundation for more efficient methods to solve the problem.

4 Complexity Discussion

The SDC problem over general graphs was shown to be $\mathcal{NP}$-complete by Vogiatzis and Camur [2019]. In this section, we provide graphs where the SDC problem can be solved in polynomial-time and prove that the SDC problem remains $\mathcal{NP}$-complete on certain networks.

4.1 Polynomial-Time Cases

Theorem 2. *The SDC problem is solvable in polynomial time on trees.*

Proof. We propose Algorithm (1) that identifies the optimal induced star with the maximum size neighborhood in $O(m)$ time for a tree. For the sake of simplicity, we assume that the given graph is connected and $n \geq 3$. The algorithm goes through each edge $(i, j) \in E$ and determines whether an adjacent node is considered a leaf node or a neighbor node. For a given edge (i, j), there exist three cases, considering each node as a center of a star.

1. If $|N(i)| > 1$ and $|N(j)| = 1$, then i would be a leaf for a star centered at j and all nodes $N(i)\backslash j$ would serve as the neighbors of the star. In this case, j would be selected as being in the neighborhood of the star centered at i since having it as a leaf would result in no additional neighbors.
2. If $|N(i)| = 1$ and $|N(j)| > 1$, then j would be leaf for a star centered as i and i would be in the neighborhood for a star centered at j.
3. If both $|N(i)|$ and $|N(j)|$ are greater than one, then they would each be a leaf for a star centered at the other. Note that after identifying a node $i \in V$ as a leaf, we can directly compute its contribution to the objective with $|N(i)| - 1$ due to the fact that the graph is acyclic.

Thus, we can conclude that the problem can be solved efficiently if the given graph is a tree.

□

Definition 2. A graph $Wd(k, n)$ where $k \geq 2$ and $n \geq 2$ is called a *windmill* graph, with n copies of K_k complete graphs with a shared universal vertex.

Algorithm 1: An algorithm to solve the SDC problem on a tree

```
Input: $G=(V,E), L, S$
$L[i] \leftarrow \emptyset$;   $\forall i \in V \mid L[i]$ : list of leaf nodes connected to center $i$
$S(i)=0$;   $\forall i \in V \mid S(i)$ : number of nodes adjacent to the star whose center is $i$
for $(i,j) \in E$ do
    if $|N(i)|>1$ and $|N(j)|=1$
        $S(i)++$;
        $L[j] \leftarrow L[j] \cup \{i\}$;
        $S(j)=S(j)+|N(i)|-1$;
    else if $|N(i)|=1$ and $|N(j)|>1$
        $L[i] \leftarrow L[i] \cup \{j\}$;
        $S(i)=S(i)+|N(j)|-1$;
        $S(j)++$;
    else
        $L[i] \leftarrow L[i] \cup \{j\}$;
        $S(i)=S(i)+|N(j)|-1$;
        $L[j] \leftarrow L[j] \cup \{i\}$;
        $S(j)=S(j)+|N(i)|-1$;
$i^* = \arg\max_{i \in V} S_i$;
return $i^*, L[i^*]$
```

Proposition 1. *Given a windmill graph $Wd(k,n)$, there exists a unique optimal solution solely containing the universal vertex for the SDC problem.*

Proof. See the online supplement. □

4.2 $\mathcal{NP}$-Complete Classes

Vogiatzis and Camur [2019] show that the SDC problem is $\mathcal{NP}$-complete via a reduction from a well-recognized combinatorial problem, the *Maximum Independent Set* (MIS). It is widely known that according to the König's theorem, the MIS can be efficiently determined if the graph is bipartite. Yet, we show that the SDC problem preserves its complexity even in a bipartite graph. We first provide the decision versions of the SDC problem and the *Set Cover Problem* (SCP) via which we perform a reduction.

Definition 3. (Star Degree Centrality) Given an undirected graph $G=(V,E)$ and an integer ℓ, does there exist a node i and an induced star C centered at i such that $|N(C)| \geq \ell$?

Definition 4. (Set Cover) Given a set of elements $U=\{u_1,u_2,\cdots,u_n\}$ (i.e., the universe), a collection of subsets, $S=\{S_1,S_2,\cdots,S_m\}$ where $\cup_{i=1}^m S_i = U$, and an integer k, does there exists a set $I \subseteq S$ such that $|I| \leq k$ and $\cup_{i\in I} S_i = U$?

Theorem 3. *The SDC problem is $\mathcal{NP}$-complete on bipartite graphs.*

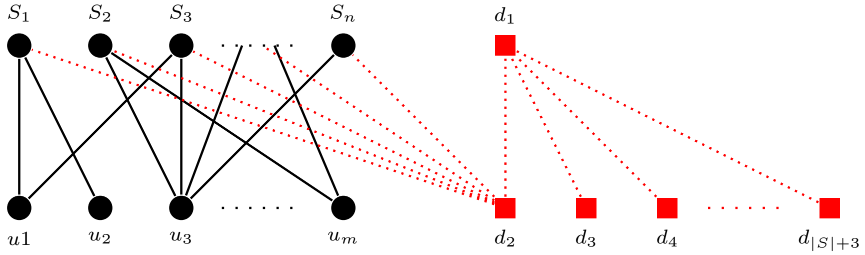
Proof. Given a potential induced star centered at node i, we must verify if any two leaf nodes share an edge to verify if it is truly an induced star. One can then verify if $|N(C)| \geq \ell$ easily. This shows that SDC problem is in $\mathcal{NP}$ if the graph is bipartite.

Now, let $<U,S,k>$ be an instance of the SCP where k represents the number of sets to cover all the elements in U. We can then construct an instance of SDC problem $<G,\ell>$ on a bipartite graph as follows:

$$V[G] = V_1 \cup V_2 \text{ where } V_1 = \{S_1,S_2,\cdots,S_m,d_1\} \text{ and } V_2 = \{u_1,u_2,\cdots,u_n,d_2,d_3,d_4\cdots,d_{|S|+3}\}$$
$$E[G] = \Big\{\cup_{i=1}^m \{\cup_{j\in S_i}(S_i,u_j)\}\Big\} \cup \{\cup_{i=1}^m (d_2,S_i)\} \cup \{(d_1,d_2)\} \cup \{\cup_{i=3}^{|S|+3}(d_1,d_i)\}.$$

The construction proposed (see Fig 3) can be explained as follows. Each set $S_i \in S$, and each element $u_i \in U$ are considered a node in V_1 and V_2, respectively. Then, we add edges between each set and all elements contained in the set. A dummy node d_2 is placed in V_2 and is connected with each $S_i \in V_1$. Another dummy node d_1 is added into V_1 and is connected to d_2. Finally, we add $|S|+1$ dummy nodes into V_2, each of which shares an edge with d_1. After this configuration, we obtain a bipartite graph. Lastly, we set $\ell = 2|S|+|U|+k-1$. We examine the potential size of the induced stars centered at five different potential nodes: a set node, an element node, d_i with $i \geq 3$, d_1, and d_2 which helps us to show that a particular choice of the star centered at d_2 corresponds to a set cover (if one exists).

Figure 3 **The transformation of *Set Cover* $<\mathbf{U},\mathbf{S},\mathbf{k}>$ to an instance $<\mathbf{G}(\mathbf{V},\mathbf{E}),\mathbf{l}>$ of *Star Degree Centrality*.**



1. If $S_i \in V_1$ is the center, then the upper bound (UB) on the size of the potential neighborhood is $(|U|-1)+(|S|-1)+1=|U|+|S|-1$ since either d_1 or d_2 can be in the neighborhood and then all other S_j and u_k nodes may be in it.
2. If $u_i \in V_2$ is the center, then the UB on the size of the potential neighborhood is $(|S|-1)+(|U|-1)+1=|U|+|S|-1$ since either d_2 can be in the neighborhood and then all other S_j and u_k nodes may be in it.
3. If a dummy node d_i where $i \geq 3$ is the center, the size of the neighborhood is $|S|+1$. Every d_j such that ($j \geq 3$ and $j \neq i$) and d_2 are neighbor nodes while d_1 is a leaf.
4. If dummy node d_1 is the center, then the size of the neighborhood is $2|S|+1$ by picking d_2 as a leaf node.
5. If dummy node d_2 is the center, then d_1 is considered a leaf and $|S|+1$ nodes become the neighbors (i.e., $\forall d_j, j \geq 3$). Every S_i node can appear as either a leaf or in the star's neighborhood. Consider a partition of the set nodes into leaves and those in the star's neighborhood. If there is a node that is a leaf such that all elements u_j in it are covered by other leaf node sets, then we can move that set node to the neighborhood of the star and increase its size. If there is a node in the neighborhood which contains one or more u_j that are not in the star's neighborhood, then we can move that node to be a leaf and either keep the size the same (if exactly one u_j is uncovered) or increase the size of the neighborhood. This latter point shows that we can create another star whose neighborhood size is greater than or equal to the size of our current star. This means that all u_j nodes should be in the neighborhood of the star.

Note that if $|U| \leq k$ in SCP, then the problem is solvable in polynomial time by verifying that each element appears in one set. We focus our analysis on situations where $|U|-k>0$. Suppose there is a set cover, I such that $|I| \leq k$. Consider the star centered at d_2 with the set of leaf nodes being $\{d_1, S_i : i \in I\}$. From Point 5, we know that all $d_j, j \geq 3$ are in the neighborhood, all $S_{i'}$ for $i' \notin I$ are in the neighborhood, and all u_j are in the neighborhood since I is a cover. This means that this star has a size of $|S|+1+|U|+|S|-|I| \geq 2|S|+1+U-k=\ell$. Alternatively, suppose we have a star whose neighborhood is greater than or equal to ℓ. This star has to be centered at d_2 by Points 1-4 above. By Point 5, we know that we can convert this star (if necessary) to one where all u_j are in the neighborhood of the same or greater size. By accounting for the dummy nodes $d_j, j \geq 3$ and the u_k nodes, we have that $|S|-k$ or more set nodes must be in the neighborhood. Note that since all u_j are in the neighborhood, this means that the set nodes that are leaves (there are at most k of these) must cover all the elements. Therefore, there exists a set cover of less than or equal to k sets.

□

Definition 5. A graph is called a *split* graph when the vertices can be partitioned into two sets where one is a clique and the other one is an independent set.

Theorem 4. *The SDC problem is $\mathcal{NP}$-complete on split graphs.*

Proof. See the online supplement. □

5 Solution Methodology

While both models proposed contain $3n$ binary variables, the number of constraints are $O(n+m)$ and $O(n)$ in [VCIP] and [NIP], respectively. Solving the IP models via a commercial solver is computationally challenging (see Section 7); especially, as the graph gets larger and/or denser. Therefore, we first examine Benders Decomposition (Benders [1962]) for both formulations. We find that the most computationally effective implementation of this decomposition approach is a branch-and-cut framework that adds violated constraints from the original problem back into the master problem. We propose to find a feasible induced star in the master problem (MP) and then check the size of the neighborhood in the subproblem (SP), i.e., the z variables move to the SP in both formulations. Hence, we are only concerned with optimality cuts.

We split the variables into (x,y) and (x,l) in the first stage for [VCIP] and [NIP], respectively. This means that we have $5n+6m$ non-zero coefficients in the MP for the method using [VCIP] and $3n+4m$ for the method based on [NIP]. Given a fixed $(\bar{y})$ or $(\bar{l},\bar{x})$, we obtain the following SPs by isolating $\vec{z}$ in the second stage:

$$\begin{aligned} \phi^{VCIP}(\bar{y}) := \max_{z} \ & \sum_{i\in V} z_i \\ s.t.\ & z_i \le 1-\bar{y}_i, && \forall i \in V \\ & z_i \le \sum_{j\in N(i)} \bar{y}_j, && \forall i \in V \\ & z \in \{0,1\}^n \end{aligned} \qquad \begin{aligned} \phi^{NIP}(\bar{l},\bar{x}) := \max_{z} \ & \sum_{i\in V} z_i \\ s.t.\ & z_i \le 1-\bar{l}_i-\bar{x}_i, && \forall i \in V \\ & z_i \le \sum_{j\in N(i)} (\bar{l}_j+\bar{x}_j), && \forall i \in V \\ & z \in \{0,1\}^n \end{aligned}$$

We first note that the primal SPs represented above are separable over each node as shown below. As a result, multiple Benders cuts can be generated at the same time.

$$\phi^{VCIP}(\bar{y}) = \sum_{i\in V} \phi_i^{VCIP}(\bar{y}) := \sum_{i\in V} \max_{z_i\in\{0,1\}} \left\{ z_i : z_i \le 1-\bar{y}_i, \quad z_i \le \sum_{j\in N(i)} \bar{y}_i \right\}$$

$$\phi^{NIP}(\bar{l},\bar{x}) = \sum_{i\in V} \phi_i^{NIP}(\bar{l},\bar{x}) := \sum_{i\in V} \max_{z_i\in\{0,1\}} \left\{ z_i : z_i \le 1-\bar{l}_i - \bar{x}_i, \quad z_i \le \sum_{j\in N(i)} (\bar{l}_j + \bar{x}_j) \right\}$$

We refer the reader Cordeau et al. [2019] for similar Benders frameworks generated for both large-scale partial set covering and maximal covering problems, where the authors discuss different ways of generating feasibility cuts (e.g., normalized and facet-defining feasibility cuts). We use the so-called *Modern Benders Decomposition approach* [Fischetti et al. 2016, 2017], where Benders cuts are added on-the-fly (if violated) when the solver identifies incumbent or fractional solutions. This is also called the *branch-and-Benders cut* approach implying that there exists only a single enumeration tree, with which the solver never visits the same candidate nodes again. Note that for our methods, the procedure to generate cuts added based on fractional and integer solutions are the same. We provide more information on the separation of fractional and integer solutions in Section 6.4..

In examining both SPs for integer incumbent solutions $(\bar{y})$ or $(\bar{l},\bar{x})$, the binary decision variables z_i are bounded by integer values. Therefore, we can solve these SPs by relaxing the z_i variables which will be helpful in deriving Benders cuts for both integer and fractional values of $(\bar{y})$ and $(\bar{l},\bar{x})$. Moreover, whenever an incumbent solution is passed to the relaxed SPs, the optional solution to these problems is indeed binary, which shows the correctness of the *traditional* Benders decomposition method to solve the problem. In particular, we can use LP duality to generate the Benders cuts.

i. For [VCIP], since $0 \le \bar{y}_i \le 1$, $(1-\bar{y}_i)$ also lies in $[0,1]$ implying $z_i \le 1$. Further, $\sum_{j\in N(i)} \bar{y}_i$ is a non-negative integer. Taking this into consideration with the fact we maximize over z_i, we *do not need to explicitly enforce* $z_i \ge 0$. Hence, we can relax the integrality and non-negativity requirements on z_i. We obtain:

$$\phi_i^{VCIP}(\bar{y}) = \max_{z_i} \left\{ z_i : z_i \le 1-\bar{y}_i, \quad z_i \le \sum_{j\in N(i)} \bar{y}_i \right\}$$

ii. For [NIP], using the same reasoning, $(1-\bar{l}_i-\bar{x}_i)$ also lies in $[0,1]$, because a node cannot be a leaf and center at the same time, implying $z_i \le 1$. The right hand side (RHS) $\sum_{j\in N(i)}(\bar{l}_j+\bar{x}_j)$ also implies a non-negative integer. Hence, we obtain:

$$\phi_i^{NIP}(\bar{l},\bar{x}) = \max_{z_i} \left\{ z_i : z_i \le 1-\bar{l}_i - \bar{x}_i, \quad z_i \le \sum_{j\in N(i)} (\bar{l}_j + \bar{x}_j) \right\}$$

Both MPs guarantee that the corresponding SP is always feasible and bounded. Therefore, the dual SP (DSP) is also feasible and bounded by strong duality. We create following DSPs for each SP introduced above.

$$\Phi_i^{VCIP}(\bar{y}) = \min_{\alpha_i,\beta_i \geq 0} \left\{ \alpha_i(1-\bar{y}_i) + \beta_i \sum_{j\in N(i)} \bar{y}_j : \alpha_i + \beta_i = 1 \right\}$$

$$\Phi_i^{NIP}(\bar{l},\bar{x}) = \min_{\lambda_i,\omega_i \geq 0} \left\{ \lambda_i(1-\bar{l}_i - \bar{x}_i) + \omega_i \sum_{j\in N(i)} \bar{l}_j + \bar{x}_j : \lambda_i + \omega_i = 1 \right\}$$

As a result, we obtain the following Benders optimality cuts from solution $\bar{y}$ for [VCIP] and from solution $(\bar{x},\bar{l})$ for [NIP]:

$$\mu_i \leq \alpha_i(1-y_i) + \beta_i \sum_{j\in N(i)} y_j, \forall i \in V$$

$$\mu_i \leq \lambda_i(1-l_i - x_i) + \omega_i \sum_{j\in N(i)} (l_j + x_j), \forall i \in V$$

Observe that the feasible region of the DSPs are independent from the upfront fixed master variables. In fact, we can analytically approach these problems rather than solving their linear programs. Let $(1-\bar{y}_i)$ and $\sum_{j\in N(i)} \bar{y}_j$ be represented by $\Phi_{i_1}^{VCIP}$ and $\Phi_{i_2}^{VCIP}$, respectively. Further, let $(1-\bar{l}_i-\bar{x}_i)$ and $\sum_{j\in N(i)} (\bar{l}_j+\bar{x}_j)$ be represented by $\Phi_{i_1}^{NIP}$ and $\Phi_{i_2}^{NIP}$, respectively. Without loss of generality, we only present Algorithm 2 which solves the primal and dual formulations presented above for [NIP] (i.e., ϕ_i^{NIP} and Φ_i^{NIP}, respectively). Note that models ϕ_i^{VCIP} and Φ_i^{VCIP} can be solved in the same way. We then show that the algorithm satisfies the LP optimality conditions.

Proposition 2. *The primal and dual variables calculated through Algorithm 2 are optimal solutions.*

Proof. See the online supplement. □

We note that the Benders cut generated through this algorithm carries the same violation characteristic independent from the value of θ. Ahat et al. [2017] provide a detailed discussion including the proof conducted on an algorithm that solves a Bender SP in a similar fashion. However, in our problem, setting θ to one of the integral bounds (i.e., 0 or 1) is preferred over fractional values as to avoid cuts with fractional coefficients.

Remark 1. In Algorithm 2, setting $\theta = 1$ produces sparser Benders cuts.

Algorithm 2: SOLUTION OF ϕ_i^{NIP} AND Φ_i^{NIP}

```
Input: $i \in V$, $0 \leq \theta \leq 1$, $\vec{l}, \vec{x}$
if $\Phi_{i_1}^{NIP} > 0$
    if $\Phi_{i_1}^{NIP} > \Phi_{i_2}^{NIP}$
        $z_i = \Phi_{i_2}^{NIP}, \lambda_i = 0, \omega_i = 1$;
    else if $\Phi_{i_1}^{NIP} < \Phi_{i_2}^{NIP}$
        $z_i = \Phi_{i_1}^{NIP}, \lambda_i = 1, \omega_i = 0$;
    else
        $z_i = \Phi_{i_1}^{NIP}, \lambda_i = \theta, \omega_i = 1 - \theta$;
else
    if $\Phi_{i_2}^{NIP} = 0$
        $z_i = 0, \lambda_i = \theta, \omega_i = 1 - \theta$;
    else
        $z_i = 0, \lambda_i = 1, \omega_i = 0$;
```

In fact, our preliminary results indicated that generating Benders cuts with $\theta = 1$ produces slightly better results compared to setting either fractional values (e.g., 0.5) or 0 values in terms of the solution time.

It is necessary to observe that setting θ between 0 and 1 yields Benders cuts that are the convex combinations of the original constraints (i.e., Constraints (1b)-(1c) and (2b)-(2c) in [VCIP] and [NIP], respectively) removed to obtain the MPs. This is due to the fact that there exist a one-to-one correspondence between variables μ_i and z_i. By setting θ to be either 0 or 1, the cuts are the original constraints from the IP models. Therefore, we refer to our decomposition approach as a general branch-and-cut method and examine common acceleration techniques used in Benders decomposition.

6 Algorithmic Enhancements

In this section, we discuss the acceleration techniques that we utilize to speed up both decomposition methods and directly solving the IP formulations.

6.1 Constraint Tightening

In the literature, there are several studies where valid inequalities based on constraint tightening are proposed with which MPs are solved more efficiently [Sherali et al. 2010, Taşkın et al. 2012, Frank and Rebennack 2015]. Here, we show that there is a valid inequality that tightens constraints (2e) in [NIP] based on the MIS problem.

Recall that, constraints (2e) make sure that no leaf node shares an edge with another leaf. The constraints also indicate that if a node i is not selected as a leaf, then any node j within its neighborhood (i.e., $j \in N(i)$) can be a potential leaf. However, it is highly likely that some nodes within $N(j)$ are connected which implies that we might determine a better bound on the RHS of the constraint.

Definition 6. Given a graph $G = (V, E)$, the independence number of G is defined as the cardinality of the maximum independent set. Formally, it can be stated as $\Theta(G) = \max\{|U| : U \subset V, (i,j) \notin E \, \forall i, j \in U\}$.

Definition 7. Given a graph $G = (V, E)$ and set of nodes $S \subset V$, the induced subgraph $G[S]$ is a graph which contains nodes in S and all the edges that connect any two nodes contained by S.

Proposition 3. *Given a graph $G = (V, E)$, the number of leaves of any star centered at some node $i \in V$ is upper bounded by $\Theta(G[N(i)])$.*

Proof. See the online supplement. □

Remark 2. For a given graph $G = (V, E)$, the total number of feasible stars can be computed by enumerating the independent sets in $G[N(i)], \forall i \in V$ (see Kleitman and Winston [1982], Samotij [2015] for discussions on how to count the number of independent sets).

We can interpret Proposition 3 in another way such that in an induced subgraph $\hat{G}$, we cannot select more leaves than $\Theta(\hat{G})$. That is why if one solves the MIS problem for the induced graph generated by the neighborhood of each node, a good bound for the RHS of Constraint (2e) is obtained. However, MIS cannot be solved efficiently due to its complexity. Yet, for each induced graph, we can place a bound for the cardinality of the MIS.

For a given network $G = (V, E)$, let I and $\Theta(G)$ be the MIS and the independence number, respectively. Then, the number of edges for the nodes included in I is bounded above by $\Theta(G)(n - \Theta(G))$. In addition, the number of edges between all the nodes $j \in V \backslash I$ and $k \in I$ is bounded above by $\binom{\Theta(G)}{2}$. Therefore, it can be stated that $m \leq \Theta(G)(n - \Theta(G)) + \binom{\Theta(G)}{2}$. Rearranging the mathematical inequality, one can obtain the following standard UB for $\Theta(G)$ stated as $\gamma(G)$ [Schiermeyer 2019]:

$$\Theta(G) \leq \gamma(G) = \frac{1}{2}(1 + \sqrt{(2n-1)^2 - 8m}) \tag{5}$$

For every node i, we first form an induced graph $G[N(i)]$. Then, we calculate the bound (i.e., $\gamma(G[N(i)])$) presented in Inequality (5) and rephrase constraints (2e);

$$\sum_{j \in N(i)} l_j \leq \gamma(G[N(i)])(1 - l_i), \quad \forall i \in V \tag{6}$$

6.2 Upper Bounds

Providing initial bounds on the objective value can help accelerate the selected solution methods. In the literature, methods to accomplish this include introducing valid inequalities [Ahat et al. 2017], solving the relaxed version of the model [Chen and Miller-Hooks 2012], using the Lagrangian relaxation [Holmberg 1994], and employing heuristic approaches [Contreras et al. 2011].

In our problem, it is also important to initially bound the objective function $\sum_{i \in V} \mu_i$ to get high quality initial solutions thereby obtaining faster convergence. The very first natural UB on the objective value is calculated as $n-1$. A star can have at most $n-1$ adjacent nodes where such star consists of a single center node. Then, the UB can be stated as:

$$\sum_{i \in V} \mu_i \leq n - 1 \tag{7}$$

Another important point is that the objective function (i.e., the size of the neighborhood of a star) is only affected by the first and second degree nodes of the center node. Hence, we can introduce another UB which changes according to the node selected as center and is calculated by the summation of the size of the first and second degree nodes of the center.

$$\sum_{i \in V} \mu_i \leq \sum_{i \in V} (|N(i)| + |\bar{N}^2(i)|) x_i \tag{8}$$

Note that once a first degree node $j \in N(i)$ is accepted as a leaf node, the RHS presented in inequality (8) decreases by one. The key observation is that if node j produces a unique path to any second degree node, then it can be considered a leaf node. In this case, we can decrease $|N(i)| + |\bar{N}^2(i)|$ by one thereby tightening the RHS. If node j is not a leaf node in a feasible solution, then its contribution would be one to the objective value, which is bounded above by the contribution of the second degree nodes uniquely reached via node j. Hence, it stays as a valid bound. Based on this argument, we propose Algorithm 3 which approximates a bound on the objective value for every candidate node as the center.

In Fig. 1, valid inequality (8) produces a RHS of $|N(1)| + |\bar{N}^2(1)| = 9$ when node 1 is selected as the center node. According to Algorithm 3, nodes 3 and 9 individually produce at least one unique path for some nodes in $\bar{N}^2(1)$. Hence, both can be considered candidate leaves nodes thereby setting the RHS as 7, which is clearly tighter than the previous bound. This is also exactly the maximum size of any open neighborhood for a star centered at 1. Note that if both 3 and 9 could not be leaf nodes in another setting where 3 and 9 were connected, then a feasible solution would consider either node as a leaf, which would keep the RHS calculated as a valid bound.

After running Algorithm 3, a new bound $\delta_i, \forall i \in V$, which is in practice tighter than the former ones, is obtained. Then, the following is a valid inequality for the IPs and MPs of the Benders decomposition algorithms:

$$\sum_{i \in V} \mu_i \leq \sum_{i \in V} \delta_i x_i \tag{9}$$

Notice that μ_i replaces z_i in the original formulations where z_i is a binary variable. Therefore, the next natural UB is to bound each single μ_i based on the binary restriction. We note that this *one-to-one* correspondence between μ_i and z_i also indicates that the Benders cuts generated are the convex combination of the original constraints removed from the model to obtain a restricted MP. In other words, our Benders framework can be viewed as a cutting-plane algorithm. The upper bound constraints are:

$$\mu_i \leq 1, \quad \forall i \in V \tag{10}$$

Although constraints (10) are the tightest UB one can obtain for each individual μ_i, we emphasize on that incorporating this UB increases the solution time and decreases solution quality in every single instance of the decomposition implementation. We believe that this is attributed to the fact that its addition changes the pre-solve and heuristic routines of the solver and that this tight UB is simple enough for the solver to identify on its own. Therefore, the benefits of its potential addition are outweighed by its drawbacks. Note that we could take a similar approach and remove the binary restriction on z_i in the IP models; however we observed that the average optimality gap across instances increases in this situation. Therefore, our discussion remains valid only for the restricted MPs.

6.3 Parameter Tuning

Tuning certain CPLEX parameters when solving the MP might yield a faster convergence [Bai and Rubin 2009, Botton et al. 2013, Dalal and Üster 2017]. In our study, we also alter

Algorithm 3: BOUND STRENGTHENING AT A GIVEN STAR-CENTER $i \in V$

```
Input: $i \in V$
$\delta_i = \sigma = 0$;
for $k \in \bar{N}^2(i)$ do
    $pred[k] = -1$;
    $visited[k] = 0$;
for $j \in N(i)$ do
    $unique[j] = |(j,k) \in E : k \in \bar{N}^2(i)|$;
    for $k \in \bar{N}^2(i)$ do
        if $(j,k) \in E$
            if $visited[k] = 0$
                $pred[k] = j$;
            else if $visited[k] = 1$
                $unique[j]--$;
                $unique[pred[k]]--$;
            else
                $unique[j]--$;
            $visited[k]++$;
for $j \in N(i)$ do
    if $unique[j] > 0$
        $\sigma++$;
if $\sigma > 0$
    $\delta_i = |N(i)| + |\bar{N}^2(i)| - \sigma$;
else
    if $|\bar{N}^2(i)| = 0$
        $\delta_i = |N(i)|$;
    else
        $\delta_i = |N(i)| + |\bar{N}^2(i)| - 1$;
return $\delta_i$
```

some default parameters to speed up the convergence of our decomposition method and these changes help to decrease the solution time by a considerable amount.

For our decomposition implementation, we switch the MIP emphasis to optimality. Since finding a feasible star is a relatively easy task, we prefer CPLEX to focus on optimality over feasibility. Second, the strategy for variable selection is changed to strong branching with which CPLEX puts more effort on identifying the most favorable branch. Note that strong branching goes through each branch to identify the best one in terms of the contribution to the objective value. In certain scenarios, this operation might be computationally challenging. Last, we set the relaxation induced neighborhood search (RINS) as 1,000 where CPLEX applies the RINS heuristic at every 1,000 nodes. When solving the IPs directly, we prefer the default CPLEX settings since no consistent improvement in terms of the solution time and/or quality is observed.

6.4 Separation of Integer and Fractional Solutions

In a branch-and-Benders implementation or, equivalently, Modern Benders decomposition, the MP is solved only once. This is in contrast to the traditional Benders method that solves each MP to optimality. Whenever the solver identifies an incumbent solution, a callback function (the generic callback in CPLEX [IBM 2017]) is triggered and the branch-and-bound tree is halted. If the incumbent solution overestimates the objective (i.e., underestimates for a minimization problem) meaning that there is a cut violated by the integer solution, then Benders cuts (i.e., lazy constraints) are generated through the dual solutions.

As suggested by Fischetti et al. [2016], one can also separate the fractional solutions where a Benders cut (i.e., a user cut) can be generated at a non-integer solution before branching. If no violated cut exists, then branching takes place as usual. Otherwise, a violated cut is generated based on a fractional solution. However, the cut generation for a fractional solution might not be as straightforward as the process for an incumbent solution. In our study, fortunately, the generation of a cut at a fractional solution can be done using the same procedure as for an incumbent solution and only requires the comparison of two objective components as shown in Algorithm 2.

6.5 Warm-Start

Several warm starting methods have been shown to be effective, especially when solution methods struggle to find incumbent solutions. Extreme points or valid cuts might be generated via solving relaxed primal problems [Adulyasak et al. 2015], deflecting the current master solution [Rahmaniani et al. 2018], or designing meta-heuristic algorithms [Emde et al. 2020]. In our experiments, we use the *ratio-based greedy* approach proposed by Vogiatzis and Camur [2019] to generate a set of high quality initial solutions. The heuristic is shown to have an approximation guarantee of $O(\Delta_i)$ for node i where Δ_i is the degree of node $i \in V$ which is the center of a candidate induced star.

The algorithm has two phases and continuously checks the ratio between the possible gain and loss of adding a node into a star in terms of the cardinality of the open neighborhood. In the first phase, we pick a node with the highest contribution to the objective where placing the node into the star does not decrease the contribution of the other candidate leaves. In the second phase, we look for a node which yields the highest ratio whose denominator keeps track of the potential loss that could occur due to the adjacent nodes. For more details about the heuristic and its pseudocode, we refer to reader to Vogiatzis and Camur [2019].

While the UBs introduced in Section 6.2 help the solver to tighten the dual bounds, our intention with using warm-start is to help with the primal bounds. It is crucial to point out that we use the valid inequalities (see Sections 6.1 and 6.2) if applicable for both IP models for a fair comparison. For the warm-start strategy, we have a set of experiments to see its impact on each model in Section 7.1.1.

7 Experimental Results

All the experiments are conducted using Java and CPLEX 12.8.1 on an Intel Core i7-6500 CPU at 3.10GHz laptop with 16 GB of RAM. During the implementation of the decomposition algorithm, we utilize the callback function feature to add the Benders cuts as lazy cuts and user defined cuts. While Algorithm 3 and ratio-based heuristic are implemented in Java, the UB (5) introduced in Section 6.2 is calculated in R using the *igraph* library. All data sets and code sources used in our study are available online at https://github.com/mcamur/SDC.

7.1 Randomly Generated Instances

We first randomly generate test cases according to three well-known models through *igraph* [Igraph 2020]: i) *Barabási–Albert (BA)* (i.e., scale-free networks), ii) *Erdös–Rényi (ER)* (i.e., random networks), and iii) *Watts–Strogatz (WS)* (i.e., small-world networks). We consider instances with $n \in \{500, 600, 700, 800, 900, 1000\}$ regardless of the model type, as each model has its own parametric settings, which are summarized in Table 1.

In the BA model, we consider g in the set $\{10, 12, 14, 16\}$. For ER model, we set pr as $\frac{i}{n}$ where $i \in \{10, 20, 30, 40, 50\}$ and $i \in \{20, 30, 40, 50, 60\}$ for $\{500, 600, 700\}$ and

Table 1 Parameter settings

Model	Parameter	Definition
BA	g	the number of edges generated at each step
ER	pr	probability of adding an edge between randomly selected two nodes
WS	r	the rewiring probability
	nei	the average degree of each node

$\{800, 900, 1000\}$ nodes, respectively. Finally, in the WS model r is pulled from the set $\{0.3, 0.5, 0.7\}$ in every instance, and nei is in the set $\{12, 14, 16\}$ and $\{14, 16, 18\}$ for $\{500, 600, 700\}$ and $\{800, 900, 1000\}$ nodes, respectively. Overall, the total number of instances generated in the BA, ER, and WS models are 24, 30, and 54, respectively.

We set a time limit of 3,600 seconds, where we also take the time required by Algorithm 3 into consideration. We first test the impact of warm-start on each solution technique and then proceed to the full set of analysis conducted on the randomly generated networks. We present the comparisons between [NIP], [VCIP], [DNIP], and [DVCIP] for each model where [DNIP] and [DVCIP] represent the decomposition implementations for the IP models [NIP] and [VCIP], respectively.

7.1.1 Warm-Start Analysis

We examine the impact of warm-start on the randomly generated networks where $n \in \{500, 700, 900\}$. The main goal is to decide whether performing the full analysis should be done with or without warm-start in each solution technique (i.e., [NIP], [VCIP], [DNIP] and [DVCIP]) should be done.

The detailed analysis regarding how warm-start impacts each solution technique can be found in the online supplement. Our results have three main findings: i) the solver does not face a difficulty in improving the primal bounds, which can also be practically observed when engine logs are analyzed, ii) warm-start does not improve the solution quality in terms of optimality gaps in many instances, and iii) one cannot reach a sharp conclusion whether warm-starting both IP models and MPs via an effective heuristic solution works well or not. As a result, we decide to move into the full analysis without using warm-start as an acceleration technique.

7.1.2 Full Analysis

In this section, we compare the performance of the solution techniques on all randomly generated networks. If the optimal solution is not obtained by the time limit (TL), we report the optimality gap provided by CPLEX. For each instance, we share: i) the time taken to reach the solution in seconds, ii) the optimality gap returned in %, and iii) the number of branch-and-bound nodes saturated by the solver. In addition, we show n, m, the density of the graph represented by D (i.e., $2m/[n(n-1)]$), and the corresponding parameters (see Table 1). Tables 3, 4 and 5 show the results for the BA, ER, and WS models, respectively.

Table 2 Summary of Results

	BA Model - 24 instances				**ER Model - 30 instances**				**WS Model - 54 instances**			
	[NIP]	**[VCIP]**	**[DNIP]**	**[DVCIP]**	**[NIP]**	**[VCIP]**	**[DNIP]**	**[DVCIP]**	**[NIP]**	**[VCIP]**	**[DNIP]**	**[DVCIP]**
Optimal	10	14	**20**	19	11	12	**14**	**14**	13	21	**35**	34
Pct	42	58	**83**	79	37	40	**47**	**47**	24	39	**65**	63
Ave Gap	8.82	7.16	**0.44**	1.66	12.06	10.97	**4.02**	4.29	24.71	20.95	**2.22**	2.61
Best	3	6	**12**	3	3	4	**14**	9	12	6	**30**	6

We start our analysis with a summary of the computational results in Table 2. For each network model, we compare all four methods in terms of: i) the number of instances solved to optimality, ii) the percentage of instances where optimal solutions were found, iii) the average optimality gap over all instances, and iv) the number of instances where a method shows the best performance. Note that the best performance is first identified based on the optimality gaps. If more than one method reaches the optimal solution for the same instance, then we compare the solution times.

We observe that the decomposition implementations significantly outperform the [NIP] and [VCIP]. We do note that [VCIP] turns out to be the slightly better IP formulation; however, our analysis indicates that [DNIP] outperforms [DVCIP].

To start with, both decomposition algorithms show a considerably high performance in the BA model where [DNIP] and [DVCIP] solve two-times and and two-third-times more instances to optimality compared to their corresponding IPs, respectively. However, when it comes to the ER model, the performance of the two algorithms worsens, yet is still better than the IPs, and they can only solve 14 of the instances, which is roughly half of the total number of ER instances. It is important to mention that the instances that cannot be solved to optimality are the same instances in both algorithms with two exceptions ($n = 800, pr = 0.038$ and $n = 1000, pr = 0.03$). Furthermore, it is worth mentioning that there is no single instance in both BA and ER models where either of the IP models reaches the optimal solution while decomposition methods do not.

The reason behind the lower performance shown via decomposition implementations in the ER model compared to the BA models can be explained from two perspectives. First,

Table 3 The computational results for the BA Model

				[NIP]			[VCIP]			[DNIP]			[DVCIP]		
n	**m**	**D**	**g**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**
500	4945	0.04	10	58.16	0	9639	**19.95**	**0**	**3144**	78.97	0	1678	140.22	0	1517
500	5922	0.048	12	**88.61**	**0**	**17839**	228.01	0	21646	133.51	0	2992	233.48	0	2482
500	6895	0.055	14	TL	11.08	238608	**309.08**	**0**	**22597**	593.65	0	9663	569.76	0	4364
500	7864	0.063	16	TL	9.48	363414	3292.34	0	162751	**1328.52**	**0**	**21795**	1668.44	0	11623
600	5945	0.033	10	**18.43**	**0**	**3910**	260.31	0	10526	89.04	0	2024	106.83	0	1053
600	7122	0.034	12	1824.26	0	139597	310.81	0	16615	**203.28**	**0**	**3361**	324.7	0	2580
600	8295	0.046	14	171.11	0	22949	**459.01**	**0**	**22185**	641.98	0	8188	624.58	0	3950
600	9464	0.053	16	TL	10.81	169044	TL	13.21	58488	**1605.87**	**0**	**24178**	2777.47	0	16317
700	6945	0.028	10	**141.95**	**0**	**13754**	363.34	0	18020	316.04	0	4811	169.49	0	1785
700	8322	0.034	12	3519.86	0	183841	485.29	0	29795	700.25	0	8734	**442.82**	**0**	**3670**
700	9695	0.04	14	TL	13.95	131019	TL	14.50	65264	**1883.1**	**0**	**23709**	2021.94	0	14561
700	11064	0.045	16	TL	13.47	148775	TL	15.82	49246	**TL**	**2.15**	**37234**	TL	3.09	18598
800	7945	0.025	10	201.33	0	9630	**51.12**	**0**	**4839**	154.34	0	2390	100.9	0	816
800	9522	0.03	12	3059.28	0	125518	TL	17.33	51590	818.19	0	7947	**596.68**	**0**	**3782**
800	11095	0.035	14	TL	19.08	102405	TL	20.75	54750	**1528.24**	**0**	**15288**	2311.62	0	10275
800	12664	0.04	16	TL	17.09	111822	TL	20.13	57051	**TL**	**1.65**	34356	TL	11.77	8614
900	8945	0.022	10	1018.83	0	58500	**122.62**	**0**	**4480**	275.33	0	3626	339.92	0	2156
900	10722	0.027	12	TL	3.00	135767	TL	10.80	49816	**961.72**	**0**	**8640**	1393.53	0	7405
900	12495	0.031	14	TL	16.41	90017	**946.41**	**0**	**36842**	1964.96	0	19329	2432.43	0	11498
900	14264	0.035	16	TL	19.04	130576	TL	17.77	41565	**TL**	**1.15**	**29232**	TL	10.71	8375
1000	9945	0.02	10	TL	20.45	82900	589.65	0	21920	631.54	0	5953	**503.15**	**0**	**2979**
1000	11922	0.024	12	TL	15.68	80103	2993.83	0	62964	**1596.08**	**0**	**16203**	1925.37	0	10223
1000	13895	0.028	14	TL	22.97	94927	TL	23.15	33999	**2416.05**	**0**	**20431**	TL	2.63	11192
1000	15864	0.032	16	TL	19.12	90223	TL	18.40	38594	**TL**	**5.66**	**20919**	TL	11.65	6166

the average edge numbers and the average graph densities are $9,656/0.036$ and $13,016/0.046$ in the BA and ER models, respectively. In other words, the problem gets harder to solve with higher edge numbers and/or a denser graph. Also, the density of graphs in the ER model increases at a faster rate than the other models for our selected parameters. Second, we examine the number of clique inequalities added by the solver. For instance, while the solver generates 184 clique inequalities on average in the BA model in [DNIP], this average drops to 10 in the ER model. For the [DVCIP], it produces, on average, 2 clique inequalities in the BA model and only 0.8 in the ER model. As a potential future research direction, one might be interested in incorporating clique inequalities for each triangle in a cutting-plane manner to test whether it would strengthen the decomposition implementations.

In the WS model, while [DNIP] solves nearly threefold the number of instances solved by [NIP], [DVCIP] solves one and a half times more than the instances solved by [VCIP]. For the instances that are not solved to optimality, [DNIP] and [DVCIP] give an average of 6.30% and 7.05% optimality gaps, respectively. While both decomposition implementations far outperform the corresponding IPs in the majority of the instances with respect to the solution status, we observe only two instances where they fail to reach the optimal solution while

Table 4 The computational results for the ER Model.

				[NIP]			[VCIP]			[DNIP]			[DVCIP]		
n	**m**	**D**	**pr**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**
500	2469	0.02	0.02	7.98	0	159	5.13	0	139	**0.52**	**0**	**0**	0.77	0	0
500	4999	0.041	0.04	52.88	0	1047	31.19	0	857	**21.61**	**0**	**231**	39.86	0	147
500	7537	0.061	0.06	TL	19.21	82190	TL	18.81	79125	**1611.1**	**0**	**12939**	2406.97	0	11377
500	9870	0.08	0.08	TL	11.13	99443	TL	11.10	72571	**TL**	**3.90**	**22869**	TL	6.15	5690
500	12466	0.1	0.1	TL	7.71	109527	TL	7.69	75616	TL	7.36	10978	**TL**	**7.30**	**3426**
600	2948	0.017	0.017	5.05	0	0	7.34	0	0	**1.01**	**0**	**0**	1.09	0	0
600	6009	0.034	0.033	**22.81**	**0**	**1215**	34.24	0	971	101.22	0	721	121.21	0	437
600	8993	0.051	0.05	TL	26.15	50422	TL	24.34	89329	2056.21	0	11162	**1578.78**	**0**	**5446**
600	11967	0.067	0.067	TL	13.97	113537	TL	15.06	55589	**TL**	**7.33**	**10120**	TL	8.55	4122
600	14993	0.084	0.083	**TL**	**7.26**	**115613**	TL	11.78	39343	TL	7.64	10802	TL	10.11	3057
700	3483	0.015	0.014	11.7	0	57	7.12	0	0	**0.78**	**0**	**0**	1.28	0	0
700	6895	0.029	0.029	35.81	0	1064	31.77	0	973	**30.94**	**0**	**186**	54.6	0	265
700	10526	0.044	0.043	TL	30.30	22403	TL	33.38	98316	3182.75	0	15534	**2024.99**	**0**	**5393**
700	13943	0.057	0.057	TL	18.36	48110	TL	17.07	33905	**TL**	**5.80**	**5886**	TL	6.47	6557
700	17713	0.073	0.071	TL	11.60	48468	TL	11.29	26307	**TL**	**7.36**	**7383**	TL	9.12	3477
800	7890	0.025	0.025	28.81	0	903	**7.34**	**0**	**0**	7.89	0	50	12.11	0	25
800	11969	0.038	0.038	TL	33.70	102977	**34.24**	**0**	**971**	TL	1.45	10888	3440.81	0	6737
800	15859	0.05	0.05	TL	24.06	62404	TL	24.34	89329	**TL**	**9.63**	**0**	TL	10.44	0
800	20003	0.063	0.063	TL	15.14	52575	TL	15.06	55589	TL	9.12	3269	**TL**	**8.48**	**2920**
800	19910	0.063	0.075	TL	14.47	36246	TL	11.78	39343	TL	8.21	3842	**TL**	**7.55**	**2873**
900	9064	0.023	0.022	50.42	0	1241	**39.3**	**0**	**1025**	50.43	0	343	60.67	0	202
900	13418	0.034	0.033	1285.1	0	14926	**265.01**	**0**	**1737**	2182.82	0	0	1957.7	0	2250
900	17979	0.045	0.044	TL	29.64	28104	TL	23.34	21991	**TL**	**8.47**	**2585**	TL	8.90	4589
900	22397	0.056	0.056	TL	19.28	17336	TL	16.33	9846	TL	8.84	3976	**TL**	**8.32**	**1829**
900	22349	0.056	0.067	TL	15.60	36047	TL	16.58	4784	TL	8.60	3742	**TL**	**8.46**	**2179**
1000	10003	0.021	0.02	35.65	0	1408	32.41	0	838	**8.28**	**0**	**34**	8.82	0	28
1000	14926	0.03	0.03	**164.97**	**0**	**5218**	333.14	0	1918	3540.07	0	5114	TL	3.26	9655
1000	20008	0.041	0.04	TL	25.81	50235	TL	33.36	11367	**TL**	**8.66**	**2083**	TL	9.69	4013
1000	24896	0.05	0.05	TL	18.53	30891	TL	20.61	3325	TL	9.84	1850	**TL**	**8.04**	**2432**
1000	25015	0.051	0.06	TL	19.97	31503	TL	17.24	3470	TL	8.43	1846	**TL**	**7.90**	**1784**

[VCIP] does (see the instances $(n = 1000, nei = 16, p = 0.5)$ and $(n = 1000, nei = 16, p = 0.7)$ in Table 5).

Note that both IP formulations show poorer performances on the WS model compared to the other network models. First, we believe that the number of clique inequalities is again a driving factor to reach the optimal solution especially in [VCIP]. For example, for the instances solved to optimality by [VCIP], the solver produces 364 clique inequalities on average. On the other hand, this number drops to 30 for instances that fail to solve to optimality. Further, we expect to have more feasible stars in WS model than in both BA and ER models. We believe this is due to the fact that the small world nature of the WS model implies that there are many stars with open neighborhoods of similar size centered at i because nodes tend to share a common neighbor. Potentially, this symmetry may cause issues in solving the IP models. One might be interested in examining symmetry breaking

Table 5 The computational results for the WS Model

					[NIP]			[VCIP]			[DNIP]			[DVCIP]		
n	**m**	**D**	**nei**	**r**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**
500	6000	0.049	12	0.3	TL	9.02	25308	TL	21.28	91678	51.7	0	227	**24.28**	**0**	**105**
500	6000	0.049	12	0.5	TL	23.43	77774	2417.97	0	128770	**574.56**	**0**	**4370**	663.63	0	3293
500	6000	0.049	12	0.7	TL	20.69	84833	2151.62	0	73650	**232.84**	**0**	**1520**	358.92	0	1173
500	7000	0.057	14	0.3	TL	28.14	76538	TL	28.50	72283	**482.08**	**0**	**3171**	612.67	0	2075
500	7000	0.057	14	0.5	TL	18.96	106315	TL	19.36	127357	**221.26**	**0**	**0**	278.13	0	1355
500	7000	0.057	14	0.7	TL	20.04	86474	TL	17.37	86253	**752.48**	**0**	**4378**	873.46	0	3004
500	8000	0.065	16	0.3	TL	24.59	195299	TL	25.43	77650	**1831.41**	**0**	**14462**	2941.25	0	12349
500	8000	0.065	16	0.5	TL	18.76	177452	TL	20.46	78010	**2915.64**	**0**	**21736**	TL	5.16	10613
500	8000	0.065	16	0.7	TL	20.17	199549	TL	20.96	89842	**3462.82**	**0**	**24098**	TL	4.97	9936
600	7200	0.041	12	0.3	**62.47**	**0**	**3084**	63.42	0	1209	240.25	0	1472	343.41	0	1123
600	7200	0.041	12	0.5	TL	20.20	62763	**63.52**	**0**	**1149**	145.35	0	681	114.59	0	353
600	7200	0.041	12	0.7	**37.89**	**0**	**2018**	60.7	0	1153	357.76	0	1785	355.04	0	1176
600	8400	0.047	14	0.3	TL	24.59	36714	TL	34.15	80105	**114.52**	**0**	**390**	118.75	0	201
600	8400	0.047	14	0.5	TL	32.60	58742	TL	30.85	110016	**2154.81**	**0**	**12498**	3193.15	0	10954
600	8400	0.047	14	0.7	TL	30.73	31864	TL	32.09	119002	2175.71	0	11386	**1162.07**	**0**	**5026**
600	9600	0.054	16	0.3	TL	36.59	69550	TL	31.87	67384	**2617.79**	**0**	**13338**	3161.54	0	11501
600	9600	0.054	16	0.5	TL	20.88	128234	TL	19.55	67171	**1370.85**	**0**	**0**	2021.53	0	0
600	9600	0.054	16	0.7	TL	21.17	108631	TL	24.01	61322	**2842.7**	**0**	**14824**	TL	2.39	8895
700	8400	0.035	12	0.3	48.16	0	1368	79.6	0	1339	**10.64**	**0**	**71**	41.4	0	173
700	8400	0.035	12	0.5	TL	20.46	73254	**93.55**	**0**	**1319**	310.13	0	1105	385.78	0	1608
700	8400	0.035	12	0.7	**43.91**	**0**	**2246**	93.88	0	1314	241.96	0	1209	327.95	0	795
700	9800	0.041	14	0.3	TL	36.23	101988	TL	50.48	86444	468.17	0	1091	**300.63**	**0**	**479**
700	9800	0.041	14	0.5	TL	31.33	63199	**183.8**	**0**	**1379**	795.9	0	2575	835.1	0	1843
700	9800	0.041	14	0.7	TL	25.24	55767	**125.84**	**0**	**1363**	195.25	0	889	513.84	0	1325
700	11200	0.046	16	0.3	TL	26.22	37692	TL	27.42	56269	**105.26**	**0**	**202**	131.25	0	170
700	11200	0.046	16	0.5	TL	33.45	40710	TL	30.36	61073	**TL**	**4.99**	**10109**	TL	5.57	6050
700	11200	0.046	16	0.7	TL	29.74	45132	TL	23.14	57484	**1399.7**	**0**	**0**	3391.95	0	2902
800	11200	0.036	14	0.3	**98.88**	**0**	**3825**	286.84	0	1602	1306.57	0	3667	1412.73	0	3405
800	11200	0.036	14	0.5	**105.97**	**0**	**6467**	172.33	0	1576	TL	4.00	7536	2785.38	0	8209
800	11200	0.036	14	0.7	**106.27**	**0**	**4124**	169.39	0	1559	1188.81	0	3737	1340.02	0	2292
800	12800	0.041	16	0.3	TL	51.67	72137	TL	51.78	52088	TL	2.45	9949	**2029.77**	**0**	**6406**
800	12800	0.041	16	0.5	TL	39.25	36231	TL	33.31	60910	**TL**	**3.48**	**5719**	TL	3.66	7985
800	12800	0.041	16	0.7	TL	32.89	58367	TL	34.33	52240	**TL**	**6.27**	**9798**	TL	8.36	6806
800	14400	0.046	18	0.3	TL	41.58	49452	TL	45.56	42441	**TL**	**6.88**	**5977**	TL	7.96	7436
800	14400	0.046	18	0.5	TL	26.99	80005	TL	26.96	41281	**TL**	**4.82**	**5382**	TL	5.50	7904
800	14400	0.046	18	0.7	TL	30.80	50056	TL	25.96	45543	**TL**	**7.06**	**4540**	TL	7.90	6258
900	12600	0.032	14	0.3	**108.51**	**0**	**3025**	280.17	0	1793	1591.75	0	3298	1441.56	0	1868
900	12600	0.032	14	0.5	**107.42**	**0**	**4075**	258.75	0	1749	1136.86	0	2300	999.72	0	3231
900	12600	0.032	14	0.7	**104.11**	**0**	**5085**	252.56	0	1733	1641.69	0	4338	1950.74	0	5006
900	14400	0.036	16	0.3	TL	56.89	90708	TL	57.94	50475	TL	3.86	6021	**TL**	**2.94**	**9397**
900	14400	0.036	16	0.5	TL	33.72	69199	TL	35.05	44805	1333.74	0	2868	**1093.58**	**0**	**3233**
900	14400	0.036	16	0.7	TL	39.68	66614	TL	42.36	47839	**TL**	**5.48**	**3897**	TL	6.89	7582
900	16200	0.041	18	0.3	TL	46.37	56945	TL	47.88	30751	**TL**	**8.94**	**4912**	TL	9.60	5537
900	16200	0.041	18	0.5	TL	34.79	74381	TL	34.74	29603	**TL**	**10.20**	**3297**	TL	10.96	3975
900	16200	0.041	18	0.7	TL	32.97	51064	TL	35.54	27628	**TL**	**5.68**	**2982**	TL	6.55	5652
1000	14000	0.029	14	0.3	**127.2**	**0**	**2867**	241.73	0	1978	2027.07	0	3646	1490.76	0	4950
1000	14000	0.029	14	0.5	**100.75**	**0**	**3070**	217.52	0	1922	1328.25	0	2301	894.33	0	2560
1000	14000	0.029	14	0.7	**84.9**	**0**	**1996**	184.2	0	1781	202.7	0	375	281.6	0	719
1000	16000	0.033	16	0.3	TL	77.90	82180	TL	75.20	35581	**TL**	**9.31**	**4730**	TL	10.46	7621
1000	16000	0.033	16	0.5	TL	50.80	113819	**505.5**6	**0**	**1992**	TL	8.35	3539	TL	8.35	6967
1000	16000	0.033	16	0.7	TL	39.52	121941	**317.29**	**0**	**1961**	TL	4.19	4891	TL	5.47	7186
1000	18000	0.037	18	0.3	TL	48.99	54205	TL	47.41	23574	**TL**	**7.15**	**4069**	TL	9.52	7427
1000	18000	0.037	18	0.5	TL	43.92	88314	TL	42.44	18365	**TL**	**10.69**	**3309**	TL	11.64	5319
1000	18000	0.037	18	0.7	TL	32.27	61227	TL	37.70	23845	**TL**	**5.87**	**2756**	TL	7.08	6828

techniques during the search process for WS networks in the future. Lastly, since [NIP] is not as tight as [VCIP] (please see the proof of Theorem 1 in the online supplement), we believe that the graphs generated by the WS model may be more challenging for [NIP].

We now look at the cases where both decomposition algorithms reach the optimal solution and make a comparison in terms of the solution time. As shown in Fig. 4, [DNIP] outperforms [DVCIP] solution-time-wise and reaches the optimal solution quicker in 12 instances. As for the ER model, we observe slightly a different trend. For the instances where both method take more than 1,000 seconds to solve (i.e., four instances), [DVCIP] performs better and outperforms [DNIP] in three instances (see Fig. 5). Even though overall [DNIP] produces a better solution time in more instances (i.e., 10 out of 13 instances), [DVCIP] is 75 seconds faster than [DNIP] on average. Lastly, as for the WS model, [DNIP] notably outperforms [DVCIP] as depicted in Figs. 6 and reaches the optimal solution faster in 22 instances out of 32. On average, [DNIP] is 139 seconds faster than [DVCIP].

Figure 4 Solution time comparison between [DNIP] and [DVCIP] in the BA model

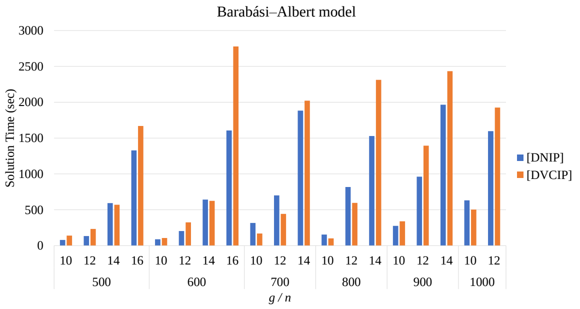


Figure 5 Solution time comparison between [DNIP] and [DVCIP] in the ER model

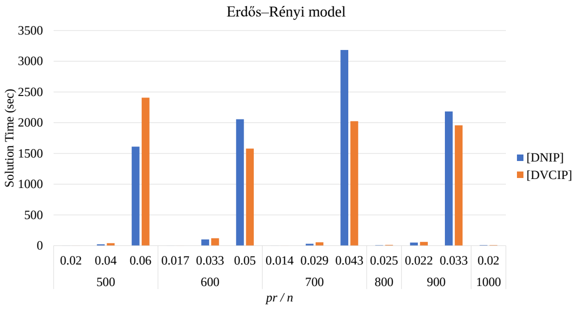
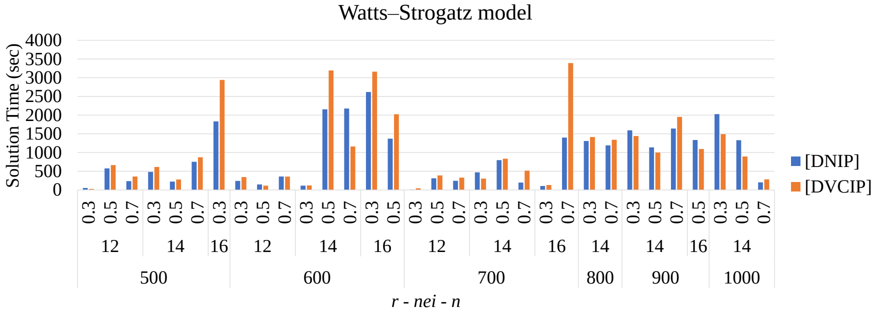


Figure 6 Solution time comparison between [DNIP] and [DVCIP] in the WS model



Although the new IP formulation [NIP] could not compete with the formulation [VCIP], the decomposition implementation [DNIP] shows a better performance compared to [DVCIP] in terms of both solution time and solution quality in more instances. First, as mentioned earlier, the number of constraints is bounded by $O(n)$ in [NIP]. and its number of non-zero coefficients are lower compared to [VCIP]. Second, the number of non-zero coefficients is further decreased in [NIP] by constraint tightening(Section 6.1). Third, when decomposing [NIP], the two constraints causing the increase in the number of non-zero coefficients – constraints (2b) and (2c) – are placed in the SP. In fact, as discussed previously, one the MPs of [VCIP] and [NIP] have i) $5n+6m$ and $3n+4m$ non-zero coefficients, and ii) $2n+m+1$ and $2n+1$ constraints, respectively. All these facts imply that the restricted MP generated via [NIP] is more efficient than the MP generated via [VCIP]. Note that even though Theorem 1 states that [VCIP] is stronger than [NIP] with respect to LP-relaxations, we observe that the root node relaxations turn out to be same in all randomly generated instances, implying that the size of the formulations likely plays an important role in the quality of solving them. Lastly, the number clique inequalities created by the solver in [DNIP] is significantly higher than [DVCIP] on average in all three network models. Taking all these into consideration, it makes sense that [DNIP] produces more fruitful results than [DVCIP].

Figure 7 The optimality gap comparisons in [NIP], [VCIP], [DNIP] and [DVCIP] in the BA model

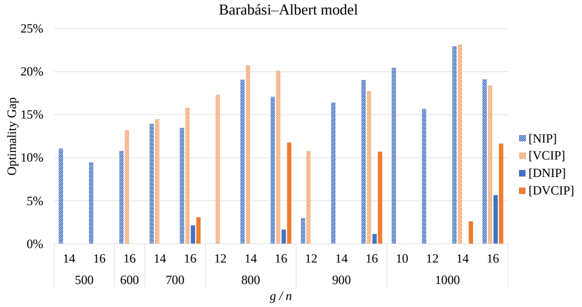


Figure 8 The optimality gap comparisons in [NIP], [VCIP], [DNIP] and [DVCIP] in the ER model

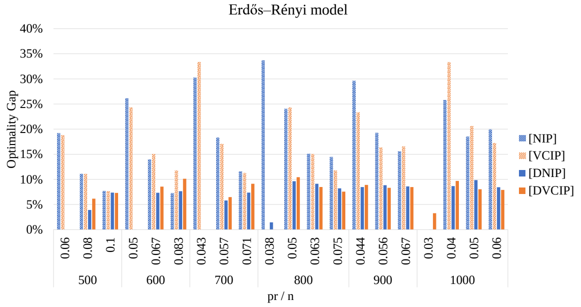
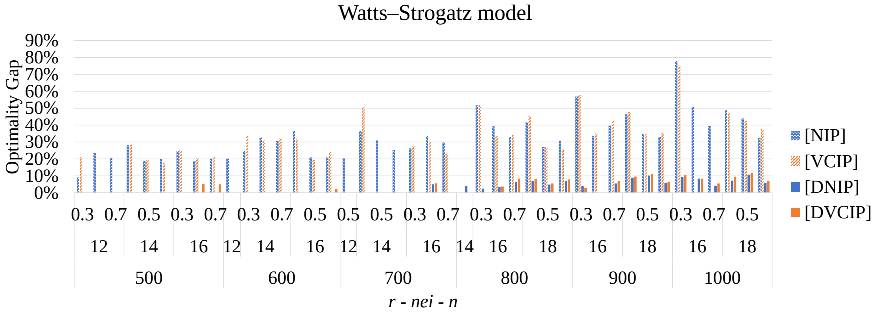


Figure 9 The optimality gap comparisons in [NIP], [VCIP], [DNIP] and [DVCIP] in the WS model



Lastly, we compare all four methods in terms of the optimality gaps to solidify our point when one of the methods cannot reach the optimal solution. We present Figs. 7, 8, and 9 where it can be clearly seen that both decomposition implementations show a better performance than their corresponding IPs. Fig. 7 illustrates that [DNIP] is the best method when we have a graph following the properties of the BA model. When we cannot reach the optimal solution with it, the optimality gap does not exceed 5.66%. On the other hand, both IP models returns over 12.5% optimality gaps for the instances shown in Fig 7. The ER model turned out to be the most challenging model where even decomposition methods had a hard time to converge to the optimal solution for certain instances (see Fig. 8) whose potential reasons are discussed earlier. Yet, [DNIP] and [DVCIP] never return an optimality gap larger than 9.84% and 10.44%, respectively. As for the WS model, Fig. 9 depicts that as the number of nodes go up, both IP models start returning poorer optimality gaps with few exceptions. On the other hand, both decomposition implementations show a strong performance with the instances up to 800. When the number of nodes is 800 or more, the average optimality gaps become 6% and 6.4% in [DNIP] and [DVCIP], respectively; in certain cases which is still better than solving the IP model directly.

7.2 Protein-Protein Interaction Networks (PPINs)

In this section, we analyze the datasets of two organisms: i) *Helicobacter Pylori (HP)* and ii) *Staphylococcus Aureus (SA)* obtained by Szklarczyk et al. [2014]. Each data set is converted into a PPIN as follows. A protein is represented by a node that is connected by an edge to all other proteins if there exists an interaction. Each interaction is associated with an interaction score defined within the range of $[0, 1000]$.

With this configuration, the networks created turn out to be highly dense graphs with diameter equal to six. The number of nodes and edges are $(n = 1,570, m = 89,507)$ and $(n = 2,852, m = 146,783)$ for HP and SA, respectively. Hence, we prune the interactions which are below a certain threshold. In this study, we set the interaction threshold κ as $\{600, 500, 400, 300\}$ and $\{500, 400, 300, 200\}$ for the organisms HP and SA, respectively. As a result, we obtain four networks per organism studied. In addition, we increase the time limit to 10,800 seconds (i..e, 3 hours) due to the size of the networks.

Table 6 The computational results for Helicobacter Pylori ($n = 1,570$)

		[NIP]			[VCIP]			[DNIP]			[DVCIP]		
κ	**m**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**
600	17735	888.09	0	7617	3117.88	0	43721	**78.19**	**0**	**595**	415.9	0	1522
500	27570	TL	16.20	59510	TL	17.28	63273	**741.73**	**0**	**2709**	3356.8	0	8329
400	33663	TL	18.32	68789	TL	21.16	55947	**9572.51**	**0**	**28965**	TL	5.69	11412
300	45123	TL	15.03	53843	TL	13.53	36859	**TL**	**4.32**	**12271**	TL	6.32	9815

We first share the computational results for HP (see Table 6). As κ decreases, the difficulty in solving the problem increases since the graph gets denser. We initially point out that [VCIP] shows the worst performance where it takes 51 minutes to reach an optimal solution when all other methods converge to optimality in under 15 minutes when $\kappa = 600$. In addition, when κ is set as 500 and 400, we obtain the worst optimality gaps employing [VCIP]. This is an interesting finding since [VCIP] showed marginally a better performance than [NIP] on the randomly generated graphs as discussed in the previous section. On the other hand, [DNIP] outperforms all three methods by reaching the optimal solution in three instances out of four. Even though none of the methods reaches the optimal solution when κ is 300, [DNIP] provided the best optimality gap (4.32%).

We now share Table 7 and the results for SA. Once again, we observe that [VCIP] shows a poorer performance compared to the others. For instance, when κ is set as 400, even though all three other methods converge to the optimal solution, [VCIP] returns an optimality gap of 16.37%. Similar to the results seen in HP, [DNIP] produces the best optimality gaps when no other method can reach the optimal solution. Yet, even though [DNIP] gives the best optimality gap when $\kappa = 200$, the result does not seem as good as the other instances (i.e., 18.09%). Therefore, it might be better to increase the solution time limit when $\kappa \leq 200$. Lastly, it is worth mentioning that [NIP] reaches the optimal solution roughly two times faster than both decomposition methods when $\kappa = 400$.

Table 7 The computational results for Staphylococcus Aureus ($n = 2,852$)

		[NIP]			[VCIP]			[DNIP]			[DVCIP]		
κ	**m**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**	**Time (sec)**	**Gap (%)**	**BB Nodes**
500	21549	65.18	0	415	89.39	0	621	94.19	0	0	**45.34**	**0**	**43**
400	30276	**202.13**	**0**	**2576**	TL	16.37	29888	429.81	0	1557	504.22	0	957
300	45645	TL	37.30	48008	TL	32.64	36084	**TL**	**3.40**	**10957**	TL	13.20	6671
200	87607	TL	26.54	21999	TL	27.93	18250	**TL**	**18.09**	**5873**	TL	27.99	2687

Our computational results on the real-world PPINs indicate that [DNIP] is the best method among all others methods where the optimal solution can be reached for the most of the instances for both organisms tested (i.e., 75% and 50% success rate for HP and SA, respectively). On the other hand, the new IP formulation showed a better performance compared to the one existing in the literature that is different than the observation made in the previous section. We can interpret this from two different points of view: i) [NIP] might be more effective in larger and denser graphs, and/or ii) [NIP] works better specifically in PPINs which carry different characteristics (e.g., following different probability distributions) than the well-known networks models.

8 Conclusion

In this study, we first introduce a new IP formulation for the SDC problem where the goal is to identify the induced star with the largest open neighborhood. We then show that while the SDC can be efficiently solved in tree graphs, it remains $\mathcal{NP}$-complete in bipartite and split graphs via a reduction performed from the set cover problem. In addition, we implement a decomposition algorithm inspired by the Benders Decomposition together with several acceleration techniques to both the new IP formulation and the existing formulation in the literature. Finally, we share extensive computational results on three well-known network models (*Barabási–Albert* , *Erdös–Rényi*, and *Watts–Strogatz model*), and large-scale PPINs generated for two organisms (*Helicobacter Pylori* and *Staphylococcus Aureus*).

Our findings include: i) the existing formulation performs better with respect to the solution time and solution quality when solving the IP models via a branch-and-cut process on randomly generated graphs; ii) the new formulation starts showing its effectiveness in real networks as the size and density increase; iii) the decomposition approaches significantly outperform both IP models in every network model; and iv) the decomposition approach based on the new IP model is shown to be a more effective decomposition framework than the one designed based on the previously proposed IP model.

In the future, it might be interesting to investigate the weighted SDC problem and analyze the impact of the weights on the identification of the essential proteins, rather than employing thresholds to cut off less frequent protein-protein interactions. In addition, from an algorithmic perspective, it could be a good direction to accelerate the decomposition implementations by: i) working on determining new valid inequalities and ii) incorporating clique inequalities especially for triangles.

References

Adulyasak Y, Cordeau JF, Jans R (2015) Benders decomposition for production routing under demand uncertainty. *Operations Research* 63(4):851–867.

Ahat B, Ekim T, Taşkın ZC (2017) Integer programming formulations and Benders decomposition for the maximum induced matching problem. *INFORMS Journal on Computing* 30(1):43–56.

Ashtiani M, Salehzadeh-Yazdi A, Razaghi-Moghadam Z, Hennig H, Wolkenhauer O, Mirzaie M, Jafari M (2018) A systematic survey of centrality measures for protein-protein interaction networks. *BMC Systems Biology* 12(1):80.

Bai L, Rubin PA (2009) Combinatorial Benders cuts for the minimum tollbooth problem. *Operations Research* 57(6):1510–1522.

Banerjee A, Chandrasekhar AG, Duflo E, Jackson MO (2013) The diffusion of microfinance. *Science* 341(6144):1236498.

Bavelas A (1948) A mathematical model for group structures. *Applied Anthropology* 7(3):16–30.

Bavelas A (1950) Communication patterns in task-oriented groups. *The Journal of the Acoustical Society of America* 22(6):725–730.

Benders JF (1962) Partitioning procedures for solving mixed–variables programming problems. *Numerische Mathematik* 4(1):238–252.

Bhowmick SS, Seah BS (2015) Clustering and summarizing protein-protein interaction networks: A survey. *IEEE Transactions on Knowledge and Data Engineering* 28(3):638–658.

Bonacich P (1972) Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology* 2(1):113–120.

Bonacich P (1987) Power and centrality: A family of measures. *American Journal of Sociology* 92(5):1170–1182.

Botton Q, Fortz B, Gouveia L, Poss M (2013) Benders decomposition for the hop-constrained survivable network design problem. *INFORMS Journal on Computing* 25(1):13–26.

Chen L, Miller-Hooks E (2012) Resilience: an indicator of recovery capability in intermodal freight transport. *Transportation Science* 46(1):109–123.

Contreras I, Cordeau JF, Laporte G (2011) Benders decomposition for large-scale uncapacitated hub location. *Operations Research* 59(6):1477–1490.

Cordeau JF, Furini F, Ljubić I (2019) Benders decomposition for very large scale partial set covering and maximal covering location problems. *European Journal of Operational Research* 275(3):882–896.

Dalal J, Üster H (2017) Combining worst case and average case considerations in an integrated emergency response network design problem. *Transportation Science* 52(1):171–188.

Dangalchev C (2006) Residual closeness in networks. *Physica A: Statistical Mechanics and its Applications* 365(2):556–564.

Emde S, Polten L, Gendreau M (2020) Logic-based benders decomposition for scheduling a batching machine. *Computers & Operations Research* 113:104777.

Estrada E (2006) Virtual identification of essential proteins within the protein interaction network of yeast. *Proteomics* 6(1):35–40.

Estrada E, Rodríguez-Velázquez JA (2005) Subgraph centrality in complex networks. *Phys. Rev. E* 71:056103.

Everett MG, Borgatti SP (1999) The centrality of groups and classes. *The Journal of Mathematical Sociology* 23(3):181–201.

Everett MG, Borgatti SP (2005) Extending centrality. *Models and Methods in Social Network Analysis* 35(1):57–76.

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(3):557–569.

Fischetti M, Ljubić I, Sinnl M (2017) Redesigning benders decomposition for large-scale facility location. *Management Science* 63(7):2146–2162.

Frank SM, Rebennack S (2015) Optimal design of mixed AC-DC distribution systems for commercial buildings: A nonconvex generalized Benders Decomposition approach. *European Journal of Operational Research* 242(3):710–729.

Freeman LC (1978) Centrality in social networks conceptual clarification. *Social Networks* 1(3):215–239.

Holmberg K (1994) On using approximations of the Benders master problem. *European Journal of Operational Research* 77(1):111–125.

IBM (2017) CPLEX User's Manual. `https://www.ibm.com/support/knowledgecenter/SSSA5P_12.8.0/ilog.odms.studio.help/pdf/usrcplex.pdf`, (Accessed on 12/04/2020).

Igraph (2020) R igraph manual pages. `https://igraph.org/r/doc`, (Accessed on 12/07/2020).

Jalili M, Salehzadeh-Yazdi A, Asgari Y, Arab SS, Yaghmaie M, Ghavamzadeh A, Alimoghaddam K (2015) Centiserver: A Comprehensive Resource, Web-Based Application and R Package for Centrality Analysis. *PLOS ONE* 10(11):1–8.

Jeong H, Mason SP, Barabási AL, Oltvai ZN (2001) Lethality and centrality in protein networks. *Nature* 411(6833):41–42.

Joy MP, Brock A, Ingber DE, Huang S (2005) High-betweenness proteins in the yeast protein interaction network. *BioMed Research International* 2005(2):96–103.

Kamath RS, Fraser AG, Dong Y, Poulin G, Durbin R, Gotta M, Kanapin A, Le Bot N, Moreno S, Sohrmann M, et al. (2003) Systematic functional analysis of the Caenorhabditis elegans genome using RNAi. *Nature* 421(6920):231–237.

Kleitman DJ, Winston KJ (1982) On the number of graphs without 4-cycles. *Discrete Mathematics* 41(2):167–172.

Leavitt HJ (1951) Some effects of certain communication patterns on group performance. *The Journal of Abnormal and Social Psychology* 46(1):38.

Nasirian F, Pajouh FM, Balasundaram B (2020) Detecting a most closeness-central clique in complex networks. *European Journal of Operational Research* 283(2):461–475.

Rahmaniani R, Crainic TG, Gendreau M, Rei W (2018) Accelerating the benders decomposition method: Application to stochastic network design problems. *SIAM Journal on Optimization* 28(1):875–903.

Rasti S, Vogiatzis C (2019) A survey of computational methods in protein–protein interaction networks. *Annals of Operations Research* 276(1-2):35–87.

Rysz M, Pajouh FM, Pasiliao EL (2018) Finding clique clusters with the highest betweenness centrality. *European Journal of Operational Research* 271(1):155–164.

Samotij W (2015) Counting independent sets in graphs. *European Journal of Combinatorics* 48:5–18.

Schiermeyer I (2019) Maximum independent sets near the upper bound. *Discrete Applied Mathematics* 266:186–190.

Sherali HD, Bae KH, Haouari M (2010) Integrated airline schedule design and fleet assignment: Polyhedral analysis and Benders' decomposition approach. *INFORMS Journal on Computing* 22(4):500–513.

Szklarczyk D, Franceschini A, Wyder S, Forslund K, Heller D, Huerta-Cepas J, Simonovic M, Roth A, Santos A, Tsafou KP, et al. (2014) String v10: protein–protein interaction networks, integrated over the tree of life. *Nucleic Acids Research* 43(D1):D447–D452.

Taşkın ZC, Smith JC, Romeijn HE (2012) Mixed-integer programming techniques for decomposing IMRT fluence maps using rectangular apertures. *Annals of Operations Research* 196(1):799–818.

Veremyev A, Prokopyev OA, Pasiliao EL (2017) Finding groups with maximum betweenness centrality. *Optimization Methods and Software* 32(2):369–399.

Vogiatzis C, Camur MC (2019) Identification of essential proteins using induced stars in protein–protein interaction networks. *INFORMS Journal on Computing* 31(4):703–718.

Vogiatzis C, Veremyev A, Pasiliao EL, Pardalos PM (2015) An integer programming approach for finding the most and the least central cliques. *Optimization Letters* 9(4):615–633.

Wang J, Peng W, Wu FX (2013) Computational approaches to predicting essential proteins: A survey. *PROTEOMICS–Clinical Applications* 7(1-2):181–192.

Wuchty S, Stadler PF (2003) Centers of complex networks. *Journal of Theoretical Biology* 223(1):45–53.

Online Supplement of “The Star Degree Centrality Problem: A Decomposition Approach”

Appendix A: Proof of Theorem 1

Given two LP formulations LP_i and LP_j, let P_i and P_j be the polyhedra defined by LP_i and LP_j, respectively. LP_j is said to be *stronger* than LP_i, if i) there exists at least once instance and one point contained by P_i while not contained by P_j, and ii) all the points contained by P_j are also contained by P_i.

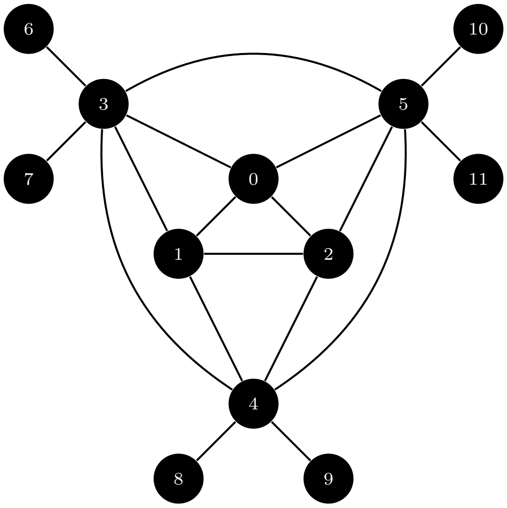
First of all, note that constraints (1g) and (2f) are equivalent, and do not need an explicit comparison. Now, let $l_i = y_i - x_i, \forall i \in V$ be the mapping from $LP_{[VCIP]}$ to $LP_{[NIP]}$ between the variables. When replacing each l_i by $y_i - x_i$ in $LP_{[NIP]}$, it is straightforward to see that constraints (1b) and (1c) imply constraints (2b) and (2c), respectively. When we replace y_i by $l_i + x_i$ in constraints (1d), they imply constraints (2d) since $y_i = l_i + x_i \leq \sum_{j \in N[i]} x_j \implies l_i \leq -x_i + \sum_{j\in N[i]} x_j = \sum_{j \in N(i)} x_j$. In addition, constraints (1e) implies the non-negativity of variables l_i due to the fact that $x_i \leq y_i \implies 0 \leq y_i - x_i \implies 0 \leq l_i$. If we rearrange constraints (1f) based on the map definition, we obtain $l_i + l_j \leq 1, \forall (i,j) \in E$. For a given node i, we then openly write constraints (1f) and aggregate them.

$$(l_i + l_{j_1}) + \cdots + (l_i + l_{j_{|N(i)|}}) \leq |N(i)| \implies \sum_{j \in N(i)} l_j \leq |N(i)|(1 - l_i)$$

It can be seen that, constraints (1f) imply constraints (2e) with a slight modification. Therefore, we can conclude that all the points contained by the polyhedron generated by $LP_{[VCIP]}$ is contained by the polyhedron generated by $LP_{[NIP]}$; in other words, $OBJ_{LP_{[VCIP]}} \leq OBJ_{LP_{[NIP]}}$.

Below we present a counter example where a solution produced by $LP_{[NIP]}$ cannot be converted a feasible solution in $LP_{[VCIP]}$.

Figure 10 A counter example where the optimal solution obtained in $\mathbf{LP_{[NIP]}}$ cannot be converted a feasible solution in $\mathbf{LP_{[VCIP]}}$.



For this example, $LP_{[NIP]}$ sets x_3, x_4 and x_5 0.2, 0.2, and 0.6, respectively while the leaf variables of the same nodes (i.e., l_i) are set as $1 - x_i$ where $i = \{3,4,5\}$ in an optimal solution. As a result, the objective value becomes nine. On the other hand, since nodes 3 and 4 share an edge, the same solution becomes infeasible in $LP_{[VCIP]}$ due to constraints (1f) (i.e., $1.6 \nleq 1.4$). The solver returns 8.5 as optimal solution in $LP_{[NIP]}$. Hence, we can conclude that $[VCIP]$ is a tighter formulation than $[NIP]$ with respect to LP-relaxations.

Appendix B: Proof of Proposition 1

By the definition of the windmill graph, there exist n identical complete graphs with k vertices each of which is connected to the universal vertex u. A star whose center is u with no selected leaves has a neighborhood of size $|V| - 1 = (k-1)n$. Note that any node selected as a leaf node decreases the objective by one since all its neighbors are already in the star's neighborhood. For any node $j \in V \setminus \{u\}$ as a center, we must have the universal node u as a leaf node in order to gain access to the nodes j does not have an edge to. If u is not a leaf node, then the maximum neighborhood would be $k-1$ (all nodes incident to j are in the neighborhood). If u is a leaf node, then the maximum neighborhood is for all nodes besides j and u to be in it, which implies the maximum size is $|V| - 2 < |V| - 1$. Hence, the optimal solution is unique and provided by the universal vertex u with no leaf nodes.

Appendix C: Proof of Theorem 4

We can create a reduction via a set cover instance in the following way.

$$V[G] = V_1 \cup V_2 \text{ where } V_1 = \{S_1, S_2, \cdots, S_m, d_1\} \text{ and } V_2 = \{u_1, u_2, \cdots, u_n, d_2, d_3, d_4 \cdots, d_{|S|+3}\}$$
$$E[G] = \left\{\cup_{i=1}^{m}\{\cup_{j\in S_i}(S_i, u_j)\}\right\} \cup \left\{\cup_{j=1}^{n}\{\cup_{p=1}^{m}(u_j, u_p)\}\right\} \cup \{\cup_{i=1}^{m}(d_2, S_i)\} \cup \{(d_1, d_2)\}$$
$$\cup\{\cup_{i=3}^{|S|+3}(d_1, d_i)\}$$

Note that we connect all the elements in the universe set with one another to create a clique instance. With this formation following the similar steps discussed to prove Theorem 3, if we solve the SDC problem, the dummy node d_2 would be the center of the star with the largest objective value implying that we obtain the solution for the set cover instance. Hence, we conclude that the SDC problem is $\mathcal{NP}$-complete when a split graph is concerned.

Appendix D: Proof of Proposition 2

First of all, since constraint $\lambda_i + \omega_i = 1$ is satisfied (i.e., tight) for every $(\lambda, \omega, \theta)$ in all the assignment cases, the algorithm produces a dual feasible solution for a given solution vector $(\bar{l}, \bar{x})$. As for the primal problem, we set $z_i = 0$ for a node i if the RHS of either constraints in $\phi_i^{NIP}(\bar{l}, \bar{x})$ is zero. On the other hand, if the RHSs of constraints are positive, then we set $z_i = \min\{1 - \bar{l}_i - \bar{x}_i, \sum\limits_{j\in N(i)} (\bar{l}_j + \bar{x}_j)\}$. Therefore, we also obtain a primal feasible solution.

In addition, the objective values of $\phi_i^{NIP}(\bar{l}, \bar{x})$ and $\Phi_i^{NIP}(\bar{l}, \bar{x})$ are the same (i.e., the strong duality holds). In the case of primal variable $z_i = (1 - \bar{l}_i - \bar{x}_i)$, we set the dual variables λ_i and ω_i accordingly to keep the contribution to the dual objective the same. When $z_i = \sum\limits_{j\in N(i)} (\bar{l}_j + \bar{x}_j)$, we set $\lambda_i = 0, w_i = 1$ which yields the same objective in $\Phi_i^{NIP}(\bar{l}, \bar{x})$. When $z_i = 0$, based on the value of $\sum\limits_{j\in N(i)} (\bar{l}_j + \bar{x}_j)$, we keep the contribution of node i to the dual objective as zero by tuning the dual variables λ_i and ω_i accordingly. Therefore, the algorithm produces primal/dual solutions that satisfy the complementary slackness. As a result, the primal and dual variables calculated are indeed optimal solutions.

Appendix E: Proof of Proposition 3

Considering the constraint that no leaf node is connected in a star, let us answer the following question: "What is the largest number of nodes that can be selected as leaf nodes within $N(i)$?". In fact, this question is equivalent to the MIS which is the maximum number of nodes such that none of which is connected to the other in a given graph. Hence, a feasible star centered at node i cannot have more leaves than the cardinality of MIS for the induced graph formed by the nodes within $N(i)$.

Appendix F: Warm-Start Results

We compare the solutions obtained with and without warm-start from two different perspectives: (i) difference between the solution times when either produces a feasible solution, and (ii) difference between the optimality gaps when either produces an optimal solution. We set thresholds of 30 seconds and 0.5% for (i) ad (ii), respectively. If the absolute value of a difference value is less than the corresponding threshold, we neglect to report such result. Note that the negative improvement in both solution time and optimality gap indicates that warm-start improves the performance of the solution technique utilized.

Figure 11 The impact of warm-start in the solution times in [NIP] in the BA model

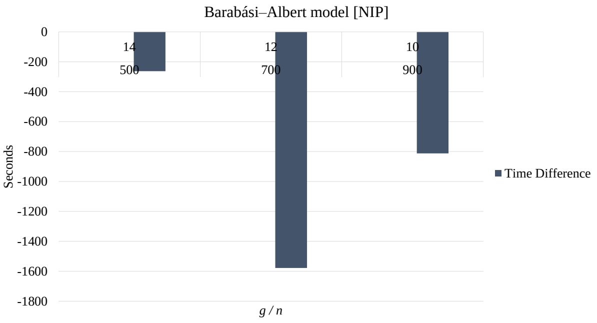


Figure 12 The impact of warm-start in the optimality gaps in [NIP] in the BA model

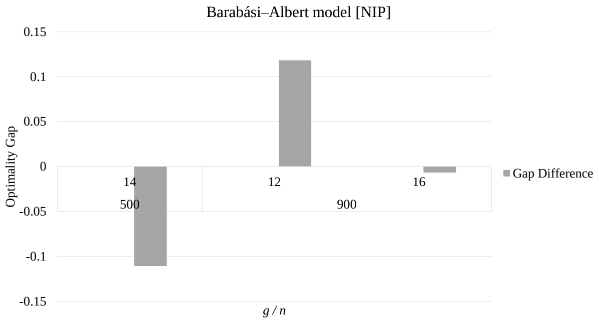


Figure 13 The impact of warm-start in the solution times in [VCIP] in the BA model

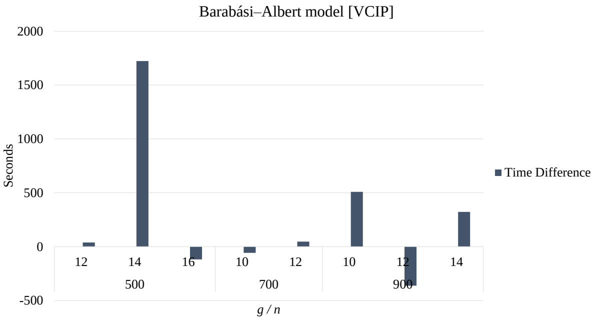
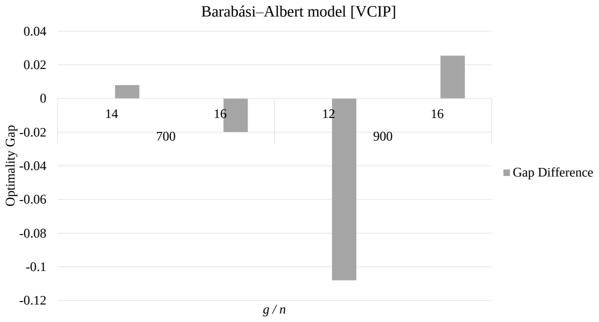


Figure 14 The impact of warm-start in the optimality gaps in [VCIP] in the BA model



In the BA model, we observe that while warm-start helps [NIP] to improve the solution time a considerable amount in three instances out of 12, an inconsistent pattern takes place in terms of optimality gaps (see Figs. 11 and 12). Furthermore, [VCIP] does not show a clear trend in both solution times and optimality gaps as depicted in Figs. 13 and 14.

Figure 15 The impact of warm-start in the optimality gaps in [NIP] in the ER model

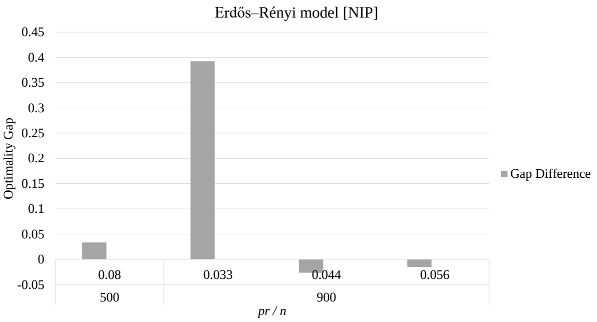
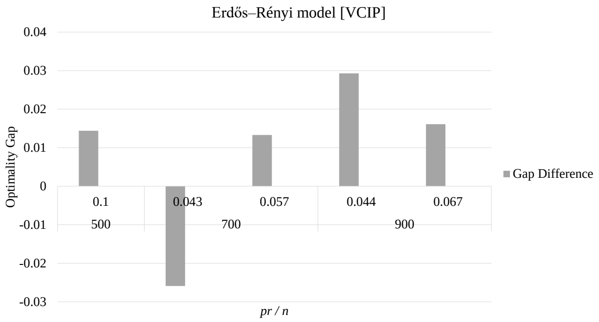


Figure 16 The impact of warm-start in the optimality gaps in [VCIP] in the ER model



In the ER model, while warm-start increases the solution time in [NIP] in solely one instance by roughly 2300 seconds (i.e., $n = 900, pr = 0.033$), we do not observe any instance where it helps with the solution time. As for [VCIP], there is no instance with respect to the solution time that meets our threshold definition of improvement (30 seconds). Furthermore, similar to the BA model, no consistent pattern appears in terms of optimality gaps in both IP models as depicted in Figs. 15 and 16.

Figure 17 **The impact of warm-start in the optimality gaps in [NIP] in the WS model**

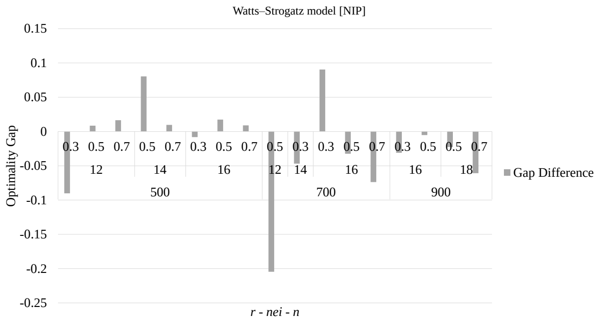
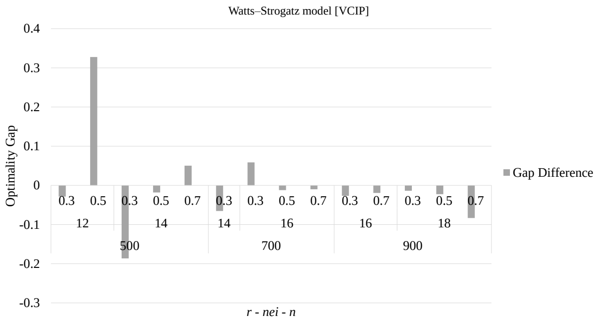


Figure 18 **The impact of warm-start in the optimality gaps in [VCIP] in the WS model**



Lastly, in the WS model, we observe that warm-start helps [NIP] with the solution time in two instances (i.e., $n = 500, nei = 12, r = 0.3$, $n = 700, nei = 12, r = 0.5$) to a great extent, which is a decrease of nearly 3500 seconds. On the other hand, while [VCIP] shows a worse performance in one instance ($n = 500, nei = 12, r = 0.5$) with an increase of around 1200 seconds via warm-start, no apparent improvement is seen in any of the instances. Similar to other network models, we cannot see a distinguishable performance with respect to the optimality gaps in both IP formulations when warm-starting (see Figs. 17 and 18). Therefore, it becomes hard to reach a solid conclusion.

As for the decomposition implementations, we do not observe big changes with respect to solution time and optimality gaps either; especially in both BA and ER models in the majority of the instances. For the changes occurring, they turn out to be more erratic patterns compared to the IP models. As an example, we share Figs. 19 and 20 which illustrate the solution time changes in the WS model with warm-start in [DNIP] and [DVCIP], respectively.

Figure 19 **The impact of warm-start in the solution times in [DNIP] in the WS model**

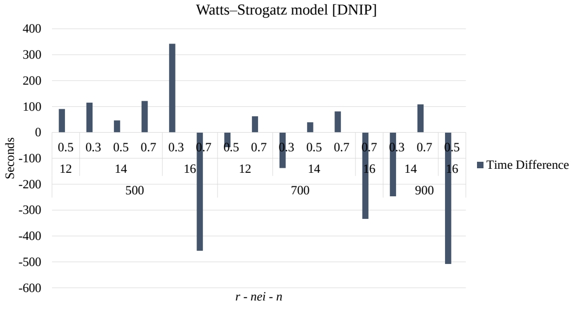
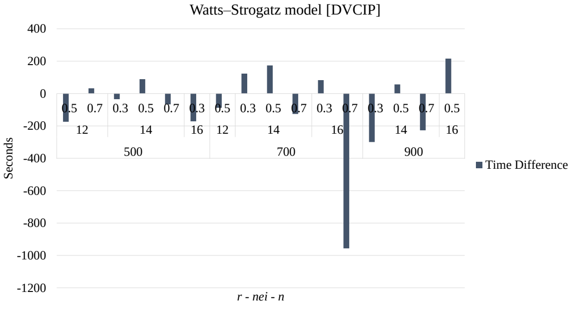


Figure 20 **The impact of warm-start in the solution times in [DVCIP] in the WS model**


Watts–Strogatz model [DVCIP]
Seconds
r - nei - n
Time Difference