Robust Critical Node Selection by Benders Decomposition

Joe Naoum-Sawaya, Christoph Buchheim

{jnaoumsa@uwaterloo.ca, christoph.buchheim@tu-dortmund.de}

Abstract

The critical node selection problem (CNP) has important applications in telecommunication, supply chain design, and disease propagation prevention. In practice, the weights on the connections are either uncertain or hard to estimate so recently robust optimization approaches have been considered for CNP. In this article, we address very general uncertainty sets, only requiring a linear optimization oracle for the set of potential scenarios. In particular, we can deal with interval uncertainty, discrete scenario based uncertainty, Gamma-uncertainty, and ellipsoidal uncertainty. For this general class of robust critical node selection problems (RCNP), we propose an exact solution method based on Benders decomposition. The Benders subproblem, which in our approach is a robust optimization problem, is essentially solved by applying the Floyd-Warshall algorithm. The presented approach is tested on 1,220 instances based on random planar and Barabási-Albert graphs with different number of nodes and graph densities, where running times are compared to CPLEX being directly applied to the robust problem formulation. The computational results show the advantage of the proposed approach in handling the uncertainty thus outperforming CPLEX most notably for the ellipsoidal uncertainty cases.

Keywords Robust Optimization, Critical Nodes, Benders Decomposition.

1 Introduction

Given an undirected graph $G(V, E)$, the objective of the critical node selection problem (CNP) is to find a subset $S \subseteq V$ of at most K nodes such that the residual graph obtained by eliminating the selected nodes has a minimum number of connected pairs $\{i, j\}$. A pair $\{i, j\}$ is said to be connected if there exists a path linking nodes i and j. In the weighted version of the CNP, we are additionally given a non-negative weight w_{ij} on each pair $\{i, j\}$ and aim at a minimum total weight of connected node pairs.

The CNP has several applications in practice. It is important in assessing the robustness of a network such as a supply chain, where the objective is to measure the number of connections that are still reliable after a "smart" attacker damages the most critical nodes in the network (Arulselvan et al., 2007). Another application arises in health-care particularly in virus immunization where limited immunization resources are to be distributed. The goal is to identify the individuals to be vaccinated in order to reduce the overall transmissibility of a virus and to control the outbreak of an epidemic (Tao et al., 2006). Similarly, the spread of a virus over a telecommunication network can be limited by identifying the critical nodes of the graph and taking them offline (Cohen et al., 2003). Another application arises in wireless network jamming, where the objective is to efficiently prevent communication over a wireless network by finding the critical nodes and jamming them (Commander et al., 2007).

Arulselvan et al. (2007) presented an integer programming (IP) formulation for the CNP and proposed a heuristic that is based on finding an independent set of nodes from the graph and then improving it through a simple 2-exchange local search. Di Summa et al. (2011) showed that the CNP is $\mathcal{NP}$-hard in the general case and then considered graphs with the special tree structure and showed that the resulting CNP is polynomially solvable for the case of unit edge weights and proposed a dynamic programming approach for its solution. Shen and Smith (2012) presented polynomial-time dynamic programming algorithm for solving the CNP on tree structures and on series-parallel graphs. Addis et al. (2013) also used dynamic programming to solve special cases of the CNP. Based on the IP model presented by Arulselvan et al. (2007), Di Summa et al. (2012) derived valid inequalities based on cliques and cycles in the graph. Di Summa et al. (2012) also proposed an integer programming formulation with a non-polynomial number of constraints and a branch-and-cut framework as a solution methodology. Valid inequalities based on dynamic programming solutions are also studied by Shen et al. (2012). Most recently, Ventresca (2012) proposed a simulated annealing heuristic with incremental learning targeting large graphs of sizes between 1000 and 5000 nodes.

While this paper focuses on CNP, we note that several important variants to the CNP have also been proposed. Arulselvan et al. (2011) consider the problem of minimizing the number of vertices whose deletion results in disconnected components which are constrained by a given cardinality. Shen and Smith (2012) consider the problem of minimizing the number of nodes that belong to the largest maximal connected subgraph. Another variant of CNP is also proposed in Shen and Smith (2012) where the number of disconnected components is maximized. Other closely related problems to CNP are the k-cut problem that partitions a vertex set of a graph into several disjoint subsets so as to minimize the total weight of the edges between the disjoint subsets (Fan and Pardalos, 2010b; Ghaddar et al., 2011) and the minimum multi-cut problem that minimizes the weight of edges whose removal disconnects each of the of source-sink node pairs (Garg et al., 1996).

The importance of a connection, modeled by the weight w_{ij} of a pair $\{i, j\}$, is often not known exactly in real-world applications. Such data uncertainties can be addressed by means of either robust or stochastic optimization. The latter approach requires a precise knowledge about probability distributions of the weights; moreover it usually leads to optimization problems that are far too hard to be solved in practice. On the other hand, the robust optimization approach has already been applied successfully to the CNP by Fan and Pardalos (2010a) and Fan et al. (2012).

In the approach that is proposed in this paper, a set of potential scenarios, the so-called uncertainty set, is defined. The objective of the robust critical node selection problem (RCNP) is then to minimize the maximum total weight of a feasible solution over all scenarios, thus aiming at a worst-case optimal solution. The choice of an appropriate uncertainty set is crucial, as a too large set leads to bad solutions in the average case and a too small set does not sufficiently take the uncertainty into account. Moreover, in most approaches the choice of the uncertainty set has a dramatic impact on the solvability of the resulting model, both from a theoretical and a practical point of view.

In this article, we present an approach for the RCNP that can deal with all common classes of uncertainty sets, including interval uncertainty, discrete scenario based uncertainty, Gamma-uncertainty, and ellipsoidal uncertainty. This is achieved through a Benders decomposition approach that explicitly handles the uncertainty in the Benders subproblem through an efficient optimization oracle. The structure of the Benders subproblem is also exploited and a specialized polynomial time algorithm that is based on Floyd-Warshall algorithm is proposed in order to achieve maximum speedup. Problems based on random planar graphs and Barabási-Albert graphs are solved to optimality and a comparison with CPLEX is conducted.

It turns out that our approach is particularly effective for ellipsoidal uncertainties, where it clearly ourperforms CPLEX on all instances considered. Compared to all other classes of uncertainty considered, the ellipsoidal case is usually the hardest one, but also the most relevant one in practice: on the one hand, unlike the interval and the Gamma uncertainty cases, it avoids to consider unlikely scenarios corresponding to vertices of the uncertainty set, which lead to overly conservative solutions. On the other hand, it does not require any discretization as in the case of finite scenario sets, while still being able to capture correlations between the weights of different connections. In fact, when assuming that all weights are jointly normally distributed, one can show that the confidence regions form ellipsoids, so that ellipsoidal uncertainty is the most natural type of uncertainty to consider. To the best of our knowledge, no other problem-specific approaches for the RCNP under ellipsoidal uncertainty have been proposed in the literature before.

Contribution of this paper. The main contributions of this paper are (a) a robust optimization model for the CNP under very general uncertainty sets, including ellipsoidal uncertainty, (b) a novel approach to address this problem through Benders decomposition while explicitly dealing with robustness in the subproblems, and (c) a computationally efficient implementation that is capable of solving problems of reasonable size.

Outline of this paper. Following this introductory section, Section 2 provides the mathematical formulation. Sections 3 presents the Benders decomposition approach. Computational results are presented in Section 4. Finally, a conclusion is given in Section 5.

2 Problem Formulation

We consider an undirected weighted graph $G(V, E)$, with node set V and edge set E. A node pair (i, j) is connected if there exists a path that links nodes i and j. To formulate the critical node selection problem, we define the following decision variables

$$x_i = \begin{cases} 1 & \text{if node } i \text{ is deleted from the graph} \\ 0 & \text{otherwise,} \end{cases}$$

$$y_{ij} = \begin{cases} 1 & \text{if node pair } (i,j) \text{ is disconnected in the residual graph} \\ 0 & \text{otherwise.} \end{cases}$$

We consider an uncertainty set $\mathcal{U}$ that contains all possible realizations of edge weights w. We make the reasonable assumption that $w \geq 0$ for all $w \in \mathcal{U}$. Apart from this natural assumption, we only require that a linear optimization oracle is given for $\mathcal{U}$. The robust critical node selection problem is then formulated as follows:

$$\text{[RCNP]:} \max_{x,y} \min_{w \in \mathcal{U}} w^\top y \tag{1}$$

$$\text{s.t. } e^\top x \leq K, \tag{2}$$

$$y_{ij} \leq x_i + x_j, \ \forall (i,j) \in E, \tag{3}$$

$$y_{ij} + y_{jk} - y_{ik} \geq 0, \forall i,j,k \in V, \ i < j < k, \tag{4}$$

$$y_{ij} - y_{jk} + y_{ik} \geq 0, \forall i,j,k \in V, \ i < j < k, \tag{5}$$

$$-y_{ij} + y_{jk} + y_{ik} \geq 0, \forall i,j,k \in V, \ i < j < k, \tag{6}$$

$$0 \leq y_{ij} \leq 1, \forall i,j \in V, \ i < j, \tag{7}$$

$$x_i \in \{0,1\}, \forall i \in V. \tag{8}$$

In the non-robust case, i.e. when $\mathcal{U}$ is a singleton, this model is identical to the one presented in Di Summa et al. (2012). The objective function maximizes the worst case weight of the disconnected node pairs. Constraint (2) is a cardinality constraint limiting the number of deleted nodes to K ($e^\top$ is a row vector of all ones). Constraints (3) ensure that if a node pair (i,j) is disconnected in the residual graph and there exists an edge (i,j) then either node i or node j should be deleted. Constraints (4)–(6) guarantee that for all triplets (i,j,k), if (i,j) is connected and (j,k) is connected, then (i,k) should be connected. For ease of presentation, in the rest of the paper we refer to constraints (3)–(7) as

$$Ay \leq Bx + b.$$

We also note that the binary condition on the y_{ij} variables can be relaxed as discussed in Di Summa et al. (2012) thus permitting the application of Benders decomposition. In the following section, we briefly describe our Benders decomposition approach for problem [RCNP].

3 Benders Decomposition

Benders decomposition (Benders, 1962; Geoffrion, 1972) splits the variables in a mixed integer program into a linear subproblem and a mixed integer master problem that contains all the integer variables. The main motivation is having a subproblem that is significantly easier to solve. The procedure iterates between solving the subproblem and a relaxed version of the mixed integer master problem, until an optimal solution is reached.

In what follows, we describe the application of Benders decomposition to problem [RCNP]. For that, we write [RCNP] as follows:

$$\max_{\substack{e^\top \overline{x} \leq K \\ \overline{x} \in \{0,1\}^V}} \max_{y} \min_{w \in \mathcal{U}} w^\top y \tag{9}$$

$$\text{s.t. } Ay \leq B\overline{x} + b. \tag{10}$$

In order to generate optimality cuts for the Benders master problem, we derive the dual of the inner problem

$$\begin{aligned}\text{[PSP]: } f(\overline{x}) = \max_{y} \min_{w \in \mathcal{U}} \; & w^\top y \\ \text{s.t. } & Ay \leq B\overline{x} + b.\end{aligned}$$

From the concavity of the function $g(y) := \min_{w \in \mathcal{U}} w^\top y$, which is independent of the set $\mathcal{U}$, we conclude that f is a concave function in $\overline{x}$. This leads to

Theorem 1 *The dual of [PSP] is*

$$\text{[DSP]: } \min_{\lambda} \ (B\overline{x} + b)^\top \lambda,$$
$$\text{s.t. } A^\top \lambda \in \mathcal{U},$$
$$\lambda \geq 0,$$

and strong duality holds.

Proof 1 *The Lagrangian dual of [PSP] is*

$$\min_{\lambda \geq 0} \ \max_{y} \left(\min_{w \in \mathcal{U}} \ w^\top y + (B\overline{x} + b - Ay)^\top \lambda \right).$$

Rearranging the variables we get

$$\min_{\lambda \geq 0} \left((B\overline{x} + b)^\top \lambda + \max_{y} \ \min_{w \in \mathcal{U}} \ (w - A^\top \lambda)^\top y \right) \ = \ \min_{\lambda, w} \ (B\overline{x} + b)^\top \lambda$$
$$\text{s.t. } A^\top \lambda = w$$
$$\lambda \geq 0, \ w \in \mathcal{U} \ .$$

By concavity of f and since all constraints in [PSP] are linear, the Slater condition guarantees strong duality. □

The Benders optimality cut can now be derived from a given optimal solution $\overline{\lambda}$ of [DSP] as

$$\theta - (B^\top \overline{\lambda})^\top x \leq b^\top \overline{\lambda} \ .$$

Hence the Benders master problem is

$$[BMP] : \max_{x, \theta} \ \theta \tag{11}$$
$$\text{s.t. } e^\top x \leq K, \tag{12}$$
$$\theta - (B^\top \overline{\lambda})^\top x \leq b^\top \overline{\lambda}, \tag{13}$$
$$x_i \in \{0, 1\}, \ \forall i \in V, \tag{14}$$

where cuts (13) are generated iteratively. Solving the Benders master problem with an additional optimality cut, a new optimal solution $\overline{x}$ is obtained which in turn gives rise to a new optimality cut. This process is repeated iteratively until an optimal solution is obtained.

3.1 Solving the Primal Benders Subproblem

In this section, we present a specialized algorithm for solving the Benders subproblem [PSP]. We do not require $\overline{x}$ to be a binary vector. Fractional values of $\overline{x}$ appear when using the analytic center cutting plane method (ACCPM) that improves over the classical Benders decomposition; this is discussed in Section 4.1. At the end of this section, we also provide a faster and simpler algorithm for the case of binary $\overline{x}$ vector.

The specialized algorithm for the fractional case is based on finding the shortest path between every pair of nodes in the graph. For that, let us define an initial set of weights h_{ij} for every edge $(i,j) \in E$ such that

$$h_{ij} = \min\{1, \overline{x}_i + \overline{x}_j\}, \ \forall (i,j) \in E,$$
$$h_{ij} = 1, \ \forall (i,j) \notin E.$$

We then compute all pairwise shortest paths between every pair of nodes (i,j) and finally set

$$\overline{y}_{ij} = \text{shortestPath}(i,j).$$

The pairwise shortest paths can be computed efficiently using the Floyd-Warshall algorithm (Floyd, 1962) which finds the shortest paths between all pairs of nodes in $O(|V|^3)$ comparisons where $|V|$ is the number of nodes in the graph.

In the following we provide a proof that the resulting solution $\overline{y}$ is an optimal solution to the nominal Benders subproblem for any fixed $w \geq 0$. We first show that $\overline{y}$ is feasible to [PSP] and finally show that it is an optimal solution.

Theorem 2 *The vector $\overline{y}$ with $\overline{y}_{ij} =$ shortestPath(i,j) is feasible to [PSP].*

Proof 2 *By definition, we have that $0 \leq h_{ij} \leq 1$ and $h_{ij} \leq \overline{x}_i + \overline{x}_j \ \forall (i,j) \in E$. Furthermore, $\overline{y}_{ij} \leq h_{ij}$ since $\overline{y}_{ij} =$ shortestPath(i,j). Therefore $\overline{y}_{ij} \leq \overline{x}_i + \overline{x}_j$ and $\overline{y}_{ij} \leq 1$ for all $(i,j) \in E$ and hence constraints* (3) *and* (7) *are satisfied. Additionally, for every set of nodes (i,j,k)*

we have that

$$\begin{aligned}
&\textit{shortestPath}(i,k) \leq \textit{shortestPath}(i,j) + \textit{shortestPath}(j,k),\\
&\textit{shortestPath}(j,k) \leq \textit{shortestPath}(i,j) + \textit{shortestPath}(i,k),\\
&\textit{shortestPath}(i,j) \leq \textit{shortestPath}(i,k) + \textit{shortestPath}(j,k),
\end{aligned}$$

hence constraints (4), (5), (6) *are satisfied respectively. Since all the constraints of [PSP] are satisfied, then* $\overline{y}_{ij}$ *is feasible to [PSP].* $\square$

Theorem 3 *The vector* $\overline{y}$ *with* $\overline{y}_{ij} = \textit{shortestPath}(i,j)$ *is optimal for [PSP].*

Proof 3 *Let* y *be any feasible solution for [PSP]. We show that* $y \leq \overline{y}$*. This implies that* $w^\top y \leq w^\top \overline{y}$ *for all* $w \in \mathcal{U}$ *since* $w \geq 0$ *and therefore*

$$\min_{w\in\mathcal{U}} \; w^\top y \;\leq\; \min_{w\in\mathcal{U}} \; w^\top \overline{y},$$

showing that $\overline{y}$ *is an optimal solution for [PSP].*

So fix $i, j \in V$ *with* $i < j$*. If* $\overline{y}_{ij} = 1$*, we obtain* $y_{ij} \leq \overline{y}_{ij}$ *from* (7)*, so that we may assume* $\overline{y}_{ij} < 1$*. Then, the shortest path from* i *to* j *contains only edges* $(i,k_1),\dots,(k_m,k_{m+1}),\dots,(k_n,j) \in E$ *so that*

$$\overline{y}_{ij} = \min\{1, \overline{x}_i + \overline{x}_{k_1}\} + \cdots + \min\{1, \overline{x}_{k_m} + \overline{x}_{k_{m+1}}\} + \cdots + \min\{1, \overline{x}_{k_n} + \overline{x}_j\}.$$

As y *is feasible for [PSP], we have*

$$y_{ik_1} \leq \min\{1, \overline{x}_i + \overline{x}_{k_1}\} \tag{15}$$

$$\vdots$$

$$y_{k_m k_{m+1}} \leq \min\{1, \overline{x}_{k_m} + \overline{x}_{k_{m+1}}\} \tag{16}$$

$$\vdots$$

$$y_{k_n j} \leq \min\{1, \overline{x}_{k_n} + \overline{x}_j\} \tag{17}$$

by (3) *and* (7) *and*

$$y_{ij} \leq y_{ik_n} + y_{k_n j} \tag{18}$$

$$y_{ik_n} \leq y_{ik_{n-1}} + y_{k_{n-1}k_n} \tag{19}$$

$$\vdots$$

$$y_{ik_2} \leq y_{ik_1} + y_{k_1 k_2} \tag{20}$$

by constraints (4)–(6)*. Adding constraints* (18)*,*(19)*, ...,* (20) *we get*

$$y_{ij} \leq y_{ik_1} + \cdots + y_{k_m k_{m+1}} + \cdots + y_{k_n j}. \tag{21}$$

Substituting (15)–(17) *in* (21) *we get*

$$\begin{aligned} y_{ij} &\leq \min\{1, \overline{x}_i + \overline{x}_{k_1}\} + \cdots + \min\{1, \overline{x}_{k_m} + \overline{x}_{k_{m+1}}\} + \cdots + \min\{1, \overline{x}_{k_n} + \overline{x}_j\} \\ &= \overline{y}_{ij} \end{aligned}$$

as desired. □

Corollary 1 *The optimal value of [PSP] is equal to*

$$\min \overline{y}^\top w \tag{22}$$

$$s.t.\ w \in \mathcal{U}\ . \tag{23}$$

For solving problem (22)–(23), we can use the linear optimization oracle that we assume is available for $\mathcal{U}$. The running time for computing $\overline{y}$ is $O(|V|^3)$ as mentioned earlier, however the running time for computing the optimal value of [PSP] depends on how quickly we can minimize the linear objective (22) over the set $\mathcal{U}$. For the case of binary $\overline{x}$, the computation of $\overline{y}$ can be accelerated:

Theorem 4 *Assume that $\overline{x} \in \{0,1\}^n$. Then a maximizer of [PSP] can be computed in $O(|V|^2)$ time.*

Proof 4 *Starting with $\overline{y} = 0$, we first remove all nodes i such that $x_i = 1$ along with their incident edges. As long as there is a node r left, we run a depth-first-search from r, resulting*

in a node set R, set $\overline{y}_{ij} = 1$ for all $i, j \in R$, and delete the nodes in R. Clearly, the resulting vector $\overline{y}$ is optimal for [PSP], and this algorithm can be implemented to run in $O(|V|^2)$ time.
$\square$

3.2 Solving the Dual Subproblem

As discussed earlier, to derive the Benders optimality cuts, the dual optimal solution $\overline{\lambda}$ must be obtained.

Lemma 1 *Let $\overline{w}$ be an optimal solution of problem (22)–(23), then an optimal solution of [DSP] can be computed by solving*

$$\min_{\lambda} \ \lambda^\top (B\overline{x} + b) \tag{24}$$

$$s.t. \ A^\top \lambda = \overline{w} \tag{25}$$

$$\lambda \geq 0. \tag{26}$$

Proof 5 *By Corollary 1, we have that*

$$\begin{array}{lll} \max_y & \min_{w\in\mathcal{U}} \ w^\top y & = \ \overline{w}^\top \overline{y} \ = \quad \max_y \qquad \overline{w}^\top y \\ \text{s.t.} & Ay \leq B\overline{x} + b & \qquad\qquad\quad \text{s.t.} \quad Ay \leq B\overline{x} + b \ . \end{array}$$

The latter problem is the dual of problem (24)–(26) and has the same objective function value due to strong duality. Since the objective function is the same for both dual problems, [DSP] and problem (24)–(26), and the feasible set of problem (24)–(26) is a subset of the feasible set of [DSP] then every optimal solution of problem (24)–(26) is an optimal solution of [DSP].
$\square$

Lemma 1 shows that solving [DSP] can be reduced to solving a linear program once the primal solution and the corresponding $\overline{w}$ have been computed. In summary, the only step in the algorithm that requires dealing with $\mathcal{U}$ explicitly is the solution of (22)–(23) to compute $\overline{w}$.

In fact, an optimal solution of this linear program can be directly derived from an optimal solution of the primal problem [PSP]. For this, assume that an optimal solution of [PSP] is computed as discussed in Section 3.1. More precisely, for each pair $\{i, j\}$, let P_{ij} be a shortest path from i to j with respect to the length function h. Without loss of generality, we can make the following assumptions:

(A1) No edge $\{i,j\}$ with $h_{ij} = 1$ is contained in any shortest path except for possibly P_{ij}.

(A2) If $|E(P_{ij})| = 1$ and $i, j \in V(P_{kl})$, then $\{i,j\} \in E(P_{kl})$.

In order to ensure (A1), we can re-define $E(P_{kl}) := \{\{k,l\}\}$ for all pairs k,l with $\{i,j\} \in E(P_{kl})$ and $h_{ij} = 1$, as $h_{kl} \leq 1 \leq \text{shortestPath}(k,l)$ in this case, so that the new path is still a shortest path from k to l. In the situation of (A2), one of the shortest paths from i to j consists of edge $\{i,j\}$, so we may assume that each shortest path containing i and j uses this edge.

Theorem 5 *A Benders optimality cut can be obtained as*

$$\theta - \sum_{h_{ij}<1} \Big(\sum_{E(P_{kl}) \ni \{i,j\}} \overline{w}_{kl} \Big)(x_i + x_j) \leq \sum_{h_{ij}=1,\ |E(P_{ij})|=1} \overline{w}_{ij} \ . \tag{27}$$

Proof 6 *The Benders primal subproblem reads*

$$\begin{array}{lrcllr}
\max_y & \overline{w}^\top y & & & & \\
\text{s.t.} & y_{ij} & \leq & \overline{x}_i + \overline{x}_j & \forall (i,j) \in E & [\lambda^1_{ij}] \\
& y_{ij} & \leq & 1 & \forall i,j \in V,\ i<j & [\lambda^2_{ij}] \\
& y_{ij} - y_{jk} - y_{ik} & \leq & 0 & \forall i,j,k \in V,\ i<j<k & [\beta^k_{ij}] \\
& -y_{ij} + y_{jk} - y_{ik} & \leq & 0 & \forall i,j,k \in V,\ i<j<k & [\beta^i_{jk}] \\
& -y_{ij} - y_{jk} + y_{ik} & \leq & 0 & \forall i,j,k \in V,\ i<j<k & [\beta^j_{ik}] \\
& y_{ij} & \geq & 0 & \forall i,j \in V,\ i<j &
\end{array}$$

We obtain the Benders dual subproblem

$$\begin{array}{lrcll}
\min_{\lambda,\beta} & \multicolumn{4}{l}{\sum_{(i,j)\in E} (\overline{x}_i + \overline{x}_j)\lambda^1_{ij} + \sum_{\{i,j\}} \lambda^2_{ij}} \\
\text{s.t.} & \lambda^1_{ij} + \lambda^2_{ij} + \sum_{k \neq i,j} (\beta^k_{ij} - \beta^j_{ik} - \beta^i_{jk}) & \geq & \overline{w}_{ij} & \forall (i,j) \in E \\
& \lambda^2_{ij} + \sum_{k \neq i,j} (\beta^k_{ij} - \beta^j_{ik} - \beta^i_{jk}) & \geq & \overline{w}_{ij} & \forall (i,j) \notin E \\
& \lambda, \beta & \geq & 0. &
\end{array}$$

Define

$$\overline{\lambda}^1_{ij} := \begin{cases} 0 & \text{if } h_{ij} = 1 \\ \sum_{E(P_{kl})\ni\{i,j\}} \overline{w}_{kl} & \text{otherwise} \end{cases}$$

and

$$\overline{\lambda}^2_{ij} := \begin{cases} \overline{w}_{ij} & \text{if } h_{ij} = 1 \text{ and } |E(P_{ij})| = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Moreover, define

$$\overline{\beta}^k_{ij} := \frac{1}{2}\Big(\sum_{E(P_{il})\ni(k,j)} \overline{w}_{il} + \sum_{E(P_{jl})\ni(k,i)} \overline{w}_{jl} \Big) \,.$$

Here, $E(P_{il}) \ni (k,j)$ means that the path P_{il} contains the vertices k and j one after another, in this order when traversing P_{ij} from i to j. Note that $j = l$ is not excluded.

Since we assume $\overline{w} \geq 0$, we have $\overline{\lambda}, \overline{\beta} \geq 0$. We next show that $\overline{\lambda}, \overline{\beta}$ satisfies all constraints of the dual problem. First note that $\sum_{k\neq i,j} \overline{\beta}^k_{ij}$ is half the sum of all $\overline{w}_{st}$ such that $i, j \in V(P_{st})$ but $\{i,j\} \notin E(P_{st})$ and such that exactly one of the end vertices of the path is i or j, plus $\overline{w}_{ij}$ if $|E(P_{ij})| \geq 2$. Moreover, $\sum_{k\neq i,j} \overline{\beta}^j_{ik}$ is half the sum of all $\overline{w}_{it}$ such that $j \in V(P_{it})$ but $\{i,j\} \notin E(P_{it})$, plus half the sum of all $\overline{w}_{st}$ such that $t \neq j$ and $\{i,j\} \in E(P_{st})$ (analogously for $\sum_{k\neq i,j} \overline{\beta}^j_{ik}$ by symmetry).

***Case** $h_{ij} = 1$**:** In this case, no shortest path except for possibly P_{ij} contains edge $\{i,j\}$ by (A1). If $|E(P_{ij})| = 1$, then by (A2) all shortest paths containing both i and j must contain edge $\{i,j\}$, so it follows that no shortest path except for P_{ij} contains both i and j. Hence $\overline{\beta}^k_{ij} = \overline{\beta}^j_{ik} = \overline{\beta}^i_{jk} = 0$ for all $k \neq i,j$ and $\overline{\lambda}^2_{ij} = \overline{w}_{ij}$. Otherwise, $\sum_{k\neq i,j}(\overline{\beta}^k_{ij} - \overline{\beta}^j_{ik} - \overline{\beta}^i_{jk}) = \overline{w}_{ij}$, as all other terms $\overline{w}_{kl}$ cancel out each other, and $\overline{\lambda}^2_{ij} = 0$. If $(i,j) \in E$, we have $\overline{\lambda}^1_{ij} = 0$, so in both cases the constraint corresponding to y_{ij} in the dual problem is tight.*

***Case** $h_{ij} < 1$**:** Here we have $(i,j) \in E$. If $|E(P_{ij})| = 1$, then again all shortest paths containing i and j must contain the edge $\{i,j\}$ by (A2), thus $\sum_{k\neq i,j} \overline{\beta}^k_{ij} = 0$ and $\overline{\lambda}^1_{ij} \geq \sum_{k\neq i,j}(\overline{\beta}^j_{ik} + \overline{\beta}^i_{jk}) + \overline{w}_{ij}$. Otherwise, $\overline{\lambda}^1_{ij} = 0$ by (A1) and $\sum_{k\neq i,j} \overline{\beta}^k_{ij} \geq \sum_{k\neq i,j}(\overline{\beta}^j_{ik} + \overline{\beta}^i_{jk}) + \overline{w}_{ij}$.*

It remains to show that the objective function values of $\overline{y}$ in [PSP] and of $\overline{\lambda}, \overline{\beta}$ in the dual *problem agree, then $\overline{\lambda}, \overline{\beta}$ is an optimal dual solution. We have*

$$\begin{aligned}
\sum_{(i,j)\in E} (\overline{x}_i + \overline{x}_j)\overline{\lambda}^1_{ij} + \sum_{\{i,j\}} \overline{\lambda}^2_{ij} &= \sum_{h_{ij}<1} (\overline{x}_i + \overline{x}_j) \sum_{E(P_{kl})\ni\{i,j\}} \overline{w}_{kl} + \sum_{\substack{h_{ij}=1\\ |E(P_{ij})|=1}} \overline{w}_{ij} \\
&= \sum_{h_{ij}<1} h_{ij} \sum_{E(P_{kl})\ni\{i,j\}} \overline{w}_{kl} + \sum_{\substack{h_{ij}=1\\ |E(P_{ij})|=1}} h_{ij}\overline{w}_{ij} \\
&= \sum_{|E(P_{ij})|=1} h_{ij}\overline{w}_{ij} + \sum_{|E(P_{kl})|\neq 1} \sum_{(i,j)\in P_{kl}} h_{ij}\overline{w}_{kl} \\
&= \sum_{|E(P_{ij})|=1} \overline{y}_{ij}\overline{w}_{ij} + \sum_{|E(P_{kl})|\neq 1} \overline{y}_{kl}\overline{w}_{kl} \\
&= \overline{w}^\top \overline{y}.
\end{aligned}$$

$\square$

Hence at each iteration, the Benders subproblem is solved as discussed in Section 3.1 and a constraint of the form (27) is generated and added to the Benders master problem.

3.3 Complete Algorithm to Compute the Benders Cut

In summary, the following steps are performed to compute a Benders cut for a given solution $\overline{x}$ of the Benders master problem [BMP]:

1. Use the Floyd-Warshall based algorithm of Section 3.1 to compute an optimal solution $\overline{y}$ of [PSP] as well as the corresponding pair-wise shortest paths. This step can be done in $O(|V|^3)$ time.

2. Given $\overline{y}$, compute the corresponding worst case scenario $\overline{w}$. This step highly depends on the uncertainty set $\mathcal{U}$. In Section 4, we provide closed form formulas for all standard uncertainty sets, i.e. interval, scenario based, Gamma, and ellipsoidal uncertainty.

3. Given $\overline{w}$ and the shortest paths of Step 1, compute the Benders cut (27).

Note that in this approach, the computation of an optimal solution $\overline{y}$ of [PSP] is the main algorithmic step needed to compute the Benders cut.

4 Computational Testing

4.1 Analytic Center Cutting Plane Method

The application of classical Benders decomposition often leads to poor performance due to slow convergence. A number of attempts have been made to alleviate this problem. The concept of Pareto-optimal cuts, introduced by Magnanti and Wong (1981), looks for a non-dominated cut to add to the master problem when the subproblem has multiple optimal solutions. Recently, Naoum-Sawaya and Elhedhli (2013) proposed integrating the Benders framework within branch-and-bound and using the analytic center cutting plane method to generate Pareto-optimal cuts thus achieving very good speed-ups compared to classical Benders decomposition. The computational results conducted by Naoum-Sawaya and Elhedhli (2013) showed that the majority of the Benders cuts are generated at the root node of the branch-and-cut tree. We hence use a modified version of the approach presented in Naoum-Sawaya and Elhedhli (2013). Rather than generating the Benders cuts at all the nodes of the branch-and-cut tree, the Benders cuts are generated only at the root node and when a new incumbent solution is found. The modified algorithm can also be seen as warm starting the Benders decomposition algorithm with a set of cuts that are generated from the LP relaxation using the analytic center cutting plane method.

4.2 Implementation

The proposed Benders decomposition algorithm was coded in C using CPLEX 12.5 callable libraries. The computations are carried out on a Lenovo C30 workstation with 2GHz CPU and 16GB of memory. We conduct the computational testing on two classes of instances. The first class of test instances is based on planar graphs that were randomly generated using the rudy graph generator (Rinaldi, 1998). The second class on test instances is based on Barabási-Albert graphs generated using the Barabási graph generator (Dreier, 2006). A total of 1,220 instances of varying sizes and varying uncertainty sets were solved. The CPU time is limited to 10,000 seconds, after which the algorithm stops and returns the best solution found. All running times are compared to solving the problem [RCNP] directly by CPLEX, with default settings.

Table 1: Computational Results for Random Planar Graphs with Interval Uncertainty.

Instance		Benders				CPLEX	
Vertices	Density	CUTS0	CPU0	CPU	Solved	CPU	Solved
100	10	19.6	0.1	0.1	10	4.1	10
	20	60	0.2	0.4	10	5.3	10
	30	119.1	0.4	1	10	16.2	10
	40	154.7	0.6	1.3	10	56.3	10
	50	157.6	0.6	1.5	10	149.1	10
	60	175.7	0.7	1.8	10	498.8	10
	70	198.3	0.8	1.9	10	436	10
	80	194	0.8	1.7	10	491.8	10
	90	189.4	0.7	1.6	10	597.3	10
	Average	140.9	0.5	1.3		250.5	
200	10	38.9	0.8	1.3	10	48.0	10
	20	90.1	2	2.9	10	72.4	10
	30	135.8	3	4.9	10	502.9	10
	40	164.6	3.8	6.7	10	3,628.8	10
	50	180.4	4.1	6.7	10	5,299.0*	6
	60	196	4.6	7	10	6,302.8*	4
	70	190.6	4.5	6.6	10	6,739.9*	1
	80	193.7	4.6	6.6	10	5,655.9*	2
	90	188.6	4.4	6.5	10	5,202.6*	1
	Average	153.2	3.5	5.4		3,716.9	

4.3 Results

We consider different uncertainty sets centered at the nominal scenario w_0, where $(w_0)_{ij} = 1$ for all node pairs (i, j), $i < j$. Parameter K in constraint (2) is set to 5. Computational results for interval uncertainty, scenario based uncertainty, Gamma uncertainty, and ellipsoidal uncertainty are presented next.

4.3.1 Interval Uncertainty

In this section, we consider the interval uncertainty case, i.e. the set

$$\mathcal{U} = \{w \colon w_0 - \hat{w} \leq w \leq w_0 + \hat{w}\}$$

is a box centered at w_0 such that $\hat{w} \geq 0$ and $w_0 - \hat{w} \geq 0$. It is easy to see that in this case the worst case scenario is $w_0 - \hat{w}$. Therefore [RCNP] is equivalent to the nominal problem

$$\begin{aligned} \max_{x,y} \; & (w_0 - \hat{w})^\top y \\ \text{s.t. } & (2) - (8) \end{aligned} \tag{28}$$

which is still $\mathcal{NP}$-hard and the proposed Benders approach is applicable without changes (except that finding $\overline{w}$ is trivial). For our computational testing, we generate random interval

Table 2: Computational Results for Barabási-Albert Instances with Interval Uncertainty.

Instance		Benders				CPLEX	
Vertices	Edges	CUTS0	CPU0	CPU	Opt	CPU	Opt
40	76	204	0.1	2.7	214.18	69.2	214.18
	111	451	0.5	9.6	156.32	96.4	156.32
	144	481	0.8	19.2	129.6	83.1	129.6
	175	493	1.1	246.4	114.1	77.0	114.1
	204	724	2.1	1,057.1	106.06	95.8	106.06
	231	1138	5.1	152.4	117.27	47.8	117.27
	256	721	2.0	57.7	120.55	27.7	120.55
	Average	601.7	1.7	220.7	Gap: 0%	71.0	Gap: 0%
50	96	351	0.4	4.2	325.17	294.1	325.17
	141	689	1.6	28.1	203.49	317.6	203.49
	184	491	0.9	97.7	170.6	330.0	170.6
	225	847	3.5	704.6	147.68	309.6	147.68
	264	1096	7.2	8,910.8	132.06	250.4	132.06
	301	749	3.2	294.7	141.17	189.3	141.17
	336	906	4.9	2,297.8	132.06	280.9	132.06
	Average	732.7	3.1	1,762.6	Gap: 0%	281.7	Gap: 0%
60	116	806	2.2	103.6	302.35	1,251.4	302.35
	171	905	3.0	196.9	231.17	1,205.8	231.17
	224	1577	13.0	1,180.0	196.36	1,290.0	196.36
	275	1659	15.7	2,930.3	196.07	1,214.6	196.07
	324	1643	19.7	>10,000	(158.39;188.59)	1,387.0	158.39
	371	2494	58.8	4,796.7	181.4	1,041.8	181.4
	416	1008	9.3	683.8	176.47	685.4	176.47
	Average	1,408.2*	17.0*	1,648.5*	Gap: 2%	1,153.7	Gap: 0%
70	136	478	1.1	10.8	544.19	2,696.6	544.19
	201	1388	10.2	4,029.2	241.55	7,079.2	241.55
	264	2305	37.8	4,009.0	239.85	4,388.5	239.85
	325	2180	39.5	>10,000	(206.81;240.45)	7,126.0	206.81
	384	3291	101.4	>10,000	(197.76;246.02)	6,875.1	197.76
	441	2066	52.2	7,093.5	192.06	4,695.5	192.06
	496	3054	95.6	>10,000	(214.42;226.52)	3,528.8	214.42
	Average	1,559.3*	25.3*	3,785.6*	Gap: 6%	5,198.5	Gap: 0%

sizes by choosing each entry of $\hat{w}$ uniformly at random from $[0, 1]$. For comparison, we apply the CPLEX MIP solver directly to problem (28).

Results for randomly generated planar graphs are given in Table 1. For each problem size, 10 different instances were generated and average results are shown. The following additional details are reported in Table 1:

$Vertices$:	Number of vertices in the graph.
$Density$:	Percentage of the maximum number of edges where the maximum number of edges for a planar graph is given by $3 \times (\lvert V \rvert - 2)$.
$CUTS0$:	Number of Benders cuts that are generated at the root node.
$CPU0$:	Computational time in seconds spent on generating Benders cuts at the root node.
CPU :	Total computational time in seconds.
$Solved$:	Number of instances solved within the time limit.

We notice that the proposed Benders approach clearly outperforms CPLEX. For instances with 100 vertices, the proposed Benders approach solved all the instances in less than 2 seconds of computational time on average while CPLEX consumed an average computational time of 250 seconds. For instances with 200 vertices, the proposed Benders approach solved all the attempted instances in less than 6 seconds of computational time while CPLEX failed in solving within the time limit 36 out of the 90 attempted instances.

Table 2 shows results for instances with Barabási-Albert graphs. For these instances, the optimal objective function values are displayed for the cases that were solved within the time limit while the best lower and upper bounds are displayed for the cases where the time limit was exceeded. In case not all instances are solved to optimality, averages are taken over all solved instances and marked with an asterisk (*).

We notice that instances with Barabási-Albert graphs are significantly harder than the ones with planar graphs. For these instances, CPLEX solved all of the 28 problems within the time limit while the proposed Benders approach reached the time limit for 4 cases however still achieved a faster computational time compared to CPLEX in 13 of the 28 test cases. We also note that all the instances in which the proposed Benders approach outperformed CPLEX had a relatively low number of edges hence indicating that the proposed Benders

Table 3: Computational Results for Random Planar Graphs with Scenario Uncertainty.

Instance		Benders				CPLEX	
Vertices	Density	CUTS0	CPU0	CPU	Solved	CPU	Solved
100	10	23.6	0.3	0.4	10	13.3	10
	20	63.2	0.7	1.1	10	23.4	10
	30	127.1	1.4	2.2	10	53.5	10
	40	157.3	1.8	2.8	10	141.0	10
	50	164.0	1.9	3.3	10	422.4	10
	60	166.2	1.9	3.5	10	854.7	10
	70	202.3	2.4	4.0	10	836.8	10
	80	186.9	2.2	3.4	10	834.1	10
	90	186.2	2.1	3.2	10	807.5	10
	Average	141.9	1.6	2.7		443.0	
200	10	49.5	2.5	3.3	10	107.6	10
	20	96.3	5.1	6.8	10	198.1	10
	30	132.0	6.9	10.1	10	778.6	10
	40	164.9	8.8	13.8	10	3,955.8*	9
	50	188.0	9.9	13.9	10	4,536.4*	5
	60	191.2	10.3	14.6	10	7,399.6*	5
	70	189.4	10.1	13.9	10	7,286.9*	2
	80	194.9	10.6	14.0	10	7,379.5*	2
	90	191.3	10.2	13.6	10	8,256.6*	1
	Average	155.3	8.3	11.6		4,433.2	

approach is computationally efficient mostly for sparse graphs, as is also the case with planar graphs.

4.3.2 Scenario based Uncertainty

In this section, we consider scenario based robustness where the uncertainty set is

$$\mathcal{U} = \{w^1, w^2, \ldots, w^n\}.$$

In this case the subproblem (22)–(23) is solved by evaluating each of the scenarios and choosing the scenario i that minimizes $(w^i)^\top \overline{y}$. For each instance in the computational experiments, we generate vectors $w^1, \ldots, w^{1000}$ each chosen uniformly at random from the ball around w_0 with radius 1. For comparison with CPLEX, we reformulate problem [RCNP] by replacing $\min\{(w^1)^\top y, (w^2)^\top y, \ldots, (w^n)^\top y\}$ with a new variable z and adding the constraint $z \leq (w^i)^\top y$ for every scenario i thus turning the problem into a MIP.

The same instances that were discussed in the interval uncertainty case are also solved and the results are reported in Tables 3 and 4. Similar to the observations in the interval uncertainty case, the proposed Benders approach significantly outperforms CPLEX for instances with planar graphs. For problems with 100 vertices, the average computational time for the Benders approach was 2.7 seconds while CPLEX achieved an average of 443 seconds

Table 4: Computational Results for Barabási-Albert Instances with Scenario based Uncertainty.

Instance		Benders				CPLEX	
Vertices	Edges	CUTS0	CPU0	CPU	Opt	CPU	Opt
40	76	203	0.3	5.7	419.49	292.0	419.49
	111	397	0.8	9.0	312.16	314.5	312.16
	144	492	1.4	24.1	250.29	298.8	250.29
	175	586	2.2	723.2	217.56	308.7	217.56
	204	623	2.6	>10,000	(183.88;218.16)	>10,000	(183.94;217.63)
	231	851	4.1	558.0	217.61	382.1	217.61
	256	569	2.0	29.8	249.82	68.2	249.82
	Average	516.3*	1.8*	224.9*	Gap: 2%	277.4*	Gap: 2%
50	96	330	1.1	5.8	646.51	501.4	646.51
	141	649	2.9	26.3	402.31	1,456.3	402.31
	184	399	1.5	362.9	320.72	1,006.0	320.72
	225	888	6.2	2,470.7	277.77	1,095.8	277.77
	264	953	7.9	>10,000	(234;313.13)	>10,000	(234.05;240.43)
	301	665	4.4	345.6	277.79	443.3	277.79
	336	1085	9.4	>10,000	(233.99;322.57)	>10,000	(233.96;240.36)
	Average	586.2*	3.2*	642.2*	Gap: 8%	900.5*	Gap: 1%
60	116	805	4.5	156.2	590.99	3,770.8	590.99
	171	856	5.8	342.0	442.63	2,949.4	442.63
	224	1394	16.9	1,239.0	390.51	3,255.9	390.51
	275	1862	31.2	4,819.1	390.74	2,507.6	390.74
	324	1959	43.1	>10,000	(284.08;398.91)	>10,000	(284.05;332.08)
	371	1800	42.2	>10,000	(337.98;394.18)	>10,000	(337.98;355.21)
	416	940	12.7	1,416.2	337.5	1,989.8	337.50
	Average	1,171.4*	14.2*	1,594.5*	Gap: 6%	2,894.7*	Gap: 3%
70	136	449	2.8	17.5	1075.93	3,360.9	1075.93
	201	1356	16.4	7,952.2	460.89	>10,000	(460.85;515.17)
	264	2318	62.4	4,422.5	460.81	9,092.1	460.81
	325	1934	51.1	>10,000	(398;493.39)	>10,000	(398;516.99)
	384	3286	140.4	>10,000	(397.94;485.86)	7,185.0	397.94
	441	2018	75.8	>10,000	(334.18;521.66)	>10,000	(334.07;407.48)
	496	2563	112.2	>10,000	(398.03;453.15)	>10,000	(398.03;422.25)
	Average	1,374.3*	27.2*	4,130.7*	Gap: 12%	6,546.0*	Gap: 8%

Table 5: Computational Results for Random Planar Graphs with Gamma Uncertainty.

Instance		Benders				CPLEX	
Vertices	Density	CUTS0	CPU0	CPU	Solved	CPU	Solved
100	10	21.5	0.1	0.2	10	4.9	10
	20	64.5	0.3	0.5	10	6.8	10
	30	127.4	0.6	1.2	10	21.2	10
	40	168.1	0.8	1.6	10	86.5	10
	50	187.0	0.9	1.9	10	211.1	10
	60	185.9	0.9	2.1	10	505.6	10
	70	218.3	1.1	2.4	10	563.7	10
	80	209.8	1.0	2.0	10	752.2	10
	90	209.1	1.0	1.9	10	660.4	10
	Average	154.6	0.7	1.5		312.5	
200	10	46.8	1.2	1.5	10	52.3	10
	20	90.0	2.3	3.3	10	80.2	10
	30	134.7	3.4	5.5	10	594.3	10
	40	165.2	4.3	7.4	10	3,097.2*	9
	50	188.9	5.0	7.6	10	4,848.6*	6
	60	183.8	5.0	7.7	10	5,794.2*	5
	70	187.7	5.0	7.4	10	6,422.7*	3
	80	195.7	5.3	7.7	10	5,639.6*	3
	90	193.9	5.2	7.3	10	5,188.8*	1
	Average	154.1	4.1	6.1		3,524.2	

of computational time. For instances with 200 vertices, the proposed Benders approach achieved 11.6 seconds while CPLEX failed in solving 38 out of the 90 attempted instances.

For the instances that are based on Barabási-Albert graphs that are shown in Table 4, the proposed Benders approach achieved a better computational time than CPLEX for 15 out of the 28 instances while CPLEX achieved a better computational time for 5 instances, while both approaches reached the time limit for the remaining 8 instances.

4.3.3 Gamma-Uncertainty

In this section, we consider the Gamma robustness as defined by Bertsimas and Sim (2004) and considered by Fan and Pardalos (2010a) in the context of the critical node selection problem. In this case, at most $\lfloor \Gamma \rfloor$ entries of w are allowed to deviate from the nominal scenario w_0, in addition to one entry (i, j) that is allowed to deviate by $(\Gamma - \lfloor \Gamma \rfloor)\hat{w}_{ij}$ for a given $\Gamma \geq 0$. The subproblem (22)–(23) can then be rewritten as

$$\begin{aligned} \bar{y}^\top w_0 - \max_{v} \ & \tilde{y}^\top v \\ \text{s.t. } & 0 \leq v \leq 1 \\ & e^\top v \leq \Gamma \end{aligned}$$

Table 6: Computational Results for Barabási-Albert Instances with Gamma Uncertainty.

Instance		Benders				CPLEX	
Vertices	Edges	CUTS0	CPU0	CPU	Opt	CPU	Opt
40	76	197	0.2	3.2	402.52	117.6	402.52
	111	464	0.6	8.2	294.62	106.8	294.62
	144	786	1.9	28.3	232.87	116.9	232.87
	175	845	2.9	863.0	200.53	108.2	200.53
	204	902	3.4	>10,000	(167.22;203.62)	2,807.8	167.38
	231	940	3.7	989.4	200.43	107.8	200.43
	256	865	3.0	69.1	232.7	38.3	232.7
	Average	682.8*	2.0*	326.9*	Gap: 3%	486.20	Gap: 0%
50	96	347	0.4	4.9	624.5	149.0	624.50
	141	735	1.8	25.3	379.89	327.0	379.89
	184	537	1.1	515.2	298.21	460.1	298.21
	225	1320	7.7	6,386.2	255.35	357.4	255.35
	264	1216	8.3	>10,000	(212.59;277.64)	>10,000	(212.71;255.25)
	301	903	4.9	854.5	255.25	161.8	255.25
	336	1379	11.3	>10,000	(212.58;285.58)	>10,000	(212.71;213.26)
	Average	768.4*	3.2*	1,557.2*	Gap: 7%	291.0*	Gap: 2%
60	116	856	2.6	140.3	563.96	1,393.0	563.96
	171	992	3.8	361.7	415.2	1,380.4	415.20
	224	1664	15.3	1,300.3	363.45	3,381.4	363.45
	275	1862	20.3	3,562.9	363.48	1,710.2	363.48
	324	2339	41.5	>10,000	(257.63;366.96)	>10,000	(257.79;268.47)
	371	1815	28.3	>10,000	(310.98;356.58)	8,750.1	310.98
	416	1325	15.6	4,591.5	310.44	832.9	310.44
	Average	1,339.8*	11.5*	1,991.3*	Gap: 6%	2,908.0*	Gap: 1%
70	136	484	1.2	12.8	1043.68	1,632.6	1043.68
	201	1489	10.8	5,551.2	428.92	>10,000	(428.81;492.21)
	264	2448	45.2	3,290.7	428.52	>10,000	(428.34;475.70)
	325	2560	65.4	>10,000	(365.84;472.64)	6,363.7	365.84
	384	3592	167.9	>10,000	(365.64;446.59)	4,510.0	365.64
	441	2746	122.6	>10,000	(303.2;504.32)	>10,000	(303.36;343.25)
	496	3570	180.0	>10,000	(366.39;473.46)	>10,000	(366.41;382.09)
	Average	1,473.7*	19.1*	2,951.6*	Gap: 15%	4,168.7*	Gap: 6%

where $\tilde{y}_{ij} = \overline{y}_{ij}\hat{w}_{ij}$. An optimal solution $\overline{w}$ is easily obtained by setting $w_{ij} = (w_0)_{ij} - \hat{w}_{ij}$ for the $\lfloor \Gamma \rfloor$ largest entries $\tilde{y}_{ij}$ and setting $w_{ij} = (w_0)_{ij} - (\Gamma - \lfloor \Gamma \rfloor)\hat{w}_{ij}$ for the next largest entry $\tilde{y}_{ij}$. For comparison, we use the approach of Bertsimas and Sim (2004) to reformulate problem [RCNP] with Gamma-uncertainty as a MIP that we solve using CPLEX.

For our computational testing, the same $\hat{w}$ as in Section 4.3.1 are used and we assume $\Gamma = \frac{|V|}{2}$. The results for the planar and Barabási-Albert graphs are reported in Tables 5 and 6, respectively. As shown in Table 5, the proposed Benders approach consistently outperformed CPLEX for planar graphs. For the instances that are based on Barabási-Albert graphs which are displayed in Table 6, the proposed Benders approach outperformed CPLEX for 11 instances, while CPLEX achieved a better computational time for 12 instances, and both approaches failed in finding an optimal solution within the time limit for the remaining 5 instances.

Table 7: Computational Results for Random Planar Graphs with Ellipsoidal Uncertainty.

Instance		Benders				CPLEX	
Vertices	Density	CUTS0	CPU0	CPU	Solved	CPU	Solved
100	10	27.5	1.1	1.5	10	61.0*	8
	20	70.1	2.8	3.6	10	81.8	10
	30	128.5	5.1	6.9	10	158.8	10
	40	167.4	6.6	8.9	10	750.0	10
	50	171.5	6.8	10.0	10	1,945.8	10
	60	175.2	7.0	10.3	10	2,480.0*	9
	70	206.5	8.2	11.3	10	2,343.0*	8
	80	193.8	7.7	10.5	10	2,038.5*	9
	90	195.7	7.7	10.0	10	1,372.0*	9
	Average	148.5	5.9	8.1		1,247.9	
200	10	48.6	27.6	35.7	10	8,027.7*	1
	20	89.7	52.6	71.8	10	-	0
	30	132.7	75.9	104.8	10	-	0
	40	162.9	91.7	130.5	10	-	0
	50	201.1	114.6	147.4	10	-	0
	60	194.3	109.5	143.3	10	-	0
	70	197.0	112.4	142.4	10	-	0
	80	200.7	114.3	143.1	10	-	0
	90	196.6	115.2	142.9	10	-	0
	Average	158.2	90.4	118.0		8,027.7	

4.3.4 Ellipsoidal-Uncertainty

In this section, we consider ellipsoidal robustness (Ben-Tal et al., 2009) where the uncertainty set is

$$\mathcal{U} = \{w\colon (w - w_0)W^{-1}(w - w_0) \leq 1\}$$

for a positive definite matrix W such that $w \geq 0$ for all $w \in \mathcal{U}$. In this case the subproblem (22)–(23) is minimizing a linear objective function over an ellipsoid and hence admits an optimal solution

$$\overline{w} = w_0 - \frac{1}{\sqrt{\overline{y}^\top W \overline{y}}} W \overline{y}.$$

For each instance in the computational experiments, we choose the matrix W as $AA^\top$ for a random matrix A. The largest eigenvalue of W is then normalized to 1.

Since $\overline{w}^\top y = w_0^\top y - \sqrt{y^\top W y}$, we can model [RCNP] with ellipsoidal uncertainty as a second-order cone program that we solve using CPLEX. The results for planar and Barabási-Albert graphs are reported in Tables 7 and 8, respectively. The proposed Benders approach remains very efficient for instances with planar graphs where all the instances are solved to optimality within an average computational time of 8.1 seconds for problems with 100 vertices and 118 seconds for problems with 200 vertices. CPLEX however failed in solving the majority of the attempted instances where only 1 instance is solved for problems with

Table 8: Computational Results for Barabási-Albert Instances with Ellipsoidal Uncertainty.

Instance		Benders				CPLEX	
Vertices	Edges	CUTS0	CPU0	CPU	Opt	CPU	Opt
40	76	214	0.3	4.0	406.88	1,964.6	406.88
	111	417	0.8	10.4	302.77	1,674.3	302.77
	144	453	1.1	21.4	242.98	1,269.0	242.98
	175	583	1.9	263.9	211.17	877.2	211.17
	204	650	2.2	>10,000	(178.40;213.72)	6,645.7	178.40
	231	927	4.3	464.7	211.17	556.4	211.17
	256	648	2.2	29.4	242.99	344.7	242.99
	Average	540.3*	1.8*	132.3*	Gap: 2%	1,904.5	Gap: 0%
50	96	344	1.2	6.4	630.46	4,547.4	630.46
	141	725	3.6	30.0	392.45	2,928.8	392.45
	184	368	1.5	457.8	312.82	4,713.3	312.82
	225	885	6.5	2,273.9	271.05	3,356.4	271.05
	264	921	8.0	>10,000	(228.32;297.47)	>10,000	(228.32;231.21)
	301	778	5.8	351.6	271.04	1,656.8	271.04
	336	1172	12.3	>10,000	(228.32;309.26)	>10,000	(228.32;231.21)
	Average	620.0*	3.7*	623.9*	Gap: 7%	3,440.5*	Gap: 0%
60	116	793	6.1	157.7	578.91	>10,000	(0;881.99)
	171	929	8.3	401.3	433.45	8,350.2	433.45
	224	1517	22.1	1,351.1	382.69	9,268.5	382.69
	275	1914	35.8	3,570.0	382.68	>10,000	(330.95;512.27)
	324	1919	43.9	>10,000	(278.25;460.43)	>10,000	(278.24;366.37)
	371	1965	52.4	>10,000	(330.96;391.27)	>10,000	(330.96;366.58)
	416	976	15.5	1,549.0	330.94	5,370.2	330.94
	Average	1,225.8*	17.5*	1,405.8*	Gap: 8%	7,663.0*	Gap: 24%
70	136	465	5.2	18.6	1056.06	>10,000	(0;1317.51)
	201	1337	22.5	9,991.8	452.61	>10,000	(0;797.41)
	264	2340	78.3	5,372.6	452.6	>10,000	(0;684.95)
	325	1829	53.3	>10,000	(390.89;477.85)	>10,000	(0;599.83)
	384	3060	142.0	>10,000	(328.2;545.41)	>10,000	(0;566.24)
	441	2247	91.9	>10,000	(328.2;554.24)	>10,000	(0;520.99)
	496	2900	119.5	>10,000	(390.89;480.99)	>10,000	(0;509.86)
	Average	1,380.7*	35.3*	5,127.7*	Gap: 17%	-	Gap: 100%

200 vertices. Similarly, the proposed Benders approach also outperformed CPLEX for the instances that are based on Barabási-Albert graphs which are shown in Table 8. Out of the 28 attempted instances, the proposed Benders approach achieved a lower computational time than CPLEX in 19 instances, CPLEX achieved a better computational time for only 1 instance, while both approaches failed in finding an optimal solution for the remaining 9 instances. Note that for the Barabási-Albert instances with 70 nodes, CPLEX could not even find a lower bounds for the problem within the time limit.

Table 9: Computational Results for Barabási-Albert Instances with Ellipsoidal Uncertainty and $K = 1, 2, 10, 20$.

		$K = 1$			
Instance		Benders		CPLEX	
Vertices	Edges	CPU	Opt	CPU	Opt
50	96	1.1	47.61	6115.0	47.61
	141	0.9	47.61	1853.1	47.61
	184	1.3	94.23	704.4	94.23
	225	8.1	47.61	4852.2	47.61
	264	10.8	47.61	4083.2	47.61
	301	7.4	47.61	2287.0	47.61
	336	15.5	47.61	1806.0	47.61
	Average	6.4	Gap: 0%	3100.1	Gap: 0%
		$K = 2$			
Instance		Benders		CPLEX	
50	96	1.5	184.57	2941.7	184.57
	141	6.5	94.24	2063.9	94.24
	184	1.8	139.89	3418.9	139.89
	225	19.2	94.24	4428.5	94.24
	264	23.7	94.24	3855.7	94.24
	301	17.6	94.24	1938.6	94.24
	336	31.8	94.24	2623.1	94.24
	Average	14.6	Gap: 0%	3038.6	Gap: 0%
		$K = 10$			
Instance		Benders		CPLEX	
50	96	65.4	1133.77	41.8	1133.77
	141	1483.2	895.75	4112.7	895.75
	184	>10,000	(645.04, 721.19)	>10,000	(645.04, 690.57)
	225	>10,000	(543.04, 771.63)	>10,000	(543.06, 553.64)
	264	>10,000	(507.11, 741.72)	>10,000	(507.11, 515.82)
	301	>10,000	(507.10, 784.19)	4445.6	507.1
	336	>10,000	(507.09, 719.74)	4412.4	507.09
	Average	774.3*	Gap: 20%	3253.1*	Gap: 1%
		$K = 20$			
Instance		Benders		CPLEX	
50	96	0.9	1189.03	1.6	1189.03
	141	173.5	1178.35	12.3	1178.35
	184	91.1	1176.41	8.5	1176.41
	225	>10,000	(1128.81, 1165.97)	779.9	1128.86
	264	>10,000	(1046.24, 1159.25)	>10,000	(1046.24, 1077.12)
	301	>10,000	(921.89, 1150.82)	>10,000	(929.66, 1070.61)
	336	>10,000	(898.58, 1145.70)	>10,000	(918.98, 1020.39)
	Average	88.5*	Gap: 22%	200.6*	Gap: 4%

4.3.5 Varying Budget

In the computational results that are presented earlier, parameter K in constraint (2) is set to 5. In this section, we consider $K = 1, 2, 10, 20$ and evaluate the impact on the computational performance. Since the proposed approach is particularly effective for ellipsoidal uncertainties, we limit the testing for this type of uncertainty sets and results for Barabási-Albert instances with 50 nodes and planar graph instances with 100 nodes are presented in Tables 9 and 10, respectively.

The results show that the proposed approach performs significantly better than CPLEX for small values of K. For Barabási-Albert instances with $K = 1$ and $K = 2$ the proposed

Table 10: Computational Results for Random Planar Graphs with Ellipsoidal Uncertainty and $K = 1, 2, 10, 20$.

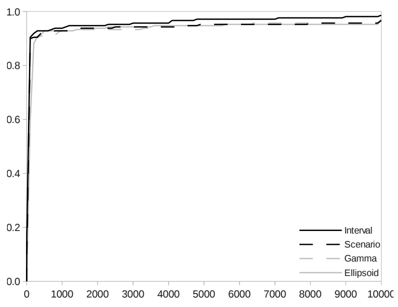
		$K = 1$				$K = 2$			
Instance		Benders		CPLEX		Benders		CPLEX	
Vertices	Density	Solved	CPU	Solved	CPU	Solved	CPU	Solved	CPU
100	10	10	0.5	10	90.4	10	0.9	10	77.6
	20	10	0.7	10	224.5	10	1.6	10	156.1
	30	10	1.2	9	3661.1*	10	2.8	8	828.0*
	40	10	1.4	3	1516.5*	10	2.6	10	3375.4
	50	10	1.9	-	0	10	3.6	2	2027.8*
	60	10	1.9	-	0	10	4.1	3	5944.1*
	70	10	2.2	-	0	10	4.4	2	8849.2*
	80	10	2.3	-	0	10	5.6	2	1028.3*
	90	10	2.9	-	0	10	8.5	-	0
	Average		1.6		1373.1*		3.8		2785.8*
		$K = 10$				$K = 20$			
100	10	10	1.1	9	40.4*	10	7.5	10	52.4
	20	10	3.5	10	56.9	10	5.1	10	52.0
	30	10	21.4	10	81.9	10	8.5	8	55.2*
	40	10	58.1	10	197.3	9	163.9	10	174.4
	50	10	98.9	3	1057.9*	6	122.6	2	4489.3*
	60	10	100.6	3	2212.9*	5	869.3	4	2108.1*
	70	10	108.1	4	3065.7*	6	1570.2	5	4100.2*
	80	10	100.9	2	3486.0*	5	1398.2	3	4121.0*
	90	10	66.5	2	3039.3*	8	3021.8	8	2120.0*
	Average		62.1		1470.9*		796.3*		1894.1*

approach found the optimal solution for all the tested instances in an average CPU time of 6.4 and 14.6 seconds respectively while CPLEX required an average of CPU time of 3100.1 and 3038.6 seconds respectively. The computational time however increases as K increases. For $K = 10$ and $K = 20$, the proposed approach fails to solve 9 of the tested Barabási-Albert instances within the time limit, while CPLEX fails in solving 6 instances.

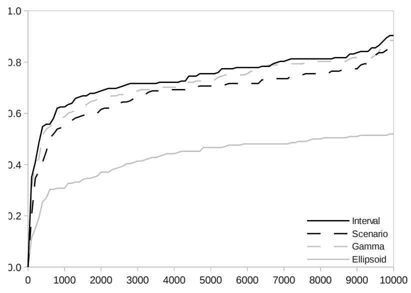
The Results for the planar graphs are displayed in Table 10. The proposed approach outperforms CPLEX for all the tested values of K, however, we notice that the computational time increases from an average of 1.6 seconds with $K = 1$ to an average of 796.3 seconds with $K = 20$.

4.4 Results Summary

This section provides a summary of the effect of uncertainty on the performance of the proposed Benders decomposition algorithm and on the performance of CPLEX. The performance profiles which are given in Figure 1 (a) show that the type of the uncertainty set has little effect on the performance of the proposed Benders decomposition algorithm. The interval uncertainty, which was shown in Section 4.3.1 to correspond to the nominal case, results in the best performance however with only a slight advantage over the remaining uncertainty



(a) Benders Decomposition



(b) CPLEX

Figure 1: Performance profiles for the 4 different uncertainty types with respect to the running times for the proposed Benders decomposition algorithm (a) and for CPLEX (b).

sets which have identical performance, thus revealing the advantage of the proposed Benders decomposition approach which handles the uncertainty sets through an efficient optimization oracle. The performance of CPLEX is however highly dependent on the type of the uncertainty set. As shown in Figure 1 (b), CPLEX achieves the best performance when interval uncertainty is used, outperforming the cases with Gamma, scenario based, and ellipsoidal uncertainty, with the latter being the most computationally challenging.

5 Conclusion

This paper provides a Benders decomposition algorithm for the robust critical node selection problem for a very general class of uncertainty sets. The novel Benders decomposition approach exploits the structure of the Benders subproblem while explicitly dealing with the uncertainty. A polynomial time algorithm based on the Floyd-Warshall approach is devised to solve the Benders subproblem. Extensive computational testing is conducted and a comparison with CPLEX is presented using four classes of uncertainty sets: interval uncertainty, scenario based uncertainty, Gamma uncertainty, and ellipsoidal uncertainty. For all classes of uncertainty sets, the proposed Benders approach achieved remarkable performance particularly for sparse graphs.

References

B. Addis, M. D. Summa, and A. Grosso. Identifying critical nodes in undirected graphs: Complexity results and polynomial algorithms for the case of bounded treewidth. *Discrete*

Applied Mathematics, 161(1617):2349 – 2360, 2013.

A. Arulselvan, C. W. Commander, P. M. Pardalos, and O. Shylo. Managing network risk via critical node identification. In N. Gülpınar and B. Rustem, editors, *Risk management in telecommunication networks*, pages 474–477. Springer, 2007.

A. Arulselvan, C. W. Commander, O. Shylo, and P. M. Pardalos. Cardinality-constrained critical node detection problem. In N. Gülpınar, P. Harrison, and B. Rustem, editors, *Performance Models and Risk Management in Communications Systems*, volume 46 of *Springer Optimization and Its Applications*, pages 79–91. Springer New York, 2011.

A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. *Robust Optimization.* Princeton Series in Applied Mathematics, 2009.

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

D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

R. Cohen, S. Havlin, and D. Ben-Avraham. Efficient immunization strategies for computer networks and populations. *Physical review letters*, 91(24):247901–247905, 2003.

C. W. Commander, P. M. Pardalos, V. Ryabchenko, S. Uryasev, and G. Zrazhevsky. The wireless network jamming problem. *Journal of Combinatorial Optimization*, 14(4):481–498, 2007.

M. Di Summa, A. Grosso, and M. Locatelli. Complexity of the critical node problem over trees. *Computers & Operations Research*, 38(12):1766 – 1774, 2011.

M. Di Summa, A. Grosso, and M. Locatelli. Branch and cut algorithms for detecting critical nodes in undirected graphs. *Computational Optimization and Applications*, 53(3):649–680, 2012.

D. Dreier. *Barabasi graph generator*, 2006. www.cs.ucr.edu/ddreier.

N. Fan and P. M. Pardalos. Robust optimization of graph partitioning and critical node detection in analyzing networks. In W. Wu and O. Daescu, editors, *Combinatorial Op-*

timization and Applications, volume 6508 of *Lecture Notes in Computer Science*, pages 170–183. Springer Berlin Heidelberg, 2010a.

N. Fan and P. M. Pardalos. Linear and quadratic programming approaches for the general graph partitioning problem. *Journal of Global Optimization*, 48(1):57–71, 2010b.

N. Fan, Q. P. Zheng, and P. M. Pardalos. Robust optimization of graph partitioning involving interval uncertainty. *Theoretical Computer Science*, 447:53–61, 2012.

R. Floyd. Algorithm 97: Shortest Path. *Communications of the ACM*, 5(6):345, 1962.

N. Garg, V. V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi) cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996.

A. M. Geoffrion. Generalized Benders decomposition. *Journal of optimization theory and applications*, 10(4):237–260, 1972.

B. Ghaddar, M. Anjos, and F. Liers. A branch-and-cut algorithm based on semidefinite programming for the minimum k-partition problem. *Annals of Operations Research*, 188 (1):155–174, 2011.

T. L. Magnanti and R. T. Wong. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981.

J. Naoum-Sawaya and S. Elhedhli. An interior-point Benders based branch-and-cut algorithm for mixed integer programs. *Annals of Operations Research*, 210(1):33–55, 2013.

G. Rinaldi. Rudy: a rudimental graph generator, 1998. http://www-user.tu-chemnitz.de/∼helmberg/rudy.tar.gz.

S. Shen and J. C. Smith. Polynomial-time algorithms for solving a class of critical node problems on trees and series-parallel graphs. *Networks*, 60(2):103–119, 2012.

S. Shen, J. C. Smith, and R. Goli. Exact interdiction models and algorithms for disconnecting networks via node deletions. *Discrete Optimization*, 9(3):172 – 188, 2012.

Z. Tao, F. Zhongqian, and W. Binghong. Epidemic dynamics on complex networks. *Progress in Natural Science*, 16(5):452–457, 2006.

M. Ventresca. Global search algorithms using a combinatorial unranking-based problem representation for the critical node detection problem. *Computers & Operations Research*, 39(11):2763 – 2775, 2012.