The Stochastic Critical Node Problem over Trees

Pierre Hosteins, Rosario Scatamacchia

The Stochastic Critical Node Problem over Trees

Pierre Hosteins *
Univ Gustave Eiffel, COSYS-ESTAS,
F-59666 Villeneuve d'Ascq, Lille, France
e-mail: pierre.hosteins@ifsttar.fr

Rosario Scatamacchia
Politecnico di Torino,
Dipartimento di Ingegneria Gestionale e della Produzione,
Corso Duca degli Abruzzi 24 - 10129 Torino, Italy
e-mail: rosario.scatamacchia@polito.it

Abstract

We tackle a stochastic version of the Critical Node Problem (CNP) where the goal is to minimize the pairwise connectivity of a graph by attacking a subset of its nodes. In the stochastic setting considered, the outcome of attacks on nodes is uncertain. In our work, we focus on trees and demonstrate that over trees the stochastic CNP actually generalizes to the stochastic Critical Element Detection Problem where the outcome of attacks on edges is also uncertain. We prove the NP-completeness of the decision version of the problem when connection costs are one, while its deterministic counterpart was proved to be polynomial. We then derive a nonlinear model for the considered CNP version over trees and provide a corresponding linearization based on the concept of *probability chains*. Moreover, given the features of the derived linear model, we devise an exact Benders Decomposition approach where we solve the slave subproblems analytically. A strength of our approach is that it does not rely on any statistical approximation such as the Sample Average Approximation, which is commonly employed in stochastic optimization. We also introduce an approximation algorithm for the problem variant with unit connection costs and unit attack costs, and a specific integer linear model for the case where all the survival probabilities of the nodes in case of an attack are equal. Our methods are capable of solving relevant instances of the problem with hundreds of nodes within one hour of computational time. With this work, we aim to foster research on stochastic versions of the Critical Node Problem, a problem tackled mainly in deterministic contexts so far. Interestingly, we also show a successful application of the concept of probability chains for problem linearizations significantly improved by decomposition methods such as the Benders Decomposition.

Keywords: Network interdiction, Critical Element Detection, Critical Node Problem, Stochastic Integer Programming, Trees, Probability Chains, Benders Decomposition, Approximation algorithm.

*Corresponding author.

1 Introduction

We consider the Critical Node Problem (CNP), as introduced in [7], where the goal is to attack a subset of the nodes of an undirected graph in order to minimize its pairwise connectivity, namely the number of pairs of nodes still connected by a path in the fragmented graph. In this paper, we tackle a stochastic version of the CNP over trees where the outcome of attacks on nodes is uncertain. For such a CNP version, we propose both theoretical results and an exact solution method based on the concept of *probability chains* from probability theory and Benders Decomposition (BD). Further approaches are proposed for specific variants of the problem. This work extends some of the results in the literature for the CNP over trees within a stochastic framework. From a practical point of view, the problem we consider might find applications in contexts where the underlying graph is a tree, for instance in networks with an intrinsic hierarchical structure where each node has "superiors" and "subordinates", such as the dismantling of a network of terrorists (see, e.g., [15]) or of a network of drug dealers as in [18]. The CNP is a particular case of the Critical Element Detection Problem (CEDP) [46] which calls for fragmenting an undirected graph as much as possible by the deletion of a subset of its edges and nodes. The aim of the CEDP/CNP is to give indications on the critical elements of a network to protect or to attack according to the application of interest. These problems are connected to Network Interdiction Problems (NIPs), which also consider the removal of network components and have been intensively investigated in the recent years [39, 40]. Real-life applications of these problems arise in many fields, including transportation networks [25], power distribution networks [36, 37], diffusion phenomena such as viral infections and their mitigation [50, 44], homeland security [11] and bioinformatics [42]. We refer to [39, 40] for a comprehensive introduction on network interdiction problems and their applications. There exist many different connectivity metrics for estimating the dismantling of a network. Earlier works on network interdiction mainly focused on the maximum flow that could be transported from a source node s to a sink node t in a graph [49], or on increasing the length of the shortest path between s and t [8, 23]. Studies [12, 13] consider further variants of maximum flow and shortest path problems involving the detection of the most vital nodes in a network. In comparison, the CEDP and the CNP focus on measures related to the cohesive properties of a graph, such as the number of maximally connected components, the maximum cardinality of the connected components or the pairwise connectivity.

The CNP has been the object of numerous publications in the recent years. The authors of [7] prove the strong NP-hardness of the problem and propose a Mixed Integer Linear Programming (MILP) formulation along with a heuristic algorithm. Improved MILP formulations have been proposed afterwards in [16, 26, 33, 45] and in [46] where the deletion of edges is also considered. Since the MILP formulations can only be reasonably applied to graphs of limited size, many heuristic approaches were proposed for dealing with large graphs. Among others, we mention the recent studies in [3, 4, 5, 34, 51]. The study of the CNP on specific graph classes such as trees attracted particular attention, as well. The authors of [15] study the complexity of the pairwise connectivity CNP over trees by analyzing different problem variants and providing polynomial and super-polynomial algorithms. The works [27] and [38] provide complexity results for the CNP over trees and other specially structured graphs using alternative connectivity metrics, such as the maximum cardinality of the connected components, the number of connected components or the maximum pairwise connectivity between all the components. Further complexity and approximation results for general or specially structured graphs are given in [1]. The study [47] discusses a generalization of the pairwise connectivity CNP where the distances between node pairs impact the objective function. A related CNP variant over trees is analyzed in [6]. More precisely, the authors of [6] establish complexity results according to specific distance functions and introduce polynomial and pseudo-polynomial algorithms for special graph classes such as paths, trees and series-parallel graphs. This version of the CNP also

corresponds to a generalized multiple source-sink shortest path interdiction problem based on node removal and with a nonlinear objective function. Few decomposition approaches have been explored to solve the CNP through advanced Mathematical Programming methods, with the notable exceptions of [20] and [48] which use Column Generation techniques. In these studies, the set of critical nodes is assumed to have a specific structure such as a path, a clique or a star. Benders Decomposition is used in [22, 31] to respectively solve a distance-based version of the CNP and a robust version with uncertain costs. We refer the interested reader to the recent survey [28] for a detailed description of different CNP variants and of the related theoretical results and algorithms available in the literature.

Stochastic versions of several interdiction problems were also considered in the literature. A stochastic version of a Max-Flow Interdiction Problem is tackled in [14]. The problem calls for minimizing the maximum flow between a source and sink in a graph by removing some of its edges where an attack on each edge can fail with a given probability. The aim is thus to minimize the *expected value* of the maximum flow over all possible scenarios of attack failures. The authors design a sophisticated and effective method to solve the problem by partitioning the scenarios into clusters and reaching solutions with a predefined accuracy. The method relies on different but equivalent formulations of the classic maximum flow interdiction problem: one formulation includes the interdiction variables in the constraints while the other one includes them in the objective function. The two formulations provide respectively upper and lower bounds on the optimal solution when scenarios are aggregated. The BD method is used to solve the model with aggregated scenarios by decomposing over the different clusters (this approach is called the *L-shaped method* in the literature), as is usually done when applying BD to stochastic problems. The algorithm allows the authors to solve the stochastic problem without sampling over an exponential set of scenarios. However, the approach exploits the specific structure of the max flow problem and can hardly be generalized to other network problems such as the CNP. The same problem is solved in [24] using a statistical approximation called the Sample Average Approximation (SAA) and the model is decomposed by scenario using a BD approach.

The study [17] tackles a stochastic version of the CNP where the presence of the edges in a given graph is uncertain. Each edge has an independent probability to be absent from the graph. This gives 2^m different scenarios to analyze where m denotes the number of edges in the input graph. The goal is to minimize the expected pairwise connectivity after removing a subset of the nodes subject to a budget constraint. A MILP model is introduced with an exponentially large number of variables due to the number of possible scenarios. A heuristic approach is proposed by solving a reduced MILP with variables and constraints aggregated. A local search procedure is then applied to improve the computed solutions. The approach uses a Fully Polynomial Randomized Approximation Scheme (FPRAS) to estimate the objective value of each computed solution within a given precision in polynomial time. Finally, a CNP variant with nondeterministic connection costs for each pair of nodes is introduced in [19, 31] and formulated as a robust optimization problem.

The *probability chains* method has been used in several works on stochastic NIPs, such as [30, 32, 29], in order to linearize stochastic problems with a nonlinear objective function. The authors of [30] study an interdiction problem to maximize the probability of detecting nuclear smugglers, where the uncertainty concerns the origin and the destination of the smugglers. The authors of [32] study a protection facility problem where a set of facilities is subject to potential disaster events. The work of [29] also deals with a facility protection problem where different levels of protection can be assigned to each facility. The problem has *decision dependent uncertainties* and a nonconvex objective function, which is linearized through the use of probability chains. The recent work of [2] applies probability chains to linearize the objective function of a stochastic assignment problem called the Weapon-Target Assignment Problem, which can also be classified

as a type of interdiction problem.

1.1 Our contributions

The contributions of our work can be summarized as follows. At first, we introduce the stochastic version of the CNP with uncertain outcome of attacks on nodes and provide an Integer Linear Programming (ILP) model with an exponential number of variables and constraints on general graphs along with valid inequalities. We then focus on trees and prove that the problem over trees is NP-complete even in cases where its deterministic counterpart is polynomially solvable. We provide a compact nonlinear formulation of the problem over trees and reformulate the nonlinear model as a MILP model using the concept of probability chains. Given the features of the MILP model, we decompose the problem by adopting a classic Benders Decomposition approach for which we derive effective analytical approaches to solve the slave subproblems. Finally, we introduce an approximation algorithm for the problem variant over trees with unit connection costs and unit attack costs, and a specific ILP model for trees where all survival probabilities of the nodes in case of an attack are equal. We evaluate the effectiveness of our solution methods by performing computational tests on a large set of instances.

We remark that the proposed methods do not rely on any statistical approximation such as the Sample Average Approximation method [21]: this is a strong feature of our approaches since it avoids the necessary evaluation of a confidence interval on the optimal objective value, which in turn calls for solving the stochastic problem at hand several times with different scenario samples.

The paper is organized as follows. We introduce a general ILP formulation for the stochastic CNP in Section 2. In Section 3, we present models and theoretical results for the problem variant over trees. We describe the proposed solution approach based on Benders Decomposition in Section 4. We discuss the results of the computational tests in Section 5. The details of the ILP model for the case with equal survival probabilities and of the approximation algorithm for the problem variant with unit costs are reported in Appendices A and B, respectively. Section 6 provides some concluding remarks.

2 Notation and problem formulation over general graphs

In the stochastic CNP, hereafter denoted as $SCNP$, we are given an undirected graph $G = (V, E)$ with a set of nodes V and a set of edges E. Let $n := |V|$ and $m := |E|$ denote the number of nodes and edges respectively. Sticking to the common terminology adopted in previous studies on the CNP, let also c_{ij} denote the cost (or weight) of a connection between nodes i and j ($i, j \in V$). This parameter can be considered a cost as the presence of a connection in the CNP penalizes the corresponding objective function. Each node i ($i \in V$) has a set of neighbors $N_i := \{k : (i, k) \in E\}$, an attack cost κ_i, and a survival probability p_i in case of an attack. We have a budget value for the attacks on nodes denoted by K. We assume that the survival probabilities are independent. This assumption is realistic in real-life applications where the vulnerability of sites in a network does not depend on the outcomes of attacks on sites in other parts of the network. For instance, when all the attacks on nodes occur simultaneously, the defender usually has no time to redefine his/her defense strategy in terms of redistribution of the resources to protect each node. We stress that our results rely heavily on the assumption of independent survival probabilities. The problem calls for minimizing the expected value of the pairwise connectivity in the graph after attacking a subset of the nodes. The $SCNP$ is a generalization of the deterministic CNP (which is strongly NP-hard [1])

where $p_i = 0$ for all $i \in V$, i.e., an attacked node is always removed from the graph. We can derive an ILP formulation for the problem as follows. As customary in Stochastic Programs, we consider the set of possible scenarios $\mathbb{S}$ where each scenario defines which nodes would survive in case of an attack. Thus, the number of scenarios is exponentially large with n, i.e., $|\mathbb{S}| = 2^n$. The probability that a scenario $\bar{s} \in \mathbb{S}$ occurs is denoted by $\pi_{\bar{s}}$, with $\pi_{\bar{s}} = \prod_{i=1}^{n} \gamma_i$, where $\gamma_i = 1 - p_i$ if node i will not survive in the scenario, or else $\gamma_i = p_i$. We then introduce binary variables v_i $(i = 1, \ldots, n)$ equal to 1 if and only if node i is attacked and binary variables $u_{ij}^{\bar{s}}$ equal to 1 if and only if two nodes i and j are connected by a path in scenario $\bar{s}$ in the induced subgraph $G[V \setminus S]$, with $S := \{i \in V : v_i = 1\}$. We also introduce parameter $\delta_i^{\bar{s}}$, which is equal to 1 if an attack on a node $i \in V$ is unsuccessful in the scenario $\bar{s} \in \mathbb{S}$. Based on the model in [45] for the deterministic CNP, we formulate the stochastic CNP as follows:

$$\min \quad \sum_{\bar{s} \in \mathbb{S}} \pi_{\bar{s}} \sum_{i<j} c_{ij} u_{ij}^{\bar{s}} \tag{1a}$$

$$\text{s.t.} \quad \sum_{i} \kappa_i v_i \leq K \tag{1b}$$

$$v_i = 0 \qquad i \in V : p_i = 1 \tag{1c}$$

$$u_{ij}^{\bar{s}} \geq 1 - (1 - \delta_i^{\bar{s}}) v_i - (1 - \delta_j^{\bar{s}}) v_j \qquad (i,j) \in E, \bar{s} \in \mathbb{S} \tag{1d}$$

$$u_{ij}^{\bar{s}} \geq \frac{1}{|N_i|} \sum_{k \in N_i} u_{kj}^{\bar{s}} - (1 - \delta_i^{\bar{s}}) v_i \qquad (i,j) \notin E : i < j, \bar{s} \in \mathbb{S} \tag{1e}$$

$$u_{ij}^{\bar{s}} \in \{0,1\} \qquad i,j \in V : i < j, \bar{s} \in \mathbb{S} \tag{1f}$$

$$v_i \in \{0,1\} \qquad i \in V \tag{1g}$$

The objective function (1a) minimizes the expected value of the pairwise connectivity by summing over all possible scenarios. Constraint (1b) represents the budget constraint for the attacks on nodes in the graph. Constraints (1c) enforce the fact that no node with probability of survival equal to one will be ever attacked, as it is always suboptimal to consume budget for attacking a node that would survive anyway. In each scenario $\bar{s}$, where both nodes i and j of an edge $(i,j) \in E$ can be successfully removed by an attack $(\delta_i^{\bar{s}} = \delta_j^{\bar{s}} = 0)$, constraints (1d) ensure that $u_{ij}^{\bar{s}} = 1$ only if both nodes i and j are not attacked $(v_i = v_j = 0)$. Note that when $\delta_i^{\bar{s}}$ (or $\delta_j^{\bar{s}}$) is equal to one, node i (or j) is functional in the graph even after it is attacked. Similarly, constraints (1e) guarantee that, for two nodes i and j not connected by an edge, we have $u_{ij}^{\bar{s}} = 1$ if, on the one hand, there is at least one path between j and a neighbor of i, and, on the other hand, node i remains functional in the graph (either $v_i = 0$ and $\delta_i^{\bar{s}} = 0$, or $\delta_i^{\bar{s}} = 1$). Constraints (1f) and (1g) define the domain of definition of the variables. Notice that the ILP formulation for the deterministic CNP where $p_i = 0$ $(i = 1, \ldots, n)$ is equivalent to model (1a)-(1g) with one scenario $\bar{s}$ with $\pi_{\bar{s}} = 1$ and $\delta_i^{\bar{s}} = 0$ for $i = 1, \ldots, n$. Notice also that model (1) can be applied to any graph.

However, we remark that the model has a number of variables and constraints which is exponential in the number of nodes n and thus it is intractable to solve even for graphs of very limited size. In small graphs with $n = 20$, the number of scenarios to handle ($2^{20} \approx 10^6$) would be already computationally prohibitive with model (1). Our goal is to overcome the limits of the generic model (1) when the graphs considered are trees by offering effective exact solution approaches to deal with larger graphs and without considering any statistical approximation such as a sampling of the scenarios.

2.1 Valid inequalities

For our algorithmic developments over trees, we generalize a result for the deterministic version of the CNP where both connection costs and deletion weights are one. Several works [38, 45, 46] point out that there always exists an optimal solution for such a CNP variant where no node $i \in V$ with only one neighbour ($|N_i|= 1$, i.e., a *leaf node*) is selected. If a solution selects a leaf node, another solution which is never worse can be obtained by replacing the leaf node with its neighbour.

We generalize this result to the SCNP with arbitrary connection costs $c_{ij} \geq 0$ and arbitrary attack costs $\kappa_i > 0$. Let D_1 denote the set of leaf nodes, i.e., $D_1 = \{i \in V : |N_i|= 1\}$. The following proposition holds.

Proposition 1: For any node pair $i \in D_1$ and $j \in N_i$, with $j \notin D_1$, if $p_j \leq p_i$ and $\kappa_j \leq \kappa_i$, then $v_i \leq v_j$ in at least one optimal solution of model (1).

Proof. We consider the value of v_i and v_j in an optimal solution. If both nodes i and j are attacked or are not attacked, we have $v_i = v_j$ and the claim holds. We then compare the two remaining cases with $v_i \neq v_j$ where only one node between i and j is attacked.

Consider first the case where only node i is attacked, i.e., $v_i = 1$ and $v_j = 0$. Let $\langle u^{(1)}_{kl} \rangle$ denote the average probability that a given pair of nodes k and l is connected by a path in this configuration, namely $\langle u^{(1)}_{kl} \rangle = \sum_{\bar{s}\in\mathbb{S}} \pi_s u^{\bar{s}}_{kl}$. The part of the objective function involving nodes i and j is given by $\sum_{k\in V:k\neq i,j} c_{kj}\langle u^{(1)}_{kj}\rangle + \sum_{k\in V:k\neq i} c_{ki}\langle u^{(1)}_{ki}\rangle$. Since j is the only neighbor of i, any path between i and another node k has to go through j and we have $\langle u^{(1)}_{ki}\rangle = p_i\langle u^{(1)}_{kj}\rangle$ for any $k \neq i, j$. Using the fact that $\langle u^{(1)}_{ij}\rangle = p_i$, we have:

$$\sum_{k\in V:k\neq i,j} c_{kj}\langle u^{(1)}_{kj}\rangle + \sum_{k\in V:k\neq i} c_{ki}\langle u^{(1)}_{ki}\rangle = p_i c_{ij} + \sum_{k\in V:k\neq i,j} (c_{kj} + p_i c_{ki})\langle u^{(1)}_{kj}\rangle. \tag{2}$$

Consider now the case where only node j is attacked, $v_i = 0$ and $v_j = 1$, and let $\langle u^{(2)}_{kl}\rangle$ denote the average probability that two nodes k and l are connected by a path. We have $\langle u^{(2)}_{ki}\rangle = \langle u^{(2)}_{kj}\rangle$ for $k \neq i, j$ and $\langle u^{(2)}_{ij}\rangle = p_j$. By also considering that $\langle u^{(2)}_{kj}\rangle = p_j\langle u^{(1)}_{kj}\rangle$ for $k \neq i, j$, the contribution in the objective function associated with nodes i and j is:

$$\sum_{k\in V:k\neq i,j} c_{kj}\langle u^{(2)}_{kj}\rangle + \sum_{k\in V:k\neq i} c_{ki}\langle u^{(2)}_{ki}\rangle = p_j c_{ij} + \sum_{k\in V:k\neq i,j} (p_j c_{kj} + p_j c_{ki})\langle u^{(1)}_{kj}\rangle. \tag{3}$$

Notice that the values of all the other terms associated with the remaining nodes in the objective function do not change when only the values of v_i and v_j change. Thus, the difference in the objective value provided by the two solutions corresponds to the difference between (2) and (3)

$$(p_i - p_j)c_{ij} + \sum_{k\in V:k\neq i,j} ((1 - p_j)c_{kj} + (p_i - p_j)c_{ki})\langle u^{(1)}_{kj}\rangle \tag{4}$$

which is nonnegative when $p_j \leq p_i$ as terms c_{ij}, c_{kj}, $\langle u^{(1)}_{kj}\rangle$ are nonnegative. Hence, since $\kappa_j \leq \kappa_i$, a solution that sets $v_i = 1$ and $v_j = 0$ can be improved by attacking node j instead of i, i.e., $v_i = 0$ and $v_j = 1$, which implies $v_i \leq v_j$. □

Although the rest of this work is devoted to the study of trees, we stress that the previous inequalities are valid for general graphs.

3 The Stochastic Critical Node Problem over Trees

We now turn our attention to the stochastic CNP over trees where the outcome of a node attack is uncertain. We denote this problem as $SCNP_{tree}$. In this section, we first prove the NP-completeness of the problem even when connection costs are one, while its deterministic counterpart was proved to be polynomial. We then present a nonlinear reformulation of the problem from which we derive a Mixed Integer Linear Programming (MILP) model and an exact approach in Section 4. A significant advantage of the proposed MILP formulation with respect to model (1) is that it has a polynomial number of variables and constraints. This allows us to solve to optimality instances of reasonable size (see Section 5) while, as discussed in Section 2, model (1) can hardly tackle even small instances with 20 nodes without resorting to some kind of statistical approximation. We conclude the section by providing a theoretical approximation result for a problem variant with unit connection costs and unit attack costs and some remarks about other stochastic problems over trees.

3.1 NP-completeness with unit connection costs

We denote as $D\text{-}SCNP_{tree}$ the decision version of the problem asking whether there exists a solution of the $SCNP_{tree}$ with a value not superior to a target value Γ. The expected cost of the connection between two nodes i and j can be computed in $O(n)$ as it only depends on the products between c_{ij} and the survival probabilities of the attacked nodes in their path. By considering all $O(n^2)$ paths in the tree, the overall objective value of each given solution can be computed in polynomial time $O(n^3)$, implying that the $D\text{-}SCNP_{tree}$ is an NP problem. Here we prove the NP-completeness of the $D\text{-}SCNP_{tree}$ even with unit connection costs, i.e., $c_{ij} = 1$ for all $i, j \in V$, while the deterministic pairwise CNP variant with unit connection costs was proved to be polynomial over trees in [15].

Proposition 2: The $D\text{-}SCNP_{tree}$ with unit connection costs is NP-complete.

Proof. We prove the theorem by a reduction from the Knapsack Problem (KP), a well-known optimization problem where a capacity value C and a set of n items with profits $\alpha_i > 0$ and weights $w_i > 0$ are given. The goal is to select a subset of the items to maximize the profits while ensuring that the weight of the selected items does not exceed C. The decision version of KP, denoted by $D\text{-}KP$, is NP-complete and asks whether there exists a feasible solution with a profit not inferior to a value $A > 0$. We can map each $D\text{-}KP$ instance into a $D\text{-}SCNP_{tree}$ instance as follows. We consider a root node with survival probability $p_1 = 0$ and deletion cost $\kappa_1 = 1$. At the next level of the tree, we introduce n intermediate nodes $2, \ldots, (n+1)$ with survival probabilities $p_{i+1} = 1$ and deletion costs $\kappa_{i+1} = 1$ for $i = 1, \ldots, n$. Then, to each of the intermediate nodes, we attach a leaf node which corresponds to an item in the knapsack instance. Each leaf node $(i+n+1)$, for $i = 1, \ldots, n$, has a deletion cost $\kappa_{i+n+1} = w_i$ and a survival probability $p_{i+n+1} = 1 - \alpha_i/\alpha_{max}$, with $\alpha_{max} = \max_i\{\alpha_i\}$. Finally, we set $K = C + 1$, $\Gamma = n - \frac{A}{\alpha_{max}}$ and $c_{ij} = 1$ for each pair of nodes i and j. Such a reduction is polynomial in n.

To prove the NP-completeness of the $D\text{-}SCNP_{tree}$, we have to show that a $D\text{-}KP$ instance is a Yes instance, i.e., it has a feasible solution with a profit not inferior to A, if and only if the related $D\text{-}SCNP_{tree}$ instance is a Yes instance, i.e., the instance admits a solution with a value not superior to Γ. Notice that in the considered settings any solution of a Yes instance of the $D\text{-}SCNP_{tree}$ must attack (and successfully delete)

the root node. Else, a solution value of at least $\frac{(n+1)((n+1)-1)}{2} \geq n > \Gamma$ would be induced by the connections involving the root node and the intermediate nodes, which cannot be removed after an attack as their survival probability is one. Since attacks on intermediate nodes only induce budget consumptions without affecting the objective value and hence are suboptimal, without loss of generality we consider solutions of the $D\text{-}SCNP_{tree}$ where the intermediate nodes are not attacked. Let us consider a Yes instance of $D\text{-}SCNP_{tree}$ and a solution that attacks the root node and a subset of the leaves. If we denote by I the index set of the attacked leaf nodes $(i+n+1)$ with $i \in I$, we get a solution with weight $1 + \sum_{i \in I} \kappa_{i+n+1} \leq K$ which implies $\sum_{i \in I} w_i \leq C$. The corresponding objective value considers only the costs of the connections between intermediate nodes and their leaves, as the probability of connection between every pair of nodes passing through the root node is zero. Thus we have

$$n - |I| + \sum_{i \in I} p_{i+n+1} = n - |I| + \sum_{i \in I} 1 - \frac{\sum_{i \in I} \alpha_i}{\alpha_{max}} = n - |I| + |I| - \frac{\sum_{i \in I} \alpha_i}{\alpha_{max}} = n - \frac{\sum_{i \in I} \alpha_i}{\alpha_{max}} \leq \Gamma$$

which implies $\sum_{i \in I} \alpha_i \geq A$. Hence, the solution of the $D\text{-}SCNP_{tree}$ instance provides also a solution of the corresponding $D\text{-}KP$ instance.

Likewise, consider a Yes instance of $D\text{-}KP$ and a solution with item set I', namely with $\sum_{i \in I'} w_i \leq C$ and $\sum_{i \in I'} \alpha_i \geq A$. We derive a solution of the $D\text{-}SCNP_{tree}$ instance by attacking the root node and the leaves $(i+n+1)$ with $i \in I'$. The corresponding weight and profit entries are $1 + \sum_{i \in I'} \kappa_{i+n+1} \leq K$ and $n - \frac{\sum_{i \in I'} \alpha_i}{\alpha_{max}} \leq n - \frac{A}{\alpha_{max}} = \Gamma$. □

The inherent difficulty of solving the stochastic CNP over trees even with unit connection costs also motivates the development of the exact algorithm introduced in Section 4.

3.2 Nonlinear reformulation

Two given nodes in a tree are connected by a unique path of nodes. Hence, the expected cost of the connection between two nodes i and j only depends on the products between c_{ij} and the survival probabilities of the attacked nodes in their path. Let $\mathcal{P}_{ij}$ denote the set of nodes, including nodes i and j, in the path between i and j. Let S_{ij} denote the set of nodes which are attacked in $\mathcal{P}_{ij}$, i.e., $S_{ij} = S \cap \mathcal{P}_{ij}$. The expected cost of the connection between i and j is equal either to c_{ij} if S_{ij} is empty or to the product $c_{ij} \prod_{k \in S_{ij}} p_k$, as we assume that the survival probabilities are independent. According to this consideration, we can state a nonlinear model for $SCNP_{tree}$ with a polynomial number of variables and constraints. By keeping the notation of model (1), we consider only binary variables v_i associated with the attack of a node i and introduce the following nonlinear reformulation:

$$\min \quad \sum_{i<j} c_{ij} \prod_{k \in \mathcal{P}_{ij}} (1 - (1 - p_k) v_k) \tag{5a}$$

$$\text{s.t.} \quad \sum_i \kappa_i v_i \leq K \tag{5b}$$

$$v_i = 0 \qquad i \in V : p_i = 1 \tag{5c}$$

$$v_i \leq v_j \qquad i \in D_1,\ j \in N_i,\ j \notin D_1,\ p_j \leq p_i,\ \kappa_j \leq \kappa_i \tag{5d}$$

$$v_i \in \{0, 1\} \qquad i \in V. \tag{5e}$$

The objective function (5a) represents the same sum over the connection costs as in objective (1a). However, we have already performed the sum over the exponential set of scenarios $\mathbb{S}$. In fact, the probability of survival of the connection between two nodes i and j corresponds to the probability of survival of their path $\mathcal{P}_{ij}$. Such a probability can be computed by multiplying the probabilities of survival of the attacked nodes. In particular, for each node $k \in \mathcal{P}_{ij}$ we have either 1 in the product $\prod_{k\in\mathcal{P}_{ij}}(1-(1-p_k)v_k)$ in (5a) if the node is not attacked ($v_k = 0$) or p_k if the node is attacked ($v_k = 1$). Therefore, the product $\prod_{k\in\mathcal{P}_{ij}}(1-(1-p_k)v_k)$ represents the product $\prod_{k\in S_{ij}} p_k$. Constraints (5b) and (5c) are the same constraints as (1b) and (1c), respectively. Finally, constraints (5d) represent the valid inequalities introduced in Section 2.1. As discussed in detail in Section 4, we propose a linear reformulation of model (5), that allows us to develop an effective exact approach based on Benders Decomposition. We also derive a linear formulation for the specific problem variant where all the survival probabilities are equal to a value p, i.e., $p_i = p$ for $i \in V$. Such a case could occur when all the nodes of a network are defended with a similar level of protection. Even though this might not be a likely situation in many practical applications, the proposed model might be of interest as it can handle instances of reasonable size with a limited computational effort (see Section 5). The model might also constitute a starting point for further developments in cases where the set of possible values of the survival probabilities is very limited. We present the corresponding ILP model in Appendix A and compare its effectiveness with the performance of the more general Benders Decomposition approach in our numerical experiments in Section 5.

3.3 An approximation result for the $SCNP_{tree}$ with unit costs

For $SCNP_{tree}$ instances where both the connection costs and attack costs are one, we derive an approximation result that is summarized in the following proposition.

> **Proposition 3:** For the $SCNP_{tree}$ with unit connection costs and unit attack costs, there exists an approximation algorithm with pseudo-polynomial time complexity and an absolute approximation bound of $\frac{n(n-1)}{2\mu}$.

We refer to Appendix B for the details of the algorithm.

3.4 Considerations on other stochastic interdiction problems over trees

We conclude this section with some remarks about other stochastic interdiction problems over trees. We first show that the considered CNP variant over trees actually generalizes to the CEDP over trees where attacks on edges as well as on nodes succeed only with a given probability. The following proposition holds.

> **Proposition 4:** The stochastic CEDP over trees with edge and node removal reduces to the $SCNP_{tree}$.

Proof. A stochastic CEDP instance has the same inputs of a $SCNP_{tree}$ instance plus a survival probability and an attack cost for each edge $(i, j) \in E$, that we denote by p'_{ij} and κ'_{ij}, respectively. To obtain a $SCNP_{tree}$ instance, it is sufficient to associate each edge $(i, j) \in E$ with a new node e_{ij} emanating two edges towards

nodes i and j respectively, giving a new tree with $2n-1$ nodes. For each new node e_{ij}, setting $p_{e_{ij}} = p'_{ij}$ and $\kappa_{e_{ij}} = \kappa'_{ij}$ completes the reduction. □

An alternative stochastic version of the problem also comes to mind where each edge is present in the graph with an independent probability ϕ_{ij}, as proposed in [17]. The problem calls for the minimization of the expected value of the pairwise connectivity over all the possible realizations of the graph. However, the following proposition shows that this CNP variant might be of limited interest over trees as it can be reduced to the deterministic CNP and solved with existing methods available in the literature.

Proposition 5: The stochastic CNP over trees with uncertainty on the existence of edges reduces to the deterministic CNP over trees.

Proof. In a tree, the probability that a connection exists between two nodes i and j is equal to $\prod_{(k,l)\in\mathcal{E}_{ij}} \phi_{kl}$ where $\mathcal{E}_{ij}$ denotes the set of edges in the unique path between i and j. Thus, it is sufficient to redefine the connection costs c_{ij} as $c'_{ij} = c_{ij} \prod_{(k,l)\in\mathcal{E}_{ij}} \phi_{kl}$ so as to obtain an instance of the deterministic CNP. □

4 An exact solution approach for the $SCNP_{tree}$

We propose an exact solution approach based on a linearization of model (5) and on Benders Decomposition. We first introduce a MILP reformulation of model (5) and then present the proposed approach.

4.1 MILP reformulation of model (5)

We can linearize the objective function of model (5) by introducing additional variables and constraints to compute the probability that a given connection exists through a recursive analysis of the corresponding attacked nodes. More precisely, we employ the concept of probability chains (see, e.g., [32]). We compute the probability of survival of each connection i-j as follows. Let us introduce continuous variables s_k^{ij}, with $0 \le s_k^{ij} \le 1$, representing the probability that a connection i-j still exists when possible attacks on the first k nodes in $\mathcal{P}_{ij}$ are considered. We also introduce continuous variables r_k^{ij}, with $0 \le r_k^{ij} \le 1$, that represent the probability of disconnecting nodes i and j by considering an attack on the $k-th$ node in the path, provided that the connection survived possible attacks on the previous nodes. From now on, we will consider the nodes in a path i-j indexed in increasing order. Correspondingly, we indicate by $v_{[1]}, v_{[2]}, \ldots, v_{[|\mathcal{P}_{ij}|]}$ the binary variables associated with attacks on the nodes in the original tree and by $p_{[1]}, p_{[2]}, \ldots, p_{[|\mathcal{P}_{ij}|]}$ the related survival probabilities. For the first node in $\mathcal{P}_{ij}$ we have:

$$r_1^{ij} = (1 - p_{[1]})v_{[1]}; \tag{6}$$

$$s_1^{ij} = 1 - r_1^{ij}. \tag{7}$$

The variable r_1^{ij} is equal to $(1-p_{[1]})v_{[1]}$ as this product represents the probability of disconnecting i and j if the first node in the path is attacked. Correspondingly, the probability s_1^{ij} that the connection still exists is computed as the complementary probability $1 - r_1^{ij}$. Moving to the second node in the path, by definition of variables r, we have:

$$r_2^{ij} = (1 - p_{[2]})v_{[2]}s_1^{ij}. \tag{8}$$

The variable r_2^{ij} represents the probability of disconnecting i and j by considering a possible attack on the second node in the path, given by the product $(1-p_{[2]})v_{[2]}$, subject to the fact that the connection survived a possible attack on the first node, which is represented by variable s_1^{ij}. The probability of survival s_2^{ij} is then equal to 1 minus the probability that at least one previous attack is successful. Such a probability is equal to $r_2^{ij} + (1 - s_1^{ij})$, namely nodes i and j can be disconnected by an attack on the second node in the path with probability r_2^{ij} or by an attack on the previous nodes (in this case only on the first node) with probability $1 - s_1^{ij}$. Notice that we can just sum up the probabilities of the two events as they are independent according to the definition of r_2^{ij}. Therefore, we have

$$s_2^{ij} = 1 - ((1 - s_1^{ij}) + r_2^{ij}) = s_1^{ij} - r_2^{ij}. \tag{9}$$

Since variables v are binary and variables r and s are between 0 and 1, the nonlinear equation (8) can be linearized by the following set of constraints:

$$r_2^{ij} \leq (1 - p_{[2]})v_{[2]}, \tag{10}$$
$$r_2^{ij} \leq (1 - p_{[2]})s_1^{ij}, \tag{11}$$
$$r_2^{ij} \geq (1 - p_{[2]})(s_1^{ij} + v_{[2]} - 1). \tag{12}$$

By using equations (8), (9) and applying the same recursive argument for all the nodes in the path, we compute the overall probability $s^{ij}_{|\mathcal{P}_{ij}|}$ that connection i-j exists, i.e., $s^{ij}_{|\mathcal{P}_{ij}|} = \prod_{k\in\mathcal{P}_{ij}}(1-(1-p_k)v_k)$. Hence, we linearize the objective function of model (5) by replacing the nonlinear terms in the objective with $s^{ij}_{|\mathcal{P}_{ij}|}$ and by adding the corresponding constraints and variables for each path i-j. We obtain the following MILP model:

$$\min \quad \sum_{i<j} c_{ij} s^{ij}_{|\mathcal{P}_{ij}|} \tag{13a}$$
$$\text{s.t.} \quad \sum_i \kappa_i v_i \leq K \tag{13b}$$
$$v_i = 0 \qquad i \in V : p_i = 1 \tag{13c}$$
$$v_i \leq v_j \qquad i \in D_1,\ j \in N_i,\ j \notin D_1,\ p_j \leq p_i,\ \kappa_j \leq \kappa_i \tag{13d}$$
$$r_1^{ij} = (1 - p_{[1]})v_{[1]} \qquad i, j \in V : i < j \tag{13e}$$
$$s_1^{ij} = 1 - r_1^{ij} \qquad i, j \in V : i < j \tag{13f}$$
$$r_k^{ij} \leq (1 - p_{[k]})v_{[k]} \qquad k = 2, \ldots, |\mathcal{P}_{ij}|,\ i, j \in V : i < j \tag{13g}$$
$$r_k^{ij} \leq (1 - p_{[k]})s^{ij}_{(k-1)} \qquad k = 2, \ldots, |\mathcal{P}_{ij}|,\ i, j \in V : i < j \tag{13h}$$
$$r_k^{ij} \geq (1 - p_{[k]})(s^{ij}_{(k-1)} + v_{[k]} - 1) \qquad k = 2, \ldots, |\mathcal{P}_{ij}|,\ i, j \in V : i < j \tag{13i}$$
$$s_k^{ij} = s^{ij}_{(k-1)} - r_k^{ij} \qquad k = 2, \ldots, |\mathcal{P}_{ij}|,\ i, j \in V : i < j \tag{13j}$$
$$r_k^{ij} \geq 0, s_k^{ij} \geq 0 \qquad k = 1, \ldots, |\mathcal{P}_{ij}|,\ i, j \in V : i < j \tag{13k}$$
$$v_i \in \{0, 1\} \qquad i \in V \tag{13l}$$

Notice that, since $0 \leq p_i \leq 1$ $(i = 1, \ldots, n)$, we can avoid adding bounds $r_k^{ij} \leq 1$, $s_k^{ij} \leq 1$ for $k = 1, \ldots, |\mathcal{P}_{ij}|$. Also, notice that for each subpath i-j' of path i-j $(j' < j)$, the related constraints are already taken

into account in the constraints for path i-j. Thus, for these subpaths, we just have to replace the term $\prod_{k\in\mathcal{P}_{ij'}}(1-(1-p_k)v_k)$ with the variable $s^{ij'}_{|\mathcal{P}_{ij'}|}$ without adding additional constraints.

Furthermore, we can show that any optimal solution of model (13) sets the value of variables r^{ij}_k equal to the minimum value of the two upper bounds given by constraints (13g) and (13h), for each path i-j. An immediate consequence of this result is that constraints (13i) can be removed from the model without loss of generality.

Proposition 6: In any optimal solution of model (13), $r^{ij}_k = (1-p_{[k]})\min\{v_{[k]},\ s^{ij}_{(k-1)}\}$ for $k = 2,\ldots,|\mathcal{P}_{ij}|$.

Proof. For a given path i-j, the minimization of variable $s^{ij}_{|\mathcal{P}_{ij}|}$ in the objective function (as $c_{ij}>0$) implies that, according to related equality constraint (13j), variable $r^{ij}_{|\mathcal{P}_{ij}|}$ will be set to the largest possible value. Hence we have $r^{ij}_{|\mathcal{P}_{ij}|} = (1-p_{[|\mathcal{P}_{ij}|]})\min\{v_{[|\mathcal{P}_{ij}|]},\ s^{ij}_{(|\mathcal{P}_{ij}|-1)}\}$ from the corresponding constraints (13g) and (13h). Due to constraint (13j) for $k=|\mathcal{P}_{ij}|$, we can show that the minimization of $s^{ij}_{|\mathcal{P}_{ij}|}$ also implies that $s^{ij}_{(|\mathcal{P}_{ij}|-1)}$ will be set to the lowest possible value (in accordance with the minimization of the cost of the related path of length $|\mathcal{P}_{ij}|-1$): if $v_{[|\mathcal{P}_{ij}|]}=0$ in an optimal solution, then $r^{ij}_{|\mathcal{P}_{ij}|}=0$ and thus $s^{ij}_{|\mathcal{P}_{ij}|}=s^{ij}_{(|\mathcal{P}_{ij}|-1)}$; if $v_{[|\mathcal{P}_{ij}|]}=1$, then $r^{ij}_{|\mathcal{P}_{ij}|}=(1-p_{[|\mathcal{P}_{ij}|]})s^{ij}_{(|\mathcal{P}_{ij}|-1)}$ and thus $s^{ij}_{|\mathcal{P}_{ij}|}=p_{[|\mathcal{P}_{ij}|]}s^{ij}_{(|\mathcal{P}_{ij}|-1)}$.

In turn, the minimization of $s^{ij}_{(|\mathcal{P}_{ij}|-1)}$ implies the maximization of $r^{ij}_{(|\mathcal{P}_{ij}|-1)}$, thus giving $r^{ij}_{(|\mathcal{P}_{ij}|-1)} = (1-p_{[|\mathcal{P}_{ij}|-1]})\min\{v_{[|\mathcal{P}_{ij}|-1]},\ s^{ij}_{(|\mathcal{P}_{ij}|-2)}\}$, as well as the minimization of $s^{ij}_{(|\mathcal{P}_{ij}|-2)}$. By considering $s^{ij}_{(|\mathcal{P}_{ij}|-2)}$ and recursively applying the same argument, we have $r^{ij}_k = (1-p_{[k]})\min\{v_{[k]},\ s^{ij}_{(k-1)}\}$ for the remaining k values $(|\mathcal{P}_{ij}|-2),\ldots,2$. Extending the same analysis to all paths completes the proof. □

Corollary 1: Constraints (13i) are redundant and can be removed from model (13).

4.2 A Benders Decomposition approach

The classic version of the Benders Decomposition [9] relies on the construction of a Master Problem (MP) involving all the binary/integer variables of a MILP model, denoted as "complicating" variables, and on the construction of a continuous linear Slave Problem (SP) induced by fixing the variables of the MP. The Master Problem is iteratively solved providing nondecreasing lower bounds (for a minimization problem) on the optimal objective value of the original model. In each iteration, an optimal solution of the MP induces a SP that can provide either a "feasibility" cut for the MP if SP is infeasible or else an optimality cut and a feasible solution (an upper bound) for the original problem. This procedure is repeated until an optimal solution is certified by the lower bounds provided by the MP. We refer the interested reader to the recent survey [35] and the references therein for a discussion about applications and algorithmic variants of the BD. We also mention the recent works [22, 31] for BD applications to variants of the CNP.

In the MILP formulation derived for $SCNP_{tree}$, any feasible assignment of values to binary variables v_i induces a linear programming model related to all paths in the tree and with variables r^{ij} and s^{ij} only. The

optimal solution of such an LP model coincides with the value of a $SCNP_{tree}$ feasible solution given by attacking nodes i with $v_i = 1$, i.e., it is equal to the sum of the average costs of the connections. A crucial observation here is that an optimal solution of the induced LP model can be determined by separately considering each subproblem associated with a path between two nodes i and j. All the subproblems for pairs (i, j) are in fact independent once the variables v_i are set. These structural aspects motivate us to apply the classic Benders Decomposition method where binary variables v_i are included in the Master Problem and all the continuous variables are projected out in the slave subproblems.

We rely on a MILP solver to compute an optimal solution of the Master Problem which is passed as input to the slave subproblems at each iteration. As a further option, we also investigate the use of the MILP solver provided with additional cuts when solving the Master Problem. Nowadays modern MILP solvers allow the insertion of problem-specific cuts (user cuts) in their branch and cut tree to violate the LP relaxation solutions computed in each node. Typically, violated cuts are iteratively added and the optimal LP solution of a node is recomputed until no violated cut can be identified. In our settings, we might derive specific cuts for the solution of each Master Problem by applying the Benders recipe to the LP solutions (with fractional v_i), which are computed by the MILP solver during the exploration of the branch and cut tree. We manage to analytically derive both these cuts and the classic optimality Benders cuts by means of an effective procedure that avoids the use of an LP solver for handling the slave subproblems. We describe the details of the proposed Benders Decomposition approach in the next subsections and then provide the corresponding pseudo-code.

4.2.1 Master Problem

For the $SCNP_{tree}$, we construct the following Master Problem. Since for any given assignment of variables v_i, we can separately analyze the paths in the tree, for each path i-j we introduce a nonnegative variable z_{ij} that relates to the cost of connection i-j through a set of constraints $\Phi_{ij}(z_{ij}, v)$. As explained next, each set $\Phi_{ij}(z_{ij}, v)$ is iteratively filled with optimality cuts induced by the slave subproblem SP_{ij} associated with path i-j. We obtain the following formulation:

MP:

$$\min \quad \sum_{i<j} z_{ij} \tag{14a}$$

$$\text{s.t.} \quad \sum_{i} \kappa_i v_i \leq K \tag{14b}$$

$$v_i = 0 \qquad i \in V : p_i = 1 \tag{14c}$$

$$v_i \leq v_j \qquad i \in D_1,\ j \in N_i,\ j \notin D_1,\ p_j \leq p_i,\ \kappa_j \leq \kappa_i \tag{14d}$$

$$\Phi_{ij}(z_{ij}, v) \qquad i, j \in V : i < j \tag{14e}$$

$$v_i \in \{0, 1\} \qquad i \in V \tag{14f}$$

$$z_{ij} \geq 0 \qquad i, j \in V : i < j \tag{14g}$$

At each iteration, an optimal solution of the MP provides a lower bound on the $SCNP_{tree}$ optimal value and an assignment of the v_i variables to the slave subproblems. The values of variables v_i also constitute a feasible solution for the $SCNP_{tree}$.

4.2.2 Slave Problems

Each Slave Problem $SP_{ij}(\hat{v})$ associated with path i-j receives as input a vector $\hat{v}$ that represents the assignment of values to variables v_i provided by the MP in each iteration. A problem $SP_{ij}(\hat{v})$ has term $c_{ij}s^{ij}_{|\mathcal{P}_{ij}|}$ in the objective function and constraints (13e)-(13h) and (13j) (constraints (13i) are not considered due to Corollary 1), where the terms involving variables v_i become constant and are placed in the right-hand sides. Even though $r^{ij}_k - (1-p_{[k]})\hat{v}_{[k]}s^{ij}_{k-1} = 0$ (for $k = 2, \ldots, |\mathcal{P}_{ij}|$) would be a linear constraint in the Slave Problem, we still need to implement it to the Slave Problem through its linearized constraints. Otherwise, the dual space would depend on a solution to the Master Problem which may violate the validity of a Benders optimality cut. We obtain the following LP formulation:

$SP_{ij}(\hat{v})$:

$$\min \quad c_{ij}s^{ij}_{|\mathcal{P}_{ij}|} \tag{15a}$$

$$\text{s.t.} \quad r^{ij}_1 = (1-p_{[1]})\hat{v}_{[1]} \qquad (\lambda^{ij}_{11}) \tag{15b}$$

$$s^{ij}_1 + r^{ij}_1 = 1 \qquad (\lambda^{ij}_{12}) \tag{15c}$$

$$r^{ij}_k \leq (1-p_{[k]})\hat{v}_{[k]} \qquad k = 2, \ldots, |\mathcal{P}_{ij}| \quad (\lambda^{ij}_{k1}) \tag{15d}$$

$$r^{ij}_k - (1-p_{[k]})s^{ij}_{k-1} \leq 0 \qquad k = 2, \ldots, |\mathcal{P}_{ij}| \quad (\lambda^{ij}_{k3}) \tag{15e}$$

$$s^{ij}_k - s^{ij}_{k-1} + r^{ij}_k = 0 \qquad k = 2, \ldots, |\mathcal{P}_{ij}| \quad (\lambda^{ij}_{k2}) \tag{15f}$$

$$r^{ij}_k \geq 0,\ s^{ij}_k \geq 0 \qquad k = 1, \ldots, |\mathcal{P}_{ij}| \tag{15g}$$

In problem $SP_{ij}(\hat{v})$ we associate dual variables $\lambda^{ij}_{k\ell}$ with constraints (15b)-(15f), with index $\ell = 1, 2$ for $k = 1$ and $\ell = 1, 2, 3$ for $k > 1$. Notice that we assign the index value $l = 2$ to constraints (15f) as they correspond to constraint (15c) for $k > 1$ and in order to write the dual problem in a more compact form. The corresponding dual formulation of the subproblem $SP_{ij}(\hat{v})$, denoted as $SP^{dual}_{ij}(\hat{v})$, is as follows:

$SP^{dual}_{ij}(\hat{v})$:

$$\max \quad (1-p_{[1]})\hat{v}_{[1]}\lambda^{ij}_{11} + \lambda^{ij}_{12} + \sum_{k=2}^{|\mathcal{P}_{ij}|}((1-p_{[k]})\hat{v}_{[k]}\lambda^{ij}_{k1}) \tag{16a}$$

$$\text{s.t.} \quad \lambda^{ij}_{11} + \lambda^{ij}_{12} \leq 0 \tag{16b}$$

$$\lambda^{ij}_{k1} + \lambda^{ij}_{k3} + \lambda^{ij}_{k2} \leq 0 \qquad k = 2, \ldots, |\mathcal{P}_{ij}| \tag{16c}$$

$$\lambda^{ij}_{k2} - (1-p_{[k+1]})\lambda^{ij}_{(k+1)3} - \lambda^{ij}_{(k+1)2} \leq 0 \qquad k = 1, \ldots, |\mathcal{P}_{ij}|-1 \tag{16d}$$

$$\lambda^{ij}_{|\mathcal{P}_{ij}|2} \leq c_{ij} \tag{16e}$$

$$\lambda^{ij}_{11} \in \mathbb{R}, \lambda^{ij}_{k2} \in \mathbb{R} :\ k = 1, \ldots, |\mathcal{P}_{ij}| \tag{16f}$$

$$\lambda^{ij}_{k1} \leq 0, \lambda^{ij}_{k3} \leq 0 :\ k = 2, \ldots, |\mathcal{P}_{ij}| \tag{16g}$$

Dual constraint (16b) is associated with variable r^{ij}_1, constraints (16c) are associated with variables r^{ij}_k ($k = 2, \ldots, |\mathcal{P}_{ij}|$), constraints (16d) with variables s^{ij}_k ($k = 1, \ldots, |\mathcal{P}_{ij}|-1$) and constraint (16e) with variable $s^{ij}_{|\mathcal{P}_{ij}|}$, respectively.

4.2.3 Benders cut separation

We remark that each $SP_{ij}(\hat{v})$ is a feasible problem for a given solution provided by the MP. Hence no feasibility cut is derived. Let us now consider the current integer optimal solution $(\hat{v}, \hat{z})$ of the MP. Denote by $\hat{\lambda}^{ij}$ the value of variable λ^{ij} in an optimal solution of the dual problem $SP_{ij}^{dual}(\hat{v})$. If the corresponding objective value in (16a) ($= c_{ij} s^{ij}_{|\mathcal{P}_{ij}|}$ by strong duality) is superior to the value of variable $\hat{z}_{ij}$, we add the following classic Benders optimality cut to set $\Phi_{ij}(z_{ij}, v)$ in the MP:

$$z_{ij} \geq (1 - p_{[1]}) v_{[1]} \hat{\lambda}^{ij}_{11} + \hat{\lambda}^{ij}_{12} + \sum_{k=2}^{|\mathcal{P}_{ij}|} ((1 - p_{[k]}) v_{[k]} \hat{\lambda}^{ij}_{k1}) \tag{17}$$

as the current optimal solution of the MP violates this constraint. We consider all the slave subproblems at each iteration and possibly add a cut for each subproblem. This strategy is often indicated in the literature as a multi-cut reformulation (see, e.g., [35]). Then the MP is solved again and the process repeats until the lower bound converges to the value of the best $SCNP_{tree}$ solution found.

The same reasoning for the generation of cuts still applies when the LP relaxation of the Master Problem is solved at each iteration, namely when the integrality constraints on variables v_i are replaced by the inclusion in $[0, 1]$. Therefore, as mentioned in Section 4.2, we might even fill a branch and cut MILP solver launched on the Master Problem with cuts (17) violating the LP solutions computed at each node of the branch and cut tree. We tested this modern implementation of Benders Decomposition in our numerical experiments. Unfortunately, in our instances, adding cuts in this manner turns out to be less efficient than solving the MP with only the classic optimality cuts added at each iteration (and so, for the sake of exposition, we decided not to report the corresponding results in Section 5).

The derivation of cuts (17) can be handled by calling an LP solver to compute an optimal solution of the dual slave problems. On the other hand, the multi-cut nature of our approach might cause a large number of calls to an LP solver. For a more effective handling of the cut generation process, we derive an analytical procedure to determine an optimal solution of the dual slave problems. The proposed procedure is described in the next subsection.

4.2.4 Analytical approaches for problems $SP_{ij}(\hat{v})$ and $SP_{ij}^{dual}(\hat{v})$

The optimal objective value of problem $SP_{ij}(\hat{v})$ (or of its dual problem $SP_{ij}^{dual}(\hat{v})$) yields the cost of the connection i-j for a feasible binary assignment of variables $\hat{v}$ provided by the MP. A straightforward optimal solution of problem $SP_{ij}(\hat{v})$ is:

$$\hat{s}^{ij}_k = \prod_{\ell \leq k} (1 - (1 - p_{[\ell]}) \hat{v}_{[\ell]}) \qquad k = 1, \ldots, |\mathcal{P}_{ij}| \tag{18}$$

$$\hat{r}^{ij}_k = (1 - p_{[k]}) \hat{v}_{[k]} \hat{s}^{ij}_{k-1} = (1 - p_{[k]}) \hat{v}_{[k]} \prod_{\ell \leq k-1} (1 - (1 - p_{[\ell]}) \hat{v}_{[\ell]}) \qquad k = 1, \ldots, |\mathcal{P}_{ij}| \tag{19}$$

The corresponding objective value is $c_{ij} \hat{s}^{ij}_{|\mathcal{P}_{ij}|} = c_{ij} \prod_{\ell \leq |\mathcal{P}_{ij}|} (1 - (1 - p_{[\ell]}) \hat{v}_{[\ell]})$. Notice that the last product in (19) is equal to 1 when $k = 1$.

Given the optimal values $(\hat{r}^{ij}_k, \hat{s}^{ij}_k)$ of the primal variables of problem $SP_{ij}(\hat{v})$, we analytically compute the optimal values $\hat{\lambda}^{ij}$ of the dual problem $SP_{ij}^{dual}(\hat{v})$ by the following procedure denoted as $ComputeSol\lambda^{ij}$.

Procedure $ComputeSol\lambda^{ij}$

Input: A solution vector $\hat{v}$ from the MP, an optimal solution $(\hat{r}_k^{ij}, \hat{s}_k^{ij})$ of problem $SP_{ij}(\hat{v})$.
Output: An optimal solution of the dual problem $SP_{ij}^{dual}(\hat{v})$.
for $i = 1$ to $i = |\mathcal{P}_{ij}|$ **do**
    if ($\hat{v}_{[i]} = 1$ and $p_{[i]} = 0$) **then return** $\hat{\lambda}_{k\ell}^{ij} := 0$ for $k = 1, \ldots, |\mathcal{P}_{ij}|$ and $\ell = 1, 2, 3$; **end if**
end for
$\hat{\lambda}_{|\mathcal{P}_{ij}|2}^{ij} := c_{ij}$;
if ($\hat{v}_{[|\mathcal{P}_{ij}|]} < \hat{s}_{|\mathcal{P}_{ij}|-1}^{ij}$) **then** $\hat{\lambda}_{|\mathcal{P}_{ij}|1}^{ij} := -\hat{\lambda}_{|\mathcal{P}_{ij}|2}^{ij}, \quad \hat{\lambda}_{|\mathcal{P}_{ij}|3}^{ij} := 0$;
else $\hat{\lambda}_{|\mathcal{P}_{ij}|1}^{ij} := 0, \quad \hat{\lambda}_{|\mathcal{P}_{ij}|3}^{ij} := -\hat{\lambda}_{|\mathcal{P}_{ij}|2}^{ij}$;
end if
for $k = |\mathcal{P}_{ij}|-1$ to $k = 2$ **do**
    $\hat{\lambda}_{k2}^{ij} := (1 - p_{[k+1]})\hat{\lambda}_{(k+1)3}^{ij} + \hat{\lambda}_{(k+1)2}^{ij}$;
    if ($\hat{v}_{[k]} < \hat{s}_{k-1}^{ij}$) **then** $\hat{\lambda}_{k1}^{ij} := -\hat{\lambda}_{k2}^{ij}, \quad \hat{\lambda}_{k3}^{ij} := 0$;
    else $\hat{\lambda}_{k1}^{ij} := 0, \quad \hat{\lambda}_{k3}^{ij} := -\hat{\lambda}_{k2}^{ij}$;
    end if
end for
$\hat{\lambda}_{12}^{ij} := (1 - p_{[2]})\hat{\lambda}_{23}^{ij} + \hat{\lambda}_{22}^{ij}$;
$\hat{\lambda}_{11}^{ij} := -\hat{\lambda}_{12}^{ij}$;
return $\hat{\lambda}_{k\ell}^{ij}$ for $k = 1, \ldots, |\mathcal{P}_{ij}|$ and $\ell = 1, 2, 3$.

Proposition 7: Procedure $ComputeSol\lambda^{ij}$ provides an optimal solution of any dual slave problem $SP_{ij}^{dual}(\hat{v})$.

Proof. Given an optimal solution of problem $SP_{ij}(\hat{v})$, we apply the complementary slackness conditions from duality theory to deduce an optimal solution of problem $SP_{ij}^{dual}(\hat{v})$. At first, we consider the case where there is at least one variable $\hat{v}_i = 1$ with $p_i = 0$ in the solution provided by the MP. This implies that nodes i and j are disconnected, namely the optimal objective value of problem $SP_{ij}(\hat{v})$ is 0. Therefore a feasible and optimal dual solution is simply obtained by setting $\hat{\lambda}_{k\ell}^{ij} = 0$ for all k and l.

In the remaining cases, no certain disconnection occurs in the path between i and j. Thus, we have $\hat{s}_k^{ij} > 0$ for $k = 1, \ldots, |\mathcal{P}_{ij}|$. Here, slackness conditions imply that an optimal dual solution must satisfy constraints (16d) and (16e) at equality. We now proceed by considering the last node in $\mathcal{P}_{ij}$ and the dual variables $\lambda_{|\mathcal{P}_{ij}|\ell}^{ij}$ ($\ell = 1, \ldots, 3$). From constraint (16e), we have $\hat{\lambda}_{|\mathcal{P}_{ij}|2}^{ij} = c_{ij}$. To derive the values of $\lambda_{|\mathcal{P}_{ij}|1}^{ij}$ and $\lambda_{|\mathcal{P}_{ij}|3}^{ij}$, we have to consider two cases: a) $\hat{v}_{[|\mathcal{P}_{ij}|]} < \hat{s}_{|\mathcal{P}_{ij}|-1}^{ij}$; b) $\hat{v}_{[|\mathcal{P}_{ij}|]} \geq \hat{s}_{|\mathcal{P}_{ij}|-1}^{ij}$.

In case a), constraint (15e) ($k = |\mathcal{P}_{ij}|$) is nonbinding. So we have $\hat{\lambda}_{|\mathcal{P}_{ij}|3}^{ij} = 0$ due to the associated complementary slackness condition. From Proposition 6, we also have $\hat{r}_{|\mathcal{P}_{ij}|}^{ij} = (1 - p_{[|\mathcal{P}_{ij}|]})\hat{v}_{[|\mathcal{P}_{ij}|]}$, with $\hat{r}_{|\mathcal{P}_{ij}|}^{ij} \geq 0$. If $\hat{r}_{|\mathcal{P}_{ij}|}^{ij} > 0$, then an optimal dual solution must satisfy constraint (16c) ($k = |\mathcal{P}_{ij}|$) with equality, thus giving $\hat{\lambda}_{|\mathcal{P}_{ij}|1}^{ij} = -\hat{\lambda}_{|\mathcal{P}_{ij}|2}^{ij}$. If $\hat{r}_{|\mathcal{P}_{ij}|}^{ij} = 0$, this means that $\hat{v}_{[|\mathcal{P}_{ij}|]} = 0$ (notice that even if $p_{[|\mathcal{P}_{ij}|]} = 1$, then still

$\hat{v}_{[|\mathcal{P}_{ij}|]} = 0$ due to the related constraint (14c)). Since variable $\lambda^{ij}_{|\mathcal{P}_{ij}|1}$ appears only in the related constraint (16c) and is not in the objective function anymore when $\hat{v}_{[|\mathcal{P}_{ij}|]} = 0$, we can again set $\hat{\lambda}^{ij}_{|\mathcal{P}_{ij}|1} = -\hat{\lambda}^{ij}_{|\mathcal{P}_{ij}|2}$ to satisfy such a constraint.

In case b), constraint (15d) ($k = |\mathcal{P}_{ij}|$) is nonbinding implying $\hat{\lambda}^{ij}_{|\mathcal{P}_{ij}|1} = 0$. In this case, we have $\hat{r}^{ij}_{|\mathcal{P}_{ij}|} = (1-p_{[|\mathcal{P}_{ij}|]})\hat{s}^{ij}_{|\mathcal{P}_{ij}|-1}$ from Proposition 6, with $\hat{r}^{ij}_{|\mathcal{P}_{ij}|} > 0$ as $\hat{s}^{ij}_{|\mathcal{P}_{ij}|-1} > 0$ and $p_{[|\mathcal{P}_{ij}|]} < 1$ (as $\hat{v}_{[|\mathcal{P}_{ij}|]} \geq \hat{s}^{ij}_{|\mathcal{P}_{ij}|-1} > 0$). This implies that the corresponding dual constraint (16c) must be satisfied at equality in an optimal dual solution, hence $\hat{\lambda}^{ij}_{|\mathcal{P}_{ij}|3} = -\hat{\lambda}^{ij}_{|\mathcal{P}_{ij}|2}$. We now iteratively consider nodes $k = |\mathcal{P}_{ij}|-1, \ldots, 2$. As $s^{ij}_k > 0$ for all k, from constraints (16d) we have

$$\hat{\lambda}^{ij}_{k2} = (1 - p_{[k+1]})\hat{\lambda}^{ij}_{(k+1)3} + \hat{\lambda}^{ij}_{(k+1)2}.$$

After computing the value of $\hat{\lambda}^{ij}_{k2}$ with the last equation, we can apply the previous analysis to compute the values of $\hat{\lambda}^{ij}_{k1}$ and $\hat{\lambda}^{ij}_{k3}$ according to the value of $\hat{v}_{[k]}$ and $\hat{s}^{ij}_{k-1}$. Correspondingly, we obtain $\hat{\lambda}^{ij}_{k1} = -\hat{\lambda}^{ij}_{k2}$ and $\hat{\lambda}^{ij}_{k3} = 0$ if $\hat{v}_{[k]} < \hat{s}^{ij}_{k-1}$ or else $\hat{\lambda}^{ij}_{k1} = 0$ and $\hat{\lambda}^{ij}_{k3} = -\hat{\lambda}^{ij}_{k2}$. When we reach the first node in the path, we have $\hat{\lambda}^{ij}_{12} = (1 - p_{[2]})\hat{\lambda}^{ij}_{23} + \hat{\lambda}^{ij}_{22}$ due to the related constraint (16d). Finally we can set $\hat{\lambda}^{ij}_{11} = -\hat{\lambda}^{ij}_{12}$ to satisfy constraint (16b) for any value of $\hat{r}^{ij}_1$.

Notice that the computed dual solution is feasible as it satisfies all the dual constraints and bounds (16g). The solution has been derived by invoking the slackness conditions on the primal variables and the corresponding dual constraints. We can easily show that it also satisfies the complementary slackness conditions on the dual variables and the related primal constraints, implying that the dual solution is optimal. These complementary slackness conditions are trivially satisfied for primal constraints (15b), (15c) and (15f) as they are equality constraints. Constraints (15d) and (15e) are either satisfied at equality as well (invoking Proposition 6) or are nonbinding. But in the latter case the corresponding dual variable is set to 0 thus satisfying complementary slackness. □

We remark that the procedure *ComputeSol*λ^{ij} also applies to fractional solutions of the linear relaxation of the MP.

We report the pseudo-code of the proposed exact approach below. After an initialization step (Lines 2-3) where we start with empty sets $\Phi_{ij}(z_{ij}, v)$, we iteratively solve the MP (Line 5) as long as the obtained lower bound LB is not within a precision value ε of the best solution value obtained so far, denoted as UB (*while-loop* in Lines 4-17). In the first iteration, we consider the optimal solution of the MP where all variables are equal to 0. In the next iterations, the values of LB and UB are updated according to the solutions given by the MP (Lines 6-10). We also add the relevant cuts to the MP given by solving the dual slave subproblems (Lines 11-16). The solution set S of the attacked nodes in the best solution is finally returned (Line 18).

Algorithm BD_{SCNP}

Input: $SCNP_{tree}$ instance, parameter ε;
$S := \{\emptyset\}$; $LB := 0$; $UB := \infty$;
$\Phi_{ij}(z_{ij}, v) := \{\emptyset\}$ for $i, j \in V$: $i < j$;
while $UB - LB > \varepsilon$ **do**
    $(\hat{z}, \hat{v}) \leftarrow$ solve MP;
    if $\sum_{i<j} \hat{z}_{ij} > LB$ **then** $LB := \sum_{i<j} \hat{z}_{ij}$; **end if**
    for $i, j \in V$: $i < j$ **do**
        $(\hat{s}^{ij}, \hat{r}^{ij}) \leftarrow$ compute an optimal $SP_{ij}(\hat{v})$ solution according to (18)-(19);
    end for
    if $\sum_{i<j} c_{ij} \hat{s}^{ij}_{|\mathcal{P}_{ij}|} < UB$ **then** $UB := \sum_{i<j} c_{ij} \hat{s}^{ij}_{|\mathcal{P}_{ij}|}$, $S := \{i \in V : \hat{v}_i = 1\}$; **end if**
    for $i, j \in V$: $i < j$ **do**
        if $\hat{z}_{ij} < c_{ij} \hat{s}^{ij}_{|\mathcal{P}_{ij}|}$ **then**
            $\hat{\lambda}^{ij} \leftarrow$ solve $SP^{dual}_{ij}(\hat{v})$;
            Add constraint (17) to set $\Phi_{ij}(z_{ij}, v)$;
        end if
    end for
end while
return S.

5 Computational results

We first describe the setting of our numerical experiments as well as the characteristics of the different instances generated. Then, we discuss the results and the performance of the proposed approaches.

5.1 Experimental conditions and instances

All algorithms have been implemented in C++, compiled with `gcc 4.1.2`, and all tests were performed on an HP ProLiant DL585 G6 server with two 2.1 GHz AMD Opteron 8425HE processors (with 12 threads) and 24 GB of RAM. We used solver CPLEX 12.8 to solve model (13) and the MP along the iterations of algorithm BD_{SCNP}. For model (13), we set the precision level (absolute gap) of CPLEX 12.8 to 0.001. The other parameters of the solver were set to their default values. In algorithm BD_{SCNP} we set $\varepsilon = 0.001$ for a fair comparison with CPLEX 12.8 launched on model (13). We generated trees with the number of nodes n ranging from 20 to 750. For each value of n, 30 different trees were generated as in [15], i.e., using Broder's algorithm for the uniform spanning tree problem [10]. The generation scheme in [15] guarantees that each tree is randomly chosen with uniform distribution among all trees with n nodes. For each instance, we generated a set of survival probabilities p_i ($i \in V$) uniformly distributed between 0 and 1 and with two decimal digits. For each tree, we considered different sets of weights. A first set of *unweighted instances* considers both unit attack costs κ_i and unit connection costs c_{ij}. Then, we generated *weighted instances* with three different sets of weights. A first set of weights (*type 1*) considers both attack costs and connection costs as integers uniformly distributed in the interval $[1, 10]$. A second set of weights (*type 2*) still considers

connections costs $c_{ij} \in [1, 10]$ but attack costs $\kappa_i \in [1, 100]$. The last set of weights (*type 3*) considers connection costs $c_{ij} \in [1, 10]$ and attack costs equal to the inverse of the survival probability of each node, i.e., $\kappa_i = 1/p_i$ for each $i \in V$, where we set $\kappa_i = 100$ if $p_i = 0$. For this last set of weights, the rationale is to generate instances which are expected to be hard to solve in the sense that nodes with a lower attack cost have also a higher survival probability. Summarizing, each instance has a unique set of probabilities p_i but four different sets of attack costs and connection costs. The budget K is equal to 10% of the sum of the attack costs of the nodes: $K = 0.1 \sum_{i \in V} \kappa_i$. We remark that even for the smallest instances with $n = 20$, the use of the general model (1) would be impractical due to the induced large number of variables and constraints given by the number of scenarios to analyze. Our approach does not explicitly require the definition of all possible scenarios. Thus, it can be applied to larger instances within reasonable computational times, as discussed in the next sections. We tested model (13) and algorithm BD_{SCNP} with a time limit of 3600 seconds.

5.2 Results for the unweighted instances

We first present the computational results for the unweighted instances in Table 1. For each given number of nodes n, the table reports the performance of model (13) solved over 30 instances in terms of average computational time (column *time*), average percentage gap $1 - \frac{LB}{UB}$ (column *gap (%)*) between the best solution found (UB) for the problem and its lower bound (LB), number of instances solved within the time limit (column *# closed*). The average values consider also the instances where the time limit is reached. The same entries are reported for algorithm BD_{SCNP} with two additional columns: the average number of iterations (column *# iter.*) executed by the algorithm and the average number of cuts added over all iterations (column *# cuts*). The best percentage gap between the two approaches is displayed in bold font. The results in Table 1 illustrate that model (13) successfully solves instances with up to 60 nodes and all the instances with 80 nodes but one. For larger instances with up to 150 nodes, the solver hardly succeeds in solving the instances within the time limit of 3600 seconds and in obtaining relatively small percentage gaps. Given the performance of the model (13) on instances with 150 nodes, we did not test the model on instances with 300 or more nodes.

Algorithm BD_{SCNP} shows up to outperform model (13). The proposed algorithm solves all 30 instances for trees with up to 80 nodes with a lower average running time. For larger instances, the running times and percentage gaps provided by algorithm BD_{SCNP} are considerably smaller than the ones given by model (13). Even on instances with 150 nodes, algorithm BD_{SCNP} closes almost two thirds of the benchmark instances while model (13) closes none of the 30 instances. We point out that in our numerical experiments, the valid inequalities of Proposition 1 tend to speed up the performance of algorithm BD_{SCNP} by a considerable factor. Indeed, on instances with up to 100 nodes, they provide a reduction by 20-30% in the running times. This effect is harder to evaluate in the instances where the algorithm runs out of time. The limits of BD_{SCNP} in our experimental setting seem to be reached for instances between 300 and 500 nodes, since the relative gap at 500 nodes starts to be large (more than 20%).

n	Model (13)			BD_{SCNP}				
	time	gap (%)	# closed	time	gap (%)	# closed	# iter.	# cuts
20	0.63	**0.00**	30	0.53	**0.00**	30	2.57	289.37
40	5.63	**0.00**	30	4.47	**0.00**	30	4.33	1939.60
60	114.47	**0.00**	30	24.07	**0.00**	30	5.43	5578.97
80	885.70	0.09	29	116.23	**0.00**	30	6.37	12159.03
100	3054.17	3.28	14	900.00	**0.24**	28	8.60	24450.40
120	3486.70	19.69	2	1261.90	**0.23**	26	7.67	32895.00
150	3600.00	48.96	0	2321.07	**0.91**	19	7.63	52272.27
300	-	-	-	3600.00	9.02	0	2.93	109030.13
500	-	-	-	3600.00	21.54	0	1.97	219325.33
750	-	-	-	3600.00	73.67	0	1.33	359221.07

Table 1: $SCNP_{tree}$ unweighted instances.

5.3 Results for the weighted instances

The results for the three types of weighted instances are reported in Tables 2-4. For model (13), we report the results without the valid inequalities of Proposition 1 in the MILP model, since the solver performs better without these additional constraints in the weighted instances. Instead, the valid inequalities are still beneficial for algorithm BD_{SCNP} as in the unweighted instances. The comparison between model (13) and algorithm BD_{SCNP} confirms the previous trend in the results. In general, algorithm BD_{SCNP} solves more instances than model (13) within the considered time limit, except for instances of type 3 with 80 nodes. For large instances with more than 120 nodes, model (13) is not capable of solving the instances with a reasonably small percentage gap, while algorithm BD_{SCNP} provides low average gaps for instances with up to 150 nodes and gaps less than 10% for instances with 300 nodes (except for type 3 instances). As we should expect, the type 3 instances turn out to be the hardest instances to solve. Nevertheless, algorithm BD_{SCNP} provides solutions for type 3 instances with 150 nodes with an average percentage gap of 6%, although the algorithm does not close any instance within the time limit. With respect to the unweighted instances, we notice that the number of iterations and the number of generated cuts of the algorithm tend to increase for weighted instances, in particular for type 3 instances.

In general, solving to optimality instances with 300 or more nodes is currently out of reach. In these large instances, the convergence process of algorithm BD_{SCNP} is heavily affected by the difficulty of solving to optimality the Master Problem when the number of optimality cuts increases. For such instances, our algorithm can be seen as a heuristic with a guarantee on the quality of the solutions given by the computed lower bounds.

n	Model (13)			BD_{SCNP}				
	time	gap (%)	# closed	time	gap (%)	# closed	# iter.	# cuts
20	0.70	**0.00**	30	1.17	**0.00**	30	2.57	315.63
40	6.13	**0.00**	30	8.93	**0.00**	30	5.60	2566.27
60	115.67	**0.00**	30	37.27	**0.00**	30	6.33	6309.97
80	704.53	**0.00**	30	175.57	**0.00**	30	7.70	13986.00
100	2919.53	1.47	19	648.03	**0.16**	28	8.03	24282.80
120	3507.63	6.69	3	1387.23	**0.05**	24	8.87	36801.63
150	3600.00	55.20	0	2668.47	**0.97**	16	8.00	57318.87
300	-	-	-	3600.00	8.55	0	3.30	127527.40
500	-	-	-	3600.00	17.51	0	2.07	234602.40
750	-	-	-	3600.00	58.16	0	1.60	427493.53

Table 2: $SCNP_{tree}$ weighted instances (type 1).

n	Model (13)			BD_{SCNP}				
	time	gap (%)	# closed	time	gap (%)	# closed	# iter.	# cuts
20	0.93	**0.00**	30	1.17	**0.00**	30	2.83	359.77
40	6.90	**0.00**	30	4.73	**0.00**	30	4.83	2263.00
60	158.73	**0.00**	30	28.40	**0.00**	30	5.50	6041.87
80	932.07	**0.00**	30	104.33	**0.00**	30	6.80	12635.27
100	2875.90	1.87	19	687.97	**0.06**	29	8.70	24405.93
120	3506.07	6.87	3	1221.00	**0.07**	26	8.03	33641.67
150	3600.00	45.48	0	2809.67	**0.69**	15	8.13	56545.30
300	-	-	-	3600.00	9.58	0	3.40	128142.57
500	-	-	-	3600.00	15.46	0	2.33	261545.87
750	-	-	-	3600.00	67.45	0	1.40	373020.47

Table 3: $SCNP_{tree}$ weighted instances (type 2).

n	Model (13)			BD_{SCNP}				
	time	gap (%)	# closed	time	gap (%)	# closed	# iter.	# cuts
20	0.93	**0.00**	30	2.43	**0.00**	30	5.47	633.27
40	11.00	**0.00**	30	22.80	**0.00**	30	8.87	4073.53
60	240.07	**0.00**	30	272.40	**0.00**	30	12.03	12815.43
80	1361.83	**0.03**	29	1014.30	0.09	27	10.93	21726.67
100	3157.00	3.52	11	2224.20	**0.96**	18	10.10	34298.77
120	3600.00	16.35	0	3055.90	**3.12**	9	8.47	46471.73
150	3600.00	41.61	0	3600.00	**6.13**	0	6.43	63291.63
300	-	-	-	3600.00	17.67	0	3.30	145304.60
500	-	-	-	3600.00	27.01	0	2.80	346410.50
750	-	-	-	3600.00	63.44	0	1.77	494929.87

Table 4: $SCNP_{tree}$ weighted instances (type 3).

5.4 Procedure $ComputeSol\lambda^{ij}$ versus solving LP problems

In order to evaluate the effectiveness of the procedure $ComputeSol\lambda^{ij}$, we also compared the analytical procedure with the alternative of solving the LP models $SP_{ij}^{dual}(\hat{v})$ with CPLEX 12.8. We performed the comparison by considering as a test-bed all unweighted and weighted instances with up to 150 nodes and by computing the overall time taken by $ComputeSol\lambda^{ij}$ and by the LP solver to solve all suproblems $SP_{ij}^{dual}(\hat{v})$ in each instance. Our procedure required much less time than the LP solver: the ratio between the time taken by $ComputeSol\lambda^{ij}$ and that taken by the LP solver was usually less than 1% with values ranging from 0.2% to 1%. These results highlight the gain in performance of the proposed procedure. In general, the computational effort needed to solve the master problems is expected to represent a large portion of the whole running time, especially for larger instances. Still, it turned out that solving the dual subproblems with the LP solver would have required a non-negligible part of the computational time. In small instances with 60 nodes or less, the time taken by the LP solver was of the same order as the time devoted to the master problems, while the ratio between the two times was equal to 18% in unweighted instances with 100 nodes and 6% in unweighted instances with 150 nodes. The same trend emerged for the weighted instances.

5.5 Results for instances with equal survival probabilities

To get a broader picture on the performance of our approaches, we also tested instances where all the survival probabilities are equal to a value p, i.e., $p_i = p$ for $i \in V$. We benchmarked algorithm BD_{SCNP} against the specific ILP model, called ILP_p, that we derived for this problem variant. The details of model ILP_p are reported in Appendix A. We considered unweighted instances and weighted instances of type 1 and two probability values: $p = 0.2$ and $p = 0.4$. We report the results for the unweighted instances in Table 5 and those for the weighted instances in Table 6. Model ILP_p was solved by means of CPLEX 12.8 with the above mentioned settings. For algorithm BD_{SCNP}, we omitted the average number of iterations and the average number of generated cuts in the tables.

The results show that model ILP_p usually outperforms algorithm BD_{SCNP} in terms of average percentage gap and running times in instances with up to 300 nodes. Further, model ILP_p is extremely fast in solving

	$p = 0.2$						$p = 0.4$					
	ILP_p			BD_{SCNP}			ILP_p			BD_{SCNP}		
n	time	gap (%)	# closed	time	gap (%)	# closed	time	gap (%)	# closed	time	gap (%)	# closed
20	0.37	**0.00**	30	2.07	**0.00**	30	0.37	**0.00**	30	2.53	**0.00**	30
40	0.63	**0.00**	30	3.27	**0.00**	30	0.73	**0.00**	30	34.60	**0.00**	30
60	1.73	**0.00**	30	13.07	**0.00**	30	1.80	**0.00**	30	687.97	0.05	29
80	3.63	**0.00**	30	66.23	**0.00**	30	3.10	**0.00**	30	2276.27	0.92	17
100	8.27	**0.00**	30	216.13	**0.00**	30	8.47	**0.00**	30	3506.13	6.42	1
120	13.23	**0.00**	30	1036.43	**0.00**	30	12.30	**0.00**	30	3600.00	12.69	0
150	27.00	**0.00**	30	2832.00	0.21	19	25.53	**0.00**	30	3600.00	17.58	0
300	958.50	**3.26**	29	3600.00	4.47	0	765.13	**0.01**	28	3600.00	26.06	0
500	3600.00	99.98	0	3600.00	**8.79**	0	3600.00	99.94	0	3600.00	**40.21**	0
750	3600.00	99.99	0	3600.00	**81.41**	0	3600.00	99.96	0	3600.00	**73.34**	0

Table 5: $SCNP_{tree}$ unweighted instances with equal survival probabilities.

	$p = 0.2$						$p = 0.4$					
	ILP_p			BD_{SCNP}			ILP_p			BD_{SCNP}		
n	time	gap (%)	# closed	time	gap (%)	# closed	time	gap (%)	# closed	time	gap (%)	# closed
20	0.43	**0.00**	30	0.27	**0.00**	30	1.77	**0.00**	30	0.50	**0.00**	30
40	1.33	**0.00**	30	1.27	**0.00**	30	3.00	**0.00**	30	6.33	**0.00**	30
60	4.70	**0.00**	30	8.53	**0.00**	30	11.77	**0.00**	30	81.57	**0.00**	30
80	10.63	**0.00**	30	27.97	**0.00**	30	25.87	**0.00**	30	529.93	0.01	28
100	31.07	**0.00**	30	74.23	**0.00**	30	49.70	**0.00**	30	1552.87	0.18	23
120	43.70	**0.00**	30	223.57	**0.00**	30	78.90	**0.00**	30	3119.27	1.76	8
150	98.73	**0.00**	30	728.97	**0.00**	30	125.33	**0.00**	30	3548.00	7.84	1
300	3209.70	56.06	8	3600.00	**1.97**	0	2307.93	**13.10**	23	3600.00	18.24	0
500	3600.00	99.98	0	3600.00	**6.21**	0	3600.00	99.94	0	3600.00	**28.37**	0
750	3600.00	99.99	0	3600.00	**23.63**	0	3600.00	99.96	0	3600.00	**51.72**	0

Table 6: $SCNP_{tree}$ weighted instances (type 1) with equal survival probabilities.

instances with up to 150 nodes. On the other hand, in instances with 500 and 750 nodes, model ILP_p suffers from the presence of a large number of variables and corresponding memory requirements. This behavior could be expected as the number of binary variables in the model is of the order of $\mathcal{O}(n^3)$. In these instances, algorithm BD_{SCNP} gives better results within the considered time limit. Finally, we notice that, for model ILP_p, the weighted instances tend to be harder to solve than the unweighted instances, while an opposite trend emerges for algorithm BD_{SCNP}.

6 Conclusions

In this work we introduced a stochastic version of the Critical Node Problem where the outcome of attacks on nodes is uncertain. We first presented an Integer Linear Programming formulation for the problem over general graphs and derived valid inequalities. Then, we focused on the problem variant over trees for which we provided theoretical results, a nonlinear model and a linear reformulation based on the concept of probability chains from probability theory. Moreover, we devised an exact Benders Decomposition approach for which we derived an analytical solution of the slave subproblems. We performed an extensive computational analysis to assess the effectiveness and robustness of the proposed approaches. The BD algorithm is capable of efficiently solving instances with up to 150 nodes and improves upon the performance of the linear

reformulation of the problem. We also introduced an ILP model and an approximation algorithm based on a dynamic program for specific problem variants of interest.

With this work, we aim to foster research on stochastic versions of the Critical Node Problem. Numerous research lines could be explored in the future. The most natural follow-up of our work would be the design of a solution approach for tackling the stochastic CNP over general graphs. A possibility is to combine modern scenario partitioning methods such as in [14, 41, 43] with a Column Generation approach. It might be also interesting to adapt the proposed algorithmic framework based on Benders Decomposition to the deterministic version of the CNP on general graphs. Also, exploring the design of efficient heuristic/metaheuristic algorithms could be a valid alternative to solve instances on large (and arbitrary) graphs. This last option would call for finding efficient procedures to compute the stochastic objective function of a given solution, since the complexity of computing this value is *a priori* exponential in the solution size. For example, we could derive a FPRAS to compute the objective function with a suitable precision, as it was done, e.g., in [17] for the stochastic CNP with edge uncertainty.

Finally, we have shown in this work a successful application of the concept of probability chains for problem linearizations significantly improved by decomposition methods such as the Benders Decomposition. In future research, it would be interesting to consider a similar approach for deriving effective solution methods for other stochastic optimization problems.

Acknowledgements

We wish to thank the two anonymous referees and the associate editor for their valuable suggestions and fruitful comments. We thank the authors of [15] for sharing their code for the generation of the instances. We are in debt to Konstantin Pavlikov for bringing the probability chains method to our attention. This work has been partially supported by "Ministero dell'Istruzione, dell'Università e della Ricerca" Award "TESUN-83486178370409 finanziamento dipartimenti di eccellenza CAP. 1694 TIT. 232 ART. 6".

References

[1] B. Addis, M. Di Summa, and A. Grosso, *Removing critical nodes from a graph: Complexity results and polynomial algorithms for the case of bounded treewidth*, Discr. Appl. Math. **16-17** (2013), 2349–2360.

[2] A.C. Andersen and K. Pavlikov, *Weapon-Target Assignment Problem: Exact and approximate solution algorithms. Available at* `https://www.researchgate.net/publication/331936954_Weapon-Target_Assignment_Problem_Exact_and_Approximate_Solution_Algorithms`, Submitted publication (2019).

[3] R. Aringhieri, A. Grosso, and P. Hosteins, *A genetic algorithm for a class of critical node problems*, The 7th International Network Optimization Conference (INOC'15), Vol. 52 of *Electronic Notes in Discrete Mathematics*, 2016, pp. 359–366.

[4] R. Aringhieri, A. Grosso, P. Hosteins, and R. Scatamacchia, *A general evolutionary framework for different classes of critical node problems*, Eng. Appl. Artificial Intelligence **55** (2016), 128–145.

[5] R. Aringhieri, A. Grosso, P. Hosteins, and R. Scatamacchia, *Local search metaheuristics for the critical node problem*, Networks **67** (2016), 209–221.

[6] R. Aringhieri, A. Grosso, P. Hosteins, and R. Scatamacchia, *Polynomial and pseudo-polynomial time algorithms for different classes of the distance critical node problem*, Discr. Appl. Math. **253** (2019), 103–121.

[7] A. Arulselvan, C.W. Commander, L. Elefteriadou, and P.M. Pardalos, *Detecting critical nodes in sparse graphs*, Comput. Oper. Res. **36** (2009), 2193–2200.

[8] M.O. Ball, B.L. Golden, and R.V. Vohra, *Finding the most vital arcs in a network*, Oper. Res. Lett. **8** (1989), 73–76.

[9] J.F. Benders, *Partitioning procedures for solving mixed-variables programming problems*, Numer. Mathematik **4** (1962), 238–252.

[10] A. Broder, *Generating random spanning trees*, 30th Annual Symposium on Foundations of Computer Science, 1989, pp. 442–447.

[11] G. Brown, M. Carlyle, J. Salmerón, and K. Wood, *Defending critical infrastructure*, Interfaces **36** (2006), 530–544.

[12] H.W. Corley and H. Chang, *Finding the n most vital nodes in a flow network*, Manage. Sci. **21** (1974), 362–364.

[13] H.W. Corley and D.Y. Sha, *Most vital links and nodes in weighted networks*, Oper. Res. Lett. **1** (1982), 157–160.

[14] K.J. Cormican, D.P. Morton, and R.K. Wood, *Stochastic network interdiction*, Oper. Res. **46** (1998), 184–197.

[15] M. Di Summa, A. Grosso, and M. Locatelli, *Complexity of the critical node problem over trees*, Comput. Oper. Res. **38** (2011), 1766–1774.

[16] M. Di Summa, A. Grosso, and M. Locatelli, *Branch and cut algorithms for detecting critical nodes in undirected graphs*, Comput. Optim. Appl. **53** (2012), 649–680.

[17] T.N. Dinh and M.T. Thai, *Assessing attack vulnerability in networks with uncertainty*, 2015 IEEE Conference on Computer Communications (INFOCOM), 2015, pp. 2380–2388.

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

[19] N. Fan and P.M. Pardalos, *Robust optimization of graph partitioning and critical node detection in analyzing networks*, Combinatorial Optimization and Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 170–183.

[20] D. Granata, G. Steeger, and S. Rebennack, *Network interdiction via a critical disruption path: Branch-and-price algorithms*, Comput. Oper. Res. **40** (2013), 2689–2702.

[21] T. Homem-de-Mello and G. Bayraksan, *Monte Carlo sampling-based methods for stochastic optimization*, Surveys Oper. Res. Manage. Sci. **19** (2014), 56–85.

[22] F. Hooshmand, F. Mirarabrazi, and S.A. MirHassani, *Efficient Benders decomposition for distance-based critical node detection problem*, Omega **93** (2019), 102037.

[23] E. Israeli and R.K. Wood, *Shortest-path network interdiction*, Networks **40** (2002), 97–111.

[24] U. Janjarassuk and J. Linderoth, *Reformulation and sampling to solve a stochastic network interdiction problem*, Networks **52** (2008), 120–132.

[25] E. Jenelius, T. Petersen, and L.G. Mattsson, *Importance and exposure in road network vulnerability analysis*, Transp. Res. Part A: Pol. Practice **40** (2006), 537–560.

[26] G. Karakose and R.G. McGarvey, *Optimal detection of critical nodes: Improvements to model structure and performance*, Networks Spatial Econ. **19** (2019), 1–26.

[27] M. Lalou, M.A. Tahraoui, and H. Kheddouci, *Component-cardinality-constrained critical node problem in graphs*, Discr. Appl. Math. **210** (2016), 150–163.

[28] M. Lalou, M.A. Tahraoui, and H. Kheddouci, *The critical node detection problem in networks: A survey*, Comput. Sci. Review **28** (2018), 92–117.

[29] H.R. Medal, E.A. Pohl, and M.D. Rossetti, *Allocating protection resources to facilities when the effect of protection is uncertain*, IIE TRANSACT **48** (2016), 220–234.

[30] D.P. Morton, F. Pan, and K. Saeger, *Models for nuclear smuggling interdiction*, IIE TRANSACT **39** (2007), 3–14.

[31] J. Naoum-Sawaya and C. Buchheim, *Robust critical node selection by Benders decomposition*, INFORMS J. Comput. **28** (2016), 162–174.

[32] J.R. O'Hanley, M.P. Scaparra, and S. Garcìa, *Probability chains: A general linearization technique for modeling reliability in facility location and related problems*, Eur. J. Oper. Res. **230** (2013), 63–75.

[33] K. Pavlikov, *Improved formulations for minimum connectivity network interdiction problems*, Comput. Oper. Res. **97** (2018), 48–57.

[34] D. Purevsuren, G. Cui, M. Chu, and N.N.H. Win, *Hybridization of GRASP with exterior path relinking for identifying critical nodes in graphs*, IAENG Int. J. Comput. Sci. **44** (2017), 157–165.

[35] R. Rahmanian, T.G. Crainic, M. Gendreau, and W. Rei, *The Benders decomposition algorithm: A literature review*, Eur. J. Oper. Res. **259** (2017), 801–817.

[36] J. Salmerón, K. Wood, and R. Baldick, *Analysis of electric grid security under terrorist threat*, IEEE Trans Power Syst **19** (2004), 905–912.

[37] J. Seo, S. Mishra, X. Li, and M.T. Thai, *Catastrophic cascading failures in power networks*, Theor. Comput. Sci. **607** (2015), 306–319.

[38] 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** (2012), 103–119.

[39] J.C. Smith, M. Prince, and J. Geunes, "*Modern network interdiction problems and algorithms*," *Handbook of Combinatorial Optimization*, P.M. Pardalos, D.Z. Du, and R.L. Graham (eds.), Springer New York, New York, NY, 2013, pp. 1949–1987.

[40] J.C. Smith and Y. Song, *A survey of network interdiction models and algorithms*, Eur. J. Oper. Res. **283** (2019), 797–811.

[41] Y. Song and J. Luedtke, *An adaptive partition-based approach for solving two-stage stochastic programs with fixed recourse*, SIAM J. Optim. **25** (2015), 1344–1367.

[42] V. Tomaino, A. Arulselvan, P. Veltri, and P.M. Pardalos, "*Studying connectivity properties in human protein–protein interaction network in cancer pathway*," *Data Mining for Biomarker Discovery*, P.M. Pardalos, P. Xanthopoulos, and M. Zervakis (eds.), Springer US, Boston, MA, 2012, pp. 187–197.

[43] W. van Ackooij, W. de Oliveira, and Y. Song, *Adaptive partition-based level decomposition methods for solving two-stage stochastic programs with fixed recourse*, INFORMS J. Comput. **30** (2018), 57–70.

[44] M. Ventresca and D. Aleman, *A randomized algorithm with local search for containment of pandemic disease spread*, Comput. Oper. Res. **48** (2014), 11–19.

[45] A. Veremyev, V. Boginski, and E.L. Pasiliao, *Exact identification of critical nodes in sparse networks via new compact formulations*, Optim. Lett. **8** (2014), 1245–1259.

[46] A. Veremyev, O.A. Prokopyev, and E.L. Pasiliao, *An integer programming framework for critical elements detection in graphs*, J. Combinatorial Optim. **28** (2014), 233–273.

[47] A. Veremyev, O.A. Prokopyev, and E.L. Pasiliao, *Critical nodes for distance-based connectivity and related problems in graphs*, Networks **66** (2015), 170–195.

[48] J.L. Walteros, A. Veremyev, P.M. Pardalos, and E.L. Pasiliao, *Detecting critical node structures on graphs: A mathematical programming approach*, Networks **73** (2018), 48–88.

[49] R.K. Wood, *Deterministic network interdiction*, Math. Comput. Modelling **17** (1993), 1–18.

[50] T. Zhou, Z.Q. Fu, and B.H. Wang, *Epidemic dynamics on complex networks*, Progress Natural Sci. **16** (2006), 452–457.

[51] Y. Zhou, J. Hao, and F. Glover, *Memetic search for identifying critical nodes in sparse graphs*, IEEE Trans. Cybernetics **49** (2018), 3699–3712.

Appendix A An ILP model with equal survival probabilities

For the simplified problem variant where all survival probabilities are equal to p, i.e., $p_i = p$ for $i \in V$, we derive an ILP formulation that implicitly takes into account the uncertainty associated with the attacks on nodes. Since the survival probabilities are equal, we only need to know the number of nodes attacked in $\mathcal{P}_{ij}$ in order to compute the average connection cost $c_{ij} \prod_{k \in S_{ij}} p_k = c_{ij} p^{|S_{ij}|}$. Therefore, we can introduce binary variables $y_r^{(ij)}$ equal to 1 only if there are r nodes attacked in path $\mathcal{P}_{ij}$. Index r can vary from 0 to

ρ_{ij}, with $\rho_{ij} := \min\{\frac{K}{\min_{i\in V}\kappa_i}, |\mathcal{P}_{ij}|\}$. We obtain the following ILP formulation (denoted as ILP_p):

ILP_p:

$$\min \sum_{i<j} c_{ij} \sum_{r\le\rho_{ij}} p^r y_r^{(ij)} \tag{20a}$$

$$\sum_i \kappa_i v_i \le K \tag{20b}$$

$$\sum_{r\le\rho_{ij}} y_r^{(ij)} = 1 \qquad i,j\in V : i<j \tag{20c}$$

$$\sum_{r\le\rho_{ij}} r y_r^{(ij)} = \sum_{k\in\mathcal{P}_{ij}} v_k \qquad i,j\in V : i<j \tag{20d}$$

$$y_r^{(ij)} \in \{0,1\} \qquad i,j\in V : i<j, r=0,\dots,\rho_{ij} \tag{20e}$$

$$v_i \in \{0,1\} \qquad i\in V \tag{20f}$$

The objective function (20a) minimizes the expected cost of the pairwise connectivity after attacking nodes in the tree. Constraint (20b) represents the budget constraint. Constraints (20c) ensure that only one value of r will be selected for each path $\mathcal{P}_{ij}$. Constraints (20d) link $y_r^{(ij)}$ variables with v_i variables. Constraints (20e) and (20f) define the domain of the variables.

Appendix B An approximation algorithm for the $SCNP_{tree}$ with unit costs

We present an approximation algorithm for the $SCNP_{tree}$ with unit connection costs and unit attack costs. The algorithm relies on limiting the numerical precision for the evaluation of each term in the objective function and has a performance guarantee in terms of the maximum deviation from the optimal objective value. We remark that the algorithm is very similar to the dynamic program we proposed in [6] for a certain class of multiplicative distance function for the Distance CNP over trees. Our goal here is to show how to apply similar techniques to derive an approximation algorithm for the problem considered. For the sake of conciseness, we will refer the reader to our previous work for the details of the algorithm and recursions. We first sketch a Dynamic Programming (DP) algorithm for the problem which constitutes the basis for the approximation algorithm. To derive a DP algorithm for the $SCNP_{tree}$, we will require that each connection cost in the objective can be scaled to an integer in the recursions. To this aim, let μ denote the minimum power of 10 such that $\mu\prod_{k\in\mathcal{P}_{ij}} p_k \in \mathbb{N}$ for any node pair $(i,j)\in V\times V$. Note that the value of parameter μ could be exponentially large depending on the value of probabilities p_k, which could compromise the performance of an algorithm based on scaling the objective to an integer value. However, as discussed below, the proposed approximation algorithm is based on limiting the value of μ.

Let $\mathcal{T}_a$ denote the subtree of tree $\mathcal{T}$ rooted at node $a\in V$, and by a_i with $i\in\{1,...,s\}$ the children of a. Also, we define as $\mathcal{T}_{a_{i\to s}}$ the subtree constituted by $\{a\}\cup_{j=i,\dots,s}\mathcal{T}_{a_j}$. For further details about this particular recursion scheme, we refer the interested reader to [6, 15]. All recursions in the dynamic programming approach are based on traversing the tree in postorder (i.e., from the leaves to the root) and from the right part of each tree level to the left one. We define the following recursion functions:

$F_a(c,k,\sigma)$:= minimum cost of a solution for subtree $\mathcal{T}_a$ when k nodes are attacked in $\mathcal{T}_a$ and the

total cost of connecting a to subtree $\mathcal{T}_a$ multiplied by μ is c. Index σ is equal to either 1 if $a \in S$ or 0 if $a \notin S$.

$G_{a_i}(c,k,\sigma)$:= minimum cost of a solution for subtree $\mathcal{T}_{a_{i\to s}}$ when k nodes are attacked in $\mathcal{T}_{a_{i\to s}}$ and the total cost of connecting a to subtree $\mathcal{T}_{a_{i\to s}}$ multiplied by μ is c. Index σ is equal to either 1 if $a \in S$ or 0 if $a \notin S$.

If it is not possible to remove k nodes from $\mathcal{T}_a$ such that the total cost of connecting a to subtree $\mathcal{T}_a$ multiplied by μ is equal to c, we have $F_a(c,k,\sigma) = \infty$ and/or $G_a(c,k,\sigma) = \infty$.

Let $C(\mathcal{T}')$ denote the connectivity cost of any subtree $\mathcal{T}'$. We derive the following recursions for a non-leaf node $a \in V$:

$$F_a(c,k,\sigma) = G_{a_1}(c,k,\sigma); \tag{21}$$

for $i < s$, if $p_a = 0$ and $a \in S$:

$$\begin{aligned} G_{a_i}(c,k,1) = \min \big\{ & F_{a_i}(b,q,\sigma') + G_{a_{i+1}}(0,k-q,1) : \\ & b = 0,\dots,\mu|\mathcal{T}_{a_i}|;\ q = 0,\dots,k-1;\ \sigma' = 0,1 \big\}, \end{aligned} \tag{22a}$$

otherwise, if $a \notin S$ or $p_a > 0$:

$$\begin{aligned} G_{a_i}(c,k,\sigma) = \min \Bigg\{ & F_{a_i}(b,q,\sigma') + G_{a_{i+1}}\left(c - p_a^\sigma(b+\mu p_{a_{i+1}}^{\sigma'}), k-q, \sigma\right) \\ & + p_a^\sigma p_{a_i}^{\sigma'} + p_{a_i}^{\sigma'}\left(\frac{c}{\mu} - p_a^\sigma\left(\frac{b}{\mu} + p_{a_{i+1}}^{\sigma'}\right)\right) + p_a^\sigma \frac{b}{\mu} + \frac{b}{\mu}\left(\frac{c}{\mu} - p_a^\sigma\left(\frac{b}{\mu} + p_{a_{i+1}}^{\sigma'}\right)\right) : \\ & b = 0,\dots,p_a^{-\sigma}c - \mu p_{a_i}^{\sigma'};\ q = 0,\dots,k-\sigma;\ \sigma' = 0,1 \Bigg\}. \end{aligned} \tag{22b}$$

The second line in Equation (22b) represents the total cost of connecting subtrees $\mathcal{T}_{a_i}$ and $\mathcal{T}_{a_{i\to s}}$. The first term in this line represents the connection cost of node a to a_i, the second term represents the connection cost between a_i and all nodes in $\mathcal{T}_{a_{i\to s}}$ and the third term the connection cost between a and all nodes in $\mathcal{T}_{a_i}$. Finally, the last term represents the connection cost between each node $u \in \mathcal{T}_{a_i} \setminus \{a_i\}$ and $v \in \mathcal{T}_{a_{i\to s}} \setminus \{a\}$. The cost for connecting a node pair (u,v) is in fact given by $E_{\bar{s}\in\mathbb{S}}[u_{uv}^{\bar{s}}] = E_{\bar{s}\in\mathbb{S}}[u_{ua_i}^{\bar{s}}]E_{\bar{s}\in\mathbb{S}}[u_{av}^{\bar{s}}]$ where $E_{\bar{s}\in\mathbb{S}}[u_{uv}^{\bar{s}}]$ is the average value of the connection between nodes u and v. Summing over all possible node pairs (u,v) gives the last term in Equation (22b). Note that if $p_a = 0$ and $\sigma = 0$, we assume in our equations that $p_a^{\pm\sigma} = 1$.

We do not provide extensive equations for the initial conditions as they are the same as those provided in [6]. The optimal solution value is given by

$$\min\{F_r(c,k,\sigma) :\ c = 0,\dots,C(\mathcal{T});\ k = 0,\dots,K;\ \sigma = 0,1\}$$

where r is the root node of the tree. As commonly done in DP algorithms, the optimal solution set of attacked nodes can be recovered by implementing a backtracking procedure, once the optimal recursion functions are obtained. We can state the following proposition.

Proposition 8: The previous DP algorithm has a time complexity of $\mathcal{O}(K^2 n^3 \mu^2)$.

Proof. The number of all possible combinations of the indices in functions F_a and G_a is bounded by $2[(n-1)\mu+1](K+1)$. The recursion step with the largest number of operations is given by Equation (22b) which requires up to $2[(n-1)\mu+1](K+1)$ operations. Thus, the running time of the DP algorithm can be bounded by $\mathcal{O}(K^2n^3\mu^2)$. □

Given the complexity stated in the above proposition, the performance of the proposed dynamic program is heavily affected by large values of μ. However, the DP algorithm can be modified to devise a reasonable approximation algorithm, which also provides a lower bound on the optimal objective value. Because of the rapid decrease of the cost function with the number of multiplying probabilities, beyond a certain number of attacked nodes the connection cost between two nodes will increase the objective function only by a negligible amount. Therefore, we modify the algorithm presented in Equations (21)-(22b) by truncating each term in the objective after a limited number of decimals corresponding to an integer ν. In the approximation algorithm, called *App*, we set $\mu = 10^\nu$ and we truncate the value of each term below the ν-th decimal. By limiting the value of ν and, correspondingly, the precision at which we want to solve the problem, we can keep the running time of the DP program under control. Moreover, algorithm *App* provides both a lower and an upper bound on the optimum solution value, as the following proposition shows.

Proposition 9: For the $SCNP_{tree}$ with unit connection costs and unit attack costs, algorithm *App* constitutes an approximation algorithm with time complexity $\mathcal{O}(K^2n^3\mu^2)$ and an approximation bound of $\frac{n(n-1)}{2\mu}$. The truncated objective of the approximate solution underestimates the optimal value by at most $\frac{n(n-1)}{2\mu}$.

Proof. The proof is formally the same as the one of Proposition 13 in [6]. Since every term in the objective function is truncated at the ν-th decimal, multiplying these terms by $\mu = 10^\nu$ is sufficient to obtain integer values, so we can use the DP algorithm presented above with complexity $\mathcal{O}(K^2n^3\mu^2)$ to obtain a heuristic solution S_{App}. As such a solution underestimates each term in the objective function by at most μ^{-1}, it underestimates the overall objective function by at most $\frac{n(n-1)}{2\mu}$. Let S^* denote an optimal solution of the original problem with objective function f^*. Since algorithm *App* underestimates the objective value of any solution, including S^*, the algorithm provides a truncated value between f^* and $f^* - \frac{n(n-1)}{2\mu}$, giving a lower bound on f^*. Besides, the non-truncated objective value f_{App} of S_{App} overestimates the corresponding truncated objective value of S_{App} by at most $\frac{n(n-1)}{2\mu}$. Hence, we have $f^* \leq f_{App} \leq f^* + \frac{n(n-1)}{2\mu}$, implying an approximation bound of $\frac{n(n-1)}{2\mu}$. □