Mathematical Programming (2023) 199:215–249
https://doi.org/10.1007/s10107-022-01824-5

FULL LENGTH PAPER

Series A

Vertex downgrading to minimize connectivity

Hassene Aissi[1] · Da Qi Chen[2] · R. Ravi[3]

Received: 25 September 2020 / Accepted: 19 April 2022 / Published online: 3 June 2022

Abstract

We consider the problem of interdicting a directed graph by deleting nodes with the goal of minimizing the local edge connectivity of the remaining graph from a given source to a sink. We introduce and study a general downgrading variant of the interdiction problem where the capacity of an arc is a function of the subset of its endpoints that are downgraded, and the goal is to minimize the downgraded capacity of a minimum source-sink cut subject to a node downgrading budget. This models the case when both ends of an arc must be downgraded to remove it, for example. For this generalization, we provide a bicriteria (4, 2)-approximation that downgrades nodes with total weight at most 4 times the budget and provides a solution where the downgraded connectivity from the source to the sink is at most 2 times that in an optimal solution. We accomplish this with an LP relaxation and rounding using a ball-growing algorithm based on the LP values. Furthermore, we show that other bicriteria approximations exist where one can worsen the approximation factor for one of the costs in order to improve the other. We further generalize the downgrading problem to one where each vertex can be downgraded to one of k levels, and the arc capacities are functions of the pairs of levels to which its ends are downgraded. We

This material is based upon work supported in part by the U. S. Office of Naval Research under Award Number N00014-21-1-2243 (N00014-18-1-2099) and the Air Force Office of Scientific Research under Award Number FA9550-20-1-0080.

An extended abstract of this paper appeared in the 17th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2020).

✉ Da Qi Chen
daqic@andrew.cmu.edu

Hassene Aissi
aissi@lamsade.dauphine.edu

R. Ravi
ravi@cmu.edu

1 Paris Dauphine University, Paris, France

2 Carnegie Mellon University, Pittsburgh, USA

3 Tepper School of Business, Carnegie Mellon University, Pittsburgh, USA

generalize our LP rounding to get a $(4k, 4k)$-approximation for this case. Trade-offs between the two approximation ratios similar to the two-level case also exist for the generalized problem. By transferring node values to edge values, we also derive new bicriteria approximation results for the vertex interdiction versions of the multiway cut problem in digraphs and multicut problems in undirected graphs.

Keywords Vertex interdiction · Vertex downgrading · Network interdiction · Approximation algorithm

Mathematics Subject Classification 90C27

1 Introduction

Interdiction problems arise in evaluating the robustness of infrastructure and networks. For an optimization problem on a graph, the interdiction problem can be formulated as a game consisting of two players: an attacker and a defender. Every edge/vertex of the graph has an associated interdiction cost and the attacker interdicts the network by modifying the edges/vertices subject to a budget constraint. The defender solves the problem on the modified graph. The goal of the attacker is to hamper the defender as much as possible. Ford and Fulkerson initiated the study of interdiction problems with the maximum flow/minimum cut theorem [7, 17, 24]. Other examples of interdiction problems include matchings [27], minimum spanning trees [20, 30], shortest paths [13, 18], st-flows [23, 26, 28] and global minimum cuts [6, 29].

Most of the interdiction literature today involves the interdiction of edges while the study of interdicting vertices has received less attention (e.g. [27, 28]). The various applications for these interdiction problems, including drug interdiction, hospital infection control, and protecting electrical grids or other military installations against terrorist attacks, all naturally motivate the study of the vertex interdiction variant. In this paper, we focus on vertex interdiction problems related to the minimum st-cut (which is equal to the maximum st-flow and hence also termed network flow interdiction or network interdiction in the literature).

For st-cut vertex interdiction problems, the setup is as follows. Consider a directed graph $G = (V(G), A(G))$ with n vertices, m arcs, an arc cost function $c : A(G) \to \mathbb{N}$, and an interdiction cost function $r : V(G) \setminus \{s, t\} \to \mathbb{N}$ defined on the set of vertices $V(G) \setminus \{s, t\}$. A set of arcs $F \subseteq A(G)$ is an st-cut if $G\backslash F$ no longer contains a directed path from s to t. Define the cost of F as $c(F) = \Sigma_{e \in F} c(e)$. For any subset of vertices $X \subseteq V(G) \setminus \{s, t\}$, we denote its interdiction cost by $r(X) = \sum_{v \in X} r(v)$. Let $\lambda_{st}(G\backslash X)$ denote the cost of a minimum st cut in the graph $G\backslash X$.

Problem 1 Weighted Network Vertex Interdiction Problem (WNVIP) and its special cases. Given two specific vertices s (source) and t (sink) in $V(G)$ and interdiction budget $b \in \mathbb{N}$, the Weighted Network Vertex Interdiction Problem **(WNVIP)** asks to find an interdicting vertex set $X^* \subseteq V(G) \setminus \{s, t\}$ such that $\sum_{v \in X^*} r(v) \leq b$ and $\lambda_{st}(G\backslash X^*)$ is minimum. The special case of WNVIP where all the interdiction costs are one will be termed **NVIP**, while the further special case when even the arc costs are one will be termed **NVIP with unit costs**.

In this paper, we define and study a generalization of the network flow interdiction problem in digraphs that we call **vertex downgrading**. Since interdicting vertices can be viewed as attacking a network at its vertices, it is natural to consider a variant where attacking a node does not destroy it completely but partially weakens its structural integrity. In terms of minimum st-cuts, one interpretation could be that whenever a vertex is interdicted, instead of removing it from the network we partially reduce the cost of its incident arcs. In this context, we say that a vertex is *downgraded*. Specifically, consider a directed graph $G = (V(G), A(G))$ and a downgrading cost $r : V(G) \setminus \{s, t\} \to \mathbb{N}$. For every arc $e = uv \in A(G)$, there exist four associated nonegative costs $c_e, c_{eu}, c_{ev}, c_{euv}$, respectively representing the cost of arc e if 1) neither $\{u, v\}$ are downgraded, 2) only u is downgraded, 3) only v is downgraded, and 4) both $\{u, v\}$ are downgraded. Note that these cost functions are independent of each other so downgrading vertex v might affect each of its incident arcs differently. However, we do impose the following conditions: $c_e \geq c_{eu} \geq c_{euv}$ and $c_e \geq c_{ev} \geq c_{euv}$. These inequalities are natural to impose since the more endpoints of an arc are downgraded, the lower the resulting arc should cost. Given a downgrading set $Y \subseteq V(G) \setminus \{s, t\}$, define $c^Y : A(G) \to \mathbb{R}_+$ to be the arc cost function representing the cost of cutting e after downgrading Y.

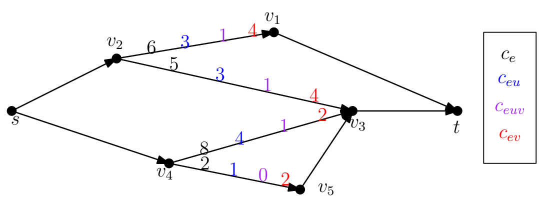
	$u, v \notin Y$	$u \in Y, v \notin Y$	$u \notin Y, v \in Y$	$u, v \in Y$
$c^Y(e) =$	c_e	c_{eu}	c_{ev}	c_{euv}

Given a set of arcs $F \subseteq A(G)$, we define $c^Y(F) = \Sigma_{e \in F} c^Y(e)$.

Problem 2 Network Vertex Downgrading Problem (NVDP). Let $G = (V(G), A(G))$ be a directed graph with a source s and a sink t. For every arc $e = uv$, we are given non-negative costs $c_e, c_{eu}, c_{ev}, c_{euv}$ as defined above. Given a (downgrading) budget b, find a set $Y \subseteq V(G) \setminus \{s, t\}$ and an st-cut $F \subseteq A(G)$ such that $\Sigma_{v \in Y} r(v) \leq b$ and minimizes $c^Y(F)$.

Figure 1 depicts an instance of NVDP. Four numbers, representing $c_e, c_{eu}, c_{euv}, c_{ev}$ respectively, are given in sequence above certain arcs. For simplicity, assume that the remaining arcs have large costs regardless of which endpoints were downgraded and thus will not be part of the final cut. If we downgrade $\{v_3, v_4\}$, the costs of the arcs v_2v_1, v_2v_3, v_4v_3, v_4v_5 are 6, 4, 1, 1 respectively, producing a final cut of cost 12. Note that if the budget was 2, then the optimal set to downgrade is $\{v_2, v_4\}$, resulting in a cut of cost 11.

Fig. 1 Example of NVDP



While it is not immediately obvious as it is for WNVIP, we can still show that detecting a zero solution for NVDP is polynomial-time solvable.

Theorem 1 *Given an instance of NVDP on graph G with budget b, there exists a polynomial time algorithm to determine if there exists $Y \subseteq V(G)$ and an st-cut $F \subseteq A(G)$ such that $\Sigma_{v \in Y} r(v) \leq b$ and $c^Y(F) = 0$.*

First we present some useful reductions between the above problems.

1. In the NFI (Network Flow Interdiction) problem defined in [7], the given graph is undirected instead of directed and the adversary interdicts edges instead of vertices. The goal is to minimize the cost of the minimum st-cut after interdiction. NFI can be reduced to the undirected version of WNVIP (where the underlying graph is undirected). Simply subdivide every undirected edge $e = uv$ with a vertex v_e. The interdiction cost of v_e remains the same as the interdiction cost of e while all original vertices have an interdiction cost of ∞ (or a very large number). The cost of the edges uv_e, $v_e v$ are equal to the cost if the original edge e.
2. The undirected version of WNVIP can be reduced to the (directed) WNVIP by replacing every edge with two parallel arcs going in opposite directions. Each new arc has the same cost as the original edge.
3. WNVIP is a special case of NVDP with costs $c_{eu} = c_{ev} = c_{euv} = 0$ for all $e = uv$.

The first two observations above imply that any hardness result for NFI in [7] also applies to WNVIP. Based on the second observation, we prove our hardness results for the (more specific) undirected version of WNVIP. As a consequence of the third observation, all these hardness results also carry over to the more general NVDP.

Our work also studies the following further generalization of NVDP. Every vertex has k possible levels that it can be downgraded to by paying different downgrading costs. Every arc has a cutting cost depending on what level its endpoints were downgraded to. More precisely, for each level $0 \leq i, j \leq k$, let $r_i(v)$ be the interdiction cost to downgrade v to level i and let $c_{i,j}(e)$ be the cost of cutting arc $e = uv$ if u, v were downgraded to levels i, j respectively. We assume that $0 = r_0(v) \leq r_1(v) \leq ... \leq r_k(v)$ since higher levels of downgrading should cost more and $c_{i,j}(e) \geq c_{i',j'}(e)$ if $i \leq i'$, $j \leq j'$ since the more one downgrades, the easier it is to cut the incident arcs. Then, given a map $L : V(G) \rightarrow \{0, ..., k\}$, representing which level to downgrade each vertex to, one can talk about the cost of performing this downgrading: $r^L := \Sigma_{v \in V(G)} r_{L(v)}(v)$, and the cost of a cut F after downgrading according to L: $c^L(F) := \Sigma_{uv \in F} c_{L(u),L(v)}(uv)$. Now, we can formally define the most general problem we address.

Problem 3 **Network Vertex Leveling Downgrading Problem (NVLDP).** Let $G = (V(G), A(G))$ be a directed graph with a source s and a sink t. For every vertex v and $0 \leq i \leq k$, we have non-negative downgrading costs $r_i(v)$. For every arc $e = uv$ and levels $0 \leq i, j \leq k$, we are given non-negative cut costs $c_{i,j}(e)$. Given a (downgrading) budget b, find a map $L : V(G) \rightarrow \{0, ..., k\}$ and an st-cut $F \subseteq A(G)$ such that $r^L \leq b$ and minimizes $c^L(F)$.

Note that when $k = 1$ we have NVDP.

Definition 1 An (α, β) **bicriteria approximation** for the interdiction (or downgrading) problem returns a solution that violates the interdiction budget b by a factor of at most β and provides a final cut (in the interdicted graph) with cost at most α times the optimal cost of a minimum cut after interdiction with a budget b.

1.1 Related works

Chestnut and Zenklusen [7] study the network flow interdiction problem (NFI), which is the undirected and edge interdiction version of WNVIP. NFI is also known to be essentially equivalent to the budgeted minimum st cut problem [22]. NFI is also a recasting of the k-route st-cut problem [8, 16], where a minimum cost set of edges must be deleted to reduce the node or edge connectivity between s and t to be k. The results of Chestnut and Zenklusen, and Chuzhoy et al. [8] show that an $(\alpha, 1)$-approximation for WNVIP implies a $2(\alpha)^2$-approximation for the notorious Densest k-Subgraph (DkS) problem. The results of Chuzhoy et al. [8] (Theorem 1.9 and Appendix section B) also imply such a hardness for NVIP even with unit edge costs. For the directed version, WNVIP is equivalent to directed NFI (by subdividing arcs or splitting vertices). As noted in [28], there is a symmetry between the interdicting cost and the capacity on each arc and thus it is also hard to obtain a $(1, \beta)$-approximation for WNVP. Furthermore, Chuzhoy et al. [8] also show that there is no $(C, 1 + \gamma_C)$-bi-criteria approximation for WNVIP assuming Feige's Random κ-AND Hypothesis (for every C and sufficiently small constant γ_C). For example, under this hypothesis, they show hardness of $(\frac{11}{10} - \epsilon, \frac{25}{24} - \epsilon)$ approximation for WNVIP.

Chestnut and Zenklusen give a $2(n - 1)$ approximation algorithm for NFI for any graph with n vertices. In the special case where the graph is planar, Philips [23] gave an FPTAS and Zenklusen [28] extended it to handle the vertex interdiction case.

Burch et al. [3] give a $(1 + \varepsilon, 1)$, $(1, 1 + \frac{1}{\varepsilon})$ pseudo-approximation algorithm for NFI. Given any $\varepsilon > 0$, this algorithm returns either a $(1 + \epsilon)$-approximation, or a solution violating the budget by a factor of $1 + \frac{1}{\epsilon}$ but has a cut no more expensive than the optimal cost. However, we do not know which case occurs *a priori*. In this line of work, Chestnut and Zenklusen [6] have extended the technique of Burch et al. to derive pseudo-approximation algorithms for a larger class of NFI problems that have good LP descriptions (such as duals that are box-TDI). Chuzhoy et al. [8] provide an alternate proof of this result by subdividing edges with nodes of appropriate costs.

We summarize other related work on multiway and multicuts in the corresponding sections later.

1.2 Our contributions

1. We define and initiate the study of multi-level node downgrading problems by defining the Network Vertex Leveling Downgrading Problem (NVLDP) and provide the first results for it. This problem extends the study in [28] of the vertex interdiction problem so as to consider a richer set of interdiction functions.

2. For the downgrading variant NVDP, we show that the problem of detecting whether there exists a downgrading set that gives a zero cost cut can be solved in polynomial time (Sect. 2).
3. We design a new LP rounding approximation algorithm that provides a $((1+\epsilon)^2, (1+1/\epsilon)^2/2)$-approximation to NVDP for any $0 < \epsilon \leq 1$. Note that this provides a range of (α, β)-approximations where $1 < \alpha \leq 4$. When $\alpha \geq 4$, the same techniques provides a slightly weaker approximation guarantee of the form $(2(1+\epsilon), 1+1/\epsilon)$ where $\epsilon \geq 1$. We use a carefully constructed auxiliary graph so that the level-cut algorithm based on ball growing for showing integrality of st-cuts in digraphs (see e.g. [9]) can be adapted to choose nodes to downgrade and arcs to cut based on the LP solution (Sect. 3).
4. In Sect. 3.2, we present a simple reduction from NVDP to WNVIP. Combining with a result from Burch et al. [3], it gives a pseudorandom approximation where for any $\epsilon > 0$, the solution is either $(2(1+\epsilon), 1)$ or $(2, (1+1/\epsilon))$ but a priori, one cannot guarantee which approximation is attainable.
5. For the most general version NVLDP with k levels of downgrading each vertex and k^2 possible downgraded costs of cutting an edge, we generalize the LP rounding method for NVDP to give a $(2(1+\epsilon)k, 2(1+1/\epsilon)k)$-approximation for any $\epsilon > 0$. The direct extension of the NVDP rounding to this case only gives an $O(k^2)$ approximation. However, we exploit the sparsity properties of a vertex optimal solution to our LP formulation to improve this guarantee to match that for the case of $k = 1$. We have also provided a zero-detection scheme to determine if it is possible to downgrade within the budget to induce a zero-cost cut (Sect. 4, 5).
6. As noted before, many previous works showed hardness in obtaining a unicriterion approximation for WNVIP, which motivates the focus on finding bicriteria approximation results. We push the hardness result further to show that it is also "DkS hard" to obtain a $(1, \beta)$-approximation for NVIP and NVIP with unit costs even in *undirected* graphs. Note that this is in sharp contrast to the edge interdiction case. NFI in undirected graphs with unitary interdiction cost and unitary cut cost can be solved by first finding a minimum cut and then interdicting b edges in that cut [29] (Sect. 6).
7. Burch et al. [3] gave a polynomial time algorithm that finds a $(1+1/\epsilon, 1)$ or $(1, 1+\epsilon)$-approximation for any $\epsilon > 0$ for WNVIP in digraphs. This was reproved more directly by Chuzhoy et al. [8] by converting both interdiction and arc costs into costs on nodes. We show that this strategy can also be extended to give a simple $(4, 4(1+\epsilon))$-bicriteria approximation for the multiway cut generalization in directed graphs and a $(2(1+\epsilon)\ln k, 2(1+\epsilon)\ln k)$-approximation for the multicut vertex interdiction problem in undirected graphs, for any $\epsilon > 0$, where k is the number of terminal nodes in the multicut problem (Sect. 7).

2 Detecting zero in NVDP in polynomial time

In this section, we show that in a given instance of NVDP, one can detect in polynomial-time whether there exists nodes to downgrade such that the downgrading cost is less than the budget and the min cut after downgrading is zero, and hence prove Theorem 1.

In order to demonstrate the main idea of the proof, we first work on a special case of NVDP. Suppose for every arc $e = uv$, $c_e = c_{eu} = c_{ev} = 1$ and $c_{euv} = 0$. In other words, every arc is unit cost and requires the downgrading of both ends in order to reduce the cost down to zero. For every vertex $v \in V(G)$, we assume the interdiction cost $r(v) = 1$. We call this the **Double-Downgrading Network Problem** (DDNP). We first prove the following.

Lemma 1 *Given an instance of DDNP on graph G with budget b, there exist a polynomial time algorithm to determine if there exists $Y \subseteq V(G)$ and an st-cut $F \subseteq A(G)$ such that $|Y| \leq b$ and $c^Y(F) = 0$.*

Proof Let $X \subseteq V(G)$ be a minimum set of vertices to downgrade such that the resulting graph contains a cut of zero cost. Let F be the set of arcs in the graph induced by X (i.e., with both ends in X). Note that F are the only arcs with cost zero and hence F is an arc cut in G. Furthermore, since X is optimal, X is the set of vertices incident to F (i.e. there are no isolated vertices in the graph induced by X). Let V_s and V_t be the set of vertices in the $G \backslash F$ that is respectively reachable by s and can reach t.

Consider the graph G^2 where we add arc uw to G if there exists $w \in V(G)$ such that $uv, vw \in A(G)$. We claim that X is a vertex cut in G^2. Suppose there is an st path in $G^2 \setminus X$ where the first arc crossing over from V_s to V_t is uv. Note that any such $u \in V_s \setminus X$ and $v \in V_t \setminus X$ are distance 3 apart and hence, do not have an arc between them in G^2, a contradiction.

Given any vertex cut Y in G^2, we claim that downgrading Y in G creates an st-cut of zero cost, by deleting the arcs induced by Y from G. Suppose for a contradiction there is an st-path in G after downgrading Y and deleting the zero-cost arcs induced by Y. Then the path cannot have two consecutive nodes in Y. Let $y \in Y$ be a single node along the path with neighbors $y^-, y^+ \notin Y$. Note that $(y^-, y^+) \in G^2$, and shortcutting over all such single node occurrences from Y in the path gives us an st-path in $G^2 \setminus Y$, a contradiction.

This proves that a minimum size downgrading vertex set Y in G whose downgrading produces a zero-cost st-cut is also a minimum vertex-cut in G^2. Then, one can check if a zero-cut solution exists with budget b for DDNP by simply checking if the minimum vertex-cut in G^2 is at most b. □

Now, to prove Theorem 1, we have to slightly modify the graph G and the construction of G^2 in order to adapt to the various costs. Our goal is still to look for a minimum vertex cut in an auxiliary graph using $r(v)$ as vertex cost.

Proof of Theorem 1 Given an instance of NVDP on G with a budget b, vertex downgrading costs $r(v)$ for every vertex v and arc costs $c_e, c_{eu}, c_{ev}, c_{euv}$, consider the following auxiliary graph H. First, we delete any arc e where $c_e = 0$ since they are free to cut anyways. For every arc $e = uv$ where $c_{euv} > 0$, subdivide e with a vertex t_e and let $r(t_e) = \infty$. In some sense, since $c_e, c_{eu}, c_{ev} \geq c_{euv} > 0$, downgrading u, v cannot reduce the cost of e to zero. Then, we should never be allowed to touch the vertex t_e. Let T be the set of all newly-added subdivided vertices. To finish constructing H, our next step is to properly simulate H^2.

We classify arcs into five types based on which of its costs are zero. Note that we no longer have any arcs where $c_e = 0$. Let $A_0 := \{e = uv : c_{eu} = c_{ev} = c_{euv} = 0\}$,

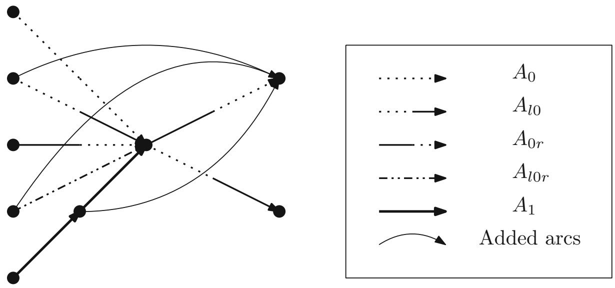


Fig. 2 Example of added arcs in H

the arcs where downgrading either ends reduce its cost to zero. Let $A_{l0} := \{e = uv : c_{eu} = c_{euv} = 0, c_{ev} > 0\}$, $A_{0r} := \{e = uv : c_{ev} = c_{euv} = 0, c_{eu} > 0\}$, $A_{l0r} := \{e = uv : c_{euv} = 0, c_{eu}, c_{ev} > 0\}$ respectively represent arcs that require the downgrading of its left tail, its right head, or both in order to reduce its cost to 0. Let A_1 be all remaining arcs, those incident to the newly subdivided vertex t_e. Now, for every path uvw of length two, we consider adding the arc uw based on the following rules (see Fig. 2 for an example of newly added arcs):

If $v \notin T$					
Add uw?	$vw \in$				
$uv \in$	A_0	A_{l0}	A_{0r}	A_{l0r}	A_1
A_0	No	No	No	No	No
A_{l0}	No	No	Yes	Yes	Yes
A_{0r}	No	No	No	No	No
A_{l0r}	No	No	Yes	Yes	Yes
A_1	No	No	Yes	Yes	Yes
If $v = t_e \in T$, do not add uw					

The idea is similar to the proof for DDNP: if $uv, vw \in A_{l0r}$, downgrading v is not enough to cut uv, vw for free. Thus we add arc uw to keep the connectivity. If $uv \in A_{0r}$, then downgrading v should reduce the cost of uv to 0. Thus, we do not want to bypass v by adding an arc uw. If $v = t_e \in T$, since $r(v)$ has high cost, we never cut it so we do not need to strengthen the connectivity by adding arcs uw.

Let (X, F) be a solution to NVDP where $\Sigma_{v \in X} r(v)$ is minimum, F is an st-cut and $c^X(F) = 0$. Let V_s be all vertices that can be reached from s in $G \backslash F$. We claim that X is a vertex cut in H. Suppose not and there exists an st-path in H and let uv be the first arc of the path leaving V_s. If $v = t_e \in T$, then arc $e \in F$, contradicting $c^X(F) = 0$. If $uv \in A(G)$, then $uv \in F$. Since $u, v \notin X$, $c^X(uv) > 0$, a contradiction. If uv is a newly added arc, then there exist $v' \in V(G)$ such that $uv'v$ is a path in G. By definition, $V_s \cap T = \emptyset$ so $u, v \notin T$. Then, there are only four cases where we add arc uv to create H. In all cases, downgrading v' does not reduce the cost of $uv', v'v$ to 0. Since at least one of $uv', v'v \in F$, it contradicts $c^X(F) = 0$.

Given a minimum vertex cut Y in H, we claim that downgrading Y in G creates an st-cut of zero cost. Note that $Y \cap T = \emptyset$ since any vertex in T is too expensive to cut. Suppose for a contradiction there is an st-path P that does not cross an arc with cost 0 after downgrading Y. Let P' be the corresponding path in H. If P contains two consecutive vertices $u, v \in Y$, then $c_{euv} > 0$ and it would have been subdivided. This implies there are no consecutive vertices of Y in P'. Let uvw be a segment of P' where $v \in Y$. Since downgrading v does not reduce its incident arcs to a cost of 0, it follows that $uv \in A_{l0} \cup A_{l0r} \cup A_1$ and $vw \in A_{0r} \cup A_{l0r} \cup A_1$. Then, it follows that $uw \in A(H)$. Then, every vertex $v \in Y \cap V(P')$ can be bypassed, a contradiction.

This implies that a minimum -weighted vertex cut in H is a downgrading set that creates a zero-cost cut in G. Then, by checking the min-vertex cut cost of H, we can determine whether a zero-solution exists for G with budget b. □

A similar zero-detection scheme exists for NVLDP where the constructed auxiliary graph is slightly more complicated. The idea is to replace each vertex with $k + 1$ different copies to simulate downgrading to different levels. The full proof is provided in Sect. 4.

3 Approximating network vertex downgrading problem (NVDP)

As an introduction and motivation to the LP model and techniques used to solve NVLDP, we focus in this section on the special case NVDP, where there is only one other level to downgrade each vertex to. Our main goal is to show the following theorem.

Theorem 2 *For NVDP, there exists a polynomial time algorithm that provides:*

1. *an* $((1+\epsilon)^2, (1+1/\epsilon)^2/2)$*-approximation for any* $0 < \epsilon < 1$,
2. *an* $2(1+\epsilon), (1+1/\epsilon)$*-approximation for any* $\epsilon \geq 1$.

3.1 LP relaxation and rounding

LP Model for Minimum st-cut. To formulate the NVDP as a LP, we begin with the following standard formulation of minimum st-cuts [14].

$$
\begin{aligned}
\min \quad & \sum_{e \in A(G)} c(e) x_e & & \\
\text{s.t.} \quad & d_v \leq d_u + x_{uv} & \forall uv \in A(G) & \quad (1)\\
& d_s = 0, d_t \geq 1 & & \\
& x_{uv} \geq 0 & \forall uv \in A(G) & \quad (2)
\end{aligned}
$$

An integer solution of this problem can be seen as setting d to be 0 for nodes in the s shore and 1 for nodes in the t shore of the cut. Constraints (1) then insist that the x-value for arcs crossing the cut to be set to 1. The potential d_v at node v can also be interpreted as a distance label starting from s and using the nonnegative values

x_{uv} as distances on the arcs. Any optimal solution of the above LP can be rounded to an optimal integer solution of no greater value by using the x-values on the arcs as lengths, growing a ball around s, and cutting it at a random threshold between 0 and the distance to t (which is 1 in this case). The expected cost of the random cut can be shown to be the LP value (see e.g., [9]), and the minimum such ball can be found efficiently using Dijkstra's algorithm. Our goal in this section is to generalize this formulation and ball-growing method to NVDP.

One difficulty in NVDP comes from the fact that every arc has four associated costs and we need to write an objective function that correctly captures the final cost of a chosen cut. One way to overcome this issue is to have a distinct arc associated with each cost. In other words, for every original arc $uv \in A(G)$, we create four new arcs $[uv]_0, [uv]_1, [uv]_2, [uv]_3$ with cost $c_e, c_{eu}, c_{euv}, c_{ev}$ respectively. Then, every arc has its unique cost and it is now easier to characterize the final cost of a cut. We consider the following auxiliary graph. See Fig. 3.

Constructing the Auxiliary Graph H. Let $V(H) = V^0(H) \cup V^1(H)$ where $V^0(H) = \{(vv) : v \in V(G)\}$ and $V^1(H) = \{(uv)_i : uv \in A(G), i = 1, 2, 3\}$. Define $A(H) = \{[uv]_0 = (uu)(uv)_1, [uv]_1 = (uv)_1(uv)_2, [uv]_2 = (uv)_2(uv)_3, [uv]_3 = (uv)_3(vv) : uv \in A(G)\}$. Essentially, the vertices $(uu) \in V^0(H)$ correspond to the original vertices $u \in V(G)$ and for every arc $uv \in A(G)$, we replace it with a path $(uu)(uv)_1(uv)_2(uv)_3(vv)$ where the four arcs on the path are $[uv]_0, [uv]_1, [uv]_2, [uv]_3$. For convenience and consistency in notation, we define $(uv)_0 := (uu)$, $(uv)_4 := (vv)$. Note that the vertices of H will always be denoted as two lowercase letters in parenthesis while arcs in H will be two lowercase letters in

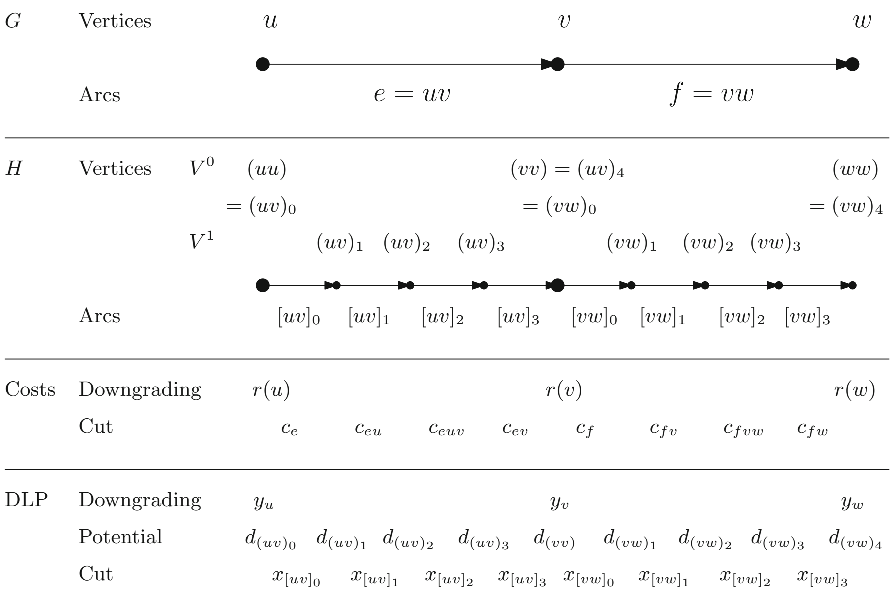


Fig. 3 Construction of the auxiliary graph H

square brackets with subscript $i = 0, 1, 2, 3$. The cost function $c : A(H) \to \mathbb{R}_{\geq 0}$ is as follows: $c([uv]_0) = c_e$, $c([uv]_1) = c_{eu}$, $c([uv]_2) = c_{euv}$, $c([uv]_3) = c_{ev}$. Since we can only downgrade vertices in V^0, to simplify the notation, we retain $r(v)$ as the cost to downgrade vertex $(vv) \in V^0$. Note that $|V(H)| = 3|A(G)| + |V(G)| = O(n+m)$.

Downgrading LP. Given the auxiliary graph H, we can now construct an LP similar to the one for st-cuts. For vertices $(vv) \in V^0(H)$ corresponding to original vertices of G, we define a downgrading variable y_v representing whether vertex v is downgraded or not in G. For every arc $[uv]_i \in A(H)$, we have a cut variable $x_{[uv]_i}$ to indicate if the arc belongs in the final cut of the graph. Lastly for all vertices $(uv)_i \in V(H)$, we have a potential variable $d_{(uv)_i}$ representing its distance from the source (ss).

The idea is to construct an LP that forces s, t to be at least distance 1 apart from each other as before. This distance can only be contributed from the arc variables $x_{[uv]_i}$. The downgrading variables y_v imposes limits on how large these distances $x_{[uv]_i}$ of some of its incident arcs can be. The motivation is that the larger y_u and y_v are, the more we should allow arc $[uv]_2$ to appear in the final cut over the other arcs $[uv]_0$, $[uv]_1$, $[uv]_3$ in order to incur the cheaper cost of c_{euv}. We consider the following downgrading LP henceforth called DLP.

Figure 3 includes the list of variables associated with H. In the LP, our objective is to minimize the cost of the final cut. Constraint (3) corresponds to the budget constraint for the downgrading variables. Constraint (4) is analogous to Constraint (1) in the LP for min-cuts.

Constraint (5) relates cut and downgrade variables. If we do not consider any constraint related to downgrading variables for a moment, the LP will naturally always want to choose the cheapest arc $[uv]_2$ over $[uv]_0$, $[uv]_1$, $[uv]_3$ when cutting somewhere between (uu) and (vv). However, the cut should not be allowed to go through $[uv]_2$ if one of u, v is not downgraded. In other words $x_{[uv]_2}$ should be at most the minimum of y_u, y_v. This reasoning gives the constraint $x_{[uv]_2}, x_{[uv]_3}, x_{[vw]_1}$ and, $x_{[vw]_2}$ all need to be $\leq y_v$ for in-arcs uv and out-arcs vw. Now consider an arc $f = vw \in E(G)$. In an integral solution, if v is downgraded, the arc vw incurs a cost of either c_{fv} or c_{fvw} but not both, since v must lie on one side of the cut. This translates to a LP solution where only one of the arcs $[vw]_1$, $[vw]_2$ is in the final cut. Thus, a better constraint to impose is $x_{[vw]_1} + x_{[vw]_2} \leq y_v$. We can also similarly insist that $x_{[uv]_2} + x_{[uv]_3} \leq y_v$ for in-arcs uv. To push this even further, consider a path uvw in G. In an integral solution, at most one of the arcs uv, vw appears in the final cut. This implies that if v is downgraded, then only one of the costs $c_{ev}, c_{euv}, c_{fv}, c_{fvw}$ is incurred. This corresponds to the tighter constraint (5). Note that for every vertex $v \in V(G)$, for every pair of incoming and outgoing arcs of v, we need to add one such constraint. Then, for every vertex in G, we potentially have to add up to n^2 many constraints. In total, the number of constraints would still only be $O(n^3)$. The last few constraints in DLP make sure s and t are 1 distance apart and cannot themselves be downgraded. The final LP relaxation is given below.

$$\min \quad \sum_{[uv]_i \in A(H)} c([uv]_i) x_{[uv]_i} \tag{DLP}$$

$$\text{s.t.} \quad \sum_{(vv) \in V^0(H)} r(v) y_v \leq b \tag{3}$$

$$d_{(uv)_{i+1}} \leq d_{(uv)_i} + x_{[uv]_i} \qquad \forall \text{arc } [uv]_i, 0 \leq i \leq 3 \qquad (4)$$

$$x_{[uv]_2} + x_{[uv]_3} + x_{[vw]_1} + x_{[vw]_2} \leq y_v \qquad \forall \text{ path } (uv)_3(vv)(vw)_1$$

$$d_{(ss)} = 0, d_{(tt)} = 1, y_s = 0, y_t = 0 \qquad (5)$$

The following lemmas shows the validity of our defined DLP for NVDP.

Lemma 2 *An optimal solution to NVDP provides a feasible integral solution to DLP with the same cost.*

Proof Given a digraph G with cost functions $c_e, c_{eu}, c_{ev}, c_{euv}$, a source s and a sink t, let $Y \subseteq V(G)$, $F \subseteq A(G)$ be an optimal solution to NVDP where $r(Y) \leq b$, F is an st-cut and $c^Y(F)$ is minimum. Then, a feasible solution (x, y, d) to DLP on the graph H can be constructed as follows:

- For the cut variables x, let
  - $x_{[uv]_0} = 1$ if $uv \in F$ and $u, v \notin Y$, 0 otherwise,
  - $x_{[uv]_1} = 1$ if $uv \in F$ and $u \in Y, v \notin Y$, 0 otherwise,
  - $x_{[uv]_2} = 1$ if $uv \in F, u, v \in Y$, 0 otherwise,
  - $x_{[uv]_3} = 1$ if $uv \in F, u \notin Y, v \in Y$, 0 otherwise.
- For the downgrading variables y, let $y_u = 1$ if $u \in Y$, 0 otherwise.
- For the potential variables d, let $d_{[uv]_i} = 0$ if $[uv]_i \in S$ and 1 otherwise,

where we define S, T as follows. Let F^* be the set of arcs in H whose x variable is 1. We claim that F^* is an st-cut in H. Note that every st-path Q in H corresponds to an st-path P in G. Then, there is an arc uv in P that is also in F. Then, it follows from construction that the x value for one of $[uv]_0, [uv]_1, [uv]_2, [uv]_3$ is 1 and thus there exists $i = 0, 1, 2, 3$ such that $[uv]_i \in F^*$. Note that $[uv]_i$ is also in Q. Therefore F^* is an st-cut in H. Then, let S be the set of vertices in $H \backslash F^*$ that is connected to the source s and let $T = V(H) \backslash S$.

Note that by construction, (x, y, d) is integral and is a feasible solution to DLP. The final objective value $\Sigma_{[uv]_i \in A(H)} c([uv]_i) x_{[uv]_i} = \Sigma_{[uv]_i \in F^*} c([uv]_i)$ and by construction, it matches the cost $c^Y(F^*)$. □

Lemma 3 *An integral solution (x^*, y^*, d^*) to DLP with objective value c^* corresponds to a feasible solution (Y^*, E^*) to NVDP such that $c^{Y^*}(E^*) \leq c^*$.*

Proof Given a directed graph G and its auxiliary graph H, let (x^*, y^*, d^*) be an optimal integral solution to DLP with an objective value of c^*. Let $F^* \subseteq A(H)$ be the set of arcs whose x^* value is 1. Let $Y^* \subseteq V^0(H)$ whose y value is 1. Let $E^* \subseteq A(G) = \{uv \in A(G) : [uv]_i \in F^* \text{ for some } i = 0, 1, 2, 3\}$ be the set of original arcs of those in F^*.

Note that by construction, Y^* does not violate the budget constraint. Every st-path in G corresponds directly to an st-path in H. Since F^* is an st-cut in H, it follows that E^* is an st-cut in G. Then it remains to show that $c^* \geq c^{Y^*}(E^*)$.

Note that

$$c^* = \Sigma_{[uv]_i \in A(H)} c([uv]_i) x^*_{[uv]_i} = \Sigma_{e=uv \in A(G)} c_e x^*_{[uv]_0} + c_{eu} x^*_{[uv]_1} + c_{euv} x^*_{[uv]_2} + c_{ev} x^*_{[uv]_3}.$$

Meanwhile, note that $c^{Y^*}(E^*) = \Sigma_{e=uv \in A(G)} c^{Y^*}(e)$. Thus, it suffices to prove the following claim. □

Claim For every arc $e = uv \in A(G)$, $\Sigma_{i=0}^{3} c([uv]_i) x^*_{[uv]_i} \geq c^{Y^*}(e)$ if $e = uv \notin E^*$.

First, note that if $e = uv \notin E^*$, then this edge is not involved in $c^{Y^*}(E^*)$ and by definition of E^* we have $x^*_{[uv]_i} = 0$ for $i = 0, 1, 2, 3$. Then, we assume in the sequel that $uv \in E^*$. This implies that there exist $i = 0, 1, 2, 3$ such that $[uv]_i \in F^*$ and $x^*_{[uv]_i} = 1$. We will now break into cases depending on whether $u, v \in Y^*$.

Suppose $u, v \notin Y^*$. Then, $y^*_u = y^*_v = 0$ and by constraint (5) in DLP, it follows that the x^* value for $[uv]_1, [uv]_2, [uv]_3$ are all 0. Then, $[uv]_0 \in F^*$ and $\Sigma_{i=0}^{3} c([uv]_i) x^*_{[uv]_i} = c_e = c^{Y^*}(e)$.

Now, assume $u \in Y^*, v \notin Y^*$. By constraint (5), $x^*_{[uv]_2} + x^*_{[uv]_3} \leq y^*_v = 0$ and thus only the x^* value for $[uv]_0, [uv]_1$ can be 1. Since we have an integral solution, it follows that $x^*_{[uv]_0} + x^*_{[uv]_1} \geq 1$, since $e \in E^*$. Note that $c_e \geq c_{eu}$. Then $\Sigma_{i=0}^{3} c_{[uv]_i} x^*_{[uv]_i} = c_e x^*_{[uv]_0} + c_{eu} x^*_{[uv]_1} \geq c_{eu}(x^*_{[uv]_0} + x^*_{[uv]_1}) \geq c_{eu} = c^{Y^*}(e)$. Note that a similar argument can be made for the case when $u \notin Y^*, v \in Y^*$.

Lastly, assume both $u, v \in Y^*$. Then $c^{Y^*}(e) = c_{euv}$. Note that $c_e, c_{eu}, c_{ev} \geq c_{euv}$. Then, $\Sigma_{i=0}^{3} c([uv]_i) x^*_{[uv]_i} \geq \Sigma_{i=0}^{3} c_{euv} x^*_{[uv]_i} \geq c_{euv}$. The last inequality is due to the fact that there exists $i = 0, 1, 2, 3$ such that $[uv]_i \in F^*$. This completes the proof of claim and thus also our lemma. □

The above two lemmas show that DLP provides a valid integer programming formulation for NVDP.

Bicriteria Approximation for NVDP. We now prove Theorem 2 by working with an optimal solution of DLP defined on the auxiliary graph H. The idea is to use a ball-growing algorithm that greedily finds cuts until one with the promised guarantee is produced. The reason this algorithm is successful is proved by analyzing a randomized algorithm that picks a number $0 \leq \alpha \leq 1$ uniformly at random and chooses a cut at distance α from the source s. Then we choose vertices to downgrade and arcs to cut based on arcs in this cut at distance α. By computing the expected downgrading cost and the expected cost of the cut arcs, the analysis will show the existence of a level cut that satisfies our approximation guarantee.

To achieve the desired result, we cannot work with the graph H directly. This is because the ball-growing algorithm works only if the probability of cutting some arc can be bounded within some range. This bound exists for the final cut arcs (as in the proof for st-cuts) but not for the final downgraded vertices. Consider a vertex v; it is downgraded if any arc of the form $[uv]_2, [uv]_3, [vw]_1, [vw]_2$ is cut in H. Thus it has the potential of being cut anywhere between the range of the vertices $(uv)_2$ and $(vw)_3$ (i.e. $[d_{(uv)_2}, d_{(vw)_3}))$. We would like to use Constraint (5) to bound this range but we cannot do this directly since we do not know the length of the arc $[vw]_0$ which also lies in this range. To circumvent this difficulty and properly employ Constraint (5), we construct a reduced graph H' obtained by reordering the edges and slightly modifying some of their lengths.

Let (x^*, y^*, d^*) be an optimal solution to DLP where the optimal cost is c^*. It follows from the validity of our model (Lemma 2) that c^* is at most the cost of an optimal integral solution.

Constructing Graph H'. We build the graph H' slightly differently depending on the value of ϵ. If $0 < \epsilon \leq 1$, for every arc $uv \in A(G)$, let $x'_{[uv]_0} = \frac{1}{2}\min((1+\epsilon)x^*_{[uv]_0}, x^*_{[uv]_0} + x^*_{[uv]_2})$, $x'_{[uv]_1} = x^*_{[uv]_1}$, $x'_{[uv]_3} = x^*_{[uv]_3}$ and $x'_{[uv]_2} = x^*_{[uv]_0} + x^*_{[uv]_2} - 2x'_{[uv]_0}$. We introduce a new arc $[uv]_{00}$ and order the five arcs as follows: $[uv]_1, [uv]_0, [uv]_2, [uv]_{00}, [uv]_3$. For $i = 1, 2, 3$, arc $[uv]_i$ has length $x'_{[uv]_i}$. The length for both $[uv]_0$ and $[uv]_{00}$ is $x'_{[uv]_0}/2$. Note that the distance under x' between any two vertices in V_0 remains the same as before (i.e. under x^* in H). This reordering allow us to bound more accurately the range in which a vertex gets downgraded.

If $\epsilon > 1$, we construct H as if $\epsilon = 1$. In other words, each arc uv is replaced with a path of arcs $[uv]_1, [uv]_0, [uv]_2, [uv]_{00}, [uv]_3$ whose lengths are $x^*_{[uv]_1}, x'_{[uv]_0}, x^*_{uv_0} + x^*_{[uv]_2} - 2x'_{[uv]_0}, x'_{[uv]_{00}}, x^*_{[uv]_3}$ respectively, where $x'_{[uv]_0} = x'_{[uv]_3} = \frac{1}{2}\min(2x^*_{[uv]_0}, x^*_{[uv]_0} + x^*_{[uv]_2})$.

The following is an observation about the new arc length x' that we use later.

Claim Assume for some arc $uv \in A(G)$, $x'_{[uv]_2} > 0$. If $0 < \epsilon \leq 1$, then $x'_{[uv]_0} + x'_{[uv]_2} = x'_{[uv]_2} + x'_{[uv]_{00}} \leq \frac{1+1/\epsilon}{2}x^*_{[uv]_2}$. If $\epsilon \geq 1$, then $x'_{[uv]_0} + x'_{uv]_2} = x'_{[uv]_2} + x'_{[uv]_{00}} \leq x^*_{[uv]_2}$.

Proof First, let us assume $0 < \epsilon \leq 1$. If $x'_{[uv]_2} > 0$, then $x^*_{[uv]_0} + x^*_{[uv]_2} > 2x'_{[uv]_0}$. It follows that $2x'_{[uv]_0} = (1+\epsilon)x^*_{[uv]_0}$ and $\frac{1}{\epsilon}x^*_{[uv]_2} > x^*_{[uv]_0}$. Then, $x'_{[uv]_0} + x'_{[uv]_2} = \frac{1}{2}(1+\epsilon)x^*_{[uv]_0} + x^*_{[uv]_0} + x^*_{[uv]_2} - (1+\epsilon)x^*_{[uv]_0} = x^*_{[uv]_2} + \frac{1-\epsilon}{2}x^*_{[uv]_0} < \frac{1+1/\epsilon}{2}x^*_{[uv]_2}$. Since $x'_{[uv]_0} = x'_{[uv]_{00}}$, the claim follows immediately.

If $\epsilon \geq 1$, by similar arguments as the previous case, it is easy to check that the claim also follows immediately. $\square$

Algorithm 1 Ball-Growing Algorithm for NVDP

Require: a graph G and its auxiliary graph H' with non-negative arc-weights x', source (ss), sink (tt), a constant $\epsilon > 0$, arc cut costs $c([uv]_i)$ and vertex downgrading costs $r(v)$

Ensure: a vertex set V' and an arc cut E' of G such that $\Sigma_{v \in V'} r(v) \leq (1+1/\epsilon)^2 b/2$, $c^{V'}(E') \leq (1+\epsilon)^2 c^*$ if $0 < \epsilon < 1$ and $\Sigma_{v \in V'} r(v) \leq (1+1/\epsilon)b$, $c^{V'}(E') \leq 2(1+\epsilon)c^*$ if $\epsilon \geq 1$,

1: initialization $V = \{(ss)\}$, $D((uv)_i) = 1$ for all $(uv)_i \in V(H')$
2: **repeat**
3: let $X' \subseteq A(H')$ be the cut induced by V
4: find $[uv]_i = ab \in X'$ minimizing $D(a) + x'_{[uv]_i}$
5: update by adding $(uv)_{i+1}$ to V, update $D(b) = D(a) + x'_{[uv]_i}$
6: let $E' = \{uv \in A(G) : \{[uv]_0, [uv]_{00}, [uv]_1, [uv]_2, [uv]_3\} \cap X' \neq \emptyset\}$ and $V' = \{v \in V(G) : \{[uv]_2, [uv]_3, [vw]_1[vw]_2\} \cap X' \neq \emptyset$ for some $u, w \in V(G)\}$
7: **until** $\Sigma_{v \in V'} r(v) \leq (1+1/\epsilon)^2 b$ and $c^{V'}(E') \leq (1+\epsilon)^2 c^*/2$
8: output the set V', E'

Algorithm 1 is simply a restatement of Dijkstra's algorithm run on H'. It follows the general ball-growing technique and looks at cuts X' at various distances from the source. Note that the algorithm adds at least one vertex to a node set V at each iteration so it runs for at most $|V(H')| = O(m)$ steps when applied to the graph H' (Recall that m denotes the number of arcs in the original graph G).

At each iteration, the algorithm computes a cut $X' \subseteq A(H')$ and considers the set E' of original arcs associated to those in X' and the vertex set V' representing the set of vertices we should downgrade based on the arcs in X'. For example, if $[uv]_2 \in X'$, then we should downgrade both u and v. Since X' is a cut in H', it follows that E' is a cut in G.

To argue the validity of the algorithm, we show that there exists a cut X' at some distance $\alpha \leq 1$ from the source such that the associated sets V', E' provides the approximation guarantee.

Lemma 4 *There exists X', V', E' such that $\Sigma_{v\in V'} r(v) \leq (1+1/\epsilon)^2 b/2, c^{V'}(E') \leq (1+\epsilon)^2 c^*$ if $0 < \epsilon < 1$ and $\Sigma_{v\in V'} r(v) \leq (1+1/\epsilon)^2 b/2, c^{V'}(E') \leq 2(1+\epsilon)c^*$ if $\epsilon \geq 1$.*

The main idea of the proof is to pick a distance uniformly at random between [0, 1] and study the cut at that distance. We claim that the extent to which an arc is cut (chosen in E' above) in the random cut is at most its x^*-value. When nodes are chosen in the random cut (in V' above) to be downgraded, we argue that the range of cutting any node is at most the maximum of the values in the left hand side of the constraints (5) corresponding to this node, which in turn is at most its y^*-value. To obtain a cut where we simultaneously do not exceed both bounds, we use Markov's inequality to argue that there is a non-zero probability of getting a cut within the two bounds, showing the existence of such cut. Then it follows that Algorithm 1 can find this cut. The detailed proof follows.

Proof of Lemma 4 Let $D(a)$ be the shortest-path distance from the source (ss) to any vertex $a \in V(H')$ viewing the x' variables as lengths. Note that the triangle-inequality holds under this distance metric where $D(b) - D(a)$ is at most the distance between a and b.

Defining the Random Variables. Let α be chosen uniformly at random from the interval [0, 1]. Consider $X_\alpha := \{[uv]_i = ab \in A(H') : D(a) \leq \alpha < D(b)\}$, the cut at distance α in H'. Let $E_\alpha = \{uv \in A(G) : [uv]_i \in X_\alpha$ for some $i \in \{0, 00, 1, 2, 3\}\}$, representing the original arcs corresponding to those in X_α. Let $V_\alpha = \{v \in V(G) : \{[uv]_2, [uv]_3, [vw]_1, [vw]_2\} \cap X_\alpha \neq \emptyset$ for some $u, w \in V(G)\}$, representing the set of vertices we should downgrade so that the final cost of the arcs E_α matches the cost associated to X_α. More precisely, we want $c^{V_\alpha}(E_\alpha) = C\Sigma_{[uv]_i\in X_\alpha} c([uv]_i)$, where $C = (1+\epsilon)^2$ if $0 < \epsilon < 1$ and $C = 2(1+\epsilon)$ if $\epsilon \geq 1$. Note that by construction E_α is an st-cut in G. Let $\mathcal{V} = \Sigma_{v\in V_\alpha} r(v)$, $\mathcal{E} = c^{V_\alpha}(E_\alpha)$. Our goal is to show that these two random variables $\mathcal{V}, \mathcal{E}$ have low expectations and obtain our approximation guarantee using Markov's inequality. In particular, we will prove that if $0 < \epsilon < 1$, then $\mathbb{E}[\mathcal{V}] \leq (1+1/\epsilon)b/2$ and $\mathbb{E}[\mathcal{E}] \leq (1+\epsilon)c^*$, and if $\epsilon \geq 1$, then $\mathbb{E}[\mathcal{V}] \leq b$ and $\mathbb{E}[\mathcal{E}] \leq 2c^*$, where c^* is the optimal value of DLP.

To understand $\mathcal{E}$, for every arc $e = uv \in A(G)$, we introduce the indicator variables $\mathcal{E}_e$ to be 1 if arc $e \in E_\alpha$ and 0 otherwise. Then $\mathcal{E} = \Sigma_{e\in A(G)} \mathcal{E}_e c^{V_\alpha}(e)$. To study the value of $\mathcal{E}_e c^{V_\alpha}(e)$, we can break into several cases depending on which arc $[uv]_i \in X_\alpha$. Note that if $[uv]_i \notin X_\alpha$ for $i \in \{0, 00, 1, 2, 3\}$, then $e \notin E_\alpha$ and $\mathcal{E}_e c^{Y_\alpha}(e) = 0$. Next, if we assume $[uv]_i \in X_\alpha$, then one can check that $c^{V_\alpha}(e) \leq c([uv]_i)$ as in the proof of Claim 3.1. Note that here, we denote $c([uv]_{00}) = c([uv]_0) = c_e$.

Slightly abusing the notation, define the indicator variable $\mathcal{E}_{[uv]_i}$ for arc $[uv]_i \in A(H)$ to be 1 if $[uv]_i \in X_\alpha$ and 0 otherwise. Then, we can upper-bound the expectation of $\mathcal{E}$ using conditional expectations of the events $\mathcal{E}_{[uv]_i} = 1$ as follows.

$$\begin{aligned}\mathbb{E}[\mathcal{E}] &= \Sigma_{e\in A(G)}\mathbb{E}[\mathcal{E}_e c^{V_\alpha}(e)]\\ &= \Sigma_{e\in A(G)}\Sigma_{i\in\{0,00,1,2,3\}}\mathbb{E}[c^{V_\alpha}(e)|\mathcal{E}_{[uv]_i} = 1]\cdot Pr[\mathcal{E}_{[uv]_i} = 1]\\ &\leq \Sigma_{e\in A(G)}\Sigma_{i\{0,00,1,2,3\}}c([uv]_i)Pr[\mathcal{E}_{[uv]_i} = 1]\end{aligned}$$

Let us first handle the case when $0 < \epsilon \leq 1$. To bound the probability of $\mathcal{E}_{[uv]_i} = 1$, note that an arc $[uv]_i = ab \in X_\alpha$ if and only if $D(a) \leq \alpha < D(b)$. Then, $Pr[[uv]_i \in X_\alpha] \leq D(b) - D(a) \leq x'_{[uv]_i} \leq x^*_{[uv]_i}$ for $i \in \{1, 2, 3\}$. For $i \in \{0, 00\}$, $Pr[[uv]_i \in X_\alpha] \leq (1+\epsilon)x^*_{[uv]_0}/2$. Combining with the previous inequalities, we see that

$$\begin{aligned}\mathbb{E}[\mathcal{E}] &\leq \Sigma_{uv\in A(G)}\Sigma_{i\in\{0,00,1,2,3\}}c([uv]_i)Pr[\mathcal{E}_{[uv]_i} = 1]\\ &\leq \Sigma_{uv\in A(G)}\Sigma_{i=1}^3 c([uv]_i)x^*_{[uv]_i} + \Sigma_{uv\in A(G)}\Sigma_{i\in\{0,00\}}c([uv]_0)(1+\epsilon)x^*_{uv]_0}/2\\ &\leq \Sigma_{uv\in A(G)}\Sigma_{i=0}^3 c([uv]_i)(1+\epsilon)x^*_{[uv]_i} = (1+\epsilon)c^*.\end{aligned}$$

When $\epsilon \geq 1$, the proof is identical to the previous case under the assumption that $\epsilon = 1$. It is easy to verify that $\mathbb{E}[\mathcal{E}] \leq 2c^*$, as desired.

Next, we show a similar result for $\mathcal{V}$. Note that $\mathbb{E}[\mathcal{V}] = \Sigma_{v\in V(G)}r(v)\cdot Pr[v \in V_\alpha]$. To determine the probability of downgrading a particular vertex v, we need to bound the range of α that includes all arcs that might cause v to be chosen in V_α. Recall that $v \in V_\alpha$ if and only if there exists a vertex u or w such that at least one of $[uv]_2, [uv]_3, [vw]_1, [vw]_2 \in X_\alpha$. Let F be the set of all arcs of the form $[uv]_2, [uv]_3, [uvw]_1$, or $[vw]_2$ whose x' value is nont 0. Let a be the tail of an arc in F such that $D(a)$ is minimum. Let b be the head of an arc in F such that $D(b)$ is maximum. It follows that $v \in V_\alpha$ only if $D(a) \leq \alpha < D(b)$. It remains to bound the distance between a and b.

First, assume $0 < \epsilon \leq 1$. Suppose a is the head of the arc $[uv]_i$. Due to the order of the arcs in H', it follows that i is either 2 or 3. We claim that in both cases, $D((vv))-D(a) \leq \frac{1+1/\epsilon}{2}(x^*_{[uv]_2}+x^*_{[uv]_3})$. If $i = 2$, then $x'_{[uv]_2} > 0$ and $D((vv))-D(a)$ is bounded by the length of the path $[uv]_2[uv]_{00}[uv]_3$. It follows from Claim 3.1, $x'_{[uv]_2} + x'_{[uv]_{00}} + x'_{[uv]_3} \leq \frac{1+1/\epsilon}{2}x^*_{[uv]_2} + x^*_{[uv]_3}$ and our claim follows. If $i = 3$, then $D(b) - D(a)$ is bounded by the length of arc $[uv]_3$ and the claim also follows immediately.

Suppose b is the head of the arc $[vw]_i$. By similar reasoning, one can show that $D(b) - D((vv)) \leq \frac{1+1/\epsilon}{2}(x^*_{[uv]_1} + x^*_{[uv]_2})$. Then, $D(b) - D(a) = D(b) - D((vv)) + D((vv)) - D(a) \leq \frac{1+1/\epsilon}{2}(x^*_{[uv]_2} + x^*_{[uv]_3} + x^*_{[vw]_1} + x^*_{[vw]_2}) \leq \frac{1+1/\epsilon}{2}y^*_v$ where the last inequality follows from Constraint (5). Thus, $Pr[v \in V_\alpha] \leq \frac{1+1/\epsilon}{2}y^*_v$. Therefore

$$\mathbb{E}[\mathcal{V}] = \Sigma_{v \in V(G)} r(v) \cdot Pr[v \in V_\alpha] \\ \leq \Sigma_{v \in V(G)} r(v) \frac{1+1/\epsilon}{2} y_v^* \leq \frac{1+1/\epsilon}{2} b.$$

If $\epsilon \geq 1$, by similar arguments as before, one can easily check that $\mathbb{E}[\mathcal{V}] = b$.

Lastly, by Markov's inequality, $Pr[\mathcal{V} \leq (1+1/\epsilon)\mathbb{E}[\mathcal{V}]] \geq 1 - \frac{\epsilon}{1+\epsilon} = \frac{1}{1+\epsilon}$, $Pr[\mathcal{E} \leq (1+\epsilon+\delta)\mathbb{E}[\mathcal{E}]] \geq 1 - \frac{1}{1+\epsilon+\delta}$ for any $\delta > 0$. Then it follows there exists $0 \leq \alpha \leq 1)$ such that $\Sigma_{v \in V_\alpha} r(v) \leq (1+1/\epsilon)\mathbb{E}[\mathcal{V}]$ and $c^{V_\alpha}(E_\alpha) \leq (1+\epsilon)\mathbb{E}[\mathcal{E}] + \delta\mathbb{E}[\mathcal{E}]$. One can choose δ such that $\delta\mathbb{E}[\mathcal{E}] < 1$. Since the cost of cutting an edge is always integral, it follows that $c^{V_\alpha}(E_\alpha) \leq (1+\epsilon)\mathbb{E}[\mathcal{E}]$, proving Lemma 4. □

Proof of Theorem 2 It is well known that the Ball Growing algorithm (which is Djikstra's algorithm run on H') selects a linear number of nested cuts that represent the set of all cuts at all distances between zero and 1 from the source. It follows from Lemma 4 that one of these cuts meets the desired guarantees. Theorem 2 is then proved by simply running Algorithm 1 on the auxiliary graph H'. □

3.2 A simple pseudorandom approximation for NVDP

In this subsection, we provide a pseudorandom approximation to NVDP by reducing the problem to WNVIP. This reduction was proposed by one of our reviewers and by combining with results by Burch et al. [3], we obtain a pseudorandom approximation for NVDP.

Theorem 3 *For any* $\epsilon > 0$*, there exists a pseudorandom* $(2(1+\epsilon), 1)$ *or* $(2, (1+1/\epsilon)$*-approximation for NVDP.*

Being a pseudorandom approximation, a priori, it is not possible to determine whether we get a $(2(1+\epsilon), 1)$ or $(2, (1+1/\epsilon)$-approximation. Note that both are incomparable to the guarantee from Theorem 2. Note that without the pseudo-random component, this theorem can only guarantee a $(2(1+\epsilon), (1+1/\epsilon))$ which is the same as the one from Theorem 2 when $\epsilon \geq 1$. However, if $\epsilon < 1$, where we desire an (α, β)-approximation such that α is strictly less than 4, $(2(1+\epsilon), (1+1/\epsilon))$ is strictly worse than the guarantee from Theorem 2. In detail, note that for any $\epsilon > 0$, if we pick ϵ' such that $2(1+\epsilon) = (1+\epsilon')^2$, then we have $(1+1/\epsilon')^2/2 \leq (1+1/\epsilon)$. Figure 4 illustrates the trade-off between α and β in the guarantees from Theorem 2 and the non-pseudomrandom guarantee of Theorem 3. Note in particular that not only Theorem 2 performs better when $\alpha \in (2, 4)$, Theorem 3 can never achieve an α value below 2 but Theorem 2 can attain an α arbitrarily close to 1. Furthermore, the reduction in the proof of Theorem 3 does not apply to approximate NVLDP while the ball-growing based method can be generalized.

Proof of Theorem 3 Given an instance of NVDP on graph G, consider the following auxiliary graph H: for every vertex v, make a copy v' and connect with bidirectional arcs with infinite cost to cut. For every arc $uv \in A(G)$, add arcs uv, uv', $u'v$ and $u'v'$ with cut costs $2c_e - c_{eu} - c_{ev} + c_{euv}$, $c_{ev} - c_{euv}$, $c_{eu} - c_{euv}$, and c_{euv} respectively.

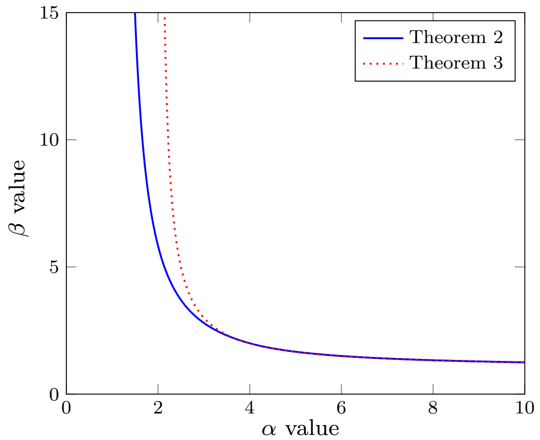


Fig. 4 The (α, β)-approximations from Theorem 2 and 3

Original vertices v have an interdiction cost of $r(v)$ and all new clones v' have an infinite interdiction cost.

Consider the instance of WNVIP on H with the above costs and budget b. It follows from [3] that there exists a pseudorandom $(1+\epsilon, 1)$, $(1, 1+1/\epsilon)$-approximation for this problem. Then it remains to show that any (α, β)-approximation for this instance of WNVIP translates to a $(2\alpha, \beta)$ approximation for the original instance of NVDP.

First, we show that if interdicting a set $Y \subseteq V(H)$ results in a minimum cut F' in H with cost c^*, then interdicting the set Y in G also results in a minimum cut of cost at most c^*. Note that since all new vertices in H have infinite interdiction cost, Y only contains original vertices and thus interdicting Y in G incurs the same interdiction cost. Since any arc between a vertex v and its clone v' have infinite cut cost, the endpoints of any arc in F' corresponds to two different original vertices in G. Furthermore, if one of uv, $u'v$ or uv' is in F, then $u'v'$ is also in F. Then, let $F := \{uv \in A(G) : u'v' \in F'\}$. It follows that F is a cut in G. Suppose $uv \in F$. We show that the cost uv contributes to F is at most the cost the arcs uv, uv', $u'v$ and $u'v'$ contributes together to F'. If u' and v' are both in Y, then $u'v' \in F'$ while uv, $u'v$, uv' are not in F'. Since $u'v'$ has cost c_{euv}, our claim is true. If $u \in Y$ but $v \notin Y$, then only arcs $u'v'$ and $u'v$ are in F'. They incur a total cost of c_{ev} corresponding to the final cost uv contributes to F in G. Similarly, if $u \notin Y$, $v \in Y$, then only arcs uv', $u'v'$ are in F', incurring a cost of c_{ev}. Lastly, if neither u nor v are in F', then all four associated arcs are in F', incurring a cost of $2c_e$. Meanwhile, since cutting uv in G only costs c_e, our claim is true and thus $c^Y(F) \leq c^*$.

Let opt_G, opt_H be the optimal value for the instance of NVDP on G and WNVIP on H. It remains to show that $opt_H \leq 2opt_G$. Let $Y \subseteq V(G)$ and F be a cut in G such that $r(Y) \leq b$ and $c^Y(F) = opt_G$. Let F' be the set of arcs in $H \backslash Y$ associated to those in F. More precisely, $F' = F_1 \cap E(H \backslash Y)$ where $F_1 = \{uv, uv', u'v, u'v' : uv \in F\}$. By construction, F' is a cut in $H \backslash Y$. One can easily check that if $uv \in F$, then the associated arcs in F' have cost at most twice of what uv contributed to $c^Y(F)$. In fact, the cost only doubles when neither u nor v are interdicted. Then, Y, F is a feasible solution for the instance of WNVIP on H with interdiction cost at most b and a final cost of at most $2opt_G$. Therefore, $2opt_G \geq opt_H$ and the theorem follows. □

4 Zero-detection for NVLDP

In this section, we we show that it is possible to detect a solution of an instance of NVLDP where downgrading a subset of vertices to certain levels creates a cut of zero cost. The strategy is similar to that for NVLP where we transform the instance into a vertex cut problem.

Lemma 5 *Given an instance of NVLDP with k levels on graph G with budget b, there exists a polynomial-time algorithm to determine if there exists a function* $L : V(G) \to \{0, ..., k\}$ *and an* st*-cut* $F \subseteq A(G)$ *such that* $r^L(V) \leq b$ *and* $c^L(F) = 0$.

Proof Consider the following auxiliary graph H: for every $v \in V(G)$, create $k+1$ copies $v_1, ..., v_{k+1}$ and add all possible arcs amongst the clones $v_i v_j$. For a given arc uv, let i_{cap}, j_{cap} represent the lowest level one can downgrade u, v respectively to reduce the cost of e to zero without downgrading the other endpoint. If no such level exist, i.e. if downgrading u (v) to level k and not touching the other endpoint does not reduce the cost to 0, set i_{cap} (j_{cap}) to $k+1$. We first add all possible arcs $u_i v_j$ in H. Now, we perform two rounds of arc deletions. First, delete all arcs $u_i v_j$ where $i > i_{cap}$ and $j > j_{cap}$. This essentially ensures that the final vertex cut does not choose any vertices larger than i_{cap}, j_{cap} since it is pointless to downgrade past those levels from uv's perspective. Next, for any $1 \leq i, j \leq k$ where $c_{i,j}(e) = 0$, remove all arcs $u_{i'} v_{j'}$ where $i' > i$ and $j' > j$. The resulting arcs define $A(H)$. □

Claim If $u_i v_j \in A(H)$, then $u_{i'} v_{j'} \in A(H)$ where $i' \leq i, j' \leq j$.

To prove this claim, suppose for the sake of contradiction that there exists $i' \leq i, j' \leq j, u_i v_j \in A(H)$ but $u_{i'} v_{j'} \notin A(H)$. Note that $i \leq i_{cap}, j \leq j_{cap}$ otherwise arc $u_i v_j$ is removed in the first round of deletion process. Then, it follows that arc $u_{i'} v_{j'}$ was removed in the second round of deletion, implying there exists $i'' \leq i', j'' \leq j'$ such that $c_{i'',j''}(e) = 0$. However, this implies arc $u_i v_j$ should also be removed in the second round, a contradiction.

Now consider assigning costs of $r_i(v) - r_{i-1}(v)$ to the vertex $v_i \in V(H)$ for $i = 1, ..., k$. Vertices of the form v_{k+1} are assigned an arbitrarily high cost of ∞. We now claim that a minimum vertex cut corresponds to the minimum downgrading cost in order to achieve a zero-cost arc-cut.

Let b^* be the minimum downgrading budget for G to obtain a zero-cost cut. We first show that b^* is an upper bound to the minimum vertex cut of H. Let L^* be an optimal downgrading function that is a witness to the budget b^* and a zero-cost cut. Consider the vertex set $Y := \{v_i : i \leq L^*(v)\}$, essentially deleting all copies of v up to level $L^*(v)$. Note that the cost of deleting these vertices is exactly b^*. It remains to show that Y is a cut-set.

Suppose for the sake of contradiction that there exists an st-path P after deleting Y in H. Let W be the underlying walk induced by P in G. Since downgrading according to L^* induces a zero-cost cut, there exists an arc $uv \in W$ such that $c^{L^*}(uv) = 0$. Then, there are no arcs in H of the form $u_i v_j$ where $i > L^*(u)$, $j > L^*(v)$. However, since we have deleted all copies of u_i, v_j where $i \leq L^*(u)$, $j \leq L^*(v)$ when deleting Y, it follows that path P does not exist, a contradiction.

Now, we show that a vertex cut solution of H can be translated to a downgrading solution for G with the same cost. Note that we may assume all vertices have non-zero costs, otherwise we can delete in H (or downgrade to the next level in G) for free. Let Y^* be a minimum vertex cut. First, we claim that if $v_i \in Y^*$, then $v_{i'} \in Y^*$ for all $i' \leq i$. Suppose for the sake of contradiction that $v_i \in Y^*$, $v_{i'} \notin Y^*$ and $i' < i$. Since v_i is important to delete, there exists an st-path P that goes through v_i but not any other vertices in $Y^* \backslash v_i$. However, by Claim 4, P can reroute through $v_{i'}$ instead, forming a new path that avoids Y^*, a contradiction.

Define $L(v)$ to be the largest i such that $v_i \in Y^*$. Note that the cost of downgrading according to L is the same as the cost of Y^*. It remains to show that L induces a zero-cost cut. Suppose not, where there exists a path P in G whose arcs have non-zero costs with respect to L. Let $uv \in P$; we claim that $u_{L(u)+1} v_{L(v)+1}$ is an arc in H. Suppose not. If it was deleted in the first round of the removal process, then $i_{cap} < L(u) + 1$, implying downgrading to $L(u)$ ensures the arc uv to have cost 0, a contradiction. If it was deleted in the second phase, then there exist $i < L(u)+1$, $j < L(v)+1$ such that $c_{i,j}(uv) = 0$. However, by the nature of the edge costs, it follows that $c_{L(u),L(v)} = 0$, a contradiction. Then, using the arcs of the form $u_{L(u)+1} v_{L(v)+1}$, we obtain an st-path that avoids Y^*, a contradiction. □

5 NVDP with *k* Levels (NVLDP)

In this section, we prove the following theorem.

Theorem 4 *There exists a polynomial-time algorithm that provides a* $(2(1+\epsilon)k, 2(1+1/\epsilon)k)$*-approximation to NVLDP.*

The strategy is similar to that for NVDP. We first create a new graph and IP model that solves NVLDP. Then we study the solution to the LP relaxation of the problem. We similarly create an auxiliary graph H and transform it slightly so that the ball-growing method can be applied on H. Then by analyzing a random algorithm and using Markov's Inequality, we can show the existence of a good approximation. Then we can find such a solution by examining all the cuts in the ball growing method. The major difference is that we need to be more careful when working with H. In NVDP, we first turn each arc into a path of length four to create H in order to better represent the four different costs associated with each arc. Then, we split the path further into two parts, aided and unaided, and kept only one of the parts before running the ball-growing algorithm. If we were to repeat this strategy here, we would turn each arc into a path of length $(k+1)^2$ (since we have $(k+1)^2$ many costs per arc) after which we need to split it properly in order to run the ball-growing algorithm. However, this splitting step is less obvious than before. The naive way is to keep only the longest arc in the path but that blows up the approximation to a $O(k^2)$ factor. Thus, further analysis is needed in order to show (see Lemma 6) that in any optimal vertex solution to the LP, most arcs have length 0 and the path only has $O(k)$ non-zero length arcs and thus keeping the longest arc is not too detrimental.

LP Model for NVLDP. Similar to NVDP, we first transform the graph G by replacing every arc uv with a path of length $(k+1)^2$. The path contains arcs $[uv]_{i,j}$ where $0 \leq i, j \leq k$ with an associated cost of $c_{i,j}(uv)$. The order of these arcs in the path does not matter but for ease of notation and consistency, we order them in lexicographic order $[uv]_{0,0}[uv]_{0,1}...[uv]_{0,k}[uv]_{1,0}...[uv]_{1,k}[uv]_{2,0}...[uv]_{k,k}$. The vertices on the path are labelled $(uv)_0, ..., (uv)_{(k+1)^2}$ and thus arc $[uv]_{i,j} = (uv)_{i(k+1)+j}(uv)_{i(k+1)+j+1}$. We similarly introduce a cut variable x for every arc in H to represent whether the arc is in the final cut. We also introduce k downgrading variables y^i for each original vertex v to represent whether it is downgraded to level i. Node potential variables d are also needed for every vertex in H. Then, we can obtain the following LP (LDLP):

$$\begin{aligned}
\min \quad & \sum_{[uv]_{i,j} \in A(H)} c_{i,j}(uv) x_{[uv]_{i,j}} \\
\text{s.t.} \quad & \sum_{i=1}^{k} \sum_{(vv) \in V(H)} r_i(v) y_v^i \leq b && (6) \\
& d_{(uv)_{i(k+1)+j+1}} \leq d_{(uv)_{i(k+1)+j}} + x_{[uv]_{i,j}} \quad \forall [uv]_{i,j} \in A(H) && (7) \\
& y_v^i \geq \sum_{j=0}^{k} x_{[uv]_{j,i}} + \sum_{j'=0}^{k} x_{[vw]_{i,j'}} \quad \forall 1 \leq i \leq k, \ \forall \text{ paths } uvw \in G \\
& d_{(ss)} = 0, d_{(tt)} = 1, y_s^0 = 0, y_t^0 = 0 && (8)
\end{aligned}$$

The intuition behind these constraints are similar to those for DLP. Constraint (6) bounds the total amount of budget for downgrading. Constraint (7) is simply the shortest-path inequality for every arc in H. Constraint (8) relates the downgrading variable y_v^i to its associated arcs. The idea is if $y_v^i = 1$, then we are paying to downgrade v to level i and thus the cost of its incident arcs uv, vw should be $c_{j,i}(uv)$, $c_{i,j'}(vw)$ respectively (subject to how u, w are downgraded). Thus, y_v^i is a natural upper bound for all arcs involving v at level i. With similar arguments, we can strengthen this to upper bound the sum of all such variables of a single $(uu)(vv)(ww)$-path, giving us constraints (8). Note that the LP does not constrain a vertex v to be downgraded to only one level. However, it remains a valid relaxation. Also, our rounding algorithm will never downgrade a vertex to more than one level by using the natural ordering of the levels in terms of their effect on the incident arcs. Note that H has a lot more vertices and arcs than before, in particular, of the order of $(k+1)^2|A(G)|$ which is still polynomial. The number of x, y, d variables are on the order of $(k+1)^2n^2, kn, (k+1)^2n^2$ respectively, where $n = |V(G)|$. Constraints (6), (7) are thus still polynomially many and for every vertex v, we have at most kn^2 many constraints of the form 8. Thus LDLP is solvable in polynomial time.

Analyzing an Optimal Solution of LDLP. Let (x^*, y^*, d^*) be an optimal solution to LDLP. We first prove the following lemma:

Lemma 6 *There exists an optimal solution such that for any $uv \in A(G)$, there are at most $2k+1$ non-zero values of $x_{[uv]_{i,j}}$.*

Proof Fix an arc $uv \in A(G)$ and let us look at all x-variables associated with this arc. Imagine the x^* values are presented in a $k+1$ square matrix M where $M_{i,j} = x^*_{[uv]_{i,j}}$. Define $row_i := \Sigma_{j=0}^{k} M_{i,j}$, $col_j := \Sigma_{i=0}^{k} M_{i,j}$ to be the row and column sum of M respectively. Now consider the following LP:

$$
\begin{aligned}
(MLP) \qquad \min \quad & \sum_{0 \le i,j \le k} c_{i,j}(uv) M_{i,j} \\
\text{s.t.} \quad & \sum_{j=0}^{k} M_{i,j} = row_i && \forall 0 \le i \le k \\
& \sum_{i=0}^{k} M_{i,j} = col_j && \forall 0 \le j \le k \\
& M_{i,j} \ge 0 && \forall 0 \le i, j \le k
\end{aligned}
$$

Note that our x^* is a feasible solution here. The converse is also true where if x' is a feasible solution here, it can also be transformed into a feasible solution for LDLP naturally. Constraint (8) remains satisfied due to the row sum and column sum constraints in MLP. The only adjustment we have to make is to the potential variables $d_{(uv)_{i(k+1)+j}}$, but this can be easily modified according to the new x' values. Since the total sum of all the x variables did not change, it does not affect the overall distance from (uu) to (vv) thus no other constraints are violated. Furthermore, it is easy to check that if x' had a better objective value than x^* in MLP, it would also provide a better objective value in LDLP.

Now, let us study the rank of the constraint matrix for MLP. There are $k+1$ row sum constraints and $k+1$ column sum constraints but they are linearly dependent since the sum of all rows equals to the sum of all columns. Since they are the only non-negative constraints, the rank of the nontrivial constraints in MLP is at most $2k+1$. Hence any basic feasible (or vertex) optimal solution to MLP has at most $2k+1$ non-zero values, thus proving our claim. □

The above technique where we isolate to study only the entries of the matrix M can actually be applied to any submatrix of M as well. Suppose N is a submatrix of M with N_r rows and N_c columns, then one can create a similar LP by restricting our attention to only looking at the variables in N and minimizing its cost subject to maintaining the row sum and column sum of N. Then, we can obtain a similar result:

Corollary 1 *Any vertex optimal solution of LDLP contains at most $N_r + N_c - 1$ non-zero variables amongst those related to any submatrix N of M.*

We will use this fact to prove the following claim and say a bit more about a solution with $2k+1$ non-zero values.

Claim In an optimal solution x^*, if x^* contains $2k+1$ non-zero variables associated with uv, then there exist $0 \leq i, j \leq k$ such that the x^* value for $[uv]_{i,0}$, $[uv]_{0,j}$ are both non-zero and they are two distinct arcs.

Proof If we apply the corollary to the submatrix without the first row, there are at most $2k$ non-zero variables. It follows there exists at least one non-zero variable in the first row say $M_{0,j}$. Symmetrically, there exists at least one non-zero value in the first column, say $M_{i,0}$. Now, as long as i, j are not both 0, we are done. Suppose that $M_{0,0}$ is the only non-zero value in the first row and the first column, then apply the corollary to the submatrix without the first row and column, it contains at most $2(k-1)+1$ non-zero values, contradicting the fact that there are $2k+1$ non-zero values to begin with. □

Now, we will use the above claims to construct our auxiliary graph H' to successfully run the ball-growing algorithm. The key is to not shrink the overall distance between s and t too much.

Constructing H'. For every uv arc in G, look at the corresponding path in H. If there are at most $2k$ non-zero x^* variables, keep only the arc with the largest x^* value and contract the rest. Otherwise, if there are exactly $2k+1$ non-zero x^* values, there exists two distinct arcs $[uv]_{i',0}$, $[uv]_{0,j'}$ with non-zero x^* values. We want to group these two arcs as one object and compare its sum $x^*_{[uv]_{i',0}} + x^*_{[uv]_{0,j'}}$ to the individual x^* values of the other arcs on this path. Once again, keep only the highest value and contract the rest, and with a slight abuse of notation, continue to denote the contracted lengths by x^*. This operation reduces every long $(uu)(vv)$ path to no smaller than $1/(2k)$ of its original length. This implies the distance from s to t is at least $1/(2k)$.

Note that for any arc $uv \in A(G)$, the corresponding path in H' is either a single arc or a path of length 2 in the form of $[uv]_{i',0}[uv]_{0,j'}$. This simplifies the analysis of bounding the range of arcs incident to v responsible for downgrading it to a certain level. Unlike the construction of H' in NVDP, we will show that it is no longer needed to do any further modification to the weights x^*.

Now we proceed with the ball-growing algorithm to find cuts at different distances.

The algorithm finds cuts X', looks at all arcs in X' that involves v, checks which level these arcs need v to be downgraded to and picks the highest one. This provides a function L' and a final cut F'. The algorithm simply greedily checks all cuts from the ball growing algorithm and its associated function L' and F' until it finds one with the promised guarantee. Since there are polynomially many vertices in H', the algorithm only needs to check polynomially many cuts. Thus as long as one of these cuts provides the proper guarantee, the algorithm can find it. In order to show that such a cut exists, we use a similar technique as NVDP by choosing a cut at random and looking at the expected interdiction cost and cut cost. Note that the algorithm never chooses an arc $[uv]_{i,j}$ that points backward, where $D(tail([uv]_{i,j})) \geq D(head([uv]_{i,j}))$, so they can be ignored in the analysis of the associated costs.

Lastly, to prove Theorem 4, it remains to show the following:

Lemma 7 *Algorithm 2 provides a solution with the desired approximation guarantee.*

Algorithm 2 Ball-Growing Algorithm for NVLDP

Require: a graph G and its auxiliary graph H' with non-negative arc-weights $x^*_{[uv]_{i,j}}$, source (ss), sink (tt), arc cut costs $c_{i,j}(uv)$ and vertex downgrading costs $r_i(v)$
Ensure: a vertex downgrading function L' and an arc cut E' of G such that $r^{L'}(V(G)) \leq 4kb$, $c^{L'}(E') \leq 4kc^*$
Initialization $V = \{(ss)\}$, $D((uu))$, $D((uv)') = 1$ for all $(uu), (uv)' \in V(H')$
repeat
  Let $X' \subseteq A(H')$ be the cut induced by V
  Find $[uv]_{i,j} = tail([uv]_{i,j})head([uv]_{i,j}) \in X'$ minimizing $D(tail([uv]_{i,j})) + x^*_{[uv]_{i,j}}$
  Update by adding $head((uv)_{i,j})$ to V, update $D(head((uv)_{i,j})) = D(tail((uv)_{i,j})) + x^*_{[uv]_{i,j}}$
  Let $E' = \{uv \in A(G) : [uv]_{i,j} \in X'\}$ for some $0 \leq i, j \leq k$ and $L'(v) = \max\{i : [uv]_{j,i}$ or $[vw]_{i,j'} \in X'$ for some $u, w \in V(G), 0 \leq j, j' \leq k\}$
until $r^{L'}(V(G)) \leq 4kb$ and $c^{L'}(E') \leq 4kc^*$
Output the set V', E'

Proof Similar to the proof of validity of the algorithm used for NVDP, we show the existence of a proper cut using a randomized algorithm. Consider randomly choosing a number α between 0 and $D((tt))$. Let X_α be the cut in H' at distance α from (ss). In other words, $X_\alpha = \{[uv]_{i,j} \in A(H') : D(tail((uv)_{i,j})) \leq \alpha < D(head((uv)_{i,j}))\}$. Let L_α, F_α be the associated level-downgrading function and cut for X_α. We now analyze the expected downgrading and cut cost of L_α, F_α respectively. Recall that $L_{\alpha(v)}$ defines the index to which v is downgraded by choosing the level cut at distance α.

Given a vertex $v \in V(G)$ and $0 \leq i \leq k$, if the algorithm downgrades v to level i then an arc associated with v at level i must be in X_α. Thus, let us examine these arcs which have the form $[uv]_{j,i}$ or $[vw]_{i,j'}$. Given an arc $[uv]_{j,i} \in A(H')$, note that due to the construction of H', it is either the only arc between $(uu), (vv)$ or it has the form $[uv]_{0,i}$. In either case, it is incident to (vv) in H'. A similar statement is true for arcs of the form $[vw]_{i,j'}$ and thus all arcs associated to v at level i in H' are incident to (vv). These arcs form a star S^i_v centered at (vv). Since the algorithm never chooses (in the cut) any arc zz' where $D(z') < D(z)$, we may remove any such arcs from the star S^i_v. Note that the relevant arcs remain as a star.[1] Let $u', w' \in V(S^i_v)$ be vertices closest and farthest respectively from s. Note that an arc in S^i_v is chosen only if $D(u') \leq \alpha < D(w')$ thus the probability an arc in S^i_v is chosen is at most $(D(w') - D(u'))/D((tt))$.The numerator is upperbounded by the sum of the x^* value of the arc between u', (vv) and (vv), w'. Note that this sum, in turn is upperbounded by y^i_v due to Constraint (8). This implies the probability of downgrading v to level i is upperbounded by $(y^i_v)^*/D((tt)) \leq (y^i_v)^*(2k)$. Then,

[1] This is the major difference compared to NVDP where we no longer need to add dummy arcs nor rescale the weights x^*. The relevant arcs in H' for NVDP forms a *subdivision* of a star. After deleting all the backward arcs, it no longer remains as a star but could have disconnected arcs. Then, the range of these arcs is no longer bounded around (vv) and thus needed other modifications.

$$\sum_{v\in V(G)} \mathbb{E}[r_{L_\alpha(v)}(v)] \leq \sum_{v\in V(G)} \sum_{i=0}^{k} \mathbb{P}[v \text{ is downgraded to level } i] r_i(v)$$
$$\leq \sum_{v\in V(G)} \sum_{i=0}^{k} r_i(v)(y_v^i)^*(2k) \leq 2kb$$

where the last inequality is due to Constraint (6).

A similar result can be obtained for the expected cost of F_α. Consider an arc $[uv]_{i,j} \in X_\alpha$. Note that L_α might end up downgrading u, v beyond level i, j respectively. This implies the cost of cutting uv in the end is at most the cost $c_{i,j}(uv)$. Thus, $c^{L_\alpha}(F_\alpha) \leq \Sigma_{[uv]_{i,j}\in X_\alpha} c_{i,j}(uv)$. Then, by linearity of expectation, it suffices to calculate the probability of an arc $[uv]_{i,j} \in A(H')$ to end up in X_α. Using similar arguments as before, this happens only if $D((uv)_{i(k+1)+j}) \leq \alpha < D((uv)_{i(k+1)+j+1})$. Thus the probability is at most $x^*_{[uv]_{i,j}}/D((tt)) \leq 2kx^*_{[uv]_{i,j}}$. Then,

$$\mathbb{E}[c^{L_\alpha}(F_\alpha)] \leq \sum_{[uv]_{i,j}\in A(H')} c_{i,j}(uv)\mathbb{P}[[uv]_{i,j} \in X_\alpha]$$
$$\leq \sum_{[uv]_{i,j}\in A(H')} c_{i,j}(uv)2kx^*_{[uv]_{i,j}} \leq 2kopt^*$$

where opt^* is the objective value of LDLP. Then, by Markov's inequality and using a similar trick as NVDP by fixing $0 < \epsilon$, the probability that $\Sigma_{v\in V(G)} r_{L_\alpha(v)}(v) \leq (1+1/\epsilon)(2k)b$ and $c^{L_\alpha}(F_\alpha) \leq (1+\epsilon)(2k)opt^*$ are both independently at least $1/2$. Thus, there exists α such that L_α, F_α provides the promised guarantee. □

Note that when $k = 1$, the guarantee here is weaker than the one for NVDP. NVDP explicitly uses the fact that there are only two levels to its advantage and achieves a better trade-off between the downgrading cost and the cut cost. It would be of great interest if such exploitation also exists for the general NVLDP problem.

6 Hardness results

Hardness results [7, 8, 26, 28] for interdiction problems typically involve a reduction from the Densest k-subgraph problem which we define next.

Definition 2 Densest k Subgraph (DkS): Given an **undirected** graph G and an integer k, find a vertex subset $Y \subset V(G)$ such that $|Y| = k$ and it maximizes the number of edges induced by Y (i.e., with both ends in Y).

DkS is a not only NP-hard but is also believed to be hard to approximate. Under certain plausible complexity assumptions (such as it is hard to refute random 3-SAT instances [10] or that there does not exist randomized subexponential time algorithms that solve NP [19]), there does not exist a PTAS for DkS. Moreover, the current best approximation algorithm known for the problem [2] has approximation ratio $O(n^{\frac{1}{4}+\epsilon})$ in an n-node graph for any $\epsilon > 0$.

A summary of the hardness results is in the table below.

Problem	Hardness of	
	$(\alpha, 1)$-approximation	$(1, \beta)$-approximation
WNVIP	DkS-Hard (Shown in [7, 8])	DkS-Hard (Theorem 6)
NVIP	DkS-Hard (Theorem 1.9 in [8])	DkS-Hard (Theorem 7)
NVIP unit cost	DkS Hard (Appendix B in [8])	DkS-Hard (Theorem 8)

Chestnut and Zenklusen [7] and Chuzhoy et. al [8] showed the following hardness of unicriterion approximation of the cut value for the network interdiction problem. Even though they proved this result for NFI, by our earlier observation, this applies to WNVIP directly.

Theorem 5 ([7], Corollary 11) *If there is an* $(\alpha(n), 1)$*-approximation for WNVIP, then there is a* $2(\alpha(n^2))^2$*-approximation for DkS.*

We complement this to show a similar hardness of unicriterion approximation of the interdiction budget. As mentioned in the introduction, since there is a simple reduction from undirected WNVIP to the directed version, we focus on the undirected version.

Theorem 6 *If there exists a* $(1, \beta(n))$*-approximation for the undirected version of WNVIP, then there is a* $2(\beta(n^2))^2$*-approximation for DkS.*

Proof Our strategy will be very similar to the one in prior work [7, 26, 28]. Given an instance of DkS, without loss of generality, we may assume that G is connected and $k < |V(G)|$. Consider auxiliary graph H: $V(H) = V_s \cup V_t \cup \{s, t\}$ where $V_s = V(G)$ corresponding to the original vertices and $V_t = \{t_e : e \in E(G)\}$ corresponding to the edges of G. Then, we add the following edges: $\{sv : v \in V_s\}$, $\{t_e t : t_e \in V_t\}$, $\{vt_e : e = uv \in E(G)\}$. This graph H is equivalent to subdividing every edge of G then connecting s to all original vertices and connecting t to all subdivided edges. Note that $|V(H)| = |V(G)| + |E(G)| + 2 \leq n^2$, $|E(H)| = |V(G)| + 3|E(G)| \leq 3n^2$ for large n. $\square$

Now, consider the following interdiction and edge cost functions:

Vertex	s	$\in V_s$	$\in V_t$	t
$r(v)$	∞	1	∞	∞
Edges between	s, V_s	V_s, V_t	V_t, t	
$c(e)$	∞	∞	1	

Let $b = k$ and consider solving WNVIP on H. For any $Y \subseteq V(G)$, denote E_Y as the edges with both endpoints in Y.

Claim Let $(Y \subseteq V(H), F \subseteq E(H))$ be a solution to WNVIP. Then $|E_Y| = |E(G)| - |F|$.

Note that $Y \subseteq V_s = V(G)$ and F are only edges between V_t and t due to the costs. First, we will show that F is not incident to any t_e where $e \in E_Y$. Let $e = uv \in E_Y$. Note that in the graph H, the neighbours of t_e are u, v, t. This implies after interdicting Y, t_e is only adjacent to t, and hence need not be included in any minimal st cut. Next, we show that for every $e \notin E_Y$, $t_e t \in F$. Suppose $e \notin E_Y$. Then, it follows e is incident to some vertex $u \notin Y$. Note that $sut_e t$ is a path in $H \backslash Y$. Since the cost of su, ut_e is too expensive, it follows that $t_e t \in F$. Since the number edges between V_t and t is exactly $|E(G)|$, the claim follows.

Let c^* be the cost of the cut in an optimal solution to WNVIP and l^* is the number of edges in a densest k-subgraph of G. It follows from the above claim that $l^* = |E(G)| - c^*$. Suppose the approximation scheme produced an interdiction set $V' \subseteq V_s$ with a final cut E' and a cost of c' where $|V'| = \Sigma_{v \in V'} r(v) \leq \beta(|V(H)|)b = \beta(n^2)k$ and $c' \leq c^*$. Then, it follows from the claim that $|E_{V'}| = |E(G)| - c' \geq |E(G)| - c^* = l^*$. Now we can apply the following lemma from [7].

Lemma 8 *Given a graph H with n nodes and m edges, there exists a deterministic polynomial algorithm that produces a subgraph on k vertices with at least $\frac{k(k-1)}{n(n-1)}m$ edges for any $k \leq n$.*

By applying Lemma 8 on the subgraph induced by V', there exists a k-vertex subgraph with at least $\frac{k(k-1)}{\beta(n^2)k(\beta(n^2)k-1)} l^* \geq \frac{l^*}{2(\beta(n^2))^2}$ edges. Then, our theorem follows. □

Our goal in this section is to build on the proof of Theorem 6 to show hardness of NVIP with unit costs. We will do this in two steps: first we consider unitary interdiction costs with general edge cut costs. Then we also transform the edges so they have unit costs. This method allows us to show hardness for one of the unicriterion approximations.

Theorem 7 *If there exists a $(1, \beta(n))$-approximation for NVIP, then there is a $4(\beta(n^2))^2$-approximation for DkS.*

Note that this is sufficient to show a $(1, \beta)$-approximation is hard to obtain for NVIP.

Proof We once again consider the same auxiliary graph H as in the proof of Theorem 6. However, we consider the case of unit interdiction costs.

Vertex	s	$\in V_s$	$\in V_t$	t
$r(v)$	∞	1	1	∞
Edges between	s, V_s	V_s, V_t	V_t, t	
$c(e)$	∞	∞	1	

Our budget $b = k$. Note that we still forbid the interdiction of s and t. This is a natural condition to impose on NVIP. The main difference here compared to the proof of Theorem 6 is that we are now allowed to interdict vertices in V_t.

Suppose $Y \subseteq V(H)$, $F \subseteq E(H)$ is a $(1, \beta)$-approximation solution for NVIP on H. Then, consider an alternate interdiction set where we still interdict every vertex in $Y \cap V_s$ but instead of interdicting any $t_e \in Y \cap V_t$ corresponding to an edge $e = uv$, we interdict s_u and s_v, the vertices in V_S corresponding to u and v. Formally, let $\bar{Y} = (Y \cap V_s) \cup \{s_u, s_v : t_e \in Y \cap V_t, e = uv\}$. Note that $|\bar{Y}| \leq 2|Y|$. Thus, $\bar{Y}, F$ is a $(1, 2\beta)$-approximation for NVIP on H. Applying Claim 6, we can obtain a $2\beta(n^2)k$ vertex subgraph with at least l^* edges where l^* is the optimal number of edges in DkS. Then, the result follows from Lemma 8. □

We can further modify the graph H to show that we cannot obtain a good unicriterion approximation for NVIP even when we have unitary costs.

Theorem 8 *If there exists a* $(1, \beta(n))$*-approximation for NVIP with unit cut cost, then there exists a* $(4(\beta(2n^2))^2)$*-approximation for DkS.*

Proof We will use the same setup as Theorem 7. Recall, given graph G, we construct auxiliary graph H with the following cost functions:

Vertex	s	$\in V_s$	$\in V_t$	t
$r(v)$	∞	1	1	∞
Edges between	s, V_s	V_s, V_t	V_t, t	
$c(e)$	∞	∞	1	

There are two edge costs we need to modify, those between s, V_s and those between V_s, V_t. To take care of those incident to s, consider adding a large clique $S = K_{n^2}$ between s and V_s. Then, add every possible edge between s, S and similarly between S and V_s. Note that even after interdicting $b < n$ nodes in S, any cut through S involves $n^2 > |E(G)|$ many edges and thus any minimum solution will not interdict any vertices in S nor cut any edges in S.

For edges between V_s, V_t, we simply set their cost to 1 and claim that an st-minimum cut would not use any of those edges either. Let F be a minimum cut after interdicting some vertices Y. Suppose there exists $v \in V_s, t_e \in V_t$ such that $vt_e \in F$. Then, removing edge vt_e and adding t_et to F is still an st-cut. Therefore, we can assume that any min-cut after interdicting any set Y only consists of edges incident to t. Then, applying the same techniques in Theorem 7, our result follows. □

7 Simple approximations for interdiction problems

7.1 A simple bicriteria algorithm for WNVIP

We first provide a polynomial time algorithm that finds a $(1 + 1/\epsilon, 1)$ or $(1, 1 + \epsilon)$-approximation for any $\epsilon > 0$ for WNVIP in digraphs using a more direct proof than the earlier methods [3, 6]. This proof was also given by Chuzhoy et al. [8] in their study of k-route cuts, but we reproduce it since we build on it later. A priori, the algorithm will not be able to tell which guarantee it provides. Note that if $\epsilon = 1$, this easily provides a (2, 2) approximation.

Theorem 9 *For any $\epsilon > 0$, there exists a polynomial time algorithm that provides either a $(1 + 1/\epsilon, 1)$ or a $(1, 1 + \epsilon)$-approximation guarantee for WNVIP in directed graphs.*

Proof Given $\epsilon > 0$, let L be our guess for the value of $\lambda_{st}(G\backslash X^*)$ for the optimal interdiction set X^* that obeys $r(X^*) \leq b$. Since we can detect whether $L = 0$ (using weighted node connectivity computation) and output an interdiction set achieving this, we assume that L is nonzero for the rest of the proof. Also, note that L is bounded above by the weight of a minimum directed st-cut of G. Consider an auxiliary digraph G' and a weight function $w : A(G') \to \mathbb{R}_+$ where we subdivide every arc e with a vertex v_e. We assign weight $\epsilon c_e b/L$ to every new vertex v_e. Every original vertex v gets weight $r(v)$. If our guess L is correct, then G' contains an st-separating vertex cut of weight $(1+\epsilon)b$ (by using the nodes in an optimal interdicting set and the subdivided nodes of the corresponding min st-cut). Now, consider a minimum weighted vertex cut X' of G' that separates all directed paths from s to t. Let $X = X' \cap V(G)$ and $Y = X'\backslash X$ be those that correspond to subdivided arcs. Note that $(1+\epsilon)b \geq w(X') = r(X) + \Sigma_{e\in Y}\epsilon c_e b/L$. In particular, $r(X) \leq (1 + \epsilon)b$ and $\Sigma_{v_e\in Y} c_e \leq (1 + 1/\epsilon)L$. Furthermore, if $r(X) > b$, then it follows that $\Sigma_{e\in Y} c_e < L$. This implies if L is the correct guess, then (X, Y) is either a $(1 + 1/\epsilon, 1)$ or a $(1, 1 + \epsilon)$-approximation.[2] □

7.2 Interdicting multiway cuts

Problem 4 Weighted Multiway Cut Vertex Interdiction Problem (WMWIP) Let G be a directed graph, $S = \{s_1, ..., s_k\} \subseteq V(G)$ and every arc e has a non-negative weight c_e and every vertex v has a non-negative interdiction cost $r(v)$. Given a budget b, find vertices X of interdiction cost at most b to interdict that minimizes the cost of separating the vertices in S. In other words, find $X \subseteq V(G)$, $F \subseteq E(G)$ such that there is no path from any vertex $s_i \in S$ to any other $s_j \in S$ in G after deleting X and F. Furthermore, $r(X) \leq b$ and $\Sigma_{e\in F} c_e$ is minimized. We will assume that the demand vertices S cannot be interdicted or equivalently that $r(s) = \infty$ for $s \in S$.

Related Work. The multiway cut problem is another natural extension of the minimum st-cut problem, and involves finding a minimum set of edges to delete in order to separate k given terminals from each other. For the problem in undirected graphs, using a geometric relaxation, Calinescu, Karloff and Rabani [4] achieved an approximation factor 3/2. Sharma and Vondrak [25] gave the current best approximation factor of 1.2965. In the vertex version of the Multiway Cut Problem, instead of deleting edges, we delete vertices to separate the k terminals, but the graph is still undirected. By

[2] We can simply try all possible L values using binary search to find the smallest value for which the condition holds. To make this search polynomial time, we can simply search over multiplicative powers of $(1 + \epsilon')$ for some small $\epsilon' > 0$ starting from 1 and up to the maximum possible cost of a cut using $\log_{1+\epsilon'} mc_{max}$ trials where m is the number of arcs and c_{max} is the largest arc cost.

subdividing edges and introducing new nodes with costs, it is easy to see that the vertex version generalizes the edge version. Garg et al. [12] gave a 2-approximation for this vertex Multiway Cut in undirected graphs by showing half-integrality of the optimal solution of a natural LP relaxation. Finally, the directed version of the problem involves deleting arcs so that there are no directed paths between any pair of terminals. It is easy to see that the vertex version of this directed problem can be reduced back to the arc version by the usual splitting of nodes into two copies: one for supporting incoming and the other for outgoing arcs with the node cost assigned to the arc from the in- to the out-copy. Naor and Zosin [21] gave the first 2-approximation algorithm for the directed multiway cut problem using a sophisticated LP formulation and rounding. Chekuri and Madan [5] later gave a simple ball growing based LP rounding algorithm for the problem that also gives a $2(1-\frac{1}{k})$ approximation where k is the number of terminals. We can adapt either algorithm to give our results for the vertex interdiction variant of multiway cuts in directed graphs.

Using similar techniques as for WNVIP, we prove the following result for the vertex interdiction version of the directed multiway cut problem.

Theorem 10 *For any $\epsilon > 0$, there exists a polynomial time algorithm that gives a $(4, 4(1+\epsilon))$-approximation to Weighted Multiway Cut Vertex Interdiction Problem (WMWIP) in directed graphs.*

Proof Let opt be the cost of an optimal solution to WMWIP. First delete all arcs of cost 0 since they are free to cut. Then, apply the 2-approximation algorithm for the node-weighted version of the directed multiway cut problem on G disregarding any edge costs. If $opt = 0$, then the algorithm should produce a set of vertices to interdict whose cost is at most $2b$ and separates all terminals from each other.

Now, we assume that $opt > 0$ and we now make a guess of a close range bounding opt. First, by scaling the costs with an appropriate factor, we may assume that all arc costs are at least 1. Fix a constant $\epsilon > 0$ and let $q = \lceil \log(\sum_{uv \in A} c_{uv}) / \log(1+\varepsilon) \rceil$. Define the sequence $U_i = (1+\varepsilon)^i$ for $i = 0, \ldots, q$. By the choice of q, we have $(1+\varepsilon)^q > \sum_{uv \in A} c_{uv}$. Clearly, $U_0 \le opt < U_q$. We improve these bounds by guessing an index i_0 such that $U_{i_0-1} \le opt \le U_{i_0}$. It follows that the following algorithm tries all $U_{i_0} = U_i$ for $i = 1, \ldots, q$ and returns the best result.

Consider an auxiliary graph G' and a weight function $w : A(G') \to \mathbb{R}_+$ where we subdivide every arc e with a vertex v_e. We assign weight $\frac{c_e b}{U_{i_0}}$ to every new vertex v_e. Every original vertex v gets weight $r(v)$. All original arcs are assumed to have infinite weight. Then G' contains a node multiway cut of weight $2b$ (by using the nodes in an optimal interdicting set and the subdivided nodes of the corresponding minimum multiway cut). Recall that the vertex version of this directed problem can be reduced back to the arc version by the usual splitting of nodes into two copies: one for supporting incoming and the other for outgoing arcs with the node cost assigned to the arc from the in- to the out-copy. Now, we use any existing 2-approximation for directed multiway cut [5, 21] to find a 2-approximation to the minimum weighted vertex multiway cut to get the set X' of G'. Let $X = X' \cap V(G)$ and $Y = X' \backslash X$ be those that corresponds to subdivided arcs. Note that $4b \ge w(X') = r(X) + \Sigma_{e \in Y} \frac{c_e b}{U_{i_0}}$. In particular, $r(X) \le 4b$ and $\Sigma_{e \in Y} c_e \le 4U_{i_0} \le 4(1+\varepsilon)opt$. $\square$

7.3 Interdicting multicuts using vertices

Problem 5 Weighted MultiCut Vertex Interdiction Problem (WMVIP) Let G be an **undirected** graph where every edge e has a non-negative weight c_e and every node v has a non-negative interdiction cost $r(v)$, and we are given demand pairs of vertices $\{s_1, t_1\}, \ldots, \{s_k, t_k\}$. Given a budget b, find nodes X of interdiction cost at most b to interdict that minimizes the cost of separating the demand pairs in the resulting graph. In other words, find $X \subseteq V(G)$, $F \subseteq E(G)$ such that for every demand pair $\{s_i, t_i\}$, vertices s_i and t_i are in different connected components of G after deleting the nodes in X and the edges in F. Furthermore, $r(X) \leq b$, and $\Sigma_{e \in F} c_e$ is minimized.

We will assume that the demand-pair vertices cannot be interdicted or equivalently that $r(v) = \infty$ for $v \in \{s_i, t_i\}$ for any demand pair i, and call such vertices terminals.

Related Work. In the multicut problem in **undirected** graphs, we are given k source-sink pairs and a multicut puts every source sink pair in different connected components. The minimum multicut problem has a well-known $2 \ln k$-approximation algorithm [11] using an LP-rounding method. An alternate proof of this result using the ideas of Calinescu et al. [4] uses a randomized Dijkstra-like ball growing and cutting method (see e.g., [15]) that we adapt in designing our approximation algorithm for the vertex interdiction variant of this problem. For completeness, we remark that multicuts in directed graphs are not that well approximable with the best known approximation ratio being $O(n^{\frac{11}{23}})$ [1] in n-node digraphs.

This subsection focuses on applying a similar technique as above on WMVIP to prove the following theorem.

Theorem 11 *For any $\epsilon > 0$, there exists a randomized polynomial time algorithm that gives a $(2(1+\epsilon)\ln k, 2(1+\epsilon)\ln k)$-approximation to Weighted MultiCut Vertex Interdiction Problem (WMVIP) in* **undirected graphs***, where k is the number of terminal pairs.*

Our strategy is to formulate and solve a linear programming relaxation of the problem by adapting the multicut formulation by incorporating node interdiction variables. We then employ a ball-growing based rounding technique used for deriving a logarithmic approximation algorithm by first transferring the LP values on the nodes to all their adjacent edges. We then observe that this transformation does not degrade the quality of the final approximation more than the claimed amount.

Consider the following linear programming relaxation for WMVIP:

$$
\begin{aligned}
\max \quad & \sum_{e \in E(G)} c_e x_e \\
\text{s.t.} \quad & \sum_{e \in E(P)} x_e + \sum_{v \in V(P)} y_v \geq 1 && \forall s_i t_i \text{ path } P \ \forall\, i \\
& \sum_{v \in V(G)} r(v) y_v \leq b
\end{aligned}
$$

$$x_e, y_v \geq 0 \qquad \forall e \in E(G), v \in V(G)\backslash S$$
$$y_v = 0 \qquad \forall \text{ terminals } v$$

Note that an optimal solution of the multicut interdiction problem (WMVIP) is a feasible solution to the above LP. Let x^*, y^* be an optimal solution to this LP and let opt^* be the optimal value. Note that if we define the distance between any two points u, v as $\min_{uv-\textbf{path } P} \Sigma_{e\in E(P)} x_e + \Sigma_{w\in V(P)\backslash\{u,v\}} y_w$, then every s_i, t_i pair is at least 1 unit apart from each other. We will now construct an auxiliary graph G' that transfers all weights on vertices to edges and preserves this distance.

To construct the auxiliary graph G', for every edge $e = uv$, subdivide it twice with vertices u_v, v_u and give weights $y_u/2, x_e, y_v/2$ to $uu_v, u_v v_u, v_u v$ respectively. This transformation can be viewed as replacing every vertex v with a star with center v and leaves v_u for every $u \in N(v)$. Note that the length of any path is clearly preserved. Thus every s_i, s_j pair is still distance 1 apart. Let $d(u, v)$ denote the distance between two vertices u, v. Let edges of the form $uu_v, v_u v$ be called *vertical* edges since they relate to the original vertex weights y_u, y_v. Denote all other edges of the form $u_v v_u$ as *true* edges.

Consider the following Dijkstra-like Ball Growing Algorithm for multicut [15] inspired by [4]: Choose a random permutation of the demand pairs and reindex the pairs according to this random order. Next, randomly choose a number r between $(0, 1)$. In increasing order i of the terminal pairs in this random ordering, draw a ball of radius r centered at s_i. Then, cut all edges at the boundary of the ball around s_i that are not cut or not entirely contained inside the ball around s_j for some $j < i$. In other words, an edge $e = uv$ is chosen as part of the cut if and only if there exists $1 \leq i \leq k$ such that $d(u, s_j), d(v, s_j) > r$ for all $j < i$ in the random permutation order of the terminal pairs and moreover, either $d(u, s_i) \leq r < d(v, s_i)$ or $d(v, s_i) \leq r < d(u, s_i)$. Then given r and the random order of the demand pairs, let F be the set of edges in G that corresponds to true edges chosen in the cut and let X be the set of vertices of G that corresponds to vertical edges in the cut.

Note that the resulting X, F does provide an integral solution to the WMVIP problem. Now, we calculate the expected cost of F and X.

Lemma 9 *The probability that a true edge $u_v v_u$ is chosen in the cut is at most $x_e \ln k$.*

Proof Let $e = u_v v_u$ be a true edge. For a demand pair (s, t), denote the distance of the edge e to the terminal s as $\min\{d(u_v, s), d(v_u, s)\}$. Then, we rank the demand pairs (s'_i, t'_i) based on the distance of e to the terminals s_i. Suppose that s'_i is the i-th closest terminal to e.

Without loss of generality, assume $d(u_v, s'_i) \leq d(v_u, s'_i)$. Then, e is on the boundary of the ball around s'_i with radius r if and only if $d(u_v, s'_i) < r < d(v_u, s'_i)$ which happens with probability $d(v_u, s'_i) - d(u_v, s'_i) \leq x_e$. Consider $j < i$ so e is closer to s'_j than s'_i. Suppose e is on the boundary of the ball around s'_i and thus $r > d(s'_i, u_v)$. Then r is larger than the distance to s'_j. This implies e is either inside or on the boundary of the ball around s'_j with radius r. Then e cannot be chosen as part of the cut by s'_i. Thus e is chosen in the cut by s'_i only if s'_i is chosen in the random ordering after s'_j

for all $j < i$. Then, the probability that e is chosen in the cut by s'_i is at most $x_e \times \frac{1}{i}$. Then, the total probability of e being chosen in the cut is at most $\Sigma_{i=1}^{k} \frac{x_e}{i} \leq x_e \ln k$. □

A similar result can be derived about interdicting a vertex.

Lemma 10 *The probability that v is interdicted is at most $y_v \ln k$.*

Proof Define $\delta_v = \min_{u \in N(v)}\{d(v_u, s)\}$ as the distance between v and a terminal s. Once again, we can rank the terminals based on their distance to v and thus assume that s'_i is the i-th closest terminal to v. Note that the maximum distance between any two vertices incident to v is y_v due to the path via v.

For similar reasons as before, the probability of a vertical edge of v to lie on the boundary of a ball centered around s'_i is y_v. However, it is chosen in the cut due to s'_i only if s'_i appears in the ordering after s'_j for all $j < i$. Then, by similar reasoning, the probability that v is interdicted is at most $y_v \ln k$. □

Now we can prove our main theorem.

Proof (Theorem 11) Note that $\mathbb{E}[\Sigma_{e \in F} c_e] = \Sigma_{e \in E(G)} c_e Pr[e \in F]$ is at most $\Sigma_{e \in E(G)} c_e x_e \ln k = opt^* \ln k$. Similarly, $E[\Sigma_{v \in X} r(v)] = \Sigma_{v \in V(G)} r(v) Pr(v \in X)$ is at most $\Sigma_{v \in V(G)} r(v) y \ln k_v = b \ln k$. Then by Markov's inequality, the probability that the corresponding F, X satisfy $\Sigma_{e \in F} c_e \leq 2(1+\epsilon) opt^* \ln k$, $\Sigma_{v \in X} r(v) \leq 2(1+\epsilon) b \ln k$ is $1 - \frac{2}{2(1+\epsilon)} = \frac{\epsilon}{1+\epsilon}$. Thus, we can find a desirable cut in polynomial time. □

8 Conclusion

We have introduced a new class of network downgrading problems inspired by extending interdiction problems from arcs to vertices in graphs. While we mainly studied cut problems in networks, the downgrading model can also be adapted to study other network problems such as connectivity structures like spanning and Steiner trees, matchings and other subgraphs for which the interdiction problem models a plausible application. Our results extend the work on interdiction of cuts by using new relaxations that are able to exploit previously used rounding techniques after appropriate adaptations. We hope a similar approach can be useful in future work on these other subgraph downgrading problems.

Acknowledgements We thank the two anonymous reviewers whose comments/suggestions helped improve and clarify this manuscript. We especially thank an anonymous reviewer for pointing out the reduction given in Sect. 3.2.

References

1. Agarwal, A., Alon, N., Charikar, M.S.: Improved approximation for directed cut problems. In: Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing. ACM, pp. 671–680 (2007)

2. Bhaskara, A., Charikar, M., Chlamtac, E., Feige, U., Vijayaraghavan, A.: Detecting high log-densities: an $\mathcal{O}(n^{\frac{1}{4}})$ approximation for densest k-subgraph. In: Proceedings of the Forty-Second ACM Symposium on Theory of Computing. ACM, pp. 201–210 (2010)
3. Burch, C., Carr, R., Krumke, S., Marathe, M., Phillips, C., Sundberg, E.: A decomposition-based pseudoapproximation algorithm for network flow inhibition. In: Woodruff, D.L. (ed.) Network Interdiction and Stochastic Integer Programming, vol. 26, pp. 51–68. Springer, Berlin (2003)
4. Călinescu, G., Karloff, H., Rabani, Y.: An improved approximation algorithm for multiway cut. J. Comput. Syst. Sci. **60**(3), 564–574 (2000)
5. Chekuri, C., Madan, V.: Simple and fast rounding algorithms for directed and node-weighted multiway cut. In: Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms. SIAM, pp. 797–807 (2016)
6. Chestnut, S.R., Zenklusen, R.: Interdicting structured combinatorial optimization problems with {0, 1}-objectives. Math. Oper. Res. **42**(1), 144–166 (2016)
7. Chestnut, S.R., Zenklusen, R.: Hardness and approximation for network flow interdiction. Networks **69**(4), 378–387 (2017)
8. Chuzhoy, J., Makarychev, Y., Vijayaraghavan, A., Zhou, Y.: Approximation algorithms and hardness of the k-route cut problem. ACM Trans. Algorithms (TALG) **12**(1), 2 (2016)
9. Chuzoy, J.: Flows, cuts and integral routing in graphs—an approximation algorithmist's perspective. In: Proceedings of of the International Congress of Mathematicians
10. Feige, U.: Relations between average case complexity and approximation complexity. In: Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing. ACM, pp. 534–543 (2002)
11. Garg, N., Vazirani, V.V., Yannakakis, M.: Approximate max-flow min-(multi) cut theorems and their applications. SIAM J. Comput. **25**(2), 235–251 (1996)
12. Garg, N., Vazirani, V.V., Yannakakis, M.: Multiway cuts in node weighted graphs. J. Algorithms **50**(1), 49–61 (2004)
13. Golden, B.: A problem in network interdiction. Naval Res. Log. Q. **25**(4), 711–713 (1978)
14. Guenin, B., Könemann, J., Tuncel, L.: A Gentle Introduction to Optimization. Cambridge University Press, Cambridge (2014)
15. Gupta, A., O'Donnell, R.: Lecture 18 on Multicuts. Lecture Notes for 15-854(B): Advanced Approximation Algorithms, Spring (2008)
16. Guruganesh, G., Sanita, L., Swamy, C.: Improved region-growing and combinatorial algorithms for k-route cut problems. In: Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, pp. 676–695 (2015)
17. Harris, T.E., Ross, F.S.: Fundamentals of a method for evaluating rail net capacities. Technical report, Santa Monica, California (1955)
18. Israeli, E., Wood, R.K.: Shortest-path network interdiction. Netw. Int. J. **40**(2), 97–111 (2002)
19. Khot, S.: Ruling out PTAS for graph min-bisection, dense k-subgraph, and bipartite clique. SIAM J. Comput. **36**(4), 1025–1071 (2006)
20. Linhares, A., Swamy, C.: Improved algorithms for MST and metric-tsp interdiction. In: Proceedings of 44th International Colloquium on Automata, Languages, and Programming, vol. 32, pp. 1–14 (2017)
21. Naor, J., Zosin, L.: A 2-approximation algorithm for the directed multiway cut problem. SIAM J. Comput. **31**(2), 477–482 (2001)
22. Papadimitriou, C.H., Yannakakis, M.: On the approximability of trade-offs and optimal access of web sources. In: Proceedings 41st Annual Symposium on Foundations of Computer Science. IEEE, pp. 86–92 (2000)
23. Phillips, C.A.: The network inhibition problem. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, STOC'93, New York, NY, USA. ACM, pp. 776–785 (1993)
24. Schrijver, A.: On the history of the transportation and maximum flow problems. Math. Program. **91**(3), 437–445 (2002)
25. Sharma, A., Vondrák, J.: Multiway cut, pairwise realizable distributions, and descending thresholds. In: STOC (2014)
26. Wood, R.: Deterministic network interdiction. Math. Comput. Model. **17**(2), 1–18 (1993)
27. Zenklusen, R.: Matching interdiction. Discrete Appl. Math. **145**(15), (2010)
28. Zenklusen, R.: Network flow interdiction on planar graphs. Discrete Appl. Math. **158**(13), (2010)
29. Zenklusen, R.: Connectivity interdiction. Oper. Res. Lett. **42**(67), 450–454 (2014)

30. Zenklusen, R.: An $\mathcal{O}(1)$ approximation for minimum spanning tree interdiction. In: Proceedings of 56th Annual IEEE Symposium on Foundations of Computer Science, pp. 709–728 (2015)