Global and Fixed-Terminal Cuts in Digraphs*†

Kristóf Bérczi[1], Karthekeyan Chandrasekaran[2], Tamás Király[3], Euiwoong Lee[4], and Chao Xu[5]

1 **MTA-ELTE Egerváry Research Group, Budapest, Hungary**
berkri@cs.elte.hu
2 **University of Illinois, Urbana-Champaign, IL, USA**
karthe@illinois.edu
3 **MTA-ELTE Egerváry Research Group, Budapest, Hungary**
tkiraly@cs.elte.hu
4 **Carnegie Mellon University, Pittsburgh, PA, USA**
euiwoonl@cs.cmu.edu
5 **University of Illinois, Urbana-Champaign, IL, USA**
chaoxu3@illinois.edu

Abstract

The computational complexity of multicut-like problems may vary significantly depending on whether the terminals are fixed or not. In this work we present a comprehensive study of this phenomenon in two types of cut problems in directed graphs: double cut and bicut.

1. Fixed-terminal edge-weighted double cut is known to be solvable efficiently. We show that fixed-terminal node-weighted double cut cannot be approximated to a factor smaller than 2 under the Unique Games Conjecture (UGC), and we also give a 2-approximation algorithm. For the global version of the problem, we prove an inapproximability bound of 3/2 under UGC.
2. Fixed-terminal edge-weighted bicut is known to have an approximability factor of 2 that is tight under UGC. We show that the global edge-weighted bicut is approximable to a factor strictly better than 2, and that the global node-weighted bicut cannot be approximated to a factor smaller than 3/2 under UGC.
3. In relation to these investigations, we also prove two results on undirected graphs which are of independent interest. First, we show NP-completeness and a tight inapproximability bound of 4/3 for the node-weighted 3-cut problem under UGC. Second, we show that for constant k, there exists an efficient algorithm to solve the minimum $\{s,t\}$-separating k-cut problem.

Our techniques for the algorithms are combinatorial, based on LPs and based on the enumeration of approximate min-cuts. Our hardness results are based on combinatorial reductions and integrality gap instances.

1998 ACM Subject Classification G.2.2 Graph Theory

Keywords and phrases Directed Graphs, Arborescence, Graph Cuts, Hardness of Approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.2

1 Introduction

The minimum two-terminal cut problem (min $s-t$ cut) and its global variant (min cut) are classic interdiction problems with fast algorithms. Generalizations of the fixed-terminal

* A full version of the paper is available at https://arxiv.org/abs/1612.00156.
† Kristóf and Tamás are supported by the Hungarian National Research, Development and Innovation Office – NKFIH grants K109240 and K120254. Chao is supported in part by NSF grant CCF-1526799.

variant, including the multi-cut and the multi-way cut, as well as generalizations of the global variant, including the k-cut, have been well-studied in the algorithmic literature [10, 14]. In this work, we study two generalizations of global cut problems to directed graphs, namely double cut and bicut (that we describe below). We study the power and limitations of fixed terminal variants of these cut problems in order to solve the global variants. In the process, we examine "intermediate" multicut problems where only a subset of the terminals are fixed, and obtain results of independent interest. In particular, we show that the undirected $\{s,t\}$-separating k-cut problem, where two of the k terminals are fixed, is polynomial-time solvable for constant k. In what follows, we describe the problems along with the results. We refer the reader to Tables 1, 2, and 3 at the end of Section 1.1 for a summary of the results. We mention that all our algorithmic/approximation results hold for the min-cost variant while the inapproximability results hold for the min-cardinality variant by standard modification of our reductions and algorithms. For ease of presentation, we do not make this distinction.

The starting point of this work is node-weighted double cut, that we describe below. We recall that an arborescence in a directed graph $D = (V, E)$ is a minimal subset $F \subseteq E$ of arcs such that there exists a node $r \in V$ with every node $u \in V$ having a unique path from r to u in the subgraph (V, F) (e.g., see [26]).

Double Cut. The input to the NodeDoubleCut problem is a directed graph and the goal is to find the smallest number of nodes whose deletion ensures that the remaining graph has no arborescence. NodeDoubleCut is a generalization of node weighted global min cut in undirected graphs to directed graphs. It is non-monotonic under node deletion. This problem is key to understanding fault tolerant consensus in networks. We briefly describe this connection.

Significance of double cut. In a recent work, Tseng and Vaidya [28] showed that *consensus* in a directed graph can be achieved in the *synchronous model* subject to the failure of f nodes *if and only if* the removal of any f nodes still leaves an arborescence in the remaining graph. Thus, the number of nodes whose failure can be tolerated for the purposes of achieving consensus in a network is *exactly* one less than the smallest number of nodes whose removal ensures that there is no arborescence in the network. So, it is imperative for the network authority to be able to compute this number.

A directed graph $D = (V, E)$ has no arborescence if and only if [1] there exist two distinct nodes $s, t \in V$ such that every node $u \in V$ can reach at most one node in $\{s, t\}$. By this characterization, every directed graph that does not contain a tournament has a feasible solution to NodeDoubleCut. This characterization motivates the following fixed-terminal variant, denoted $\{s,t\}$-NodeDoubleCut: Given a directed graph with two specified nodes s and t, find the smallest number of nodes whose deletion ensures that every remaining node u can reach at most one node in $\{s, t\}$ in the resulting graph. An instance of $\{s,t\}$-NodeDoubleCut has a feasible solution provided that the instance has no edge between s and t. An efficient algorithm to solve/approximate $\{s,t\}$-NodeDoubleCut immediately gives an efficient algorithm to solve/approximate NodeDoubleCut.

[1] We believe that this characterization led earlier authors [3] to coin the term *double cut* to refer to the edge deletion variant of the problem. We are following this naming convention.

Edge-weighted case. In the edge-weighted version of the problem, $\{s,t\}$-EdgeDoubleCut, the goal is to delete the smallest number of edges to ensure that every node in the graph can reach at most one node in $\{s,t\}$. Similarly, in the global variant, denoted EdgeDoubleCut, the goal is to delete the smallest number of edges to ensure that there exist nodes s,t such that every node u can reach at most one node in $\{s,t\}$, i.e. the graph has no arborescence. The fixed-terminal variant $\{s,t\}$-EdgeDoubleCut is solvable in polynomial time using maximum flow and, consequently, EdgeDoubleCut is also solvable in polynomial time (see e.g. [3]).

Results for double cut. Our main result on the fixed-terminal variant, namely $\{s,t\}$-NodeDoubleCut, is the following hardness of approximation.

▶ **Theorem 1.** *$\{s,t\}$-NodeDoubleCut is NP-hard, and has no efficient $(2-\epsilon)$-approximation for any $\epsilon > 0$ assuming the Unique Games Conjecture.*

We also give a 2-approximation algorithm for $\{s,t\}$-NodeDoubleCut, which leads to a 2-approximation for the global variant.

▶ **Theorem 2.** *There exists an efficient 2-approximation algorithm for $\{s,t\}$-NodeDoubleCut and NodeDoubleCut.*

While we are aware of simple combinatorial algorithms to achieve the 2-approximation for $\{s,t\}$-NodeDoubleCut, we present an LP-based algorithm since it also helps to illustrate an integrality gap instance which is the main tool underlying the hardness of approximation (Theorem 1) for the problem. Next we focus on the complexity of NodeDoubleCut. We note that the NP-hardness of the fixed-terminal variant does not necessarily mean that the global variant is also NP-hard.

▶ **Theorem 3.** *NodeDoubleCut is NP-hard, and has no efficient $(3/2-\epsilon)$-approximation for any $\epsilon > 0$ assuming the Unique Games Conjecture.*

Bicuts offer an alternative generalization of min cut to directed graphs. The approximability of the fixed-terminal variant of bicut is well-understood while the complexity of the global variant is unknown. In the following we describe these bicut problems and exhibit a dichotomic behaviour between the fixed-terminal and the global variant.

Bicut. The edge-weighted two-terminal bicut, denoted $\{s,t\}$-EdgeBiCut, is the following: Given a directed graph with two specified nodes s and t, find the smallest number of edges whose deletion ensures that s cannot reach t and t cannot reach s in the resulting graph. Clearly, $\{s,t\}$-EdgeBiCut is equivalent to 2-terminal multiway-cut (the goal in k-terminal multiway cut is to delete the smallest number of edges to ensure that the given k terminals cannot reach each other). This problem has a rich history and has seen renewed interest in the last few months culminating in inapproximability results matching the best-known approximability factor: $\{s,t\}$-EdgeBiCut admits a 2-factor approximation (by simple combinatorial techniques) and has no efficient $(2-\epsilon)$-approximation assuming the Unique Games Conjecture [19, 5]. In the global variant, denoted EdgeBiCut, the goal is to find the smallest number of edges whose deletion ensures that there exist two distinct nodes s and t such that s cannot reach t and t cannot reach s in the resulting digraph.

The dichotomy between global cut problems and fixed-terminal cut problems in undirected graphs is well-known. For concreteness, we recall Edge-3-Cut and Edge-3-way-Cut. In Edge-3-Cut, the goal is to find the smallest number of edges to delete so that the resulting

graph has at least 3 connected components. In Edge-3-way-Cut, the input is an undirected graph with 3 specified nodes and the goal is to find the smallest number of edges to delete so that the resulting graph has at least 3 connected components with at most one of the 3 specified nodes in each. While Edge-3-way-Cut is NP-hard [10], Edge-3-Cut is solvable efficiently [14]. However, such a dichotomy is unknown for directed graphs. In particular, it is unknown whether EdgeBiCut is solvable efficiently. Our next result shows evidence of such a dichotomic behaviour.

Results for bicut. While $\{s,t\}$-EdgeBiCut is inapproximable to a factor better than 2 assuming UGC, we show that EdgeBiCut is approximable to a factor strictly better than 2.

▶ **Theorem 4.** *There exists an efficient* $(2 - 1/448)$*-approximation algorithm for* EdgeBiCut.

We also consider the node-weighted variant of bicut, denoted NodeBiCut: Given a directed graph, find the smallest number of nodes whose deletion ensures that there exist nodes s and t such that s cannot reach t and t cannot reach s in the resulting graph. Every directed graph that does not contain a tournament has a feasible solution to NodeBiCut. NodeBiCut is non-monotonic under node deletion, and it admits a 2-approximation by a simple reduction to $\{s,t\}$-EdgeBiCut. We show the following inapproximability result.

▶ **Theorem 5.** NodeBiCut *is NP-hard, and has no efficient* $(3/2 - \epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

We observe that our approximability and inapproximability factors for NodeDoubleCut and NodeBiCut coincide – 2 and $(3/2 - \epsilon)$ respectively (Theorems 2, 3 and 5).

1.1 Additional Results on Sub-problems and Variants

In what follows, we describe additional results that concern sub-problems in our algorithms/hardness results, and also variants of these sub-problems which are of independent interest.

Node weighted 3-Cut. We show the NP-hardness of NodeDoubleCut in Theorem 3 by a reduction from the node-weighted 3-cut problem in undirected graphs. In the node weighted 3-cut problem, denoted Node-3-Cut, the input is an undirected graph and the goal is to find the smallest subset of nodes whose deletion leads to at least 3 connected components in the remaining graph. A classic result of Goldschmidt and Hochbaum [14] showed that the edge-weighted variant, denoted Edge-3-Cut (see above for definition) – more commonly known as 3-cut – is solvable in polynomial time. Intriguingly, the complexity of Node-3-Cut remained open until now. We present the first results on the complexity of Node-3-Cut.

▶ **Theorem 6.** Node-3-Cut *is NP-hard, and has no efficient* $(4/3 - \epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

The inapproximability factor of 4/3 mentioned in the above theorem is tight: the 4/3-approximation factor can be achieved by guessing 3 terminals that are separated and then using well-known approximation algorithms to solve the resulting node-weighted 3-terminal cut instance [13].

$(s, *, t)$-EdgeLin3Cut. As a sub-problem in the algorithm for Theorem 4, we consider the following, denoted $(s, *, t)$-EdgeLin3Cut (abbreviating edge-weighted linear 3-cut): Given a directed graph $D = (V, E)$ and two specified nodes $s, t \in V$, find the smallest number of edges to delete so that there exists a node r with the property that s cannot reach r and t, and r cannot reach t in the resulting graph. This problem is a global variant of (s, r, t)-EdgeLin3Cut, introduced in [11], where the input specifies three terminals s, r, t and the goal is to find the smallest number of edges whose removal achieves the property above. A simple reduction from Edge-3-way-Cut shows that (s, r, t)-EdgeLin3Cut is NP-hard. The approximability of (s, r, t)-EdgeLin3Cut was studied by Chekuri and Madan [5]. They showed that the inapproximability factor coincides with the flow-cut gap of an associated *path-blocking linear program* assuming the Unique Games Conjecture.

There exists a simple combinatorial 2-approximation algorithm for (s, r, t)-EdgeLin3Cut. A 2-approximation for $(s, *, t)$-EdgeLin3Cut can be obtained by guessing the terminal r and using the above-mentioned approximation. For our purposes, we need a strictly better than 2-approximation for $(s, *, t)$-EdgeLin3Cut; we obtain the following improved approximation factor.

▶ **Theorem 7.** *There exists an efficient* $3/2$*-approximation algorithm for* $(s, *, t)$-EdgeLin-3Cut.

$\{s, t\}$-SepEdge$k$Cut. In contrast to (s, r, t)-EdgeLin3Cut, we do not have a hardness result for $(s, *, t)$-EdgeLin3Cut. Upon encountering cut problems in directed graphs, it is often insightful to consider the complexity of the analogous problem in undirected graphs. Our next result shows that the following analogous problem in undirected graphs is in fact solvable in polynomial time: given an undirected graph with two specified nodes s, t, remove the smallest subset of edges so that the resulting graph has at least 3 connected components with s and t being in different components. More generally, we consider $\{s, t\}$-SepEdge$k$Cut, where the goal is to delete the smallest subset of edges from a given undirected graph so that the resulting graph has at least k connected components with s and t being in different components. The complexity of $\{s, t\}$-SepEdge$k$Cut was posed as an open problem by Queyranne [25]. We show that $\{s, t\}$-SepEdge$k$Cut is solvable in polynomial-time for every constant k.

▶ **Theorem 8.** *For every constant* k*, there is an efficient algorithm to solve* $\{s, t\}$-SepEdge-kCut.

$\{s, *\}$-EdgeBiCut. While Theorem 4 shows that EdgeBiCut is approximable to a factor strictly smaller than 2, we do not have a hardness result. We could prove hardness for the following intermediate problem, denoted $\{s, *\}$-EdgeBiCut: Given a directed graph with a specified node s, find the smallest number of edges to delete so that there exists a node t such that s cannot reach t and t cannot reach s in the resulting graph. $\{s, *\}$-EdgeBiCut admits a 2-approximation by guessing the terminal t and then using the 2-approximation for $\{s, t\}$-EdgeBiCut. We show the following inapproximability result:

▶ **Theorem 9.** $\{s, *\}$-EdgeBiCut *is NP-hard, and has no efficient* $(4/3 - \epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

Due to space constraints, we outline our techniques for the proof of Theorem 4 and for the hardness of approximation results in Sections 2 and 3, and refer the reader to the complete version of this work [2] for all complete proofs. The proofs of Theorems 7 and 8 are presented in Section 4.

Table 1 Global Variants in Directed Graphs. Text in gray refer to known results while text in black refer to the results from this work. All hardness of approximation results are under UGC. Hardness results for Node weighted $(s, *, t)$-Lin3Cut are based on the fact that it is as hard to approximate as Node weighted $\{s, t\}$-Sep3Cut by bidirecting the edges (Table 3).

Problem	Edge-deletion	Node-deletion
DoubleCut	Poly-time [3]	2-approx (Thm 2) $(3/2 - \epsilon)$-inapprox (Thm 3)
BiCut	$(2 - 1/448)$-approx (Thm 4)	2-approx $(3/2 - \epsilon)$-inapprox (Thm 5)
$(s, *)$-BiCut	2-approx $(4/3 - \epsilon)$-inapprox (Thm 9)	2-approx $(3/2 - \epsilon)$-inapprox
$(s, *, t)$-Lin3Cut	3/2-approx (Thm 7)	2-approx $(4/3 - \epsilon)$-inapprox

Table 2 Fixed-Terminal Variants in Directed Graphs. Text in gray refer to known results while text in black refer to the results from this work. All hardness of approximation results are under UGC. We include $\{s, t\}$-BiCut and (s, r, t)-Lin3Cut for comparison with the global variants in Table 1.

Problem	Edge-deletion	Node-deletion
(s, t)-DoubleCut	Poly-time [3]	2-approx (Thm 2) $(2 - \epsilon)$-inapprox (Thm 1)
(s, t)-BiCut	2-approx $(2 - \epsilon)$-inapprox [4, 19]	[Equivalent to edge-deletion]
(s, r, t)-Lin3Cut	2-approx $(\alpha - \epsilon)$-inapprox [5] (where α is the flow-cut gap)	[Equivalent to edge-deletion]

Table 3 Global Variants in Undirected Graphs. Text in gray refer to known results while text in black refer to the results from this work. All hardness of approximation results are under UGC.

Problem	Edge-deletion	Node-deletion
k-cut (where k is constant)	Poly-time [14, 18]	$(2 - 2/k)$-approx [13] $(2 - 2/k - \epsilon)$-inapprox (Thm 6)
$\{s, t\}$-Sep$k$Cut (where k is constant)	Poly-time (Thm 8)	$(2 - 2/k)$-approx [13] $(2 - 2/k - \epsilon)$-inapprox (Thm 6)

1.2 Related Work

In recent work, Bernáth and Pap [3] studied the problem of deleting the smallest number of arcs to block all minimum cost arborescences of a given directed graph. They gave an efficient algorithm to solve this problem through combinatorial techniques. However, their techniques fail to extend to the node weighted double cut problem.

The node-weighted 3-cut problem – Node-3-Cut – is a generalization of the classic Edge-3-Cut. Various other generalizations of Edge-3-Cut have been studied in the literature showing the existence of efficient algorithms. These include the edge-weighted 3-cut in hypergraphs [30, 12] and the more general submodular 3-way partitioning [31, 24]. However, none of these known generalizations address Node-3-Cut as a special case. Feasible solutions to Node-3-Cut are also known as shredders in the node-connectivity literature.

In the unit-weight case, shredders whose cardinality is equal to the node connectivity of the graph play a crucial role in the problem of min edge addition to augment node connectivity by one [6, 15, 20, 29]. There are at most linear number of such shredders and all of them can be found efficiently [6, 15]. The complexity of finding a min cardinality shredder was open until our results (Theorem 6).

In the edge-weighted multiway cut in undirected graphs, the input is an undirected graph with k terminal nodes and the goal is to find the smallest cardinality subset of edges whose deletion ensures that there is no path between any pair of terminal nodes. For $k = 3$, a 12/11-approximation is known [7, 16], while for constant k, the current-best approximation factor is 1.2975 due to Sharma and Vondrák [27]. These results are based on an LP-relaxation proposed by Călinescu, Karloff and Rabani [9], known as the CKR relaxation. Manokaran, Naor, Raghavendra and Shwartz [21] showed that the inapproximability factor coincides with the integrality gap of the CKR relaxation. Recently, Angelidakis, Makarychev and Manurangsi [1] exhibited instances with integrality gap at least $6/(5 + (1/k - 1)) - \epsilon$ for every $k \geq 3$ and every $\epsilon > 0$ for the CKR relaxation.

The node-weighted multiway cut in undirected graphs exhibits very different structure in comparison to the edge-weighted multiway cut. It reduces to edge-weighted multiway cut in hypergraphs. Garg, Vazirani and Yannakakis [13] gave a $(2 - 2/k)$-approximation for node-weighted multiway cut by exploiting the extreme point structure of a natural LP-relaxation.

The edge-weighted multiway cut in directed graphs has a 2-approximation, due to Naor and Zosin [23], as well as Chekuri and Madan [4]. Matching inapproximability results were shown recently for $k = 2$ [19, 5]. The node-weighted multiway cut in directed graphs reduces to the edge-weighted multiway cut by exploiting the fact that the terminals are fixed. Such a reduction is unknown for the global version.

1.3 Preliminaries

Let $D = (V, E)$ be a directed graph. For two disjoint sets $X, Y \subset V$, we denote $\delta(X, Y)$ to be the set of edges (u, v) with $u \in X$ and $v \in Y$ and $d(X, Y)$ to be the cut value $|\delta(X, Y)|$. We use $\delta^{in}(X) := \delta(V \setminus X, X)$, $\delta^{out}(X) := \delta(X, V \setminus X)$, $d^{in}(X) := |\delta^{in}(X)|$ and $d^{out}(X) := |\delta^{out}(X)|$. We use a similar notation for undirected graphs by dropping the superscripts. For two nodes $s, t \in V$, a subset $X \subset V$ is called an *$\overline{st}$-set* if $t \in X \subseteq V - s$. The *cut value* of an $\overline{st}$-set X is $d^{in}(X)$.

We frequently use the following characterization of directed graphs with no arborescence for the purposes of double cut.

▶ **Theorem 10** (e.g., see [3]). *Let $D = (V, E)$ be a directed graph. The following are equivalent:*

1. *D has no arborescence.*
2. *There exist two distinct nodes $s, t \in V$ such that every node u can reach at most one node in $\{s, t\}$ in D.*
3. *There exist two disjoint non-empty sets $S, T \subset V$ with $\delta^{in}(S) \cup \delta^{in}(T) = \emptyset$.*

2 Overview of approximation for EdgeBiCut

In this section, we present the high-level ideas of the $(2 - 1/448)$-approximation algorithm for EdgeBiCut (Theorem 4). We sketch the argument for a $(2 - \epsilon)$-approximation for some small enough ϵ; the full algorithm and the proof of its approximation ratio are presented in the complete version of this work [2].

Let D be a digraph. For two disjoint sets $X, Y \subset V$, we define $\delta_D(X,Y)$ to be the set of edges (u,v) with $u \in X$ and $v \in Y$ and $d(X,Y)$ to be the cut value $|\delta_D(X,Y)|$. We use $\delta_D^{in}(X) := \delta_D(V \setminus X, X)$, $\delta_D^{out}(X) := \delta_D(X, V \setminus X)$. We drop the subscripts when the graph D is clear from context.

Two sets A and B are called *uncomparable* if $A \setminus B \neq \emptyset$ and $B \setminus A \neq \emptyset$. Given a directed graph $D = (V, E)$, EdgeBiCut is equivalent to finding an uncomparable pair $A, B \subseteq V$ with minimum $|\delta^{in}(A) \cup \delta^{in}(B)|$. Indeed, if A and B are uncomparable and we remove $\delta^{in}(A) \cup \delta^{in}(B)$ from the directed graph, then nodes in $A \setminus B$ cannot reach nodes in $B \setminus A$ and vice versa. On the other hand, if s cannot reach t and t cannot reach s, then the set of nodes that can reach s and the set of nodes that can reach t are uncomparable, and have in-degree 0.

▶ **Definition 11.** For $A, B \subseteq V$, let $\beta(A,B) := |\delta^{in}(A) \cup \delta^{in}(B)|$ and let $\sigma(A,B) := |\delta^{in}(A)| + |\delta^{in}(B)|$. Furthermore, let

$$\beta := \min\{\beta(A,B) \mid A \text{ and } B \text{ are uncomparable}\},$$
$$\sigma := \min\{\sigma(A,B) \mid A \text{ and } B \text{ are uncomparable}\}.$$

As σ can be computed efficiently, we immediately have a $(2-\epsilon)$-approximation if $\sigma \leq (2-\epsilon)\beta$. Also, for fixed $Z \subseteq V$, we can efficiently find an uncomparable pair (A,B) satisfying $A \cap B = Z$ that minimizes $\beta(A,B)$ among pairs with this property, because this is an EdgeDoubleCut problem. The same holds when $V \setminus (A \cup B)$ is fixed. In particular, if there is a pair (A,B) that minimizes $\beta(A,B)$ and $|A \cap B| \leq 2$ or $|V \setminus (A \cup B)| \leq 2$, then we can find the minimizer efficiently. Therefore we assume that every minimizer (A,B) for $\beta(A,B)$ satisfies $|A \cap B| \geq 3$ and $|V \setminus (A \cup B)| \geq 3$. Let us fix one such minimizer (A,B).

In the algorithm, we guess nodes $x \in A \setminus B$, $y \in B \setminus A$, $w_1, w_2 \in V \setminus (A \cup B)$, and $z_1, z_2 \in A \cap B$ (the reason for guessing *two* nodes in the intersection and in the complement of the union is highly technical, and not relevant to this overview). We use the notation $X = A \setminus B$, $Y = B \setminus A$, $W = V \setminus (A \cup B)$, and $Z = A \cap B$.

The algorithm proceeds by making several attempts at finding pairs (A', B') that give a $(2-\epsilon)$-approximation. Each unsuccessful attempt implies some structural property of the minimum bicut. For example, the first candidate is (X', Y'), where X' is the sink-side of the minimum $\{w_1, w_2, y\} \to \{x, z_1, z_2\}$-cut, and Y' is the sink-side of the minimum $\{w_1, w_2, x\} \to \{y, z_1, z_2\}$-cut. Notice that $\sigma(X', Y') \leq \sigma(A,B)$. If the attempt is unsuccessful, i.e. $\beta(X', Y') > (2-\epsilon)\beta(A,B)$, then $d(W,Z) > (1-\epsilon)\beta(A,B) = (1-\epsilon)\beta$.

Our subsequent attempts are more complex. In our next attempt, we try to expand X' and Y' by the same node set Z' to find $(A' = X' \cup Z', B' = Y' \cup Z')$. Also, we prefer not to have many edges of $E[X'] \cup E[Y']$ in the new bicut (A', B'), because they enter only one among the two sets A' and B', so we make these edges more expensive by duplicating them. Let D_1 be the digraph obtained by duplicating the edges in $E[X'] \cup E[Y']$, and let Z' be the sink-side of the minimum $\{w_1, w_2, x, y\} \to \{z_1, z_2\}$-cut in D_1. It can be shown that the pair $(X' \cup Z', Y' \cup Z')$ is a $(2-\epsilon)$-approximation unless $|\delta_{D_1}^{in}(Z)| > (2-3\epsilon)\beta$.

An analogous attempt can be made by shrinking instead of expanding. Let D_2 be the digraph obtained by duplicating the edges in $E[V \setminus X'] \cup E[V \setminus Y']$, and let W' be the source-side of the minimum $\{w_1, w_2\} \to \{x, y, z_1, z_2\}$-cut in D_2. We obtain that the pair $(X' \setminus W', Y' \setminus W')$ is a $(2-\epsilon)$-approximation unless $|\delta_{D_2}^{out}(W)| > (2-3\epsilon)\beta$.

If the attempts so far are unsuccessful, then $|\delta_{D_1}^{in}(Z)| > (2-3\epsilon)\beta$ and $|\delta_{D_2}^{out}(W)| > (2-3\epsilon)\beta$. From these, it can be shown that all but $O(\epsilon\beta)$ edges in $\delta^{in}(X') \cup \delta^{in}(Y') \cup \delta^{out}(W) \cup \delta^{in}(Z)$ are as positioned in Figure 1.

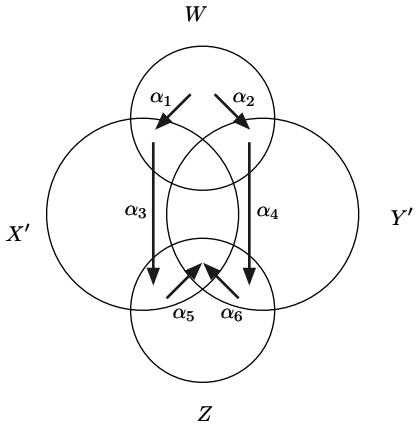

Figure 1 The quantities $\alpha_1, \ldots, \alpha_6$.

Let $\alpha_1, \ldots, \alpha_6$ be the number of edges in each position indicated in Figure 1. We can further show that the quantities $\alpha_1, \alpha_3, \alpha_5$ are within $O(\epsilon\beta)$ of each other, and so are $\alpha_2, \alpha_4, \alpha_6$. Furthermore, $(1 - O(\epsilon))\beta \leq \alpha_3 + \alpha_4 \leq (1 + O(\epsilon))\beta$. W.l.o.g. we may assume $\alpha_3 \geq \alpha_4$.

Our final attempt at obtaining a good bicut is by adding some nodes in $X' \setminus Y'$ to Y' and removing some other nodes of $X' \setminus Y'$ from X'. In other words, our candidate is a pair $(B', Y' \cup A')$ for some $X' \cap Y' \subseteq A' \subsetneq B' \subseteq X'$ (we need the condition $A' \subsetneq B'$ because B' and $Y' \cup A'$ should be uncomparable). When choosing A' and B', we ignore the edges whose contribution does not depend on A' and B'. Let H be the digraph obtained by removing the edges in $E[Y' \cup (V \setminus X')]$. Our aim is to minimize $|\delta^{in}_H(B') \cup \delta^{in}_H(Y' \cup A')|$. However, this quantity differs by $O(\epsilon\beta)$ from $|\delta^{in}_H(A') \cup \delta^{in}_H(B')|$, so we may instead aim to minimize the latter.

The crucial observation is that this minimization problem is an instance of $(s, *, t)$-Edge-Lin3Cut. While we do not know how to solve $(s, *, t)$-EdgeLin3Cut optimally, we can efficiently obtain a 3/2-approximation by Theorem 7. By the reformulation of $(s, *, t)$-Edge-Lin3Cut in Lemma 13, we get a pair of subsets (A', B') for which $X' \cap Y' \subseteq A' \subsetneq B' \subseteq X'$ and which is a 3/2-approximation. In particular, $|\delta^{in}_H(A') \cup \delta^{in}_H(B')| \leq (3/2)|\delta^{in}_H((X' \cap (Z \cup Y')) \cup \delta^{in}_H(X' \setminus (W \setminus Y'))| \leq 3(\alpha_3 + O(\epsilon)\beta)/2$. Using this and the relationship between the α_i values, we can derive $\beta(B', Y' \cup A') \leq (7/4 + O(\epsilon))\beta$, concluding the proof.

3 Overview of the results on hardness of approximation

Our hardness results include Theorem 1 for $\{s, t\}$-NodeDoubleCut, Theorem 3 for Node-DoubleCut, Theorem 5 for NodeBiCut, Theorem 6 for Node-3-Cut, and Theorem 9 for $\{s, *\}$-EdgeBiCut. We obtain all of our NP-hardness results by reducing from VertexCover on k-regular Graphs, where the input is an undirected k-regular graph, and the goal is to find the smallest subset S of nodes such that every edge in the graph has at least one end-vertex in S. It is APX-hard even for $k = 3$ [8].

We use VertexCover on k-partite Graphs as an intermediate problem, where the input is an undirected k-partite graph $G = (V_1 \cup \cdots \cup V_k, E)$ (we emphasize that the partitioning $V_1, \ldots, V_k$ is specified explicitly in the input) and the goal is to find the smallest subset $S \subset V_1 \cup \cdots \cup V_k$ such that every edge in E has at least one end-vertex in S. Our hardness results are structured as follows.

1. We first show approximation-preserving (combinatorial) reductions from VertexCover on k-regular Graphs (for $k = 3$ or 4) to the above-mentioned problems. These reductions prove all the NP-hardness results. Note that we also get an inapproximability factor of 100/99 and 53/52 respectively under the assumption that $P \neq NP$.
2. For improved hardness of approximation results, we show that VertexCover on k-partite Graphs is hard to approximate within a factor of $2 - 2/k - \epsilon$ for any $\epsilon > 0$ assuming the Unique Games Conjecture. Considering $k = 3$ and $k = 4$, this result in conjunction with the combinatorial reductions show $(4/3-\epsilon)$-inapproximability for NodeDoubleCut and $\{s,*\}$-EdgeBiCut, and $(3/2-\epsilon)$-inapproximability for NodeBiCut assuming the Unique Games Conjecture.
3. We further improve the hardness of approximation for NodeDoubleCut and $\{s,t\}$-NodeDoubleCut by directly reducing from UniqueGames via the *length-control dictatorship tests* introduced in [19]. We obtain $(3/2-\epsilon)$-inapproximability for NodeDoubleCut and $(2-\epsilon)$-inapproximability for $\{s,t\}$-NodeDoubleCut.

In the following section, we sketch the ideas behind the hardness result for $\{s,t\}$-NodeDoubleCut assuming the Unique Games Conjecture.

3.1 $(2-\epsilon)$-Inapproximability for $\{s,t\}$-NodeDoubleCut

Our results for $\{s,t\}$-NodeDoubleCut and NodeDoubleCut are based on *length-control dictatorship tests* introduced by Lee [19]. Length-control dictatorship tests provide a systematic way to convert integrality gap instances for a natural LP relaxation to dictatorship tests that can be used to prove matching hardness of approximation under the Unique Games Conjecture. In this section we illustrate the high-level ideas behind this conversion for $\{s,t\}$-NodeDoubleCut.

Consider the integrality gap instance $D_{a,b} = (V_D, A_D)$ introduced in Section A (Lemma 20) which shows that the integrality gap of a natural Path-Blocking-LP for $\{s,t\}$-NodeDoubleCut is $2-o(1)$. We note that $V_D = \{s,t\} \cup ([a] \times [b])$. Let $r = b - 2a + 1$, and $I_D = ([a] \times [b])$ be the set of internal vertices. Furthermore, a good fractional feasible solution as obtained in the proof of Lemma 20 sets $d_v := 1/r$ for every internal vertex v while every integral feasible solution has at least $2a-1$ vertices in it.

Based on $D_{a,b}$, we define the *dictatorship test* graph $\mathcal{D}^{\mathsf{st}}_{a,b,R,\epsilon} = (V, A)$ as follows, for a positive integer R and $\epsilon > 0$. Consider the probability space (Ω, μ) where $\Omega := \{0, \ldots, r-1, *\}$, and $\mu : \Omega \to [0,1]$ with $\mu(*) = \epsilon$ and $\mu(x) = (1-\epsilon)/r$ for $x \neq *$.

1. We define $V := \{s,t\} \cup \{v^\alpha_x\}_{\alpha \in I_D, x \in \Omega^R}$. Let v^α denote the set of vertices $\{v^\alpha_x\}_{x \in \Omega^R}$. We also call each v^α as a *hypercube*.
2. For $\alpha \in I_D$ and $x \in \Omega^R$, define the weight as $c(v^\alpha_x) = \prod_{i=1}^{R} \mu(x_i)$. We note that the weight of each hypercube is 1, and the sum of weight of all vertices except s and t is ab. Define the weight of the terminals s and t to be infinite.
3. For each arc between s and $\alpha \in I_D$ in A_D, for each $x \in \Omega^R$, add an arc with the same direction between s and v^α_x. Do the same for each arc between t and $\alpha \in I_D$ in A_D.
4. For each arc $(\alpha, \beta) \in A_D$ with $\alpha = (\alpha_1, \alpha_2), \beta = (\beta_1, \beta_2) \in I_D$ and $x, y \in \Omega^R$, we have an arc from v^α_x to v^β_y according to the following rule (note that $\alpha_2 \neq \beta_2$).
   a. $\alpha_2 < \beta_2$: add an arc if for any $1 \leq j \leq R$: $[y_j = (x_j + 1) \mod r]$ or $[y_j = *]$ or $[x_j = *]$. Call them *forward* arcs.
   b. $\alpha_2 > \beta_2$: add an arc if for any $1 \leq j \leq R$: $[y_j = (x_j - 1) \mod r]$ or $[y_j = *]$ or $[x_j = *]$. Call them *backward* arcs.
   c. If $(\alpha, \beta) \in A_D$ is a jumping arc (as defined in Lemma 20), call (v^α_x, v^β_y) also a jumping arc.

We provide some intuitions behind this conversion. First, we replace each internal vertex $v \in I_D$ by a R-dimensional *hypercube* $v^\alpha = \{v^\alpha_x\}_{x\in\Omega^R}$. Intuitively, our dictatorship test $\mathcal{D}^{\mathsf{st}}_{a,b,R,\epsilon} = (V, A)$, as an instance of $\{s,t\}$-NodeDoubleCut, needs to satisfy the following properties:

1. Completeness: there exists an integral solution $C^* \subseteq V$ of low weight that *reveals an influential coordinate* for each hypercube.
2. Soundness: a subset $C \subseteq V$ is an integral solution of low weight only if it *reveals an influential coordinate* for some hypercube.

To formalize the notion of influence, given $C \subseteq V$ and a hypercube v^α for some $\alpha \in I_D$, let $f = f_{C,\alpha} : \Omega^R \to \{0,1\}$ be such that $f(x) = 1$ if and only if $v^\alpha_x \in C$. Then for each $i \in [R]$, the influence of the ith coordinate is defined by

$$\mathsf{Inf}_i[f] := \mathbb{E}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_R}[\mathsf{Var}_{x_i}[f(x_1,\ldots,x_R)]],$$

where $x_1,\ldots,x_R$ are independently sampled from (Ω,μ). For example, suppose C and α are such that $f(x) = 1$ if and only if $x_i = 0$ for some $i \in [R]$ (i.e., f only depends on the ith coordinate), then $\mathsf{Inf}_i[f] = \mu(0) - \mu(0)^2 = \Omega(1/r)$ and $\mathsf{Inf}_j[f] = 0$ for all $j \neq i$, so i is the only influential coordinate. In contrast, suppose C and α are such that $f(x) = 1$ if and only if $x_1 + \cdots + x_R \leq K$ for some K (ignoring x_i with $x_i = *$), then f depends on all coordinates equally and $\mathsf{Inf}_i[f] \to 0$ for all i as $R \to \infty$. We say that C reveals an influential coordinate for the hypercube v^α if $\mathsf{Inf}_i[f_{C,\alpha}]$ is *large* for some $i \in [R]$. Since we will eventually take R to be a sufficiently large constant, a *large influence* means that the influence is some positive constant that does not depend on R.

Given these intuitions, the completeness and soundness properties can be formalized and proved as follows.

Completeness. For the completeness requirement, we need to argue that there exists an integral solution $C^* \subseteq V$ of low weight that reveals an influential coordinate for each hypercube. In particular, we show that a set of vertices that correspond to *dictators* behaves the same as the fractional solution that gives $1/r$ to every vertex and moreover has low weight. For any $q \in [R]$, let $V_q := \{v^\alpha_x : \alpha \in I_D, x_q = * \text{ or } 0\}$. We note that the total weight of V_q is

$$ab\left(\epsilon + \frac{1-\epsilon}{r}\right) \leq ab\epsilon + \frac{ab}{b-2a}.$$

▶ **Lemma 12.** *After removing vertices in V_q, no vertex in V can reach both s and t.*

The proof appears in the full version [2]. The basic intuition is that for any arc from v^α_x to v^β_y for some $\alpha = (\alpha_1,\alpha_2), \beta = (\beta_1,\beta_2), x \in \Omega^R, y \in \Omega^R$, we have $x_q = y_q + 1$ if this arc is going forward ($\alpha_2 < \beta_2$), and $x_q = y_q - 1$ if this arc is going backward ($\alpha_2 > \beta_2$). This relies on our construction and the fact that we removed all v^α_x with $x_q = *$ or $x_q = 0$ since they are in V_q. Since $x_q \in \{1,\ldots,r-1\}$, it means that for any path p,

$$|(\text{number of forward arcs in } p) - (\text{number of backward arcs in } p)| < r.$$

As a consequence, this integral solution V_q behaves similar to the fractional solution in D where each internal vertex gets $1/r$, and we can conclude that no vertex can reach both s and t.

Soundness. Suppose that we removed some vertices C such that no vertex $w \in V \setminus C$ can reach both s and t. Our soundness property requires that C either reveals an influential coordinate for some hypercube v^α or $c(C) \geq (2a-1)(1-\epsilon)$. Formally, let τ, d be some constants that depend only on ϵ and r (not R). We prove that if C is a feasible integral solution of $\{s,t\}$-NodeDoubleCut, then either $c(C) \geq (2a-1)(1-\epsilon)$, or $\mathsf{Inf}_i^{\leq d}[f_{C,\alpha}] \geq \tau$ for some $\alpha \in I_D$ and $i \in [R]$. For technical reasons, we use *low-degree influence* $\mathsf{Inf}_i^{\leq d}$ instead of Inf_i.

The main component of the proof is that if there is an arc from α to β in the integrality gap instance $D_{a,b}$, and both $f_{C,\alpha}$, $f_{C,\beta}$ do not reveal an influential coordinate, then we can always find an arc from $v^\alpha \setminus C$ to $v^\beta \setminus C$ in $\mathcal{D}^{\mathsf{st}}$ unless C almost completely contains both v^α and v^β (i.e., $c(C \cap v^\alpha) > 1-\epsilon$ and $c(C \cap v^\beta) > 1-\epsilon$). The proof involves interpreting the set of arcs between two hypercubes as a suitably designed correlated probability space, and using the invariance principle by Mossel [22].

Suppose that C does not reveal an influential coordinate for any hypercube v^α. Then the above fact ensures that for a hypercube v^α, unless it is almost completely contained in C (i.e., $c(C \cap v^\alpha) > 1-\epsilon$), it behaves as if no vertices were contained in C. This observation shows that $c(C)$ must be as large as that of an integral solution in the gap instance $D_{a,b}$. Using the fact that any integral solution of $D_{a,b}$ contains at least $2a-1$ vertices, we conclude that $c(C) \geq (2a-1)(1-\epsilon)$.

In summary, in the completeness case, there exists a subset of vertices of total weight at most $ab\epsilon + ab/(b-2a)$, so that after removing the subset, no vertex can reach both s and t. In the soundness case, unless we reveal an influential coordinate or we remove vertices of total weight at least $(2a-1)(1-\epsilon)$, there exists a vertex that can reach both s and t. The gap between the two cases is at least

$$\frac{(2a-1)(1-\epsilon)}{ab\epsilon + ab/(b-2a)},$$

which approaches to 2 as a increases, by setting $b = a^2$ and $\epsilon = 1/a^4$.

4 EdgeLin3Cut problems

Given a directed graph $D = (V, E)$, a feasible solution to (s,r,t)-EdgeLin3Cut in D is a subset F of arcs whose deletion from the graph eliminates all directed $s \to r$, $r \to t$ and $s \to t$ paths. One of our main tools used in the approximation algorithm for EdgeBiCut is a 3/2-approximation algorithm for $(s,*,t)$-EdgeLin3Cut. We present this algorithm now. For two sets $A, B \subseteq V$, let $\beta(A,B) := |\delta^{in}(A) \cup \delta^{in}(B)|$.

Proof of Theorem 7. We first rephrase the problem in a more convenient way.

▶ **Lemma 13.** $(s,*,t)$-EdgeLin3Cut *in a directed graph* $D = (V,E)$ *is equivalent to*

$$\min\{\beta(A,B) : t \in A \subset B \subseteq V - \{s\}\}.$$

Proof. Let $F \subseteq E$ be an optimal solution for $(s,*,t)$-EdgeLin3Cut in D and let $(A,B) := \operatorname{argmin}\{\beta(A,B) : t \in A \subset B \subseteq V - s\}$. Fix an arbitrary node $r \in B - A$. Since the deletion of $\delta^{in}(A) \cup \delta^{in}(B)$ results in a graph with no directed path from s to r, from r to t and from s to t, the edge set $\delta^{in}(A) \cup \delta^{in}(B)$ is a feasible solution to (s,r,t)-EdgeLin3Cut in D, thus implying that $|F| \leq \beta(A,B)$.

On the other hand, F is a feasible solution for (s,r,t)-EdgeLin3Cut in D for some $r \in V - \{s,t\}$. Let A be the set of nodes that can reach t in $D-F$, and R be the set of nodes that can reach r in $D-F$. Then, $F \supseteq \delta^{in}(A)$. Moreover, $F \supseteq \delta^{in}(R \cup A)$ since $R \cup A$ has in-degree 0 in $D-F$, and s is not in $R \cup A$ because it cannot reach r and t in $D-F$. Therefore, taking $B = R \cup A$ we get $F \supseteq \delta^{in}(A) \cup \delta^{in}(B)$. ◀

Our algorithm for determining an optimal pair $(A,B) := \text{argmin}\{\beta(A,B) : t \in A \subset B \subseteq V - s\}$ proceeds as follows: We build a chain $\mathcal{C}$ of $\overline{s}t$-sets with the property that, for some value $k \in \mathbb{Z}_+$,

(i) $\mathcal{C}$ contains only cuts of value at most k, and

(ii) every $\overline{s}t$-set of cut value strictly less than k is in $\mathcal{C}$.

We start with k being the minimum $\overline{s}t$-cut value and $\mathcal{C}$ consisting of a single minimum $\overline{s}t$-cut. In a general step, we find two $\overline{s}t$-sets: a minimum $\overline{s}t$-cut Y compatible with the current chain $\mathcal{C}$, i.e. $\mathcal{C} \cup \{Y\}$ forming a chain, and a minimum $\overline{s}t$-cut Z not compatible with the current chain $\mathcal{C}$, i.e. crossing at least one member of $\mathcal{C}$. These two sets can be found in polynomial time. Indeed, let $t \in C_1 \subset \ldots, \subset C_q \subseteq V - s$ denote the members of $\mathcal{C}$. Find a minimum cut Y_i with $C_i \subseteq Y_i \subseteq V \setminus C_{i+1}$ for $i = 1, \ldots, q$, and choose Y to be a minimum one among these cuts. Concerning Z, for each pair x, y of nodes with $y \in C_i \subseteq V - x$ for some $i \in \{1, \ldots, q\}$, find a minimum cut Z_{xy} with $\{t,x\} \subseteq Z_{xy} \subseteq V - \{s,y\}$, and choose Z to be a minimum one among these cuts. If $d^{in}(Y) \leq d^{in}(Z)$, then we add Y to $\mathcal{C}$, and set k to $d^{in}(Y)$; otherwise we set k to $d^{in}(Z)$, and stop.

Let $\mathcal{C}$ denote the chain constructed by the algorithm, and let Y be an arbitrary set crossing some of its members.

▶ **Claim 14.** *$d^{in}(Y) \geq d^{in}(C)$ for all $C \in \mathcal{C}$.*

Proof. Suppose indirectly that $d^{in}(Y) < d^{in}(C)$ for some $C \in \mathcal{C}$. Let $\mathcal{C}'$ denote the chain consisting of those members of $\mathcal{C}$ that were added before C. As C is a set of minimum cut value compatible with $\mathcal{C}'$, Y crosses at least one member of $\mathcal{C}'$. Hence, by $d^{in}(Y) < d^{in}(C)$, the algorithm stops before adding C, a contradiction. ◀

The claim implies that $\mathcal{C}$ satisfies (1) and (2) with the k obtained at the end of the algorithm. Indeed, (1) is obvious from the construction, while (2) follows from the claim and the fact that $\mathcal{C}$ contains all cuts of value strictly less than k that are compatible with $\mathcal{C}$.

By the above, the procedure stops with a chain $\mathcal{C}$ containing all $\overline{s}t$-sets of cut value less than k, and an $\overline{s}t$-set Z of cut value exactly k which crosses some member X of $\mathcal{C}$. If the optimum value of our problem is less than k, then both members of the optimal pair (A,B) belong to the chain $\mathcal{C}$, and we can find them by taking the minimum of $\beta(A',B')$ where $A' \subset B'$ with $A', B' \in \mathcal{C}$.

We can thus assume that the optimum is at least k. As $d^{in}(Z) = k$ and $d^{in}(X) \leq k$, the submodularity of the in-degree function implies $d^{in}(X \cap Z) + d^{in}(X \cup Z) \leq d^{in}(Z) + d^{in}(X) \leq 2k$. Hence at least one of $d^{in}(X \cap Z) \leq k$ and $d^{in}(X \cup Z) \leq k$ holds. As $d(X \setminus Z, X \cap Z) + d(Z \setminus X, X \cap Z) \leq d^{in}(X \cap Z)$ and $d(V \setminus (X \cup Z), X \setminus Z) + d(V \setminus (X \cup Z), Z \setminus X) \leq d^{in}(X \cup Z)$, at least one of the following four possibilities is true:

1. $d^{in}(X \cap Z) \leq k$ and $d(X \setminus Z, X \cap Z) \leq \frac{1}{2}k$. Choose $A = X \cap Z$, $B = X$. Then $\beta(A,B) = d(X \setminus Z, X \cap Z) + d^{in}(X) \leq \frac{1}{2}k + k = \frac{3}{2}k$.
2. $d^{in}(X \cap Z) \leq k$ and $d(Z \setminus X, X \cap Z) \leq \frac{1}{2}k$. Choose $A = X \cap Z$, $B = Z$. Then $\beta(A,B) = d(Z \setminus X, X \cap Z) + d^{in}(Z) \leq \frac{1}{2}k + k = \frac{3}{2}k$.
3. $d^{in}(X \cup Z) \leq k$ and $d(V \setminus (X \cup Z), X \setminus Z) \leq \frac{1}{2}k$. Choose $A = Z$, $B = X \cup Z$. Then $\beta(A,B) = d^{in}(Z) + d(V \setminus (X \cup Z), X \setminus Z) \leq k + \frac{1}{2}k = \frac{3}{2}k$.
4. $d^{in}(X \cup Z) \leq k$ and $d(V \setminus (X \cup Z), Z \setminus X) \leq \frac{1}{2}k$. Choose $A = X$, $B = X \cup Z$. Then $\beta(A,B) = d^{in}(X) + d(V \setminus (X \cup Z), Z \setminus X) \leq k + \frac{1}{2}k = \frac{3}{2}k$.

Thus a pair (A, B) can be obtained by taking the minimum among the four possibilities above and $\beta(A', B')$ where $A' \subset B'$ with $A', B' \in \mathcal{C}$, concluding the proof of the theorem. ◀

Next, we show that $\{s,t\}$-SepEdge$k$Cut is solvable in polynomial time if k is a fixed constant.

Let $G = (V, E)$ be an undirected graph. Let the minimum size of an $\{s,t\}$-cut in G be denoted by $\lambda_G(s,t)$. For two subsets of nodes X, Y, let $d(X,Y)$ denote the number of edges between X and Y and let $d(X) := d(X, V \setminus X)$. The cut value of a partition $\{V_1, \ldots, V_q\}$ of V is defined to be the total number of crossing edges, that is, $(1/2)\sum_{i=1}^{q} d(V_i)$, and is denoted by $\gamma(V_1, \ldots, V_q)$. Let $\gamma^q(G)$ denote the value of an optimum Edge-q-Cut in G, i.e.,

$$\min\{\gamma(V_1, \ldots, V_q) : V_i \neq \emptyset \ \forall\ i \in [q], V_i \cap V_j = \emptyset \ \forall\ i, j \in [q], \cup_{i=1}^{q} V_i = V\}.$$

Proof of Theorem 8. Let γ^* denote the optimum value of $\{s,t\}$-SepEdge$k$Cut in $G = (V, E)$ and let H denote the graph obtained from G by adding an edge of infinite capacity between s and t. The algorithm is based on the following observation (we recommend the reader to consider $k = 3$ for ease of understanding):

▶ **Proposition 15.** *Let $\{V_1, \ldots, V_k\}$ be a partition of V corresponding to an optimal solution of $\{s,t\}$-SepEdge$k$Cut, where s is in V_{k-1} and t is in V_k. Then $\gamma(V_1, \ldots, V_{k-2}, V_{k-1} \cup V_k) \leq 2\gamma^{k-1}(H)$.*

Proof. Let $W_1, \ldots, W_{k-1}$ be a minimum $(k-1)$-cut in H. Clearly, s and t are in the same part, so we may assume that they are in W_{k-1}. Let U_1, U_2 be a minimum $\{s,t\}$-cut in $G[W_{k-1}]$. Then $\{W_1, \ldots, W_{k-2}, U_1, U_2\}$ gives an $\{s,t\}$-separating k-cut, showing that

$$\gamma^* \leq \gamma(W_1, \ldots, W_{k-2}, U_1, U_2) = \gamma^{k-1}(H) + \lambda_{G[W_{k-1}]}(s,t). \tag{1}$$

By Menger's theorem, we have $\lambda_G(s,t)$ pairwise edge-disjoint paths $P_1, \ldots, P_{\lambda_G(s,t)}$ between s and t in G. Consider one of these paths, say P_i. If all nodes of P_i are from $V_{k-1} \cup V_k$, then P_i has to use at least one edge from $\delta(V_{k-1}, V_k)$. Otherwise, P_i uses at least two edges from $\delta(V_1 \cup \cdots \cup V_{k-2}) \cup \bigcup_{\substack{i,j \leq k-2 \\ i \neq j}} \delta(V_i, V_j)$. Hence the maximum number of pairwise edge-disjoint paths between s and t is

$$\lambda_G(s,t) \leq d(V_{k-1}, V_k) + \frac{1}{2}\left(d(V_1 \cup \cdots \cup V_{k-2}) + \sum_{\substack{i,j \leq k-2 \\ i \neq j}} d(V_i, V_j)\right).$$

Thus, we have

$$\begin{aligned}
\gamma^* &= d(V_{k-1}, V_k) + d(V_1 \cup \cdots \cup V_{k-2}) + \sum_{\substack{i,j \leq k-2 \\ i \neq j}} d(V_i, V_j) \\
&\geq \lambda_G(s,t) + \frac{1}{2}\left(d(V_1 \cup \cdots \cup V_{k-2}) + \sum_{\substack{i,j \leq k-2 \\ i \neq j}} d(V_i, V_j)\right) \\
&= \lambda_G(s,t) + \frac{1}{2}\gamma(V_1, \ldots, V_{k-2}, V_{k-1} \cup V_k) \\
&\geq \lambda_{G[W_{k-1}]}(s,t) + \frac{1}{2}\gamma(V_1, \ldots, V_{k-2}, V_{k-1} \cup V_k)
\end{aligned}$$

that is,

$$\gamma^* \geq \lambda_{G[W_{k-1}]}(s,t) + \frac{1}{2}\gamma(V_1,\ldots,V_{k-2},V_{k-1}\cup V_k). \tag{2}$$

By combining (1) and (2), we get $\gamma(V_1,\ldots,V_{k-2},V_{k-1}\cup V_k) \leq 2\gamma^{k-1}(H)$, proving the proposition. ◀

Karger and Stein [18] showed that the number of feasible solutions to Edge-k-cut in G with value at most $2\gamma^k(G)$ is $O(n^{4k})$. All these solutions can be enumerated in polynomial-time for fixed k [18, 17]. This observation together with Proposition 15 gives the following algorithm for finding an optimal solution to $\{s,t\}$-SepEdge$k$Cut:

Step 1. Let H be the graph obtained from G by adding an edge of infinite capacity between s and t. In H, enumerate all feasible solutions to Edge-$(k-1)$-Cut – namely the vertex partitions $\{W_1,\ldots,W_{k-1}\}$ – whose cut value $\gamma_H(W_1,\ldots,W_{k-1})$ is at most $2\gamma^{k-1}(H)$. Without loss of generality, assume $s,t \in W_{k-1}$.

Step 2. For each feasible solution to Edge-$(k-1)$-Cut in H listed in Step 1, find a minimum $\{s,t\}$-cut in $G[W_{k-1}]$, say U_1,U_2.

Step 3. Among all feasible solutions $\{W_1,\ldots,W_{k-1}\}$ to Edge-$(k-1)$-Cut listed in Step 1 and the corresponding U_1,U_2 found in Step 2, return the k-cut $\{W_1,\ldots,W_{k-2},U_1,U_2\}$ with minimum $\gamma(W_1,\ldots,W_{k-2},U_1,U_2)$.

The correctness of the algorithm follows from Proposition 15: one of the choices enumerated in Step 1 will correspond to the partition $(V_1,\ldots,V_{k-2},V_{k-1}\cup V_k)$, where $(V_1,\ldots,V_k)$ is the partition corresponding to the optimal solution. ◀

Acknowledgements. Karthik would like to thank Chandra Chekuri, Neil Olver and Chaitanya Swamy for helpful discussions at various stages of this work.

References

1. H. Angelidakis, Y. Makarychev, and P. Manurangsi. An Improved Integrality Gap for the Călinescu-Karloff-Rabani Relaxation for Multiway Cut. Preprint arXiv:1611.05530, 2016. URL: `https://arxiv.org/abs/1611.05530`.
2. K. Bérczi, K. Chandrasekaran, T. Király, E. Lee, and C. Xu. Global and fixed-terminal cuts in digraphs. Preprint arXiv:1612.00156, 2017. URL: `https://arxiv.org/abs/1612.00156`.
3. A. Bernáth and G. Pap. Blocking optimal arborescences. In *Proceedings of the 16th International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 74–85, 2013.
4. C. Chekuri and V. Madan. Simple and fast rounding algorithms for directed and node-weighted multiway cut. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'16, pages 797–807, 2016.
5. C. Chekuri and V. Madan. Approximating multicut and the demand graph. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'17, 2017.
6. J. Cheriyan and R. Thurimella. Fast algorithms for k-shredders and k-node connectivity augmentation. *Journal of Algorithms*, 33(1):15–50, 1999.
7. K. Cheung, W. Cunningham, and L. Tang. Optimal 3-terminal cuts and linear programming. *Mathematical Programming*, 106(1):1–23, 2006.
8. M. Chlebík and J. Chlebíková. Complexity of approximating bounded variants of optimization problems. *Theoretical Computer Science*, 354(3):320–338, 2006.

9 G. Călinescu, H. Karloff, and Y. Rabani. An improved approximation algorithm for multiway cut. *Journal of Computer and System Sciences*, 60(3):564–574, 2000.

10 E. Dahlhaus, D. Johnson, C. Papadimitriou, P. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

11 R. Erbacher, T. Jaeger, N. Talele, and J. Teutsch. Directed multicut with linearly ordered terminals. Preprint arXiv:1407.7498, 2014. URL: `https://arxiv.org/abs/1407.7498`.

12 T. Fukunaga. Computing minimum multiway cuts in hypergraphs. *Discrete Optimization*, 10(4):371–382, 2013.

13 N. Garg, V. Vazirani, and M. Yannakakis. Multiway cuts in node weighted graphs. *Journal of Algorithms*, 50(1):49–61, 2004.

14 O. Goldschmidt and D. Hochbaum. A polynomial algorithm for the k-cut problem for fixed k. *Math. Oper. Res.*, 19(1):24–37, Feb 1994.

15 T. Jordán. On the number of shredders. *Journal of Graph Theory*, 31(3):195–200, 1999.

16 D. Karger, P. Klein, C. Stein, M. Thorup, and N. Young. Rounding algorithms for a geometric embedding of minimum multiway cut. *Mathematics of Operations Research*, 29(3):436–461, 2004.

17 D. Karger and R. Motwani. Derandomization through approximation. In *Proceedings of the 26th annual ACM symposium on Theory of computing*, STOC'94, pages 497–506, 1994.

18 D. Karger and C. Stein. A new approach to the minimum cut problem. *Journal of ACM*, 43(4):601–640, July 1996.

19 E. Lee. Improved Hardness for Cut, Interdiction, and Firefighter Problems. Preprint arXiv:1607.05133, 2016. URL: `https://arxiv.org/abs/1607.05133`.

20 G. Liberman and Z. Nutov. On shredders and vertex connectivity augmentation. *Journal of Discrete Algorithms*, 5(1):91–101, 2007.

21 R. Manokaran, J. Naor, P. Raghavendra, and R. Schwartz. SDP Gaps and UGC Hardness for Multiway Cut, 0-extension, and Metric Labeling. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, STOC'08, pages 11–20, 2008.

22 E. Mossel. Gaussian bounds for noise correlation of functions. *Geometric and Functional Analysis*, 19(6):1713–1756, 2010.

23 J. Naor and L. Zosin. A 2-approximation algorithm for the directed multiway cut problem. *SIAM Journal on Computing*, 31(2):477–482, 2001.

24 K. Okumoto, T. Fukunaga, and H. Nagamochi. Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. *Algorithmica*, 62(3):787–806, 2012.

25 M. Queyranne. On Optimum k-way Partitions with Submodular Costs and Minimum Part-Size Constraints. Talk Slides, 2012. URL: `https://smartech.gatech.edu/bitstream/handle/1853/43309/Queyranne.pdf`.

26 A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency.* Algorithms and Combinatorics. Springer, 2003.

27 A. Sharma and J. Vondrák. Multiway cut, pairwise realizable distributions, and descending thresholds. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC'14, pages 724–733, 2014.

28 L. Tseng and N. Vaidya. Fault-Tolerant Consensus in Directed Graphs. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing (PODC 2015)*, pages 451–460, 2015.

29 L. Végh. Augmenting undirected node-connectivity by one. *SIAM J. Discrete Math.*, 25(2):695–718, 2011.

30 M. Xiao. Finding minimum 3-way cuts in hypergraphs. *Information Processing Letters*, 110(14):554–558, 2010.

31 L. Zhao, H. Nagamochi, and T. Ibaraki. Greedy splitting algorithms for approximating multiway partition problems. *Mathematical Programming*, 102(1):167–183, 2005.

A Approximation for NodeDoubleCut

In this section, we present an efficient 2-approximation algorithm for $\{s,t\}$-NodeDoubleCut which also leads to a 2-approximation for NodeDoubleCut by guessing the pair of nodes s,t.

Remark. Our algorithm is LP-based. Although, alternative combinatorial algorithms can be designed for this problem, we provide an LP-based algorithm since it also helps to illustrate an integrality gap instance which is the main tool underlying the hardness of approximation for the problem. Furthermore, it is also easy to round an *optimum* solution to our LP to obtain a solution whose cost is at most twice the *optimum* LP-cost (using complementary slackness conditions). Here, we present a rounding algorithm which starts from *any feasible solution* to the LP (not necessarily optimal) and gives a solution whose cost is at most twice the LP-cost of *that feasible solution*.

At the end of this section, we give an example showing that the integrality gap of the LP nearly matches the approximation factor achieved by our rounding algorithm.

Proof of Theorem 2. We recall the problem: Given a directed graph $D=(V,E)$ with two specified nodes $s,t\in V$ and node costs $c:V\setminus\{s,t\}\to\mathbb{R}_+$, the goal is to find a least cost subset $U\subseteq V\setminus\{s,t\}$ of nodes such that every node $u\in V\setminus U$ can reach at most one node in $\{s,t\}$ in the subgraph $D-U$. We will denote a path P by the set of nodes in the path and the collection of paths from node u to node v by $\mathcal{P}^{u\to v}$. For a fixed function $d:V\to\mathbb{R}_+$, the d-distance of a path P is defined to be $\sum_{u\in P}d_u$ and the shortest d-distance from node u to node v is the minimum d-distance among all paths from node u to node v. We use the following LP-relaxation, where we have a variable d_u for every node $u\in V$:

$$\min \sum_{v\in V\setminus\{s,t\}} c_v d_v \qquad \text{(Path-Blocking-LP)}$$

$$\sum_{v\in P} d_v + \sum_{v\in Q} d_v - d_u \geq 1 \;\forall\; P\in\mathcal{P}^{u\to s}, Q\in\mathcal{P}^{u\to t},\; \forall\; u\in V$$

$$d_s, d_t = 0$$

$$d_v \geq 0 \;\forall\; v\in V$$

We first observe that Path-Blocking-LP can be solved efficiently. The separation problem is the following: given $d:V\to\mathbb{R}_+$, verify if there exists a node $u\in V$ such that the sum of the shortest d-distance path from u to s and the shortest d-distance path from u to t is at most $1+d_u$. Thus, the separation problem can be solved efficiently by solving the shortest path problem in directed graphs.

Let $d:V\to R_+$ be a feasible solution to Path-Blocking-LP. We now present a rounding algorithm that achieves a 2-factor approximation. We note that our algorithm rounds an arbitrary feasible solution d to obtain an integral solution whose cost is at most twice the LP-cost of the solution d. For a subset U of nodes, let $\Delta^{in}(U)$ be the set of nodes $v\in V\setminus U$ that have an edge to a node $u\in U$.

The rounding algorithm in Figure 2 can be implemented to run in polynomial-time. We first show the feasibility of the solution returned by the rounding algorithm. We use the following claim.

▶ **Claim 16.** *For every $\theta\in(0,1/2)$, we have $\mathbb{B}^{in}(s,\theta)\cap\mathbb{B}^{in}(t,\theta)=\emptyset$.*

Rounding Algorithm for $\{s,t\}$-NodeDoubleCut

1. Pick θ uniformly from the interval $(0, 1/2)$.
2. Let $\mathbb{B}^{in}(s,\theta)$ and $\mathbb{B}^{in}(t,\theta)$ be the set of nodes whose shortest d-distance to s and t respectively, is at most θ.
3. Return $U := \Delta^{in}(\mathbb{B}^{in}(s,\theta)) \cup \Delta^{in}(\mathbb{B}^{in}(t,\theta))$.

Figure 2 The rounding algorithm for $\{s,t\}$-NodeDoubleCut.

Proof. Say $u \in \mathbb{B}^{in}(s,\theta) \cap \mathbb{B}^{in}(t,\theta)$. Then there exists a path $P \in \mathcal{P}^{u\to s}$ and a path $Q \in \mathcal{P}^{u\to t}$ such that $\sum_{v\in P} d_v + \sum_{v\in Q} d_v \leq 2\theta < 1$, a contradiction to the fact that d is feasible for Path-Blocking-LP. ◀

▶ **Claim 17.** *The solution U returned by the algorithm is such that every node $u \in V \setminus U$ can reach at most one node in $\{s,t\}$ in the subgraph $D - U$.*

Proof. Suppose not. Then there exists $u \in V \setminus U$ that can reach both s and t in $D - U$. If $u \notin \mathbb{B}^{in}(s,\theta)$, then u cannot reach s in $D - U$ since $\mathbb{B}^{in}(s,\theta)$ has no entering edges in $D - U$. Thus, $u \in \mathbb{B}^{in}(s,\theta)$. Similarly, $u \in \mathbb{B}^{in}(t,\theta)$. However, this contradicts the above claim that $\mathbb{B}^{in}(s,\theta) \cap \mathbb{B}^{in}(t,\theta) = \emptyset$. ◀

We next bound the expected cost of the solution returned by the rounding algorithm. Let $\bar{d}(v,a)$ denote the shortest d-distance from node v to node a in D. We use the following claim.

▶ **Claim 18.** *Let $\theta \in (0,1/2)$. If $v \in \Delta^{in}(\mathbb{B}^{in}(s,\theta))$ then $\theta < \bar{d}(v,s) \leq \theta + d_v$ and $d_v \neq 0$.*

Proof. If $\bar{d}(v,s) \leq \theta$, then $v \in \mathbb{B}^{in}(s,\theta)$, a contradiction to $v \in \Delta^{in}(\mathbb{B}^{in}(s,\theta))$. If $\bar{d}(v,s) > \theta + d_v$, then $v \notin \Delta^{in}(\mathbb{B}^{in}(s,\theta))$, a contradiction. If $d_v = 0$, then $\theta < \bar{d}(v,s) \leq \theta + d_v = \theta$, a contradiction. ◀

▶ **Claim 19.** *For every $v \in V$, the probability that v is chosen in U is at most $2d_v$.*

Proof. The claim holds if $v \in \{s,t\}$. Let us fix $v \in V \setminus \{s,t\}$. By the claim above, if $v \in \Delta^{in}(\mathbb{B}^{in}(s,\theta))$ then $\theta < \bar{d}(v,s) \leq \theta + d_v$ and $d_v \neq 0$. Similarly, if $v \in \Delta^{in}(\mathbb{B}^{in}(t,\theta))$, then $\theta < \bar{d}(v,t) \leq \theta + d_v$ and $d_v \neq 0$. Now, the probability that v is in U is at most

$$\mathsf{Pr}\left(\theta \in \left(\bar{d}(v,s) - d_v, \min\{\bar{d}(v,s), 1/2\}\right) \cup \left(\bar{d}(v,t) - d_v, \min\{\bar{d}(v,t), 1/2\}\right)\right).$$

Without loss of generality, let $\bar{d}(v,s) \leq \bar{d}(v,t)$. We may assume that $d_v > 0$ and $\bar{d}(v,s) - d_v < 1/2$, since otherwise, the probability that v is in U is 0 and the claim is proved. Now, by the feasibility of the solution d to Path-Blocking-LP, we have that $\bar{d}(v,s) + \bar{d}(v,t) - d_v \geq 1$ and hence $\bar{d}(v,t) \geq 1/2$. Therefore,

$$\begin{aligned}
\mathsf{Pr}(v \in U) &\leq \mathsf{Pr}\left(\theta \in \left(\bar{d}(v,s) - d_v, \min(\bar{d}(v,s), 1/2)\right)\right) + \mathsf{Pr}\left(\theta \in \left(\bar{d}(v,t) - d_v, 1/2\right)\right) \\
&= \frac{1}{(1/2)}\left(1/2 - \bar{d}(v,s) + d_v + 1/2 - \bar{d}(v,t) + d_v\right) \\
&= 2\left(1 - (\bar{d}(v,s) + \bar{d}(v,t) - d_v) + d_v\right) \\
&\leq 2d_v.
\end{aligned}$$

The first equality in the above is because θ is chosen uniformly from the interval $(0,1/2)$ while the last inequality is because of the feasibility of the solution d to Path-Blocking-LP. ◀

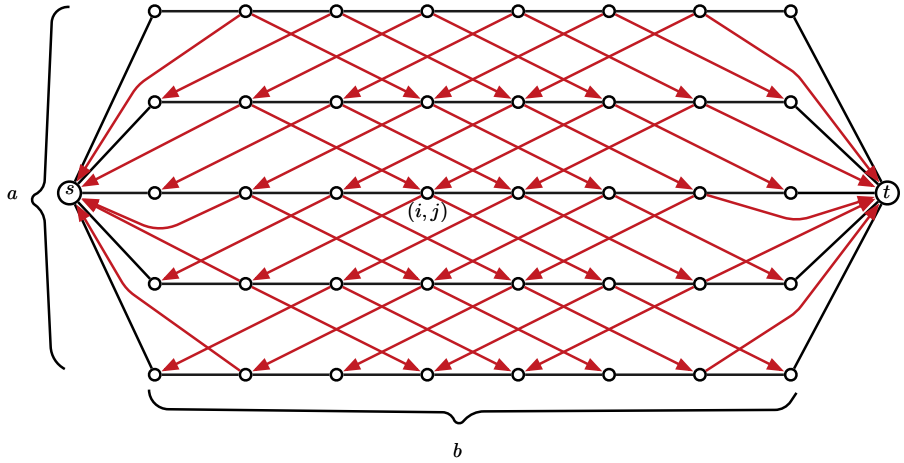

Figure 3 $D_{a,b}$ in the proof of Lemma 20 and $(2-\epsilon)$-inapproximability of $\{s,t\}$-NodeDoubleCut.

By the above claim, the expected cost of the returned solution is

$$\mathbb{E}\left(\sum_{v\in U} c_v\right) = \sum_{v\in V} \mathsf{Pr}(v \in U)c_v \leq 2\sum_{v\in V} c_v d_v.$$

Although our rounding algorithm is a randomized algorithm, it can be derandomized using standard techniques. ◀

Our next lemma shows a lower bound on the integrality gap that nearly matches the approximation factor achieved by our rounding algorithm.

▶ **Lemma 20.** *The integrality gap of the Path-Blocking-LP for directed graphs containing n nodes is at least $2 - 7/n^{1/3}$.*

Our integrality gap instance is also helpful in understanding the hardness of approximation of $\{s,t\}$-NodeDoubleCut. So, we define the instance below and summarize its properties which will be used in the proof of Lemma 20 as well as in the proof of hardness of approximation.

For two integers $a, b \in \mathbb{N}$, consider the directed graph $D_{a,b} = (V_D, A_D)$ obtained as follows (see Figure 3): Let $V_D := \{s,t\} \cup ([a] \times [b])$. There are $ab+2$ nodes. Let $I_D := [a] \times [b]$ and call them as the *internal nodes*. The set of arcs A_D are as follows:

1. For each $1 \leq i \leq a$, there is a bidirected arc between s and $(i,1)$, and a bidirected arc between (i,b) and t.
2. For each $1 \leq i \leq a$ and $1 \leq j < b$, there is a bidirected arc between (i,j) and $(i,j+1)$.
3. For each $1 \leq i < a$ and $2 \leq j \leq b-1$, there is an arc from (i,j) to $(i+1,j-2)$, and an arc from (i,j) to $(i+1,j+2)$ (let $(i,0) := s$ and $(i,b+1) := t$ for every i). Call them *jumping arcs*.

▶ **Lemma 21.** *$D_{a,b}$ has the following properties:*

1. *For each internal node $\alpha = (\alpha_1, \alpha_2) \in I_D$, each $\alpha \to s$ path has at least $\alpha_2 - a$ internal nodes other than α. Similarly, each $\alpha \to t$ path has at least $b - \alpha_2 - a + 1$ internal nodes other than α.*
2. *If $S \subseteq I_D$ is such that the subgraph induced by $V_D \setminus S$ has no node v that has paths to both s and t, then $|S| \geq 2a - 1$.*

Proof.

1. Jumping arcs are the only arcs that change α_2 by 2 while all other arcs change α_2 by 1. However, a path to s can use at most $a-1$ jumping arcs because they strictly increase α_1. The first property follows from these observations.
2. Suppose that $S \subseteq I_D$ is such that the subgraph induced by $V_D \setminus S$ has no node v that has paths to both s and t. For $i = 1, \ldots, a$, let $s_i := |S \cap \{\{i\} \times [b]\}|$. We note that $s_i \geq 1$ for each i, otherwise s can reach t and t can reach s.
   Suppose $s_i = 1$ for some $1 < i \leq a$ and let j be such that $S \cap \{\{i\} \times [b]\} = (i, j)$. If $j = 1$, then $(i, 2) \in V_D \setminus S$ and $(i, 2)$ can reach both s and t. If $j = b$, then $(i, b-1) \in V_D \setminus S$ and $(i, b-1)$ can reach both s and t. Therefore, we have $1 < j < b$. Then $s_{i-1} \geq 3$ because $(i-1, j-1), (i-1, j), (i-1, j+1)$ can reach both s and t using one jumping arc followed by regular arcs in the ith row.
   Therefore, $|S| = \sum_{i=1}^{a} s_i \geq 1 + 2(a-1) = 2a - 1$. ◀

Proof of Lemma 20. The integer optimum of Path-Blocking-LP on $D_{a,b}$ is at least $2a-1$ by the second property of Lemma 21. Let $r := b - 2a + 1$. We set $d_v := 1/r$ for every internal node v. The resulting solution is feasible to Path-Blocking-LP: Indeed, consider $\alpha = (\alpha_1, \alpha_2)$. By the first property of Lemma 21, any $\alpha \to s$ path and $\alpha \to t$ path have to together traverse at least $\alpha_2 - a + (b - \alpha_2 - a + 1) = r$ internal nodes.

Setting $b = a^2$, the integrality gap is at least $(2a-1)/(a^3/r) = 2 - 1/a^3 + 4/a^2 - 5/a \geq 2 - 6/a$ for $a \geq 2$. Using the fact that $a = (|V(D_{a,b})| - 2)^{1/3}$, we get the desired bound on the integrality gap. ◀