Complexity of most vital nodes for Independent Set in graphs related to tree structures

Cristina Bazgan[1], Sonia Toubaline[1], and Zsolt Tuza[2⋆]

[1] Université Paris-Dauphine, LAMSADE, France
{bazgan,toubaline}@lamsade.dauphine.fr
[2] Computer and Automation Institute, Hungarian Academy of Sciences, Budapest and Department of Computer Science and Systems Technology, University of Veszprém, Hungary tuza@sztaki.hu

Abstract. Given an undirected graph with weights on its vertices, the k most vital nodes independent set problem consists of determining a set of k vertices whose removal results in the greatest decrease in the maximum weight of independent sets. We also consider the complementary problem, minimum node blocker independent set that consists of removing a subset of vertices of minimum size such that the maximum weight of independent sets in the remaining graph is at most a specified value. We show that these problems are NP-hard on bipartite graphs but polynomial-time solvable on unweighted bipartite graphs. Furthermore, these problems are polynomial also on graphs of bounded treewidth and cographs. A result on the non-existence of a ptas is presented, too.

Keywords: most vital nodes, independent set, complexity, NP-hard, ptas, bipartite graph, bounded treewidth, cograph

Mathematics Subject Classification: 05C85, 05C69

1 Introduction

In many applications involving the use of communication or transportation networks, we often need to identify vulnerable or critical infrastructures. By critical infrastructure we mean a set of nodes/lines whose damage causes the largest increase in the cost within the network. Modeling the network by a weighted graph, identifying a vulnerable infrastructure amounts to finding a subset of vertices/edges of a given size whose removal from the graph causes the largest inconvenience to a particular property of the graph in question. In the literature this problem is referred to as the *k most vital nodes/edges* problem. A complementary problem consists of determining a set of vertices/edges of minimum size whose removal involves that the cost within the network is at most a given

⋆ Research supported in part by the Hungarian Scientific Research Fund, OTKA grant 81493

value. In the literature this problem is referred to as the *min node/edge blocker* problem.

The problems of k most vital nodes/edges and min node/edge blocker have been studied for various problems, including shortest path, spanning tree, maximum flow, assignment, and maximum matching. The k most vital edges problem with respect to shortest path was proved NP-hard [1]. Later, k most vital edges/nodes shortest path (and min node/edge blocker shortest path) were proved not 2-approximable (not 1.36-approximable, respectively) if P$\neq$ NP [8]. For spanning tree, k most vital edges is NP-hard [5] and $O(\log k)$-approximable [5] and randomized 2-approximable [15]. In [17] it is proved that k most vital edges maximum flow is NP-hard. Also k most vital edges and min edge blocker assignment are proved NP-hard and not 2-approximable (not 1.36-approximable, respectively) if P$\neq$NP [2]. For maximum matching, min edge blocker is NP-hard even for bipartite graphs [16], but polynomial for grids and trees [14].

In this paper, we are interested in determining a subset of k *vertices* of the graph whose deletion causes the largest decrease in the maximum weight of an independent set. This problem is referred to as k Most Vital Nodes Independent Set. We also consider the complementary version of this problem, where given a threshold, we have to determine a subset of vertices of minimum cardinality that has to be removed such that in the resulting graph the maximum weight of an independent set is at most this threshold. This problem is referred to as Min Node Blocker Independent Set.

In Section 3 we consider bipartite graphs. It turns out that a substantial jump in complexity occurs between unweighted and weighted graphs for these problems. More precisely we show that the unweighted versions are polynomial while the weighted versions are NP-hard and the most vital nodes problem even has no ptas, unless P=NP. In Section 4 we deal with graphs with weights on their vertices, which have either a tree-like structure or a representation associated with trees. These include trees themselves, cycles, more generally graphs of bounded treewidth, and cographs (graphs containing no induced P_4). For these classes we design polynomial-time algorithms for the problems mentioned above.

In fact, trees and cycles have treewidth 1 and 2, respectively, therefore our general algorithm for bounded treewidth works for the former classes, too. Nevertheless, the algorithms on trees and cycles are simpler and this is why we include them here. It should be noted further that for k fixed, there are only polynomially many subsets of k removable vertices, therefore k Most Vital Nodes Independent Set is solvable efficiently on every graph class where the largest independent set is tractable. On the other hand if $k \to \infty$ then a formula expressing the present problems in second-order monadic logic would have unbounded length. Consequently, the general approach to linear-time algorithms via MSOL is not applicable here. This fact is relevant for both treewidth and cliquewidth.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph, with $V = \{v_1, \ldots, v_n\}$, where each vertex v_i has a weight w_i. For an edge $v_iv_j \in E$, we could write $v_i, v_j \in e$ and if $v_i, v_j \in V'$ then we consider that $e \subset V'$. When removing a set V' of vertices from G, let us denote the remaining graph by $G - V'$. If H is a subgraph of G then $V(H)$ denotes the vertex set of H. Moreover, for a subset V' of vertices from G, the subgraph induced by V' is denoted by $G[V']$. A maximum-weight independent set of G is a subset of vertices of maximum total weight where any two vertices are nonadjacent. A minimum-weight vertex cover of G is a subset of vertices of minimum total weight where every edge of G has at least one vertex in the set. We denote by $\alpha(G)$ the maximum weight of an independent set and by $\tau(G)$ the minimum weight of a vertex cover. Moreover, $\alpha(k)$ represents the minimum of $\alpha(G - V')$ after removing any set of vertices V' of size k. A matching is a set of mutually vertex-disjoint edges. The largest number of edges in a matching is denoted by $\nu(G)$.

In this paper we are interested in the complexity of the following problems.

k Most Vital Nodes Independent Set
Input: An undirected graph $G = (V, E)$ where each vertex v_i has a weight w_i, and an integer k.
Output: A subset $V' \subseteq V$ of size k such that the maximum weight $\alpha(G - V')$ of an independent set in $G - V'$ is minimum.

Min Node Blocker Independent Set
Input: An undirected graph $G = (V, E)$ where each vertex v_i has a weight w_i, and an integer U.
Output: A subset $V' \subseteq V$ of minimum cardinality such that the maximum weight $\alpha(G - V')$ of an independent set in $G - V'$ is at most U.

Remark 1. The exact versions of k Most Vital Nodes Independent Set and Min Node Blocker Independent Set are polynomial-time equivalent. Indeed, if an algorithm $\mathcal{A}_k$ solves k Most Vital Nodes Independent Set for all $1 \leq k \leq n$, then we can run $\mathcal{A}_k$ for $k = 1, \ldots, n$ and choose the smallest k yielding optimum at most U. Conversely, if an algorithm $\mathcal{B}_U$ solves Min Node Blocker Independent Set with any bound U, we can apply binary search to locate the smallest U that requires the removal of at most k vertices.

Theorem 1. *If there exists an algorithm that solves the k most vital nodes version of an optimization problem $\mathcal{P}$ on graphs with n vertices in $O(t)$ time, then the min node blocker version of $\mathcal{P}$ can be solved in $O(t \log\log n)$ time.*

Proof. If the value of an optimum solution is at most U then the optimum size is 0. Otherwise, we combine the algorithm for the k most vital nodes version with an accelerated version of *approximate binary search*. On the size k of a min node blocker we maintain a lower bound ℓ and an upper bound u, initialized to

$\ell_0 = 1$ and $u_0 = n$. Instead of using a standard binary search with $v = \frac{\ell+u}{2}$, we iteratively set $v = \sqrt{\ell u}$, as suggested in [7]. More precisely, although computing the exact value $\sqrt{\ell u}$ can be time consuming, it is shown in [7] that an approximate value of $\sqrt{\ell u}$ can be computed without affecting the time complexity. The number of tests for obtaining a lower bound ℓ and an upper bound u such that $u = \ell + 1$ is $O(\log\log\frac{u_0}{\ell_0})$ (see [7] for more details), which means $O(\log\log n)$ iterations in our case. Since one iteration takes $O(t)$, finding the smallest k for which the solution has value at most U takes total running time $O(t\log\log n)$.□

For the proof concerning the non-existence of a ptas (polynomial-time approximation scheme) we shall use an approximation-preserving reduction, called L-reduction, which was introduced by Papadimitriou and Yannakakis in [12]. Let A and B be two optimization problems. Then A is said to be *L-reducible* to B if there are two constants $a, b > 0$ such that

1. there exists a function, computable in polynomial time, which transforms each instance x of A into an instance x' of B such that $opt_B(x') \leq a \cdot opt_A(x)$,
2. there exists a function, computable in polynomial time, which transforms each solution y' of x' into a solution y of x such that $|val(x,y) - opt_A(x)| \leq b \cdot |val(x',y') - opt_B(x')|$.

For us the important property of this reduction is that if A is L-reducible to B and A has no ptas then B has no ptas.

3 Complexity on bipartite graphs

Maximum-weight independent set is polynomial-time solvable on bipartite graphs. We show in this section that the k most vital nodes or min node blocker versions become NP-hard on bipartite graphs, and most vital nodes has no ptas. Nevertheless, these problems remain polynomial-time solvable in the unweighted case. We first prove this latter fact.

Theorem 2. k Most Vital Nodes Independent Set *and also its complementary problem* Min Node Blocker Independent Set *are polynomial for unweighted bipartite graphs. Moreover, if a largest matching and a smallest vertex cover are given with the input, these problems are solvable in linear time.*

Proof. Let $G = (V, E)$ be a bipartite input graph on n vertices. From Kőnig's theorem [10] we know that $\tau(G) = \nu(G)$ holds; let us denote here their common value by t. The classical proof of the equality $\tau = \nu$ is algorithmic and also yields a maximum matching $M = \{e_1, \ldots, e_t\}$ and a minimum vertex cover $X = \{v_1, \ldots, v_t\}$ in polynomial time. Moreover, we have $\alpha(G) = n - t$ (known as a Gallai-type identity) and $V \setminus X$ is a largest independent set in G. Let us introduce the further notation $R = V \setminus V(M)$ and $r = |R| = n - 2t$; i.e., the number and the set of vertices not contained in any of the matching edges in M.

We can show now that these problems are solvable in linear time, as follows.

k Most Vital Nodes Independent Set

If $k \leq |R|$, we remove any k vertices from R. Since the remaining graph (of order $n-k$) still contains the matching M of size t, the independence number cannot be larger than $n-k-t$. It is also clear that α cannot be decreased by more than k if we remove just k vertices, hence the solution obtained is optimal.

If $k > |R|$, we remove the entire R and the vertices of $\lfloor (k-r)/2 \rfloor$ edges from M, and one further vertex if $k-r$ is odd. This decreases the size of M by $\lceil (k-r)/2 \rceil$ and the independence number by $\lfloor (k+r)/2 \rfloor$, and hence the new value is $\lceil (n-k)/2 \rceil$ (originally we had $\alpha(G) = (n+r)/2$). This decrease is optimal, because after the removal of k vertices at least half of the remaining $n-k$ belong to the same vertex class.

Min Node Blocker Independent Set

If $U \geq n-t$, no vertices need to be removed. If $t \leq U < n-t$, we remove $n-t-U$ vertices of R. If $U = t-\ell$ where $1 \leq \ell \leq t$, we remove the entire R and the 2ℓ vertices of ℓ arbitrarily chosen edges from M. All these choices are optimal, as follows from the proof concerning most vital nodes. □

We show in the following that these problems become NP-hard in the weighted case. The following notion will be of essence.

Definition 1. *Let $G = (V, E)$ be an undirected graph. The* bipartite incidence graph *of G is the bipartite graph H whose vertex set is $V \cup E$ and there is an edge in H between $v \in V$ and $e \in E$ if and only if e is incident to v in G.*

Theorem 3. *k* Most Vital Nodes Independent Set *and* Min Node Blocker Independent Set *are strongly* NP*-hard even for bipartite graphs.*

Proof. We first prove hardness for k Most Vital Nodes Independent Set. Let $G = (V, E)$ be an instance of the decision problem associated to Independent Set with n vertices and m edges, and an integer ℓ; and let H denote the bipartite incidence graph of G. The construction of H from G requires linear time only. Each vertex of E in H has weight 1 and each vertex of V in H has weight n^2. Due to this rather unbalanced weighting, the unique maximum-weight independent set in H is V; i.e., $\alpha(H) = n^3$.

We show in the following that if there is an independent set of size at least ℓ in G then H contains a set S of ℓ vertices such that $\alpha(H-S) = (n-\ell)n^2$, and otherwise removing any subset S of ℓ vertices from H, we have $\alpha(H-S) \geq (n-\ell)n^2+1$. Since vertices from V have weight n^2 and those from E have weight 1, in order to have a maximum-weight independent set as small as possible after removing a set S of size ℓ, S has to be included in V.

If G contains an independent set S of size ℓ, then removing S from the vertex set of H, we obtain a graph whose maximum-weight independent set is $V \setminus S$. This set has weight $(n-\ell)n^2$.

If G contains no independent set of size ℓ, then any $S \subset V$ of size ℓ contains at least an edge $e \in E$ in G, and this e in H is nonadjacent to the entire $V \setminus S$. Thus, when we remove any set S of ℓ vertices from H, $\alpha(H-S) \geq (n-\ell)n^2+1$.

Due to Remark 1, MIN NODE BLOCKER INDEPENDENT SET is also strongly NP-hard. □

We are going to prove an approximation hardness result, too. In the reduction, the following problem will be used.

MAX k VERTEX COVER
Input: A graph $G = (V, E)$ with $k \leq |V|$.
Output: The maximum number of edges in G that can be covered by a subset $V' \subseteq V$ of cardinality k.

MAX k VERTEX COVER-B is the version of MAX k VERTEX COVER where the maximum degree of the graph is at most B.

We shall apply the following version of some known results.

Lemma 1. *For appropriately chosen B,* MAX $n/2$ VERTEX COVER-B *has no ptas on graphs $G = (V, E)$ with $m = \Theta(n)$ and $\alpha(G) = \tau(G) = n/2$, where $n = |V|$ and $m = |E|$, unless P=NP.*

Proof. An approximation algorithm for VERTEX COVER on graphs with $\tau(G) \geq |V(G)|/2$ is also an approximation algorithm with the same ratio for general instances of VERTEX COVER [11]. Using the APX-hardness of VERTEX COVER-B [12] and the gap reduction from VERTEX COVER-B to MAX k VERTEX COVER-B [13] for $k \geq n/2$, we conclude that MAX k VERTEX COVER-B has no ptas on graphs with n vertices when $k \geq \tau(G) \geq n/2$. We can reduce this last problem to the same problem on instances with $k \geq \tau = n/2$ by inserting $2\tau - n$ isolated vertices. Moreover, these last instances are reducible to instances where $k = 2\tau = n/2$ by inserting $k - \tau$ isolated edges. □

We extract the key points of the reduction in the following lemma on independent sets.

Lemma 2. *Let $G = (V, E)$ be a graph with n vertices and m edges, and let H be the bipartite incidence graph of G. Then the following properties are valid.*

(*a*) *Suppose that G has maximum degree at most B, and the weights in H are $w_v = B + 1$ for all $v \in V$ and $w_e = 1$ for all $e \in E$. Then, for any $V' \subset V$ and any independent set S disjoint from V' in H, there exists an independent set S' of H such that $w(S') \geq w(S)$ and $S' \cap V = V \setminus V'$. Thus, if S' is maximal, then*

$$S' = (V \setminus V') \cup \{e \in E \mid e \subset V'\}$$

and, in particular, $\alpha(H - V') \geq (B + 1) \cdot (n - |V'|)$.

(*b*) *Under the conditions of (a), for any $V' \subset V \cup E$ with $|V'| < |V|$ there exists a $V'' \subset V$ such that $|V''| = |V'|$ and the maximum weight of an independent set in $H - V''$ is not larger than that in $H - V'$. As a consequence,*

$$\alpha(H - V') \geq \alpha(H - V'') = (B + 1) \cdot (n - |V'|) + |\{e \in E \mid e \subset V''\}|.$$

Moreover, the set V'' can be found efficiently.

Proof. (a) If S contains all vertices of $V \setminus V'$, then we have nothing to prove. Otherwise we modify S step by step, keeping it independent and not decreasing its value, until it contains the entire $V \setminus V'$. Hence, assume that $v \in V$ is a vertex such that $v \notin V' \cup S$. If v has no neighbor in $S \cap E$, then $S \cup \{v\}$ is a proper extension. Suppose that this is not the case; i.e., there is an edge $e \in E \cap S$ such that $v \in e$. We now modify S to $(S \setminus N_H(v)) \cup \{v\}$, where $N_H(v)$ denotes the set of vertices adjacent to v in H, that is the set of edges incident to v in G. In this way we have removed at most B neighbors of v from S, each of weight 1, and inserted v of weight $B+1$, hence the total weight of the modified set is at least $w(S)$. Moreover, the set remains independent because all neighbors of v have been removed. Thus, after $|(V \setminus V') \setminus S|$ steps, the required set S' is obtained.

(b) If $V' \subset V$, then $V'' = V'$ is a proper choice. Hence suppose $V' \cap E \neq \emptyset$. Let us introduce the notation $n' = |V' \cap V|$, $m' = |E(G[V' \cap V]) \setminus (V' \cap E)|$. By (a) we see that $\alpha(H - V') = (B+1) \cdot (n - n') + m'$ holds. Choose $e \in V' \cap E$ and $v \in V \setminus V'$, and modify V' to the set $(V' \setminus \{e\}) \cup \{v\}$. This keeps cardinality unchanged, while the first term $(B+1) \cdot (n - n')$ decreases by precisely $B+1$. Moreover, since G has maximum degree at most B, the second term m' can increase by at most B when we insert v into the set, and can further increase by at most 1 when we omit e. Thus, sum does not increase. Repeatedly eliminating all $e \in E$ from V', the required V'' is obtained. Then (a) implies that the independent set of maximum weight in $H - V''$ consists of all $v \notin V''$ and all $e \subset V''$. □

Theorem 4. k Most Vital Nodes Independent Set *has no ptas even for bipartite graphs if $P \neq NP$.*

Proof. We prove the non-existence of a ptas for $k = n/2$, constructing an L-reduction from Max $n/2$ Vertex Cover-B to $n/2$ Most Vital Nodes Independent Set, where instances of the former problem are restricted to graphs G of maximum degree at most B and also satisfying $\alpha(G) = \tau(G) = n/2$. In this case, let H denote the bipartite incidence graph of the input graph $G = (V, E)$, the latter having n vertices and m edges. The vertices of H have weight $w_v = B + 1$ for all $v \in V$ and $w_e = 1$ for all $e \in E$.

Consider first an optimum solution V' in G. As $\tau(G) = n/2$ has been assumed, $opt_1 = m$ holds and V' covers all edges of G. Then removing $V \setminus V'$ from the vertex set of H, we obtain a graph in which the maximum weight of an independent set is $((B+1)/2) \cdot n$, as implied by part (a) of Lemma 2. On the other hand, parts (a) and (b) together yield that after the removal of any $n/2$ vertices from H, there always remains an independent set of at least that large weight, thus

$$opt_2 = \frac{B+1}{2} \cdot n \leq (B+1) \cdot opt_1,$$

the upper bound being valid since $opt_1 \geq n/2$ surely holds by the assumption $\tau(G) = n/2$.

Consider now any subset V' of $n/2$ vertices in H, and denote $val_2 = \alpha(H - V')$. Now we apply part (b) of Lemma 2 to obtain an appropriate set V'' of $n/2$ vertices, which is a subset of V. We view $V \setminus V''$ as a solution in G and denote its value by val_1. In this way we obtain

$$val_2 - opt_2 \geq \alpha(H - V'') - opt_2 = ((B+1) \cdot (n - |V''|) + |E(G[V''])|) - \frac{B+1}{2} \cdot n$$

$$= |E(G[V''])| = opt_1 - val_1,$$

the last equation being valid because $opt_1 = m$ and $E(G[V''])$ is precisely the set of edges not covered by the vertices of $V \setminus V''$. This completes the proof of the theorem. □

4 Graph classes related to tree structures

In this section we consider graph classes representable over tree structures, and prove that they admit algorithms solving the considered problems in polynomial time. Efficient solvability for the graph classes in the first two subsections are implied by the results of the third subsection, too, but the methods for the former are simpler.

4.1 Trees

Theorem 5. *k Most Vital Nodes Independent Set is polynomial on trees. On trees of order n the algorithm runs in $O(nk^2)$ time, for any $k \geq 1$.*

Proof. Our general approach is to find not only a set of k most vital nodes but simultaneously also the value of a corresponding largest independent set. For this purpose we view the input as a *rooted* tree with an arbitrarily chosen root, and organize computation according to a postorder traversal.

Consider any tree T with vertices $v_1, \ldots, v_n$. Each vertex v_i can have three positions in a solution, that we shall denote by marks $+, -, 0$ as follows:

- '+' means that v_i is selected into an independent set;
- '−' means that v_i is selected for deletion;
- '0' means that v_i is none of the above two types.

In a solution exactly k marks '−' have to occur.

The subtree rooted in v_i is denoted by T_i. For each $i = 1, \ldots, n$, each $* \in \{+, -, 0\}$, and each $j = 0, 1, \ldots, k$, a value $z_i(j, *)$ will be computed. This $z_i(j, *)$ represents the minimum achievable weight of a largest independent set on T_i under the conditions that *exactly* j vertices are removed from T_i and v_i has mark $*$. For the recursive computation the children of v_i with degree d will be denoted by $v_{i_1}, \ldots, v_{i_d}$. We traverse T in postorder and apply dynamic programming.

Recursion. If v_i is marked '+', then all its children must have '−' or '0', since otherwise two vertices selected for the independent set would be adjacent. Moreover, $z_i(j, *)$ requires that the total number of vertices marked '−' should be

exactly j in T_i. On the other hand, we have one and only one way to make the final result as small as possible: decide which of the vertices should be marked with '$-$'. Once this has been decided, the distribution of '$+$' and '0' positions aims at maximizing the total weight of '$+$'. This leads to the following general recursions:

$$z_i(j,+) = w_i + \min_{\substack{j_1,\ldots,j_d\geq 0\\ j_1+\ldots+j_d=j}} \sum_{\ell=1}^{d} \min\left(z_{i_\ell}(j_\ell,-), z_{i_\ell}(j_\ell,0)\right),$$

$$z_i(j,-) = \min_{\substack{j_1,\ldots,j_d\geq 0\\ j_1+\ldots+j_d=j-1}} \sum_{\ell=1}^{d} \min\left(z_{i_\ell}(j_\ell,-),\, \max\left(z_{i_\ell}(j_\ell,+), z_{i_\ell}(j_\ell,0)\right)\right),$$

$$z_i(j,0) = \min_{\substack{j_1,\ldots,j_d\geq 0\\ j_1+\ldots+j_d=j}} \sum_{\ell=1}^{d} \min\left(z_{i_\ell}(j_\ell,-),\, \max\left(z_{i_\ell}(j_\ell,+), z_{i_\ell}(j_\ell,0)\right)\right),$$

For a leaf v_i we clearly have $z_i(0,+) = w_i$ and $z_i(1,-) = z_i(0,0) = 0$. Further, to indicate that all other combinations of $j \in \{0,1,\ldots,k\}$ and $* \in \{+,-,0\}$ are infeasible, we set a dummy symbol $z_i(j,*) = \mathsf{NIL}$ for them. In the recursive step, terms with value NIL on the right-hand side are neglected, except when all terms are the same, and in this case we define $z_i(j,*) = \mathsf{NIL}$, too.

Finding an optimal solution. Assuming that T has root v_{i_0}, after the removal of k properly chosen vertices the smallest possible value of α is

$$\min\left(z_{i_0}(k,-),\, \max\left(z_{i_0}(k,+), z_{i_0}(k,0)\right)\right).$$

In fact, inserting a new vertex v_0 with weight $w_0 = 0$ as new root and parent for v_{i_0} does not change the optimum, and then we would have $z_0(k,+) \leq opt = z_0(k,0) \leq z_0(k,-)$. A set of k most vital nodes can also be determined in $O(n)$ additional steps if we make a little more administration. At the recursive step for each $z_i(j,*)$ we register for each edge $v_iv_{i_\ell}$ the corresponding value of j_ℓ in the optimal distribution $(j_1,\ldots,j_d)$ for j and also the mark $* \in \{+,-,0\}$ of i_ℓ which gave the optimum for v_i. Once these data are available for all v_i and all pairs $(j,*)$, we can traverse T in preorder and select the vertices having '$-$' mark for the most vital set.

Efficient implementation. The key point is to find in polynomial time a best distribution $(j_1,\ldots,j_d)$ for the 'max' and 'min' functions acting on the sums. This can be done, despite that the number of possibilities can even be exponential if d is proportional to n.

If $d = 2$ then we have at most $j+1$ combinations of feasible pairs j_1, j_2. Hence, optimal choice can be made in $O(k)$ steps for any one particular j, and in $O(k^2)$ steps for all $0 \leq j \leq k$. If d is larger, we can split the children of v_i into two sets of (nearly) equal size, $\{v_\ell \mid 1 \leq \ell \leq \lfloor d/2 \rfloor\}$ and $\{v_\ell \mid \lfloor d/2 \rfloor + 1 \leq \ell \leq d\}$, make all computation in them separately, and then combine the results for v_i. (Splitting corresponds to inserting a 'supernode' above each of the two sets,

which has weight zero and becomes a virtual child of v_i.) This requires $d-1$ rounds for v_i. Since T is a tree, those $d-1$ sum up to $n-2$, thus the overall running time is $O((k^2+1)n)$, and never exceeds $O(n^3)$. (Here '+1' is needed for $k=0$.) Note that there are no 'hidden large constants' in the 'O' notation. □

Theorem 6. Min Node Blocker Independent Set *is polynomial on trees. On trees with n vertices the algorithm runs in $O(n^3 \log\log n)$ time.*

Proof. The above algorithm in one iteration for any $1 \leq v \leq n$ runs in $O(v^2 n) = O(n^3)$ time. Hence, using Theorem 1, finding the smallest k for which the solution has value at most U takes total running time $O(n^3 \log\log n)$. □

Remark 2. The algorithm proposed in Theorem 5 solves the k Most Vital Nodes Independent Set problem on *paths* in $O(kn)$ time. In fact, in the general time bound $O(nk^2)$ for trees, the factor k^2 occurs due to the presence of vertices with more than one child. This observation implies further that the algorithm proposed in Theorem 6 solves Min Node Blocker Independent Set on paths in $O(n^2 \log\log n)$ time.

4.2 Cycles

Theorem 7. k Most Vital Nodes Independent Set *is polynomial on cycles. On cycles of order n the algorithm runs in $O(kn^2)$ time, for any $k \geq 1$.*

Proof. Let $S^* = \{v_1, \ldots, v_r\} \subset V$ be a maximum-weight independent set of a given cycle $C = (V, E)$. An optimal solution $V' \subset V$ of k Most Vital Nodes Independent Set must contain at least one node of S^*, since otherwise $\alpha(C - V')$ is not smaller than $\alpha(C)$. Thus, for each $v_j \in S^*$, $j = 1, \ldots, r$, we determine the $k-1$ further nodes to remove in the resulting path as follows. We delete v_j from C and determine a maximum-weight independent set in the resulting path $C - v_j$ by applying the algorithm given in Theorem 5 in order to find an optimal solution $R_j^* \subset V \setminus \{v_j\}$ of $k-1$ Most Vital Nodes Independent Set on the path $C - v_j$. Then, an optimal solution for k Most Vital Nodes Independent Set on C is $R_\ell^* \cup \{v_\ell\}$ such that $\alpha(C - v_\ell - R_\ell^*) = \min_{1 \leq j \leq r} \alpha(C - v_j - R_j^*)$. If the root is chosen to be an endpoint of the path, the complexity of the algorithm given in Theorem 5 for path $C - v_j$ is $O(kn)$. Since $|S^*| \leq n$, in this way k Most Vital Nodes Independent Set is solved in $O(kn^2)$. □

Theorem 8. Min Node Blocker Independent Set *is polynomial on cycles. On cycles of order n the algorithm runs in $O(n^3 \log\log n)$ time.*

Proof. The theorem follows from Theorem 7 and Theorem 1. □

4.3 Graphs of bounded treewidth

A *tree decomposition* of a graph $G = (V, E)$ without isolated vertices is a pair $(T, \mathcal{X})$ where

- $T = (X, F)$ is a tree graph with a set $X = \{x_1, \ldots, x_m\}$ of *nodes* and a set F of *lines*;
- $\mathcal{X} = \{X_1, \ldots, X_m\}$ is a set system over V (i.e., over the vertex set of G), where each X_q is associated with node x_q of T;
- each edge $v_i v_j \in E$ of G is contained in at least one X_q for some $1 \leq q \leq m$;
- for any $v_i \in V$, if $v_i \in X_{q'}$ and $v_i \in X_{q''}$, then $v_i \in X_q$ for all q such that x_q lies on the $x_{q'}$–$x_{q''}$ path in T.

The width of $(T, \mathcal{X})$ is $\max_{1 \leq q \leq m} |X_q| - 1$, and the *treewidth* of G, denoted by $tw(G)$, is the smallest integer t for which G admits a tree decomposition of width t. For undefined details on tree decomposition we refer to [9].

Theorem 9. *k Most Vital Nodes Independent Set is polynomial on bounded treewidth graphs. On graphs of order n, the algorithm runs in $O(nk^2)$ time for any $k \geq 1$.*

Due to space limitation, the proof of this result is omitted and will appear in the extended version of the paper.

Theorem 10. *Min Node Blocker Independent Set is polynomial on bounded treewidth graphs. On graphs of order n the algorithm runs in $O(n^3 \log\log n)$ time.*

Proof. The result follows from Theorem 9 and Theorem 1. □

4.4 Cographs

To each cograph G with n vertices, we can associate a rooted tree T, called the *cotree* of G. Leaves of T correspond to vertices of the graph G and internal nodes of T are labeled with either '$\cup$' (union-node) or '$\times$' (join-node). A subtree rooted at node '$\cup$' corresponds to the union of the subgraphs defined by the children of that node, and a subtree rooted at node '$\times$' corresponds to the join of the subgraphs defined by the children of that node; that is, we add an edge between every two vertices corresponding to leaves in different subtrees. Cographs can be recognized in linear time and the cotree representation can be obtained efficiently [4, 6]. Moreover, this cotree can easily be transformed in linear time to a binary cotree with $O(n)$ nodes.

Theorem 11. *k Most Vital Nodes Independent Set is polynomial on cographs. On cographs of order n, the algorithm runs in $O(nk^2)$ time, for any $k \geq 1$.*

Proof. Consider a cograph G with n vertices $v_1, \ldots, v_n$. Given a binary cotree representation T of G, we show in the following how to solve the k Most Vital Nodes Independent Set using dynamic programming.

Let $x_1, \ldots, x_t$ be the nodes of T where x_r is its root and t is in $O(n)$. For $i = 1, \ldots, t$, denote by T_i the subtree rooted at x_i, G_i the subgraph induced by the vertices corresponding to the leaves of T_i, and V_i these vertices.

Recursion. We associate a $(k+1)$-vector to each node x_i of T, $i = 1, \ldots, t$. In the following, a $(k+1)$-*vector* is simply call a vector. For each i and each $j = 0, 1, \ldots, k$, we compute $z_i(j)$ that is the minimum weight of a maximum independent set on G_i where exactly j vertices are removed from G_i. These vectors are computed "bottom-up" in the cotree. So, we start by computing vectors of leaves and after that the vector of an internal node if the vectors of its two children are already computed.

Given a node x_i of the cotree, the corresponding vector is obtained as follows:

- If x_i is a union-node with two children x_ℓ and x_r, we have no edges between G_ℓ and G_r. Then the maximum independent set in G_i is the union of those in G_ℓ and G_r. Thus, since we want to find a maximum-weight independent set as small as possible, the best choice is given by $z_i(j) = \min_{j_1+j_2=j} (z_\ell(j_1) + z_r(j_2))$.
- If x_i is a join-node with two children x_ℓ and x_r, every vertex in V_ℓ is adjacent to every vertex in V_r. Then each independent set in G_i is entirely contained either in G_ℓ or in G_r. So, $z_i(j) = \min_{j_1+j_2=j} (\max(z_\ell(j_1), z_r(j_2)))$.
- If x_i is a leaf then $z_i(0) = w_i$, $z_i(1) = 0$, and $z_i(j) = \mathsf{NIL}$ for $j = 2, \ldots, k$ which means that the latter configurations are infeasible. In the recursive step, terms with value NIL on the right-hand side are neglected, except when all terms are the same, and in this case we define $z_i(j) = \mathsf{NIL}$, too.

Finding an optimal solution. An optimal solution is obtained at the root x_r of T and its weight is equal to $z_r(k)$. Moreover, an optimal set of k removed vertices can be computed step by step in the recursion. Indeed, let $S_i^-(j)$ be the subset of j removed vertices in G_i. For a leaf x_i we have $S_i^-(0) = \emptyset$, $S_i^-(1) = \{v_i\}$ and $S_i^-(j) = \emptyset$ for $j = 2, \ldots, k$. For a union-node or a join-node x_i with two children x_ℓ and x_r, recursion yields $S_i^-(j) = S_\ell^-(j_1^*) \cup S_r^-(j_2^*)$ where j_1^* and j_2^* are the indices that realize the minimum for $z_i(j)$.

Time analysis. For k Most Vital Nodes Independent Set, vectors are computed in $O(k)$ for each leaf and in $O(k^2)$ for each union-node and each join-node. Since $t = O(n)$, the algorithm runs in $O(nk^2)$. □

Theorem 12. Min Node Blocker Independent Set *is polynomial on cographs. On cographs of order n, the algorithm runs in $O(n^3 \log\log n)$ time.*

Proof. The theorem follows from Theorem 11 and Theorem 1. □

5 Conclusion

In this paper we studied the complexity of the k most vital nodes and min node blocker versions of the maximum-weight independent set problem. While maximum-weight independent set is polynomial on bipartite graphs, the k most vital nodes and min node blocker versions become NP-hard. Nevertheless the unweighted versions remain polynomial on bipartite graphs. In a graph, a maximum-weight independent set is the complementary set of a minimum-weight vertex

cover. In sharp contrast to this, however, concerning the k most vital nodes or min node blocker versions an optimum solution for maximum-weight independent set may be substantially different from an optimum solution for minimum-weight vertex cover. Our results on the latter will be included in an extended paper. We show in this paper that the k most vital nodes version has no ptas. An interesting open question would be to establish other positive and negative results concerning the approximability of these versions. In particular it remains open to decide weather min node blocker on bipartite graphs has a ptas.

Another interesting perspective is to study the complexity of the k most vital nodes and min node blocker versions of the maximum-weight independent set problem for graphs of bounded cliquewidth and graphs of bounded NLC-width, that generalize cographs. Moreover, the study of the complexity and the approximation of these versions for further classes of graphs for which maximum-weight independent set and minimum-weight vertex cover are polynomial is also of interest.

References

1. A. Bar-Noy, S. Khuller, and B. Schieber, The complexity of finding most vital arcs and nodes, *Technical Report CS-TR-3539, Department of Computer Science, University of Maryland*, 1995.
2. C. Bazgan, S. Toubaline, D. Vanderpooten, Détermination des éléments les plus vitaux pour le problème d'affectation, *Actes de la 10ème Conférence de la Société Française de Recherche Opérationelle et d'Aide à la Décision (ROADEF 2010)*, 2010.
3. H. L. Bodlaender, A linear time algorithm for finding tree-decompositions of small treewidth, *SIAM Journal on Computing*, 25, 1305–1317, 1996.
4. D. G. Corneil, Y. Perl, and L.K. Stewart, A linear recognition algorithm for cographs, *SIAM Journal on Computing*, 14(4), 926–934, 1985.
5. G. N. Frederickson and R. Solis-Oba, Increasing the weight of minimum spanning trees, *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 1996)*, 539–546, 1996.
6. M. Habib and C. Paul, A simple linear time algorithm for cograph recognition, *Discrete Applied Mathematics*, 145(2), 183–197, 2005.
7. R. Hassin. Approximation schemes for the restricted shortest path. *Mathematics of Operations Research*, 17(1), 36–42, 1992.
8. L. Khachiyan, E. Boros, K. Borys, K. Elbassioni, V. Gurvich, G. Rudolf, and J. Zhao, On short paths interdiction problems: total and node-wise limited interdiction, *Theory of Computing Systems*, 43(2), 204–233, 2008.
9. T. Kloks, Treewidth, computations and approximations, *Lecture Notes in Computer Science*, 842, 1994.
10. D. Kőnig, Graphs and matrices, *Math. Fiz. Lapok* 38, 116–119, 1931 (in Hungarian).
11. G. L. Nemhauser and L. E. Trotter, Vertex packing: structural properties and algorithms, *Mathematical Programming*, 8, 232–248, 1975.
12. C. Papadimitriou and M. Yannakakis, Optimization, approximation and complexity classes, *Journal of Computer and System Science*, 43(3), 425–440, 1991.

13. E. Petrank, The hardness of approximation: gap location, *Computational Complexity* 4, 133–157, 1994.
14. B. Ries, C. Bentz, C. Picouleau, D. de Werra, M. Costa, and R. Zenklusen, Blockers and Transversals in some subclasses of bipartite graphs: When caterpillars are dancing on a grid, *Discrete Mathematics*, 310(1), 132–146, 2010.
15. H. Shen, Finding the k most vital edges with respect to minimum spanning tree, *Acta Informatica*, 36, 405–424, 1999.
16. R. Zenklusen, B. Ries, C. Picouleau, D. de Werra, M. Costa and C. Bentz, Blockers and Transversals, *Discrete Mathematics*, 309(13), 4306–4314, 2009.
17. R. K. Wood, Deterministic network interdiction, *Mathematical and Computer Modeling*, 17(2), 1–18, 1993.