On Kernelization and Approximation for the Vector Connectivity Problem

Stefan Kratsch* and Manuel Sorge†

Technical University Berlin, Germany, 
{stefan.kratsch, manuel.sorge}@tu-berlin.de

November 3, 2014

Abstract

In the vector connectivity problem we are given an undirected graph $G = (V, E)$, a demand function $\phi: V \rightarrow \{0, \ldots, d\}$, and an integer $k$. The question is whether there exists a set $S$ of at most $k$ vertices such that every vertex $v \in V \setminus S$ has at least $\phi(v)$ vertex-disjoint paths to $S$. The problem is NP-hard already for instances with $d = 4$ (Cicalese et al., manuscript ’14), admits a log-factor approximation (Boros et al., Networks ’14), and is fixed-parameter tractable in terms of $k$ (Lokshtanov, private communication ’14).

We prove several results regarding kernelization and approximation for vector connectivity and vector $d$-connectivity where the upper bound $d$ on demands is a fixed constant. Our main result is a vertex-linear kernelization for vector $d$-connectivity($k$). We complement this result by proving that vector connectivity($k$) has no polynomial kernelization, unless $\text{NP} \subseteq \text{coNP/poly}$, even for parameter $d + k$. Thus, size $f(d) \cdot \text{poly}(k)$ for a superpolynomial function $f$ is best possible. As a stepping stone for the kernelization we give a reduction rule that reduces the total demand in the input, which may be of independent interest. The rule also applies to vector connectivity and we directly obtain approximation algorithms of ratio $\text{opt}^2 + 1$ for vector connectivity and constant-factor $d^2$ for vector $d$-connectivity.

1 Introduction

In the vector connectivity problem we are given an undirected graph $G = (V, E)$, a demand function $\phi: V \rightarrow \{0, \ldots, d\}$, and an integer $k \in \mathbb{N}$.

*Supported by the Emmy Noether-program of the German Research Foundation (DFG), KR 4286/1-1.
†Supported by the German Research Foundation (DFG), project DAPA (NI 369/12-2).
The question is whether there exists a set $S$ of at most $k$ vertices of $G$ such that every vertex $v \in V$ is either in $S$ or has at least $\phi(v)$ vertex-disjoint paths to vertices in $S$ (where the paths may share the vertex $v$ itself). The study of the VECTOR CONNECTIVITY problem was initiated recently by Boros et al. [1, 5] motivated by its similarity to well-studied domination and connectivity problems. In particular, if paths to $S$ are restricted to length one then we obtain the VECTOR DOMINATION problem, which generalizes VERTEX COVER and DOMINATING SET amongst others. Boros et al. give polynomial-time algorithms for trees, cographs, and split graphs. Moreover, they obtain an ln $n + 2$-factor approximation algorithm for the general case.

More recently, Cicalese et al. [7] continued the study of VECTOR CONNECTIVITY and proved that it is APX-hard (and NP-hard) on general graphs, even when all demands are upper bounded by four. In fact, this holds even for the special case that each demand $\phi(v)$ is either 0 or $d$ for some fixed $d \geq 4$ (Milanič [20]). Cicalese et al. [7] also obtain NP-hardness on several restricted graph classes and a polynomial-time algorithm for block graphs. In a related talk during a recent Dagstuhl seminar, Milanič asked whether VECTOR CONNECTIVITY is fixed-parameter tractable with respect to the maximum solution size $k$ (see Section 2 for definitions). This was answered affirmatively by Lokshtanov [16].

Motivated by the fixed-parameter tractability result we study the kernelizability of VECTOR CONNECTIVITY, with parameter $k$, i.e., whether there is an efficient algorithm that preprocesses instances of VECTOR CONNECTIVITY to size bounded in terms of $k$. Our interest stems from the progress on kernelization (and also fixed-parameter algorithms) that has been made while studying cut, feedback, and connectivity problems (see, e.g., [6, 19, 8, 15]). As a byproduct of our work we also obtain new approximability results for VECTOR CONNECTIVITY and a variant thereof where demands are upper-bounded by a fixed constant.

**VECTOR CONNECTIVITY**

**Instance:** A graph $G = (V, E)$, a function $\phi: V \to \{0, \ldots, d\}$, and $k \in \mathbb{N}$.

**Question:** Is there a set $S$ of at most $k$ vertices such that each vertex $v \in V \setminus S$ has $\phi(v)$ vertex-disjoint paths with endpoints in $S$?

The value $\phi(v)$, $v \in V$ is also called the demand of vertex $v$. We call a set $S$ a vector connectivity set for $(G, \phi)$, if it fulfills the requirements above. We do not formally distinguish between decision and optimization version; clearly, $k$ is not needed in the input for the latter.

--

1Dagstuhl Seminar 14071 on “Graph Modification Problems.”
Our results. Our main result is a vertex-linear kernelization for vector $d$-connectivity($k$), where the maximum demand $d$ is a problem-specific constant and with parameter $k$. The result uses ideas similar to Marx' notion of important separators [17] and the protrusion-replacement strategy used in meta kernelization results (e.g., [2, 13]), but neither finds direct application. The size of the kernel is $O(k^2)$ and the number of vertices in a kernelized instance is $O(k)$; the hidden constants are superpolynomial in $d$. We complement this result by proving that vector connectivity($k$), i.e., with unrestricted demand values, admits no polynomial kernelization unless $NP \subseteq coNP/poly$. This holds also under the more restrictive parameterization by $k + d$, which supports the superpolynomial dependence on $d$ in our kernelization for vector $d$-connectivity($k$).

As a crucial subroutine of our kernelization we provide a reduction rule that reduces (and in a sense sparsifies) the total demand in the input, which may be of independent interest. The rule is independent of the value of $k$ and preserves all feasible solutions, rather than maintaining only the status of the decision problem (whether there is at least one solution of value at most $k$). We prove that reduced instances have a total number of nonzero demand vertices (i.e., $\phi(v) > 0$) that is at most $d^2 opt$, where $opt$ denotes the (unknown) minimum size of a feasible solution. This yields a constant-factor $d^2$ approximation for vector $d$-connectivity and an $(opt^2 + 1)$-factor approximation for vector connectivity. Note that APX- and NP-hardness of vector $d$-connectivity for $d = 4$ are proven by a nontrivial but fairly direct reduction from vertex cover [7], making the problem an interesting generalization of vertex cover that still enjoys constant-factor approximability.

Finally, the reduction rule can also be used to reduce to at most $k^3 + k$ nonzero demand vertices (or correctly answer NO). Using this we derive an alternative FPT algorithm for vector connectivity($k$) using an algorithm of Marx [18] that finds a set of size $k$ that is independent in each of $\ell$ given linear matroids in time that is FPT with respect to $k + \ell$.

Related work. Considering the vertex-linear kernelization for vector $d$-connectivity($k$) there is some similarity to the mentioned meta-kernelization results for problems restricted to sparse graph classes [2, 13]. Upon closer inspection, however, the two approaches differ substantially since we assume no restriction on the input graphs and we make no use of treewidth-related techniques (nor would it seem that they apply).

Our lower bound ruling out polynomial kernels for vector connectivity($k$) is by reduction from hitting set parameterized by the number $m$ of sets. Apart from this, for the standard parameterization (with or without restricted demand/set size) the problems behave very differently: hitting set($k$) is $W[2]$-complete and $d$-hitting set($k$) admits an essentially opti-
nal kernel of size $O(k^d)$ [11 19]. Vector connectivity($k$) is FPT [16] and vector $d$-connectivity($k$) has a vertex-linear kernelization of size $O(k^2)$.

**Organization.** In Section 2 we recall some definitions from graph algorithms and properties of so-called closest sets, which we frequently use. In Section 3 we present the reduction rule and the obtained approximation results. Section 4 contains the vertex-linear kernelization for vector $d$-connectivity($k$). In Section 5 we prove the kernelization lower bound for vector connectivity($k$). We conclude in Section 6. For background on parameterized complexity we refer to the literature [11 12 21].

## 2 Preliminaries

We consider only undirected, simple graphs $G = (V, E)$ where $V$ is a finite set and $E \subseteq \binom{V}{2}$; we use $V(G) := V$ and $E(G) := E$. (Here, $\binom{V}{2}$ denotes the family of all size-two subsets of $V$.) The open neighborhood $N_G(v)$ of a vertex $v \in V(G)$ is defined by $N_G(v) := \{u \mid \{u, v\} \in E(G)\}$; the closed neighborhood is $N_G[v] := N_G(v) \cup \{v\}$. For vertex sets $X \subseteq V(G)$ we define $N_G[X] := \bigcup_{v \in X} N_G(v)$ and $N_G(X) := N_G[X] \setminus X$. For $X \subseteq V(G)$ we define the subgraph induced by $X$, denoted $G[X]$, as the graph $G' = (X, E')$ with $E' := \{\{u, v\} \mid \{u, v\} \in E(G) \land u, v \in X\}$. By $G - X$ we mean the graph $G[V(G) \setminus X]$.

Due to the nature of the Vector Connectivity problem we are frequently interested in disjoint paths from a vertex $v$ to some vertex set $S$, where the paths are vertex-disjoint except for sharing $v$. The natural counterpart, in the spirit of Menger’s theorem, are $v, S$-separators $C \subseteq V(G) \setminus \{v\}$ such that in $G - C$ no vertex of $S \setminus C$ is reachable from $v$; the maximum number of disjoint paths from $v$ to $S$ that may overlap in $v$ equals the minimum size of a $v, S$-separator. Throughout, by disjoint paths from $v$ to $S$, or $v, S$-separator (for any single vertex $v$ and any vertex set $S$) we mean the mentioned path packings and separators with special role of $v$. (In a network flow interpretation of the paths between $v$ and $S$, this essentially corresponds to giving $v$ unbounded capacity, whereas all other vertices have unit capacity.) Several proofs use the function $f : 2^V \to \mathbb{N} : U \mapsto |N(U)|$, which is well-known to be submodular, i.e., for all $X, Y \subseteq V$ it holds that $f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y)$.

So-called closest sets will be used frequently; these occur naturally in cut problems but appear to have no generalized name. As an example, if $C$ is a minimum $s,t$-vertex cut that, amongst such cuts, has the smallest connected component for $s$ in $G - C$ then $C$ is also closest. Formally, without restricting to cuts specifically, we define a vertex set $C$ to be closest to $v$ if $C$ is the unique $v, C$-separator of size at most $|C|$, where $v, C$-separator is in the above sense. The following proposition captures some properties of
closest sets, mostly specialized to \(v, S\)-separators (see also \([15]\)).

**Proposition 1.** Let \(G = (V, E)\), let \(v \in V\), and let \(S \subseteq V \setminus \{v\}\). The following holds.

1. If \(C\) is a closest \(v, S\)-separator and \(X\) the connected component of \(v\) in \(G - C\), then for every set \(X' \subseteq X\) with \(v \in X' \subseteq X\) we have \(|N(X')| > |N(X)|\).

2. The minimum \(v, S\)-separator \(C\) minimizing the size of the connected component of \(v\) in \(G - C\) is unique. The set \(C\) is also the unique minimum \(v, S\)-separator closest to \(v\).

3. If \(C_1\) and \(C_2\) are minimum \(v, S\)-separators and \(C_1\) is closest to \(v\), then the connected component of \(v\) in \(G - C_1\) is fully contained in the connected component of \(v\) in \(G - C_2\).

4. If \(C \subseteq V \setminus \{v\}\) is closest to \(v\) then so is every subset \(C'\) of \(C\).

**Proof.** Let \(G = (V, E)\), let \(v \in V\), and let \(S \subseteq V \setminus \{v\}\). We give simple proofs for all claims; some of them use submodularity of \(f : 2^V \to \mathbb{N} : U \mapsto |N(U)|\).

1. Note that \(N(X) = C\) or else \(|N(X)| < |C|\) and \(N(X)\) is a \(v, C\)-separator, contradicting closeness of \(C\). Let \(X' \subseteq X\) with \(v \in X' \subseteq X\). Clearly \(X' \cap C \subseteq X \cap C = \emptyset\), making \(N(X')\) a \(v, C\)-separator. Since \(X' \subseteq X\) and \(X\) is connected, it follows that \(N(X') \cap X = \emptyset\) and, thus, that \(|N(X')| > |C| = |N(X)|\). Since \(C\) is closest to \(v\), we must have \(|N(X')| > |C| = |N(X)|\) or else \(N(X')\) would be a \(v, C\)-separator of size at most \(|C|\) but different from \(C = N(X)\).

2. Assume that both \(C_1\) and \(C_2\) are minimum \(v, S\)-separators that furthermore minimize the size of the connected component of \(v\) in \(G - C_1\); let \(X_i\) denote those components. Note that \(N(X_i) = C_i\) since \(C_i\) is a minimum \(v, S\)-separator. We have

\[
f(X_1) + f(X_2) \geq f(X_1 \cap X_2) + f(X_1 \cup X_2).
\]

Clearly, \((X_1 \cup X_2) \cap S = \emptyset\), implying that \(N(X_1 \cup X_2)\) is a \(v, S\)-separator. It follows that

\[
|N(X_1 \cup X_2)| = f(X_1 \cup X_2) \geq f(X_2),
\]

which implies that \(f(X_1) \geq f(X_1 \cap X_2)\). It is easy to see that \(N(X_1 \cap X_2)\) is also a \(v, S\)-separator of size at most \(f(X_1) = |C_1|\) and, if \(C_1 \neq C_2\), then \(X_1 \neq X_2\) and \(X_1 \cap X_2 \subseteq X_1\); a contradiction to \(X_1\) being a minimum size connected component. Thus, we have \(C_1 = C_2\), proving the first part of the claim.

For the second part, assume first that \(C_3\) is a \(v, C_1\)-separator of size at most \(|C_1|\) and let \(X_3\) the connected component of \(v\) in \(G - C_3\). Since every path from \(v\) to \(S\) contains a vertex of \(C_1\), each such path must
also contain a vertex of \( C_3 \) (to separate \( v \) from \( C_1 \)). Thus \( C_3 \) is also a \( v, S \)-separator, but then \( C_3 \) is also minimum (as \( |C_3| \leq |C_1| \)). Note that, since \( C_3 \) is a \( v, C_1 \)-separator we have \( X_3 \cap C_1 = \emptyset \). It follows that \( X_3 \subseteq X_1 \). If \( X_3 \nsubseteq X_1 \) then \( C_3 \) would violate the choice of \( C_1 \) as minimum \( v, S \)-separator with minimum component size for \( v \). But then we must have \( X_3 = X_1 \) and \( C_1 = C_2 \), which proves closeness of \( C_1 \) to \( v \).

Finally, assume that \( C_4 \) is another minimum \( v, S \)-separator closest to \( v \); for uniqueness we want to show \( C_1 = C_4 \). If \( X_1 \) and \( X_4 \) are the corresponding connected components, then \( C_i = N(X_i) \). By submodularity of \( f \) and the same arguments as above we find that \( |N(X_i \cap X_4)| \leq |C_i| \).

If \( X_1 \nsubseteq X_4 \) then \( v \in X_1 \cap X_4 \nsubseteq X_1 \) which, by property \( \mathbb{I} \) implies \( |N(X_i \cap X_4)| > |N(X_i)| = |C_i| \); a contradiction. Else, if \( X_1 \subseteq X_4 \), then \( C_1 = N(X_1) \) is a \( v, C_4 \)-separator of size \( |C_1| = |C_4| \); this implies \( C_1 = C_4 \) by closeness of \( C_4 \).

3. Let \( C_1 \) and \( C_2 \) minimum \( v, S \)-separators and let \( C_1 \) be closest to \( v \). Let \( X_i \) the connected component of \( v \) in \( G - C_i \). Note that \( N(X_i) = C_i \) since \( C_i \) is minimum. Assume, for contradiction, that \( X_1 \nsubseteq X_2 \), implying that \( X_1 \cap X_2 \nsubseteq X_1 \). By property \( \mathbb{I} \) we have \( f(X_1 \cap X_2) = |N(X_1 \cap X_2)| > |C_1| = |C_2| = f(X_2) \). Because

\[
    f(X_1) + f(X_2) \geq f(X_1 \cap X_2) + f(X_1 \cup X_2),
\]

we have \( |N(X_1 \cup X_2)| < f(X_1) = |C_1| = |C_2| \). However, \( N(X_1 \cup X_2) \) is also a \( v, S \)-separator; this contradicts \( C_1 \) and \( C_2 \) being minimum \( v, S \)-separators.

4. Let \( C \) closest to \( v \) and let \( C' \subseteq C \). If \( C' \) is not closest to \( v \) then there is a \( v, C' \)-separator \( C'' \) of size at most \( |C'| \) and with \( C'' \neq C' \). Consider \( \tilde{C} = C'' \cup (C \setminus C'') \) and note that \( |\tilde{C}| \leq |C''| + |C\setminus C'| \leq |C'| + |C\setminus C'| = |C| \). Observe that \( \tilde{C} \) is a \( v, C \)-separator since \( C'' \subseteq \tilde{C} \) separates \( v \) from \( C' \), and \( C \setminus C' \) is contained in \( \tilde{C} \). Thus, \( |\tilde{C}| < |C| \) would contradict \( C \) being closest to \( v \) because it implies that \( \tilde{C} \neq C \) in addition to \( \tilde{C} \) being a \( v, C \)-separator of size at most \( |C| \). Thus, \( |\tilde{C}| = |C| \) and by closeness of \( C \) to \( v \), we have \( \tilde{C} = C \). The latter can hold only if \( C'' \supseteq C' \), because \( \tilde{C} = C'' \cup (C \setminus C') \), but then \( C'' = C' \) because \( |C''| \leq |C'| \), contradicting \( C'' \neq C' \). Thus \( C' \) is indeed closest to \( v \).

This completes the proof. \( \square \)

3 Reducing the number of demand vertices

In this section we introduce a reduction rule for vector connectivity that reduces the total demand. We prove that the reduction rule does not affect
the solution space, which makes it applicable not only for kernelization but also for approximation and other techniques. In the next section, we will use this rule in our polynomial kernelization for VECTOR d-CONNECTIVITY(k).

In the present section, as more direct applications, we get approximation results for VECTOR CONNECTIVITY and VECTOR d-CONNECTIVITY. Finally, we sketch an alternative FPT algorithm for VECTOR CONNECTIVITY(k), using a result of Marx [15].

**Reduction Rule 1.** Let \((G, \phi, k)\) be an instance of VECTOR CONNECTIVITY. If a vertex \(v \in V\) has at least \(\phi(v)\) vertex-disjoint paths to vertices different from \(v\) with demand at least \(\phi(v)\) then set the demand of \(v\) to zero.

We prove that the rule does not affect the space of feasible solutions for any instance.

**Lemma 1.** Let \((G, \phi, k)\) be an instance of VECTOR CONNECTIVITY and let \((G, \phi', k)\) be the instance obtained via a single application of Rule 1. For every \(S \subseteq V(G)\) it holds that \(S\) is a solution for \((G, \phi, k)\) if and only if \(S\) is a solution for \((G, \phi', k)\).

**Proof.** Let \(v\) denote the vertex whose demand was set to zero by the reduction rule and define \(r := \phi(v)\). Clearly, \(\phi(u) = \phi'(u)\) for all vertices \(u \in V(G) \setminus \{v\}\), and \(\phi'(v) = 0\). It suffices to show that if \(S\) fulfills demands according to \(\phi'\) then \(S\) fulfills also demands according to \(\phi\) since \(\phi(u) \geq \phi'(u)\) for all \(u \in V(G)\). This in turn comes down to proving that \(S\) fulfills the demand of \(r\) at \(v\) assuming that it fulfills demands according to \(\phi'\). If \(v \in S\) then the demand at \(v\) is trivially fulfilled so henceforth assume that \(v \notin S\).

Let \(w_1, \ldots, w_r\) denote vertices different from \(v\) with demand each at least \(r\) such that there exist \(r\) vertex-disjoint paths from \(v\) to \(\{w_1, \ldots, w_r\}\), i.e., a single path to each \(w_i\). Existence of such vertices is required for the application of the rule.

Assume for contradiction that \(S\) does not satisfy the demand of \(r\) at \(v\) (recall that \(v \notin S\), by assumption), i.e., that there are no \(r\) vertex-disjoint paths from \(v\) to \(S\) that overlap only in \(v\). It follows directly that there is a \(v, S\)-separator \(C\) of size at most \(r - 1\). (Recall that \(C\) may contain vertices of \(S\) but not the vertex \(v\).) Let \(R\) denote the connected component of \(v\) in \(G - C\), then the following holds for each vertex \(w_i \in \{w_1, \ldots, w_r\}\):

1. If \(w_i \in S\) then \(w_i \notin R\): Otherwise, we would have \(S \cap R \supseteq \{w_i\} \neq \emptyset\) contradicting the fact that \(v\) can reach no vertex of \(S\) in \(G - C\).

2. If \(w_i \notin S\) then \(w_i \notin R\): Since \(S\) fulfills demands according to \(\phi'\) there must be at least \(r\) vertex-disjoint paths from \(w_i\) to \(S\) that overlap only in \(w_i\). However, since \(w_i \in R\) the set \(C\) is also a \(w_i, S\)-separator; a contradiction since \(C\) has size less than \(r\).
Thus, no vertex from $w_1, \ldots, w_r$ is contained in $R$. This, however, implies that $C$ separates $v$ from $\{w_1, \ldots, w_r\}$, contradicting the fact that there are $r$ vertex-disjoint paths from $v$ to $\{w_1, \ldots, w_r\}$ that overlap only in $v$. It follows that no such $v, S$-separator $C$ can exist, and, hence, that there are at least $r = \phi(v)$ vertex-disjoint paths from $v$ to $S$, as claimed. Thus, $S$ fulfills the demand of $r$ at $v$ and hence all demand according to $\phi$. (Recall that the converse is trivial since $\phi(u) \geq \phi'(u)$ for all vertices $u \in V(G)$.)

We have established that applications of Rule 1 do not affect the solution space of an instance while reducing the number of vertices with nonzero demand.

**Lemma 2.** Rule 1 can be exhaustively applied in polynomial time.

**Proof.** To check whether the rule applies to some vertex $v$ with demand $r$ it suffices to perform one maximum flow computation for source $v$ (with unbounded capacity) and using all vertices with demand at least $r$ as sinks (with capacity one). Finding a vertex suitable for applying Rule 1 thus takes only polynomial time and each application reduces the number of nonzero demand vertices decreases by one limiting the number of iterations to $|V(G)|$.

To analyze the impact of Rule 1 we will now bound the number of nonzero demand vertices in an exhaustively reduced instance in terms of the optimum solution size $\text{opt}$ and the maximum demand $d$. To this end, we require the following technical lemma about the structure of reduced instances as well as some notation.

If $(G, \phi, k)$ is reduced according to Rule 1 then for each vertex $v$ with demand $r = \phi(v) \geq 1$ there is a cut set $C$ of size at most $r - 1$ that separates $v$ from all other vertices with demand at least $r$. We fix for each vertex $v$ with demand at least one a vertex set $C'(v)$, by picking the unique closest minimum $v, D'$-separator where $D' = \{u \in V \setminus \{v\} \mid \phi(u) \geq \phi(v)\}$. Furthermore, for such vertices $v$, let $R(v)$ denote the connected component of $v$ in $G - C'(v)$.

Intuitively, any solution $S$ must intersect every set $R(v)$ since $|C'(v)| < \phi(v)$. The following lemma shows implicitly that Rule 1 limits the amount of overlap of sets $R(v)$.

**Lemma 3.** Let $(G, \phi, k)$ be reduced under Rule 1. Let $u, v \in V(G)$ be distinct vertices with $\phi(u) = \phi(v) \geq 1$. If $R(u) \cap R(v) \neq \emptyset$ then $u \in C'(v)$ or $v \in C(u)$.

**Proof.** Assume for contradiction that we have $u, v$ with $\phi(u) = \phi(v) \geq 1$ and with $R(u) \cap R(v) \neq \emptyset$ and $u \notin C'(v)$ and $v \notin C(u)$. We will show that this implies that at least one of $C(u)$ and $C(v)$ is not a closest minimum cut, giving a contradiction. By definition of cuts $C(u)$ and $C(v)$ as separating $u$
resp. \( v \) from all other vertices of at least the same demand, we have \( u \notin R(v) \) and \( v \notin R(u) \); furthermore \( u \notin C(u) \) and \( v \notin C(v) \), by definition.

Let \( C = C(u) \cup C(v) \) and note that \( u, v \notin C \). Let \( I, J \) denote the connected components of \( u, v \) in \( G - C \). Note that \( I \subseteq R(u) \) since \( C \supseteq C(u) \) and, thus, \( v \notin I \). Similarly we have \( u \notin J \) and thus \( I \) and \( J \) are two different connected components in \( G - C \). As the next step, we show that

\[
|C(u)| + |C(v)| \geq |N(I)| + |N(J)|. \tag{1}
\]

To this end, let us first note that \( N(I) \cup N(J) \subseteq C = C(u) \cup C(v) \) by definition of \( I \) and \( J \) as connected components of \( G - C \). Thus, every vertex \( p \) that appears in exactly one of \( N(I) \) and \( N(J) \) contributes value one to the right-hand side of (1) and at least value one to the left-hand side since it must be contained in \( C(u) \cup C(v) \). Now, for vertices \( p \in N(I) \cap N(J) \) we see that they contribute value two to the right-hand side of (1). Note that each such vertex is contained in a path from \( u \) to \( v \) whose other interior vertices are disjoint from \( C = C(u) \cup C(v) \). Thus, \( p \) must be contained in both \( C(u) \) and \( C(v) \) since otherwise the corresponding set would fail to separate \( u \) from \( v \) (or vice versa), which is required since \( \phi(u) = \phi(v) \). Therefore, if any vertex contributes a value of two to the right-hand side, then it also contributes two to the left-hand side. This establishes Equation (1).

Now, from (1) we immediately get that at least one of \( |N(I)| \leq |C(u)| \) or \( |N(J)| \leq |C(v)| \) must hold. W.l.o.g., let \( |N(I)| \leq |C(u)| \). Recall that \( u \in I \). Furthermore, we can see that \( I \subseteq R(u) \) by using the fact that \( R(u) \cap R(v) \neq \emptyset \): Let \( q \in R(u) \cap R(v) \subseteq R(u) \). If \( q \in I \) then \( u \) can reach \( q \) in \( G - C \) but, in particular, also in \( G - C(v) \). By definition of \( R(v) \) and using \( q \in R(v) \), we know that \( v \) can reach \( q \) in \( G - C(v) \), implying that there is a path from \( u \) to \( v \) in \( G - C(v) \) (since there is a walk through \( q \)), violating the fact that \( C(v) \) separates \( v \) from \( u \) (amongst others). Thus \( q \notin I \) and since \( I \subseteq R(u) \) follows from \( C \supseteq C(u) \), we get that \( I \subseteq R(u) \). Since \( |N(I)| \leq |C(u)| \) we find that \( N(I) \) is of at most the same size as \( C(u) \) but with a smaller connected component \( I \) for \( u \), contradicting the fact that \( C(u) \) is the unique minimum closest set that separates \( u \) from all other vertices of demand at least \( \phi(u) \). This completes the proof of the lemma.

Now, we can give the promised bound on the number of nonzero demand vertices.

**Lemma 4.** Let \((G, \phi, k)\) be an instance of vector connectivity that is reduced according to Rule 1 and let \( \text{opt} \) denote the minimum size of feasible solutions \( S \subseteq V \) for this instance. Then there are at most \( d^2 \text{opt} \) nonzero demand vertices in \( G \).

**Proof.** For analysis, let \( S \subseteq V \) denote any feasible solution of size \( \text{opt} \), i.e., such that every \( v \) with \( \phi(v) \geq 1 \) has \( v \in S \) or there are \( \phi(v) \) vertex-disjoint paths from \( v \) to \( S \) that overlap only in \( v \). We will prove that for all
$r \in \{1, \ldots, d\}$ there are at most $2r - 1$ vertices of demand $r$ in $G$ (according to $\phi$). Fix some $r \in \{1, \ldots, d\}$ and let $D_r$ denote the set of vertices with demand exactly $r$. For each $v \in D_r$ the solution $S$ must contain at least one vertex of $R(v)$ since $C(v) = N(R(v))$ has size at most $r - 1$. (Otherwise, $C(v)$ would be a $v, S$-separator of size less than $r$.) Fix some vertex $p \in S$ and let $v_1, \ldots, v_\ell$ denote all vertices of demand $r$ that have $p \in R(v_i)$. We will prove that $\ell \leq (2r - 1)$ and $|D_r| \leq \text{opt}(2r - 1)$.

At most $r - 1$ vertices are contained in $C(v_i)$ for every $i \in \{1, \ldots, \ell\}$. Thus, on the one hand the total size $\sum |C(v_i)|$ of these sets is at most $(r - 1)\ell$. On the other hand, for every pair $v_i, v_j$ with $1 \leq i < j \leq \ell$ we know that $v_i \in C(v_j)$ or $v_j \in C(v_i)$ by Lemma 4 since $R(v_i) \cap R(v_j) \supseteq \{p\} \neq \emptyset$. Thus, every pair contributes at least a value of one to the total size of the sets $C(v_i)$. (To see that different pairs have disjoint contributions note that, e.g., $v_i \in C(v_j)$ uniquely defines pair $v_i, v_j$.) We get the following inequality:

$$(r - 1)\ell \geq \sum_{i=1}^{\ell} |C(v_i)| \geq \left(\frac{\ell}{2}\right).$$

Thus, $\frac{1}{4}\ell(\ell - 1) \leq (r - 1)\ell$, implying that $\ell \leq 2r - 1$. Since there are exactly $\text{opt}$ choices for $p \in S$ and every set $R(v)$ for $v \in D_r$ must be intersected by $S$, we get an upper bound of

$$\text{opt} \cdot \ell \leq \text{opt}(2r - 1)$$

for the size of $D_r$. If we sum this over all choices of $r \in \{1, \ldots, d\}$ we get an upper bound of

$$\sum_{r=1}^{d} \text{opt}(2r - 1) = \text{opt} \sum_{r=1}^{d} 2r - 1 = \text{opt} \cdot d^2$$

for the number of vertices with nonzero demand. This completes the proof.

Lemma 4 has several nice consequences for both kernelization and approximation. Let us first discuss approximability. The key point in this regard is that taking all vertices of nonzero demand constitutes a feasible solution. After having reduced the number of these vertices via Rule and using Lemma 4 this constitutes a viable approximation algorithm.

**Theorem 1.** **Vector $d$-connectivity** has a constant-factor approximation algorithm with ratio $d^2$. **Vector connectivity** has an approximation algorithm with ratio $(\text{opt}^2 + 1)$, i.e., the algorithm guarantees to find a solution of cost at most $\text{opt}^3 + \text{opt}$. 

10
Proof. Both algorithms proceed by first applying Rule 1 exhaustively, which uses polynomial time, and then simply returning the set of all nonzero demand vertices as the proposed solution. In the case of \textsc{vector $d$-connectivity}, Lemma 4 already guarantees that this set has size at most $d^2 \text{opt}$ and, as mentioned earlier, it is a feasible solution. For \textsc{vector connectivity} a little more discussion is necessary.

Let $(G, \phi, k)$ denote an instance of \textsc{vector connectivity} that is reduced according to Rule 1 and let $S \subseteq V(G)$ a feasible solution of size $\text{opt}$. From the proof of Lemma 4 we know that $G$ contains at most $\text{opt}(2r - 1)$ vertices of demand exactly $r$. For $r > \text{opt}$ we have a better bound: Observe that $S$ must contain all vertices of demand greater than $\text{opt}$, since their demand for vertex-disjoint paths exceeds the size of $S$. Thus, there are at most $\text{opt}$ vertices of demand greater than $\text{opt}$ in $G$ and at most

$$
\sum_{r=1}^{\text{opt}} \text{opt}(2r - 1) = \text{opt} \sum_{r=1}^{\text{opt}} (2r - 1) = \text{opt} \cdot \text{opt}^2 = \text{opt}^3,
$$

vertices of demand up to $\text{opt}$. Thus, the returned set of nonzero demand vertices is a feasible solution of size at most $\text{opt}^3 + \text{opt}$, which corresponds to a ratio of $(\text{opt}^2 + 1)$.

We now turn to kernelization and derive a reduction rule from Lemma 4. The crux is that a large number of nonzero demand vertices in a reduced instance implies that $\text{opt}$ is large; if it is guaranteed to be larger than $k$ then we may safely reject the instance.

**Reduction Rule 2.** Let $(G, \phi, k)$ be reduced according to Rule 1 with $\phi: V(G) \to \{0, \ldots, d\}$. If there are more than $d^2 k$ vertices of nonzero demand then return NO.

Note that for the purpose of finding a solution of size at most $k$ we may safely reject an instance if it has more than $k$ vertices of demand exceeding $k$, since all of these are part of the solution. Thus, similarly to the approximation for \textsc{vector connectivity}, an instance may be rejected if it has more than $k^3 + k$ vertices with nonzero demand. We point out, however, that we will only use the reduction rule for the case of \textsc{vector $d$-connectivity} $(k)$ where $k$ is the sole parameter and $d$ is constant.

**Lemma 5.** Rule 2 is safe.

*Proof.* Lemma 4 guarantees that there are at most $d^2 \text{opt}$ nonzero demand vertices in instances that are reduced under Rule 1. Thus, if there are more than $d^2 k$ such vertices it immediately follows that $\text{opt} > k$. Thus, it is correct to answer NO in this case.
The combination of Rules 1 and 2 also allows an efficient reduction to at most \( k^3 + k \) nonzero demand vertices. Using a result of Marx \[18\] on fixed-parameter tractability of a parameterized (and generalized) matroid parity problem, this gives an alternative proof of fixed-parameter tractability for vector connectivity\((k)\). Concretely, this permits us to search for an independent set of size \( k \) in \( k^3 + k \) linear matroids over the same ground set. (A matroid is linear if it contains as independent sets precisely the sets of linearly independent columns of a fixed matrix. For more information see, e.g., Oxley’s book \[22\].)

**Theorem 2** (Lokshtanov \[16\]). *Vector connectivity*\((k)\) *is fixed-parameter tractable.*

**Proof.** We sketch an alternative proof for fixed-parameter tractability of vector connectivity\((k)\). W.l.o.g., input instances \((G, \phi, k)\) are already reduced to at most \( k^3 + k \) nonzero demand vertices (else apply the reduction rules); let \( D = \{v \in V(G) \mid \phi(v) \geq 1\} \). Clearly, if the instance is \textsc{yes} then there exist also solutions of size exactly \( k \) (barring \(|V(G)| < k \) which would be trivial).

**Algorithm.** As a first step, we guess the intersection of a solution \( S^* \) of size \( k \) with the set \( D \); there are at most \((k^3 + k)^k\) choices for \( S_0 = D \cap S^* \). Note that all vertices of demand exceeding \( k \) must be contained in \( S_0 \) for \( S^* \) to be a solution (we ignore \( S_0 \) if this is not true).

For each \( v \in D \setminus S_0 \), we construct a matroid \( M_v \) over \( V' = V \setminus D \) as follows.

- Build a graph \( G_v \) by first adding to \( G \) additional \( c - 1 \) copies of \( v \), called \( v_2, \ldots, v_c \), where \( c = \phi(v) \), and use \( v_1 := v \) for convenience. Second, add \( r = k - c \) universal vertices \( w_1, \ldots, w_r \) (i.e., neighborhood \( V \cup \{v_2, \ldots, v_c\} \)).

- Let \( M'_v \) denote the gammoid on \( G_v \) with source set \( T = \{v_1, \ldots, v_c, w_1, \ldots, w_r\} \) and ground set \( V' \cup S_0 \). Recall that the independent sets of a gammoid are exactly those subsets \( I \) of the ground set that have \(|I| \) vertex-disjoint paths from the sources to \( I \). It is well known that gammoids are linear matroids and that a representation over a sufficiently large field can be found in randomized polynomial time (cf. Marx \[18\]).

- Create \( M_v \) from \( M'_v \) by contracting \( S_0 \), making its ground set \( V' \). If \( S_0 \) is independent in \( M'_v \) then any \( I \) is independent in \( M_v \) if and only if \( S_0 \cup I \) is independent in \( M'_v \).

Use Marx’s algorithm \[18\] to search for a set \( I^* \) of size \( k - |S_0| \) that is independent in each matroid \( M_v \) for \( v \in D \setminus S_0 \). If a set \( I^* \) is found then test whether \( S_0 \cup I^* \) is a vector connectivity set for \((G, \phi, k)\) by appropriate polynomial-time flow computations. If yes then return the solution \( S_0 \cup I^* \).
Otherwise, if \( S_0 \cup I^* \) is not a solution or if no set \( I^* \) was found then try the next set \( S_0 \), or answer \( \text{NO} \) if no set \( S_0 \) is left to check.

It remains to prove that the algorithm is correct and to (briefly) consider the runtime.

**Correctness.** Clearly, if \((G, \phi, k)\) is \( \text{NO} \) then the algorithm will always answer \( \text{NO} \) as all possible solutions \( S_0 \cup I^* \) are tested for feasibility.

Assume now that \((G, \phi, k)\) is \( \text{YES} \), let \( S^* \) a solution of size \( k \), and let \( S_0 = D \cap S^* \). Note that \( S^* \subseteq V' \cup S_0 \). Pick any \( v \in D \setminus S_0 \). It follows that there are \( c = \phi(v) \) paths from \( v \) to \( S^* \) in \( G \) that are vertex-disjoint except for \( v \). Thus, in \( G_v \) we get \( c \) (fully) vertex-disjoint paths from \( \{v_1, \ldots, v_c\} \) to \( S^* \), by giving each path a private copy of \( v \). We get additional \( r = k - c \) paths from \( \{w_1, \ldots, w_r\} \) to the remaining vertices of \( S^* \) since \( S^* \subseteq V' \cup S_0 \subseteq N(w_i) \). Thus, the set \( S^* \) is independent in each gammoid \( M'_v \). Therefore, in each \( M'_v \) also \( S_0 \subseteq S^* \) is independent. This implies that in \( M_v \) (obtained by contraction of \( S_0 \)) the set \( S^* \setminus S_0 \) is independent and has size \( k - |S_0| \). Moreover, any \( I \) is independent in \( M_v \) if and only if \( I \cup S_0 \) is independent in \( M'_v \). It follows, from the former statement, that Marx’ algorithm will find some set \( I \) of size \( k - |S_0| \) that is independent in all matroids \( M_v \) for \( v \in D \setminus S_0 \).

We claim that \( I \cup S_0 \) is a vector connectivity set for \((G, \phi, k)\). Let \( v \in D \setminus S_0 \). We know that \( I \) is independent in \( M_v \) and, thus, \( S := I \cup S_0 \) is independent in \( M'_v \). Thus, in \( G_v \) there are \( |S| = k \) paths from \( T \) to \( S \). This entails \( c = \phi(v) \) vertex-disjoint paths from \( \{v_1, \ldots, v_c\} \) to \( S \) that each contain no further vertex of \( T \) since \( |T| = k \). By construction of \( G_v \), we directly get \( \phi(v) \) paths from \( v \) to \( S \) in \( G \) that are vertex-disjoint except for overlap in \( v \). Thus, \( S \) satisfies the demand of any \( v \in D \setminus S_0 \). Since \( S \supseteq S_0 \), we see that \( S \) satisfies all demands. Thus, the algorithm returns a feasible solution, as required.

**Runtime.** Marx’ algorithm for finding a set of size \( k' \) that is independent in \( \ell \) matroids has \( \text{FPT} \) running time with respect to \( k' + \ell \). We have \( k' \leq k \) and \( \ell \leq |D| \leq k^3 + k \) in all iterations of the algorithm and there are at most \((k^3 + k)^k \) iterations. This gives a total time that is \( \text{FPT} \) with respect to \( k \), completing the proof sketch.

4 Vertex-linear kernelization for constant demand

In this section we prove a vertex-linear kernelization for \textsc{vector d-connectivity}(\( k \)), i.e., with \( d \) a problem-specific constant and \( k \) the parameter. We recall the problem definition.

**\textsc{vector d-connectivity}(\( k \))**

**Instance:** A graph \( G = (V, E) \), a function \( \phi: V \to \{0, \ldots, d\} \), and an integer \( k \in \mathbb{N} \).

**Parameter:** \( k \).
Question: Is there a set $S$ of at most $k$ vertices such that each vertex $v \in V \setminus S$ has $\phi(v)$ vertex-disjoint paths with endpoints in $S$?

The starting point for our kernelization are Reduction Rules 1 and 2 and a result of Cicalese et al. [7] that relates vector connectivity sets for $(G, \phi)$ to hitting sets for a family of connected subgraphs of $G$: Intuitively, if the neighborhood of $X \subseteq V$ in $G$ is smaller than the largest demand of any $v \in X$, then every solution must select at least one vertex in $X$ to satisfy $v$ (by Menger’s Theorem). We begin by introducing notation for such a set family but additionally restrict it to (inclusionwise) minimal sets $X$ where the demand of some vertex in $X$ exceeds $|N(X)|$. We then state the result of Cicalese et al. [7] using our notation.

**Definition 1 ($\mathcal{X}(G, \phi)$).** Let $G = (V, E)$ and let $\phi : V \rightarrow \mathbb{N}$. The family $\mathcal{X}(G, \phi)$ contains all minimal sets $X \subseteq V$ such that

1. $G[X]$ is connected and
2. there is a vertex $v \in X$ with $\phi(v) > |N(X)|$.

Using this notation, the result of Cicalese et al. [7] is as follows.

**Proposition 2** (adapted from Cicalese et al. [7, Proposition 1]). Let $G = (V, E)$, let $\phi : V \rightarrow \mathbb{N}$, and let $X := \mathcal{X}(G, \phi)$. Then every set $S \subseteq V$ is a vector connectivity set for $(G, \phi)$ if and only if it is a hitting set for $X$, i.e., it has a nonempty intersection with each $X \in \mathcal{X}$.

**Proof.** Cicalese et al. [7] proved Proposition 2 without the minimality restriction, allowing for a larger family, say $\mathcal{X}^+ \supseteq \mathcal{X}$. Clearly, hitting sets for $\mathcal{X}^+$ are also hitting sets for $\mathcal{X}$. Conversely, since $\mathcal{X}^+ \setminus \mathcal{X}$ contains only supersets of sets in $\mathcal{X}$, hitting sets for $\mathcal{X}$ are also hitting sets for $\mathcal{X}^+$. 

Note that for the general case of VECTOR CONNECTIVITY with unrestricted demands the size of $\mathcal{X}(G, \phi)$ can be exponential in $|V(G)|$; for vector $d$-connectivity there is a straightforward bound of $|\mathcal{X}| = \mathcal{O}(|V(G)|^d)$ since $|N(X)| \leq d - 1$. However, even for vector $d$-connectivity, the sets $X \in \mathcal{X}$ are not necessarily small and, thus, we will not take a hitting set approach for the kernelization; in fact, we will not even materialize the set $\mathcal{X}$ but use it only for analysis.

We will leverage Reduction Rules 1 and 2 throughout this section. Hence, we will, sometimes tacitly, assume that all instances of VECTOR CONNECTIVITY are reduced with respect to these rules.

As a first step, we prove that instances $(G, \phi, k)$ of vector $d$-connectivity($k$) that are reduced under Rule 1 have the property that every set $X \in \mathcal{X}(G, \phi)$ contains at most $d^3$ vertices with nonzero demand. For ease of presentation we define $D(G, \phi) := \{v \in V(G) \mid \phi(v) \geq 1\}$, and use the shorthand $D = D(G, \phi)$ whenever $G$ and $\phi$ are clear from context.
Lemma 6. For all $X \in \mathcal{X}$ we have $|X \cap D| \leq (d - 1)d^2 \leq d^3$.

Proof. Recall the definition of $C(v)$ as the unique closest minimum $v, D'(v)$-separator, where $D'(v) = \{ u \in V \setminus \{ v \} \mid \phi(u) \geq \phi(v) \}$, and the definition of $R(v)$ as the connected component of $v$ in $G - C(v)$, from Section 3. Fix some $r \in \{ 1, \ldots, d \}$, define $D_r = \{ v \in D \mid \phi(v) = r \}$, and consider the relation of $R(v)$ with $X$ for $v \in D_r \cap X$. Note that $D_r \setminus \{ v \} \subseteq D'(v)$.

If $R(v) \subseteq X$, then this would contradict minimality of $X$. By reducedness under Rule 1 we have $|C(v)| < \phi(v)$ or else the rule would apply to $v$. But then $R(v)$ with $N(R(v)) = C(v)$ fulfills the conditions for being in $\mathcal{X}$, except possibly for minimality. This would prevent $X \supseteq R(v)$ from being included in $\mathcal{X}$. Else, if $R(v) = X$, then no further vertex of $D_r$ is in $X$, since $C(v)$ is also a $v, D_r \setminus \{ v \}$-separator as $D_r \setminus \{ v \} \subseteq D'(v)$. In this case we get, $|X \cap D_r| = 1$.

In the remaining case, there is no $v \in D_r \cap X$ with $R(v) \subseteq X$. It follows that for all $v \in D_r \cap X$ we have $R(v) \cap X \neq \emptyset$ but $R(v) \not\subseteq X$. This implies $R(v) \cap N(X) \neq \emptyset$ since both $G[X]$ and $G[R(v)]$ are connected. We will use this fact to bound the number of vertices with demand $r$ in $X$.

Let $w \in N(X)$ and let $W \subseteq D_r \cap X$ contain those vertices $v$ of demand $r$ whose set $R(v)$ contains $w$; each $R(v)$ must contain at least one vertex in $N(X)$. Thus, for any two vertices $u, v \in W$ we find that their sets $R(u)$ and $R(v)$ have a nonempty intersection, since they share at least $w$. We can now repeat the same analysis as used in the proof of Lemma 4 to get that $|W| \leq 2r - 1$. Over all choices of $w$ we get an upper bound of $(d - 1)(2r - 1)$ vertices of demand $r$ in $X$.

We showed that for each choice of $r \in \{ 1, \ldots, d \}$ we have at most $(d - 1)(2r - 1)$ vertices of demand $r$ in $X$. Summing over all $r \in \{ 1, \ldots, d \}$ this yields an upper bound of $(d - 1)d^2 \leq d^3$ for $|X \cap D|$, as claimed.

To arrive at our kernelization we will later establish a, somewhat technical, reduction rule that shrinks connected subgraphs with small boundary and bounded number of demand vertices to constant size. This is akin to protrusion-based reduction rules, but there is no treewidth involved. Applying this reduction to the family $\mathcal{X}$, that is, reducing graphs $G[X], X \in \mathcal{X}$, to constant size, would yield polynomial total size but not necessarily a vertex-linear upper bound. To achieve the latter, we introduce a new family $\mathcal{Y}$ with larger but (as it will turn out) fewer sets, and apply the reduction process to graphs $G[Y]$ with $Y \in \mathcal{Y}$ instead.

Definition 2 ($\mathcal{Y}(G, \phi, d)$). Let $G = (V, E)$, let $d \in \mathbb{N}$, and let $\phi: V \rightarrow \{ 0, \ldots, d \}$. The family $\mathcal{Y}(G, \phi, d)$ contains all sets $Y \subseteq V$ with

1. $G[Y]$ is connected,
2. $|Y \cap D| \leq d^3$, i.e., $Y$ contains at most $d^3$ vertices $v$ with nonzero demand $\phi(v)$,
3. \(|N(Y)| \leq d|, i.e., Y has at most d neighbors, and

4. there is a vertex \(v \in Y \cap D\), i.e., \(\phi(v) \geq 1\), such that \(N(Y)\) is the unique closest minimum \(v\), \(D \setminus Y\)-separator.

For \((G, \phi, d)\), we relate \(X = X(G, \phi)\) and \(Y = Y(G, \phi, d)\) by proving that every set \(X \in X\) is contained in at least one \(Y \in Y\). Intuitively, this proves that all “interesting” parts of the instance are contained in subgraphs \(G[Y]\).

**Lemma 7. Let \(G = (V, E)\) a graph, \(d \in \mathbb{N}\), and \(\phi : V \to \{0, \ldots, d\}\). Let \(X := X(G, \phi)\) and \(Y := Y(G, \phi, d)\). Then for all \(X \in X\) there exists \(Y \in Y\) with \(X \subseteq Y\).**

**Proof.** Let \(X \in X\) and pick \(v \in X\) with \(\phi(v) > |N(X)|\). Let \(D_1 = D \setminus X\), i.e., those nonzero demand vertices that are not in \(X\). Now, let \(Z \subseteq V \setminus \{v\}\) the minimum \(v\), \(D_1\)-separator that is closest to \(v\), and let \(Y\) be the connected component of \(v\) in \(G - Z\); thus \(Z = N(Y)\). We claim that \(X \subseteq Y\) and \(Y \in Y\).

First, assume for contradiction that \(X \not\subseteq Y\). We use submodularity of \(f : 2^V \to \mathbb{N} : U \mapsto |N(U)|\), which implies

\[
f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y). \tag{2}
\]

Note that \(D_1 \cap (X \cup Y) = (D_1 \cap X) \cup (D_1 \cap Y) = \emptyset\) and that \(v \in X \cup Y\). It follows that \(N(X \cup Y)\) is also a \(v\), \(D_1\)-separator and, using that \(Z\) is minimum, we get that \(f(X \cup Y) = |N(X \cup Y)| \geq |Z| = |N(Y)| = f(Y)\). Plugging this into (2) we obtain \(f(X) \geq f(X \cap Y)\). Note that \(v \in X \cap Y\) and \(D_1 \cap (X \cap Y) = \emptyset\), implying that \(N(X \cap Y)\) is also a \(v\), \(D_1\)-separator of size at most \(|N(X)| = f(X)\). From \(X \not\subseteq Y\) we get \(X \cap Y \subseteq X\). But then \(X \cap Y\) is a proper subset of \(X\) containing \(v\) and having \(|N(X \cap Y)| \leq |N(X)| < \phi(v)\). It follows that the connected component of \(v\) in \(G[X \cap Y]\) also has neighborhood size (in \(G\)) less than \(\phi(v)\). This contradicts the fact that \(X\) is a minimal set fulfilling the properties of Definition [1] which is required for \(X \in X\). We conclude that, indeed, \(X \subseteq Y\).

Second, let us check that \(Y\) fulfills the requirements for \(Y \in Y = Y(G, \phi, d)\) (as in Definition [2]). Note that \(Z\) separates \(v\) from all nonzero demand vertices that are not in \(X\), since \(D_1 = D \setminus X\). Thus, every nonzero demand vertex in \(Y\) is also contained in \(X\), i.e., \(D \cap Y \subseteq D \cap X\), which bounds their number by \(d^3\) using Lemma [6]. It also follows that \(D \setminus X = D \setminus Y\), since \(X \subseteq Y\) implies \(D \cap X \subseteq D \cap Y\). Furthermore, \(G[Y]\) is connected and \(N(Y)\) is a minimum \(v\), \(D \setminus Y\)-separator that is closest to \(v\); note that \(D_1 = D \setminus X = D \setminus Y\). Finally, since \(N(X)\) is also a \(v\), \(D_1\)-separator and \(N(Y)\) is minimum, we conclude that \(|N(Y)| \leq |N(X)| < \phi(v) \leq d\). Thus, indeed, \(Y \in Y\) as claimed.

We prove that the number of sets \(Y \in Y\) is linear in \(k\) for every fixed \(d\). Thus, by later shrinking the size of sets in \(Y\) to some constant we get \(O(k)\) vertices in total over sets \(Y \in Y\).
Lemma 8. Let \((G, \phi, k)\) an instance of vector \(d\)-connectivity \((k)\) with \(\phi: V(G) \to \{0, \ldots, d\}\) and let \(\mathcal{Y} := \mathcal{Y}(G, \phi, d)\). Then \(|\mathcal{Y}| \leq d^2 k \cdot 2^{d^3 + d}\).

Proof. We prove the lemma by giving a branching process that enumerates all sets \(Y \in \mathcal{Y}\) within the leaves of a branching tree and by showing that the tree has at most \(d^2 k \cdot 2^{d^3 + d}\) leaves in which sets \(Y\) are found. Given \(G = (V, E), \phi: V \to \{0, \ldots, d\},\) and \(k \in \mathbb{N},\) the process works as follows. (Recall \(D = \{v \in V \mid \phi(v) \geq 1\}.\))

1. As a first step, branch on choosing one vertex \(v \in D\). Recall that an instance reduced with respect to Rule 2 has \(|D| \leq d^2 k\). Hence, this branching step incurs a factor of \(|D| \leq d^2 k\) to the number of leaves and creates one node for each choice.

2. Maintain disjoint sets \(D_0, D_1 \subseteq D\), starting with \(D_0 := \{v\}\) and \(D_1 := \emptyset\), and the minimum \(v, D_1\)-separator \(Z\) that is closest to \(v\); initially \(Z = \emptyset\). Throughout, use \(Y\) to refer to the connected component of \(v\) in \(G - Z\).

3. All further branchings work as follows: Pick an arbitrary vertex \(p \in D \setminus (D_0 \cup D_1)\) that is reachable from \(v\) in \(G - Z\). Branch on either adding this vertex to \(D_0\) or to \(D_1\), creating a child node for each choice. In the branch where \(p\) is added to \(D_1\) update the minimum \(v, D_1\)-separator \(Z\) closest to \(v\) and update the connected component \(Y\) of \(v\) in \(G - Z\).

4. Terminate a branch if any of the following three termination conditions occurs.

(a) The size of \(D_0\) exceeds \(d^3\).
(b) The size of \(Z\) exceeds \(d\).
(c) No vertex of \(D \setminus (D_0 \cup D_1)\) is reachable from \(v\) in \(G - Z\).

We will now analyze this process. First, we show that every set of \(Y\) occurs as the set \(Y\) in some leaf node of the process, i.e., a node to which a termination condition applies. Second, we show that the number of such leaf nodes is bounded by \(|D| \cdot 2^{d^3 + d} \leq d^2 k \cdot 2^{d^3 + d}\).

Each set of \(Y\) occurs in some leaf. We show that every set \(Y^* \in \mathcal{Y}\) is found as the set \(Y\) of at least one leaf of the branching tree. To this end, fix an arbitrary set \(Y^* \in \mathcal{Y}\) and let \(Z^* := N_G(Y^*)\). Furthermore, let \(D_0^* := Y^* \cap D\). Let \(v \in Y^* \cap D\) such that \(Z^* = N(Y^*)\) is the unique minimum, closest \(v, D \setminus Y^*\)-separator. In the first branching step the process can clearly pick \(v\) for its choice of vertex in \(D\); in this case it continues with \(D_0 = \{v\}, D_1 = \emptyset, Z = \emptyset,\) and \(Y\) is the connected component of \(v\) in \(G\). Consider nodes in the branching process with current sets \(Y, D_0, D_1,\) and \(Z = N(Y)\) fulfilling the requirements that
that $D_Y$ violating the definition of $|D_Y|$ and the sets therein. Similarly, we cannot have cut size for $v, D_Z$ is the minimum closest vertex of $D_Y$ to $v$. It follows directly that $Z = \{v\}$, $D_0 = \emptyset$, $Z = \emptyset$, and $Y$ equal to the component of $v$ in $G$ fulfills the requirements, so we can indeed always find such a node by starting at this one and following child nodes fulfilling the requirements until reaching a leaf or until both child nodes do not fulfill the requirements.

Leaf node fulfilling the requirements. If the chosen node is a leaf then one of the three termination conditions must hold. Clearly, we cannot have $|D_0| > d^3$ since that would imply $|D_0^*| > d^3$, violating the definition of $Y$ and the sets therein. Similarly, we cannot have $|Z| > d$: In this regard, note that $D_1 \cap D_0^* = \emptyset$ implies that $D_1 \subseteq D \setminus D_0^* = D \setminus (D \cap Y^*) = D \setminus Y^*$. It follows directly that $Z^*$, which separates $v$ from $D \setminus Y^*$, also separates $v$ from $D_1$. But then the size of $Z^*$ is an upper bound for the minimum cut size for $v, D_1$-separators and $|Z| > d$ would imply $|Z^*| \geq |Z| > d$, again violating the definition of $Y$.

Thus, in case of a leaf node the only remaining option is that no further vertex of $D \setminus (D_0 \cup D_1)$ is reachable from $v$ in $G - Z = G - N(Y)$. Recall that $Z$ is the minimum closest $v, D_1$-separator. By termination $Z$ also separates $v$ from $D \setminus (D_0 \cup D_1)$, making it a closest $v, D \setminus D_0$-separator. Since $D_0 \subseteq D_0^*$, it follows that $D \setminus Y^* = D \setminus D_0^* \subseteq D \setminus D_0$. Thus, the size of the minimum $v, D \setminus Y^*$-separator $Z^*$ is upper-bounded by $|Z|$ since $Z$ is also a $v, D \setminus Y^*$-separator. Recall that we have $Y \supseteq Y^*$, which implies that $Z^* = N(Y^*)$ also separates $v$ from $Z = N(Y)$. Now, because $Z$ is closest to $v$, it is the unique $v, Z$-separator of size at most $|Z|$, which implies $Z = Z^*$ since we just derived that $|Z^*| \leq |Z|$. Thus, $Y = Y^*$ since both are identified as the connected component of $v$ in $G - Z = G - Z^*$. We conclude that $Y^*$ is equal to $Y$ in the chosen node if it is a leaf.

Internal node fulfilling the requirements. Now, let us consider the case that the chosen node is not a leaf of the branching tree. We want to check that at least one possible branch must lead us to a child node that also fulfills our restrictions; this would contradict our choice of node that is either a leaf or such that neither child node fulfills the requirements, implying that we necessarily pick a leaf node. Thus, for any $Y^* \in \mathcal{Y}$ there is a leaf of the branching tree with $Y = Y^*$. For clarity, in the following discussion we will use $Y, D_0, etc.$ for the current node and $Y', D_0', etc.$ for the considered child node in the branching tree.

Since we are not in a leaf in this case, the process chooses an arbitrary vertex $p \in D \setminus (D_0 \cup D_1)$ that is reachable from $v$ in $G - Z$ to branch on. If $p \in D_0^*$ then the child node corresponding to adding $p$ to $D_0$ has $D_0' = D_0 \cup \{p\} \subseteq D_0^*$ and $D_1' = D_1$. Thus, $Z' = Z$ and, hence, $Y' = Y \supseteq Y^*$. Thus, the child node fulfills all requirements; a contradiction.
Otherwise, if \( p \notin D_0^* \), then \( p \in D \setminus D_0^* = D \setminus Y \). Thus, the child node\( \) corresponding to adding \( p \) to \( D_1 \) has \( D_0' = D_0 \subseteq D_0^* \) and \( D_1' = D_1 \cup \{ p \} \), implying that \( D_1' \cap D_0' = D_1 \cap D_0^* = \emptyset \). For the desired contradiction it remains to prove that \( Y' \supseteq Y^* \) since we assumed that neither child fulfills the requirements.

Assume that \( Y^* \notin Y' \). Thus, \( Y^* \cap Y' \subseteq Y^* \). We again use submodularity of the function \( f : 2^V \to \mathbb{N} : U \mapsto |N(U)| \) and get

\[
f(Y^*) + f(Y') \geq f(Y^* \cap Y') + f(Y^* \cup Y').
\] (3)

Since \( v \in Y^* \cap Y' \subseteq Y^* \) and \( N(Y^*) \) is closest to \( v \) it follows that \( f(Y^* \cap Y') = |N(Y^* \cap Y')| > |N(Y^*)| = f(Y^*) \), by Proposition I. Plugging this into (3) yields \( f(Y^* \cup Y') < f(Y') \); let us check that this violates the fact that \( Z' = N(Y') \) is a minimum \( v, D_1 \)-separator: We have \( v \in Y^* \cup Y' \) and

\[
D_1' \cap (Y^* \cup Y') = (D_1' \cap Y^*) \cup (D_1' \cap Y') = (D_1' \cap Y^*) \cap D = D_1' \cap D_0^* = \emptyset.
\]

Thus, indeed, we find that \( N(Y^* \cup Y') \) is a \( v, D_1 \)-separator and we know from \( f(Y^* \cup Y') < f(Y') \) that it is smaller than the assumed minimum \( v, D_1 \)-separator \( Z' = N(Y') \); a contradiction. Hence, by our choice of node that fulfills the requirements and is a leaf or neither child fulfills the requirements, we must obtain a leaf with set \( Y \) equal to \( Y^* \).

**Number of leaves containing some \( Y^* \in \mathcal{Y} \).** We have seen that every set \( Y^* \in \mathcal{Y} \) is equal to the set \( Y \) of some leaf node fulfilling certain requirements. Furthermore, leaves with \( |D_0| > d^3 \) or \( |Z| > d \) were shown not to correspond to any \( Y \in \mathcal{Y} \). We will now analyze the number of leaf nodes with \( |D_0| \leq d^3 \) and \( |Z| \leq d \).

Crucially, we show that each branching increases \( |D_0| + |Z| \). For adding \( p \) to \( D_0 \) this is obvious, for adding \( p \) to \( D_1 \) we prove this next. Concretely, we prove that adding \( p \) to \( D_1 \) increases the minimum size of \( v, D_1 \)-separators by at least one. (This is essentially folklore but we provide a proof for completeness.) Use \( D_1' = D_1 \cup \{ p \} \) and let \( Z \) and \( Z' \) denote minimum \( v, D_1 \)- and \( v, D_1' \)-separators closest to \( v \); use \( Y \) and \( Y' \) for the components of \( v \) in \( G - Z \) and \( G - Z' \), respectively. We use again the submodular function \( f : 2^V \to \mathbb{N} : U \mapsto |N_G(U)| \) and obtain

\[
f(Y) + f(Y') \geq f(Y \cup Y') + f(Y \cap Y').
\] (4)

Both \( Z \) and \( Z' \) separate \( v \) from \( D_1 \). Thus, \( D_1 \cap (Y \cup Y') = \emptyset \) and \( N(Y \cup Y') \) is also a \( v, D_1 \)-separator, since it creates the component \( Y \cup Y' \) for \( v \) in \( G - N(Y \cup Y') \). Since \( Z \) is a minimum \( v, D_1 \)-separator we must have \( f(Y \cup Y') = |N(Y \cup Y')| \geq |Z| = f(Y) \). Plugging this into (4) yields \( f(Y \cap Y') \leq f(Y') \). If \( f(Y) = |Z| \leq |Z| = f(Y) \), i.e., if adding \( p \) to \( D_1 \) does
not increase the minimum size of $v, D_1$-separators then $f(Y \cap Y') \leq |Z|$ and $N(Y \cap Y')$ is also a $v, D_1$-separator of size at most $|Z|$. However, as $p \notin Y'$, since $Z'$ separates $v$ from $D'_1 = D_1 \cup \{p\}$, the set $Y \cap Y'$ is a strict subset of $Y$; this is a contradiction to $Z$ being closest to $v$. As a consequence, the size of closest, minimum $v, D_1$-separators increases whenever we branch into adding a so far reachable vertex to $D_1$.

Thus, every child node has $|D'_0| + |Z'| \geq |D_0| + |Z| + 1$ and, clearly, neither value decreases when branching. Thus, the process can only reach leaf nodes with $|D_0| \leq d^3$ and $|Z| \leq d$ via internal nodes where $|D_0| + |Z| \leq d^3 + d$. It follows that the number of such leaf nodes is upper bounded by $|D| \cdot 2^{d^3 + d} \leq d^3 k \cdot 2^{d^3 + d}$. (Once $|D_0| \geq d^3$ or $|Z| \geq d$ at most one child node can lead to such a leaf, since the other child violates the restriction on $|D_0|$ or $|Z|$; these branches are not counted.) Thus, $|Y| \leq d^3 k \cdot 2^{d^3 + d}$, as claimed.

Reducing the size of sets in $\mathcal{Y}$. In this part, we explain and prove how to reduce the size of sets $Y \in \mathcal{Y}$ through modifications on the graph $G$. At a high level, this will be achieved by replacing subgraphs $G[Y]$ by “equivalent” subgraphs of bounded size. When this is done, we know that the total number of vertices in sets $Y \in \mathcal{Y}$ is $O(k)$. Since this part is somewhat technical and long, let us try to illustrate it first.

Consider a set $Y \in \mathcal{Y}$ and its (small) neighborhood $Z := N_G(Y)$. Think of deciding whether $(G, \phi, k)$ is YES as a game between two players, Alice and Bob. Alice sees $G[Y \cup Z]$ and wants to satisfy the demands of all vertices in $Y$, and Bob sees $G - Y$ and wants to satisfy the demands of the vertices in $V \setminus Y$. To achieve a small solution the players must cooperate and exchange information about paths between vertices in $Z$, or between $Z$ and vertices of a partial solution, that they can provide or that they require.

Since our goal is to simplify $G[Y]$, all notation is given using Alice’s perspective. Crucially, we know that there are only constantly many nonzero demand vertices in $Y$, which can be seen to imply that the intersection of optimal solutions with $Y$ is bounded (Lemma [10]). Thus, Alice can try all partial solutions $S_Y \subseteq Y$ of bounded size and determine what facilities each $S_Y$ provides for Bob, and what requirements she has on Bob to satisfy her demand vertices using $S_Y$ and further paths through $G - Y$.

Let us be slightly more concrete about facilities and requirements, before making them fully formal. If we fix some partial solution $S_Y \subseteq Y$ then Alice can offer (as facilities) to Bob to connect some subsets of $Z$ to $S_Y$ by disjoint paths in $G[Y \cup Z]$, and to, additionally, provide paths connecting certain pairs of vertices in $Z$. There can be a large number of such options for each $S_Y$. Similarly, to fulfill demands in $Y$, Alice may need (as requirements)

---

2These values coincide with $f(Y)$ and $f(Y')$ since we chose $Z = N(Y)$ and $Z' = N(Y')$ as minimum $v, D_1$- and $v, D'_1$-separators.
that Bob can provide paths from certain subsets of \(Z\) to a solution and, additionally, paths connecting pairs of vertices in \(Z\). (Note that there is some asymmetry here since Bob’s part is too large to fully analyze, but this will be no problem.) Fortunately, while there may be many choices for \(S_Y\), and many facilities and requirements relative to a single \(S_Y\), it will turn out that the overall number of things to keep track of is bounded (in \(d\)); this will be called the \textit{signature} of \(G[Y \cup Z]\). Ultimately, we will be able to replace \(G[Y \cup Z]\) by a bounded-size graph \(G[Y' \cup Z]\) with the same signature.

Let us now make this approach formal. In the following, let \(G\) and \(\phi\) represent the graph and demand function of an instance of \textsc{vector} \(d\)-\textsc{connectivity}(\(k\)). To decrease the necessary notation we also fix a set \(Y \subseteq V(G)\) such that \(|Y \cap D| \leq d^3\) and \(|N(Y)| \leq d\). (In the kernelization procedure, the role of \(Y\) will be assumed by some set in \(Y\).) We denote the neighborhood \(N(Y)\) by \(Z\). The edges in \(G[Z]\) will be administered by Bob. Hence, Alice will play on \(G[Y \cup Z] - E(G[Z])\); this is a somewhat arbitrary choice. As a shorthand, we use \(G_Z\) for \(G - E(G[Z])\).

We will first take care of the requirements that Alice has. The facilities will be treated later. As mentioned above, a requirement comprises several sets of paths outside of \(G[Y]\) one set of which needs to be provided by Bob (and his part of the solution) in order to satisfy the demand of a vertex \(v \in D \cap Y\). To this end, we define a \textit{satisfying connector}. In the following, all paths contain at least one edge.

**Definition 3** (Satisfying connector). Let \(H\) be a graph on vertex set \(Y \uplus Z\) such that \(Z\) is independent in \(H\), let \(v \in Y\), let \(d_v \neq 0\) be a positive integer, and let \(S_Y \subseteq Y\) be a partial solution.\(^3\) We define a \textit{satisfying connector} of \(v\) with respect to \(S_Y\) as a tuple \((T, F)\) consisting of a vertex set \(T\) and an edge set \(F \subseteq \left(\binom{Z}{2}\right)\). Tuple \((T, F)\) is a satisfying connector for \(v\) if in \(H\), after adding the edges \(F\), there are \(d_v\) vertex-disjoint paths from \(v\) to \(S_Y \cup T\). The set of all satisfying connectors of \(v\) with demand \(d_v\) is denoted by \(Sat(H, Z, d_v, S_Y, v)\).

Now we can formally define the requirement of Alice. Intuitively, Bob should provide paths that “hit” all the sets of satisfying connectors.

**Definition 4** (Requirement). Let \(H\) be a graph on vertex set \(Y \uplus Z\) such that \(Z\) is independent in \(H\), let \(\phi\) be a demand function on \(Y\), and let \(S_Y \subseteq Y\) be a partial solution. The \textit{requirement} \(Req(H, Z, \phi, S_Y)\) is the collection \(\{Sat(H, Z, \phi(v), S_Y, v) \mid v \in D(H, \phi) \cap Y\}\).

Let us show that, the demand of each vertex in \(D(\phi) \cap Y\) can be met if and only if the requirement of Alice is met by suitable paths provided by Bob.

**Lemma 9.** Let \(v \in D \cap Y\), \(S \subseteq V(G)\), and \(S_Y = S \cap Y\). There are \(\phi(v)\) vertex-disjoint paths from \(v\) to \(S\) in \(G\) if and only if \(^3\)Here, \(Y \uplus Z\) denotes a disjoint union.
1. there is a satisfying connector \((T, F) \in \text{Sat}(G_Z[Y \cup Z], Z, \phi(v), S_Y, v)\) such that

2. there are \(|T \setminus S| + |F|\) paths in \(G - Y\) with no internal vertices from \(Z\) that can be partitioned into a set \(P_1\) of \(|T \setminus S|\) pairwise vertex-disjoint paths that run between \(T \setminus S\) and \(S \setminus (Z \cup Y)\), and a set \(P_2\) of \(|F|\) paths that yield \(F\) when, for each of them, contracting all but one of its edges. Furthermore, the inner vertices of each path in \(P_2\) need to be disjoint to the inner vertices of every other path in \(P_1 \cup P_2\).

Proof. “\(\Leftarrow\)”: Assume that \((T, F) \in \text{Sat}(G_Z[Y \cup Z], Z, \phi(v), S_Y, v)\), such that sets \(P_1\) and \(P_2\) of paths exist as described above. By the definition of satisfying connector, there is a set \(P\) of \(\phi(v)\) paths from \(v\) to \(S_Y \cup T\) in \(G_Z[Y \cup Z] + F\). Append to each path in \(P\) that ends in \(T \setminus S\) a path in \(P_1\) from the corresponding vertex in \(T \setminus S\) to a vertex in \(S \setminus (Z \cup Y)\). Since the paths in \(P_1\) are contained in \(G - (Y \cup (Z \setminus T))\) and they are pairwise vertex-disjoint, the resulting set \(P\) of walks is indeed a set of \(\phi(v)\) vertex-disjoint paths in \(G + F\). Furthermore, every path in \(P\) now has one endpoint in \(v\) and one in \(S\). (Note that some paths may end in \(T \cap S\).) Next, replace each edge \(e \in F\) used in a path in \(P\) by the corresponding path in \(P_2\) that yields \(e\) when contracting all but one of its edges. Since the inner vertices of all paths in \(P_2\) do intersect neither \(Y\) nor the inner vertices of any other path in \(P_1 \cup P_2\), the paths in \(P\) are pairwise vertex-disjoint. Hence, after the replacement we obtain \(\phi(v)\) pairwise vertex-disjoint paths between \(v\) and \(S\) in \(G\).

“\(\Rightarrow\)”: Let \(P\) be a set of \(\phi(v)\) paths from \(v\) to \(S\) of shortest possible total length. Note that no path in \(P\) contains a vertex \(u \in S\) as an inner vertex since otherwise we may shortcut this path so that \(u\) is an endpoint. We construct a satisfying connector \((T, F)\) for \(v\). Consider the paths in \(P\) and their maximal subpaths in \(G - Y\) that do not contain any vertex from \(Z\) as an inner vertex. There are two possible types of such subpaths (containing at least one edge): a set \(P_2\) of paths that start and end in \(Z\), and a set \(P_1\) of paths that start in \(Z\) and end in \(S \setminus (Z \cup Y)\). Let us define \(T\) as the union of \(S \setminus Z\) and the set \(T'\) of endpoints of paths in \(P_1\) that are also in \(Z\), and let us define \(F\) as the set of edges in \((\frac{Z}{2})\) obtained by contracting all but one edge in each path in \(P_2\). We claim that \((T, F) \in \text{Sat}(G_Z[Y \cup Z], Z, \phi(v), S_Y, v)\) and \(P_1\) and \(P_2\) are the sets of paths required in statement 2 of the lemma.

Let us first prove the second part of the claim. Clearly, the inner vertices of the paths in \(P_1 \cup P_2\) are disjoint to \(Z\). Note that \(T'\) does not contain any vertex of \(S\); otherwise there was a path in \(P_1\) that contains two vertices of \(S\) which is a contradiction to our choice of \(P\). Hence, there are \(|T'| = |T \setminus S|\) paths in \(P_1\). The set \(P_2\) of paths contracts to \(F\) per definition and hence, \(|F| = |P_2|\). It remains to show the disjointness of their vertex (sub)sets. First, since the paths in \(P\) are pairwise vertex-disjoint, the inner vertices of paths in \(P_1 \cup P_2\) are also pairwise disjoint. Furthermore, no two paths
in \( \mathcal{P}_1 \) share an endpoint in \( Z \) since otherwise, \( \mathcal{P} \) contains a path with two endpoints in \( S \) which is absurd.

It remains to show that \((T, F) \in \text{Sat}(G_Z[Y \cup Z], Z, \phi(v), S_Y, v)\). We claim that modifying \( \mathcal{P} \) as follows yields a set of paths that witness that \((T, F)\) is the desired satisfying connector. We first contract all edges that occur in paths in \( \mathcal{P}_1 \) and then, for each path in \( \mathcal{P}_2 \), we contract all but one edge. Note that, as \( v \in Y \) and all contracted edges are not contained in \( Y \cup Z \), we obtain \( \phi(v) \) vertex-disjoint paths. Moreover, these paths are contained in \( G_Z[Y \cup Z] + F \). Finally, each path either still has an endpoint in \( S \cap Y = S_Y \), it still has an endpoint in \( S \cap Z \), or it now ends at an endpoint of a path in \( \mathcal{P}_1 \); hence, the paths run from \( v \) to \( S_Y \cup (S \cap Z) \cup T' = S_Y \cup T \), as required.

Note that, in order to be able to replace \( G[Y] \) by a different graph, we need to know which requirements it imposes for any relevant choice of the partial solution \( S_Y \). Since we are aiming for a polynomial-time kernelization, we also need to be able to compute them in polynomial time. For this, we first bound the size of \( S_Y \).

**Lemma 10.** For each vector connectivity set \( S \) of \((G, \phi)\), there is a vector connectivity set \( S' \) such that \(|S'| \leq |S|\) and such that \(|Y \cap S'| \leq d^3 + d\).

**Proof.** Assume that \(|Y \cap S| \geq d^3 + d\). Remove all vertices in \((Y \cup Z) \cap S\) from \( S \) and add all vertices in \( D \cap Y \) as well as all vertices in \( Z \). Denote the resulting vertex set by \( S' \). Recall that \(|D \cap Y| \leq d^3\) by definition of \( Y \). Hence, \(|Y \cap S'| \leq |D \cap Y| + |N_G(Y)| \leq d^3 + d\) and, thus, \(|S'| \leq |S|\). Clearly, \( S' \) satisfies all demands of vertices in \( Y \) and \( Z \) since \((D \cap Y) \cup Z \subseteq S'\). For vertices \( v \in D \setminus (Y \cup Z) \), note that at most \(|Z|\) paths used for reaching \( S \) from \( v \) can traverse \( Z \), and all of those can be shortened to end in \( Z \subseteq S' \); all other paths avoid \( Z \) and thereby \( Y \cup Z \), implying that they still end in vertices of \( S \setminus (Y \cup Z) = S' \setminus (Y \cup Z) \).

We are ready to compute the requirements.

**Lemma 11.** The set \( \{\text{Req}(G_Z[Y \cup Z], Z, \phi|_Y, S_Y) \mid S_Y \subseteq Y \land |S_Y| \leq d^3 + d\} \) is computable in polynomial time. Here, \( \phi|_Y \) denotes \( \phi \) restricted to \( Y \).

**Proof.** By enumerating all \( S_Y \subseteq Y \) of size at most \( d^3 + d \) in \( n^{O(d^3)} \) time, i.e. polynomial time, the task reduces to computing \( \text{Req}(G_Z[Y \cup Z], Z, \phi|_Y, S_Y) \) for a given \( S_Y \). To do this, we compute the set of satisfying connectors for each vertex \( v \) in \( D \cap Y \). This in turn we do by simply iterating over all tuples \((T, F)\) with \( T \subseteq Z \) and \( F \subseteq \binom{Z}{2} \) and checking whether it is contained in \( \text{Sat}(G_Z[Y \cup Z], Z, \phi(v), S_Y, v) \). Note that \(|Z| \leq d\), hence, there are at most \( 2^{O(d^3)} \in O(1) \) different tuples to check. Thus, it remains to check whether in \( G_Z[Y \cup Z] + F \) there are \( \phi(v) \) vertex-disjoint paths from \( v \) to \( S_Y \cup T \). This can be done in polynomial time by computing a maximum flow.
So far we have only talked about the requirements that Alice has on Bob. Now let us come to the facilities that Alice provides. As mentioned, a facility models the sets of paths inside of $G[Y]$ that Alice, using her part of the solution, provides to Bob so to satisfy the demand of each vertex in $D \setminus (Y \cup Z)$.

**Definition 5 (Provided connector).** Let $H$ be a graph on vertex set $Y \cup Z$ such that $Z$ is independent in $H$, and let $S_Y \subseteq Y$ be a partial solution. A provided connector of $S_Y$ in $H$ is a tuple $(T, F)$ consisting of a vertex set $T \subseteq Z$, and an edge set $F \subseteq \binom{Z}{2}$. Tuple $(T, F)$ is a provided connector if there are $|T| + |F|$ paths in $H$ that contain no vertices of $Z$ as inner vertices and that can be partitioned into $|T|$ paths $P_1$ from $T$ to $S_Y$ that are pairwise vertex-disjoint and $|F|$ paths $P_2$ such that, for each of these, contracting all but one of its edges yields exactly $F$. Furthermore, the inner vertices of each path in $P_2$ need to be disjoint to the inner vertices of any other path in $P_1 \cup P_2$.

We now prove that all demands of vertices in $D \setminus (Y \cup Z)$ can be met if and only if Alice provides suitable paths to Bob. The proof is similar to Lemma 9 where the roles of Alice and Bob are reversed. However, note that Bob has the edges of $G[Z]$ at his disposal and also knows the vertices in $S \cap Z$, which imposes subtle differences.

**Lemma 12.** Let $v \in D \setminus (Y \cup Z)$, and let $S \subseteq V(G)$ with $S \cap Y = S_Y$. There are $\phi(v)$ vertex-disjoint paths from $v$ to $S$ in $G$ if and only if

1. there is a connector $(T, F)$ provided by $S_Y$ in $G_Z[Y \cup Z]$ such that
2. there are $\phi(v)$ vertex-disjoint paths in $(G - Y) + F$ from $v$ to $T \cup (S \setminus S_Y)$.

**Proof.** “$\Leftarrow$”: Let $P_1$ and $P_2$ denote the sets of paths guaranteed by the provided connector and let $P'$ denote the set of paths guaranteed by statement 2 of the lemma. Then append to each path in $P'$ that ends in $T$ a path in $P_1$ from the corresponding vertex in $T$ to $S_Y$. Since the paths in $P_1$ do not contain vertices from $Z$ as inner vertices and are pairwise vertex-disjoint, the resulting set of walks $P'$ is indeed a set of pairwise vertex-disjoint paths in $G + F$. Next, replace all edges $e \in F$ used in a path of $P'$ by the path in $P_2$ that contracts to $e$. Since the inner vertices of paths in $P_2$ are disjoint to those of paths in $P_1$ and paths in $P_2$ do not contain any vertices from $Z$ as inner vertices, the replacement results in a set of $\phi(v)$ vertex-disjoint paths from $v$ to $S$ in $G$.

“$\Rightarrow$”: Assume that there is a set $P$ of $\phi(v)$ vertex-disjoint paths from $v$ to $S$ in $G$. Consider the set $P'$ of maximal subpaths of $P$ in $G_Z[Y \cup Z]$ that do not contain any vertex from $Z$ as an inner vertex. There are two possible types of paths in $P'$: a set $P'_2$ of paths that start and end in $Z$ and a set $P'_1$ of paths that start in $Z$ and end in $S_Y \subseteq Y$. We claim that these two
sets of paths witness that \((T, F)\) is a provided connector, where \(T\) is the set of endpoints in \(Z\) of the paths in \(P'_{1}\) and \(F\) is the set of edges obtained by contracting all but one edge in each path in \(P'_{2}\). Clearly, the paths in \(P'_{1} \cup P'_{2}\) are in \(G_{Z}[Y \cup Z]\) and they do not contain vertices from \(Z\) as inner vertices. It remains to show the disjointness of their vertex (sub)sets. First, note that the paths in \(P'_{1}\) are pairwise vertex-disjoint, since otherwise, there was a path from a vertex in \(S_{Y}\) to another vertex in \(S_{Y}\) in \(P'\) which is absurd. Furthermore, the inner vertices of any path in \(P'_{2}\) do not intersect with the inner vertices of any other path in \(P'_{1} \cup P'_{2}\) because that would mean that the paths in \(P'\) are not pairwise vertex-disjoint. (The endpoints of paths in \(P'_{2}\) may overlap with the endpoints of other paths, however.) This proves that \((T, F)\) is a provided connector. Now, for each path in \(P'_{2}\) contract all but one edge in the corresponding path in \(P\) and remove the vertices of the paths in \(P'_{1}\), except the vertices in \(T\). This yields the set of paths claimed to exist in statement 2.

We take special care of vertices in \(D \cap Z\), as they may need multiple paths into \(G[Y]\).

**Definition 6 (Provided special connector).** Let \(H\) be a graph on vertex set \(Y \uplus Z\) such that \(Z\) is independent in \(H\), and let \(S_{Y} \subseteq Y\) be a partial solution. A provided special connector of \(S_{Y}\) in \(H\) is a tuple \((z, i, T, F)\) of the form \(z \in Z, i \in \{0, 1, \ldots, d\}, T \subseteq Z \setminus \{z\}\), and \(F \subseteq \binom{Z}{2}\). A tuple \((z, i, T, F)\) is a provided special connector if there are \(i + |T| + |F|\) paths in \(H\) that can be partitioned into \(i\) paths \(P_{1}\) from \(z\) to \(S_{Y}\), \(|T|\) paths \(P_{2}\) from \(T\) to \(S_{Y}\), and \(|F|\) paths \(P_{3}\) such that, for each of these, contracting all but one of its edges yields exactly \(F\). Furthermore, none of these paths may contain vertices from \(Z\) as inner vertices, the paths in \(P_{1} \cup P_{2}\) need to be pairwise vertex-disjoint except for \(z\), and the inner vertices of each path in \(P_{3}\) may not intersect with the inner vertices of any path in \(P_{1} \cup P_{2}\).

**Lemma 13.** Let \(z \in D \cap Z\), and let \(S \subseteq V(G)\) with \(S \cap Y = S_{Y}\). There are \(\phi(z)\) vertex-disjoint paths from \(z\) to \(S\) in \(G\) if and only if

1. there is a special connector \((z, i, T, F)\) provided by \(S_{Y}\) in \(G_{Z}[Y \cup Z]\) such that
2. there are \(\phi(z) - i\) vertex-disjoint paths in \(G - Y + F\) from \(z\) to \(T \cup (S \setminus S_{Y})\).

The proof is analogous to the proof of Lemma 12 and we omit it. We are now ready to give a formal definition of the facilities provided by Alice.

**Definition 7 (Facility).** Let \(H\) be a graph on vertex set \(Y \uplus Z\) such that \(Z\) is independent in \(H\), and let \(S_{Y} \subseteq Y\) be a partial solution. The facility \(\text{Fac}(H, Z, S_{Y})\) of \(S_{Y}\) in \(H\) is the set of all provided connectors and provided special connectors of \(S_{Y}\).
Similarly to requirements, we need an efficient algorithm for computing the facilities.

**Lemma 14.** The set \( \{ \text{Fac}(G_Z[Y \cup Z], Z, S_Y) \mid S_Y \subseteq Y \land |S_Y| \leq d^3 + d \} \) is computable in polynomial time.

**Proof.** We first enumerate all \( S_Y \subseteq Y \) with \( |S_Y| \leq d^3 + d \) in \( n^{O(d^3)} \) time and, for each such \( S_Y \), we compute all provided connectors and provided special connectors. This is done by iterating over all possible tuples \( (T, F) \) and \( (z, i, T, F) \) (there are at most \( 2d^3 \cdot 2^d \cdot 2^d \in O(1) \) of them) and checking whether they indeed are provided (special) connectors. This is done mainly by using Robertson and Seymour’s \( O(|V(G)|^3) \)-time algorithm for deciding whether for a fixed number of given pairs in \( (V(G))^2 \) there are pairwise vertex-disjoint paths joining the pairs \([23]\). Let us focus on provided special connectors as the case of ordinary provided connectors is simpler.

To decide whether a given tuple \( (z, i, T, F) \) is a provided special connector, we introduce \( i - 1 \) copies \( z_1, \ldots, z_{i-1} \) of \( z \) into \( G_Z[Y \cup Z] \) (adjacent to all neighbors of \( z \)). Furthermore, for each edge \( e = \{p, q\} \in F \subseteq (Z^2) \) we add “private” copies \( p_e, q_e \) of \( p, q \) and replace \( e \) in \( F \) by \( \{p_e, q_e\} \). We delete all vertices of \( Z \) that are not in \( F \cup \{z, z_1, \ldots, z_{i-1}\} \) from the graph, in order to avoid their appearance as internal vertices of paths. Then, for all size- \( (i + |T|) \) matchings \( M \) of \( S_Y \) on the one hand, and \( z, z_1, \ldots, z_{i-1} \) and \( T \) on the other hand, we test whether the pairs \( M \cup F \) have pairwise vertex-disjoint paths joining them using Robertson and Seymour’s algorithm. We have \( |M \cup F| \leq 2d + d^2 \in O(1) \), and there are at most \( (d^3 + d)^{2d} \in O(1) \) matchings \( M \). Thus, the overall procedure runs in polynomial time. \( \square \)

Now we can precisely define the signature of \( G[Y \cup Z] \) that we mentioned earlier.

**Definition 8 (Signature).** Let \( H \) be a graph on vertex set \( Y \cup Z \) such that \( Z \) is independent in \( H \), let \( \phi \) be a demand function on \( Y \), and let \( S_Y \subseteq Y \) be a partial solution. The *signature* of \( H \) is the set

\[
\text{Sig}(H, Z, \phi) := \{|S_Y|, \text{Req}(H, Z, \phi, S_Y), \text{Fac}(H, Z, S_Y)\},
\]

where \( S_Y \) ranges over all \( S_Y \subseteq Y \) such that \( |S_Y| \leq d^3 + d \).

We claim that we can replace \( G_Z[Y \cup Z] \) with any graph \( G'[Y' \cup Z] \) that has the same signature and that there is a graph \( G'[Y' \cup Z] \) of constant size with the same signature.

**Lemma 15.** There is a polynomial-time algorithm that receives \( G_Z[Y \cup Z] \), \( Y \) and \( \phi|_Y \) as input and computes a graph \( G' \) on vertex set \( Y' \cup Z \) such that \( Z \) is independent in \( G' \) and a demand function \( \phi' : Y' \rightarrow \{0, \ldots, d\} \) such that \( \text{Sig}(G_Z[Y \cup Z], Z, \phi|_Y) = \text{Sig}(G', Z, \phi') \) and \( |D(G', \phi')| \leq d^3 \). Moreover, the resulting \( G' \) and \( \phi' \) are encoded using at most \( f(d) \) bits and \( G' \) has at most \( f(d) \) vertices, for some computable function \( f \) depending only on \( d \).
Proof. The algorithm is as follows. First, compute $\text{Sig}(G_Z[Y \cup Z], Z, \phi|_Y)$ in polynomial time using Lemmas 11 and 14. Then, generate all graphs containing $Z$ as an independent set in the order of increasing number of vertices (break the remaining ties arbitrarily). For each graph $G'$, iterate over all possible $\phi' : V(G') \setminus Z \rightarrow \{0, \ldots, d\}$ and check whether $\text{Sig}(G_Z[Y \cup Z], Z, \phi|_Y) = \text{Sig}(G', Z, \phi')$ as well as $|D(G', \phi')| \leq d^3$. Clearly, this procedure terminates and finds the required tuple $(G', \phi')$, because $(G_Z[Y \cup Z], \phi|_Y)$ witnesses its existence.

Without loss of generality, we may assume $Z = \{1, \ldots, |Z|\}$. Note now that both requirements and facilities contain only set systems and tuples of elements of $Z$, and numbers in $\{1, \ldots, O(d^3)\}$. Hence, $\text{Sig}(G_Z[Y \cup Z], Z, \phi|_Y)$ can be encoded using at most $g(|Z|) \leq g(d)$ bits, where $g$ is some monotone ascending, computable function. Thus, since $\text{Sig}(G_Z[Y \cup Z], Z, \phi|_Y)$ is the only input to the procedure that finds $G'$ and $\phi'$, the procedure terminates after at most $f'(g(d))$ steps for some computable function $f'$, meaning that $(G', \phi')$ is of size at most $f'(g(d))$. \hfill \Box

Now we can make the notion of replacing $G[Y]$ more precise; it involves an operation commonly referred to as glueing of graphs.

**Definition 9** (Glueing). Let $G_1, G_2$ be two graphs, both containing $Z$ as a subset of their vertices. Glueing $G_1$ and $G_2$ on $Z$ results in a graph denoted by $G_1 \oplus_Z G_2 := (V(G_1) \cup V(G_2), E(G_1) \cup E(G_2))$, where $V(G_1)$ and $V(G_2)$ are treated as being disjoint except for $Z$.

We arrive at the reduction rule aiming at reducing the size of $Y$.

**Reduction Rule 3.** Let $(G, \phi, k)$ be an instance of vector $d$-connectivity($k$) that is reduced with respect to Reduction Rules 2 and 3. Let $Y \in \mathcal{Y}$, where $\mathcal{Y}$ is as in Definition 2, and let $Z = N_G(Y)$. Furthermore, let $(G', \phi')$ be as in Lemma 15. If $|Y| > |V(G') \setminus Z|$, then replace $G$ by $G' \oplus_Z (G - Y)$ and replace $\phi$ by $\phi'' : (V(G') \cup V(G)) \setminus Y \rightarrow \{0, \ldots, d\}$ where $\phi''(v) = \phi(v)$ if $v \in V(G) \setminus Y$ and $\phi''(v) = \phi'(v)$ otherwise.

**Lemma 16.** Rule 3 is safe.

Proof. We claim that an even stronger statement holds. Namely, let $G_1, G_2, \hat{G}$ be three graphs, each containing $Z$ as a vertex subset, where $|Z| \leq d$ and $Z$ is independent in $G_1$ and $G_2$. Furthermore, let $\phi_1 : V(G_1) \setminus Z \rightarrow \{0, \ldots, d\}$, $\phi_2 : V(G_2) \setminus Z \rightarrow \{0, \ldots, d\}$, and $\phi_{\hat{G}} : V(\hat{G}) \rightarrow \{0, \ldots, d\}$ be three demand functions, such that $|D(G_1, \phi_1)|, |D(G_2, \phi_2)| \leq d^3$ and such that $\text{Sig}(G_1, Z, \phi_1) = \text{Sig}(G_2, Z, \phi_2)$. Then $G_1 \oplus_Z \hat{G}$ has a vector connectivity set of size $k$ with respect to $\phi_1 \cup \phi_{\hat{G}}$ if and only if $G_2 \oplus_Z \hat{G}$ has a vector connectivity set of size $k$ with respect to $\phi_2 \cup \phi_{\hat{G}}$. Below we only glue on $Z$, so we will omit the index $Z$ in the $\oplus$ operation.
To see that our claim implies the lemma, set $G_1 := G_Z[Y \cup Z], G_2 := G', \hat{G} := G - Y$ and define the demand functions $\phi_1, \phi_2$ and $\hat{\phi}_G$ accordingly. Then our claim implies that $G_1 \oplus \hat{G} = G_Z[Y \cup Z] \oplus (G - Y) = G$ has a vector connectivity set of size $k$ if and only if $G_2 \oplus \hat{G} = G' \oplus (G - Y)$ has such a set.

Let us prove the claim. Note that, by swapping the names of $G_1$ and $G_2$, as well as $\phi_1$ and $\phi_2$, it suffices to prove one direction. Assume hence that $G_1 \oplus \hat{G}$ has a vector connectivity set $S$ of size $k$ with respect to $\phi_1 \cup \hat{\phi}_G$. Since $|D(G_1, \phi_1)| \leq d^3$ and $|N_{G_1 \oplus \hat{G}}(V(G_1) \setminus Z)| \leq |Z| \leq d$ we may apply Lemma 10 with $Y = V(G_1) \setminus Z$ and hence, we may assume that $S_1 := (V(G_1) \cap S) \setminus Z$ contains at most $d^2 + d$ elements. Thus, we have

\[(|S_1|, \text{Req}(G_1, Z, \phi_1, S_1), \text{Fac}(G_1, Z, S_1)) \in \text{Sig}(G_1, Z, \phi_1)\.
\]

Since $\text{Sig}(G_1, Z, \phi_1) = \text{Sig}(G_2, Z, \phi_2)$, there is a set $S_2 \subseteq V(G_2) \setminus Z$ with $|S_1| = |S_2|$, $\text{Req}(G_1, Z, \phi_1, S_1) = \text{Req}(G_2, Z, \phi_2, S_2)$ and $\text{Fac}(G_1, Z, S_1) = \text{Fac}(G_2, Z, S_2)$. We claim that $S' := (S \setminus S_1) \cup S_2$ is a vector connectivity set for $G_2 \oplus \hat{G}$ with respect to $\phi_2 \cup \hat{\phi}_G$; clearly $|S| = |S'|$.

Let us check that this is true indeed. For each $v_2 \in D(G_2, \phi_2)$, there is at least one vertex $v_1 \in D(G_1, \phi_1)$ with $\text{Sat}(G_1, Z, \phi_1, S_1, v_1) = \text{Sat}(G_2, Z, \phi_2, S_2, v_2)$. Let us show that the demand of $v_2$ is satisfied in $G_2 \oplus \hat{G}$. Since $S$ is a vector connectivity set, by Lemma 9 there is a satisfying connector $(T, F) \in \text{Sat}(G_1, Z, \phi_1, S_1, v_1)$ such that there is a set of paths $P$ in $(G_1 \oplus \hat{G}) - (V(G_1) \setminus Z)$ as defined in statement 2 of Lemma 9. Note that $(G_1 \oplus \hat{G}) - (V(G_1) \setminus Z)$ is a subgraph of $\hat{G}$. Since also $(T, F) \in \text{Sat}(G_2, Y, \phi_2, S_2, v_2)$ and since the paths $P$ are also in $\hat{G}$, by Lemma 9 there are $\phi_2(v_2)$ paths between $S'$ and $v_2$. Similarly, using Lemma 12 all the demands of vertices in $D(\phi_G) \setminus Z$ are satisfied by $S'$ and, using Lemma 13 all the demands of vertices in $D(\hat{\phi}_G) \cap Z$ are satisfied.

**Putting things together.** We can now state our kernelization procedure for instances $(G, \phi, k)$ of vector $d$-connectivity($k$). The only missing piece is to argue why and how we may reduce vertices in $G$ that are not contained in any set of $\mathcal{Y}(G, \phi, d)$.

**Theorem 3.** Vector $d$-connectivity($k$) has a vertex-linear polynomial kernelization.

**Proof.** Given an instance $(G, \phi, k)$ of vector $d$-connectivity($k$) the kernelization proceeds as follows. Throughout, we refer to the current instance by $(G, \phi, k)$ and recall the use of $D = \{v \in V(G) \mid \phi(v) \geq 1\}$.

1. Apply Rule 1 exhaustively and then apply Rule 2 (this may return answer NO if we have more than $d^2k$ demand vertices).

2. Apply Rule 3 once if possible.

28
3. Return to Step 1 if Rule 3 was applied.

4. Let \( W := D \cup \bigcup_{Y \in \mathcal{Y}} N[Y] \). Perform the torso operation on \( W \) in \( G \) to obtain \( G' \):

   (a) Start with \( G' = G[W] \).

   (b) For every pair \( u, v \in W \), if there is a \( u, v \)-path in \( G \) with internal vertices from \( V \setminus W \) then add the edge \( \{u, v\} \) to \( G' \).

5. Return \((G', \phi', k)\) as the kernelized instance, where \( \phi' = \phi|_W \) is \( \phi \) restricted to \( W \).

**Correctness.** We already know that Rules 1 through 3 are correct; it remains to discuss the effect of the torso operation: Proposition 2 implies that minimal solutions \( S \) for \((G, \phi, k)\) are completely contained in the union of sets \( X \in \mathcal{X} \), since only such vertices can contribute to \( S \) being a hitting set for \( \mathcal{X} \). It follows, by Lemma 7, that every minimal solution \( S \) is also contained in \( W \). Thus, if before the torso operation every vertex \( v \in D \) has \( \phi(v) \) paths to \( S \) then the same is true after the operation since there are shortcut edges for all paths with internal vertices from \( V \setminus W \). The converse is more interesting.

Assume that \((G, \phi, k)\) is NO and fix an arbitrary set \( S \subseteq W = V(G') \subseteq V(G) \) of size at most \( k \); we will show that \( S \) is not a solution for \((G', \phi', k)\).

By assumption \( S \) is not a solution for \((G, \phi, k)\) and, by Proposition 2, it is not a hitting set for \( \mathcal{X} = \mathcal{X}(G, \phi) \). Accordingly, fix a set \( X \in \mathcal{X} \) with \( S \cap X = \emptyset \). By definition of \( \mathcal{X} \), let \( v \in X \) with \( \phi(v) > |N(X)| \). By Lemma 7 there is a \( Y \in \mathcal{Y}(G, \phi, d) \) with \( X \subseteq Y \). Thus, \( N[X] \subseteq N[Y] \subseteq W \). If there were \( \phi(v) \) paths from \( v \) to \( S \) in \( G' \) then at least one of them avoids \( N(X) \), since \( |N(X)| < |\phi(v)| \); let \( P' \) denote a path from \( v \) to \( S \) in \( G' \) that avoids \( N(X) \). Undoing the torso operation, we get a walk \( P \) in \( G \), with additional interval vertices from \( V(G) \setminus W \). Since \( (V(G) \setminus W) \cap N(X) = \emptyset \), this walk also avoids \( N(X) \) and implies that \( X \) contains at least one vertex of \( S \); a contradiction to \( S \cap X = \emptyset \). Thus, no \( S \subseteq V(G') \) of size at most \( k \) is a solution for \((G', \phi', k)\), implying that \((G', \phi', k)\) is NO, as required.

**Size.** The output graph \( G' \) has vertex set \( W = D \cup \bigcup_{Y \in \mathcal{Y}} N[Y] \) where \( D \) and \( \mathcal{Y} \) correspond to a fully reduced instance \((G, \phi, k)\). By Rule 2 we have \(|D| \leq d^2 k \). By Lemma 8 the set \( \mathcal{Y} \) contains at most \( 2^{d^3 + d^2 k} \) sets, each of size bounded by \( f(d) \) for some computable function depending only on \( d \). By Definition 2 the neighborhood \( N(Y) \) of each set \( Y \) has size at most \( d \). It follows that \( G' \) has \( O(k) \) vertices, as claimed. Clearly, the total encoding size for an instance can be bounded by \( O(k^2) \) since \( d \) is a constant.

**Runtime.** By Lemma 2 Rule 1 can be applied exhaustively in polynomial time. Clearly, Rule 2 can be applied in polynomial time as it only checks the number of non-zero-demand vertices. Finding one application of Rule 3 can be done by iterating over \( Y \in \mathcal{Y} \) and applying Lemma 15 to
each \( Y \) until we find a replacement subgraph that is strictly smaller; in total this takes polynomial time. Furthermore, repeating these steps whenever Rule 3 has been applied gives only a polynomial factor because each time the instance shrinks by at least one vertex. Finally, it is easy to implement the torso operation in polynomial time.

\section{Kernelization lower bound}

In this section, we prove that \textsc{vector connectivity}(\( k \)) admits no polynomial kernelization unless \( \text{NP} \subseteq \text{coNP}/\text{poly} \). We give a reduction from \textsc{hitting set}(\( m \)), i.e., \textsc{hitting set} with parameter number of sets, which also makes a polynomial Turing kernelization unlikely (cf. \cite{v15}). Since demands greater than \( k+1 \) can be safely replaced by demand \( k+1 \), implying \( d \leq k+1 \), the lower bound applies also to parameterization by \( d+k \).

\begin{theorem}
\textsc{vector connectivity}(\( k \)) does not admit a polynomial kernelization unless \( \text{NP} \subseteq \text{coNP}/\text{poly} \) and the polynomial hierarchy collapses.
\end{theorem}

\begin{proof}
We give a polynomial parameter transformation from \textsc{hitting set}(\( m \)) to \textsc{vector connectivity}(\( k \)), which is known to imply the claimed lower bound (cf. \cite{v2}). Let \(( U, \mathcal{F}, k )\) be an instance of \textsc{hitting set}(\( m \)) with parameter \( m = |\mathcal{F}| \); w.l.o.g. \( k \leq m \). Let \( n := |U| \). We construct a graph \( G \) on \( 2(k+1)m+n \) vertices that has a vector connectivity set of size at most \( k' = (k+1)m+k = \mathcal{O}(m^2) \) if and only if \(( U, \mathcal{F}, k )\) is \textsc{yes} for \textsc{hitting set}(\( m \)).

\textbf{Construction.} Make one vertex \( x_u \) for each element \( u \in U \), and make \( 2(k+1) \) vertices \( y_{1,F}, \ldots, y_{k+1,F}, y'_{1,F}, \ldots, y'_{k+1,F} \) for each set \( F \in \mathcal{F} \). We add the following edges:

1. Add \( \{y_{i,F}, y'_{i,F}\} \) for all \( i \in \{1, \ldots, k+1\} \) and \( F \in \mathcal{F} \).
2. Add \( \{x_u, y_{i,F}\} \) for all \( i \in \{1, \ldots, k+1\} \), \( F \in \mathcal{F} \), and \( u \in F \).
3. Make the set of all vertices \( y_{i,F} \) a clique (not including any \( y'\)-vertex).

Set the demand \( \phi \) of each \( y'_{i,F} \) vertex to 2 and of each \( y_{i,F} \) vertex to \( (k+1)m+1 \); all \( x \)-vertices have demand zero. Set the budget \( k' \) to \( (k+1)m+k \). This completes the construction of an instance \(( G, \phi, k' )\), which can be easily performed in polynomial time.

\textbf{Correctness.} Assume first that \(( G, \phi, k' )\) is \textsc{yes} and let \( S \) a vector connectivity set of size at most \( k' \). Note that \( S \) must contain all vertices \( y'_{i,F} \) since they have demand of 2 but only one neighbor (namely \( y_{i,F} \)). This accounts for \( (k+1)m \) vertices in \( S \); there are at most \( k \) further vertices in \( S \). Let \( T \) contain exactly those elements \( u \in U \) such that \( x_u \in S \); thus \( |T| \leq k \). We claim that \( T \) is a hitting set for \( \mathcal{F} \). Let \( F \in \mathcal{F} \) and assume that \( T \cap F = \emptyset \). It follows that \( S \) contains no vertex \( x_u \) with \( u \in F \). Since
at most $k$ vertices in $S$ are not $y'$-vertices, we can choose $i \in \{1, \ldots, k + 1\}$ such that $S$ does not contain $y_{i,F}$. Consider the set $C$ consisting of all $y$-vertices other than $y_{i,F}$ as well as the vertex $y'_{i,F}$. In $G - C$ we find a connected component containing $y_{i,F}$ and all $x_u$ with $u \in F$ but no further vertices. Crucially, all other neighbors of $y_{i,F}$ are $y'_{i,F}$ and all $y$-vertices, and $x$-vertices only have $y$-vertices as neighbors. By assumption $S$ contains no vertex of this connected component. This yields a contradiction cause $C$ is of size $(k + 1)m$ and separates $y_{i,F}$ from $S$, but since $S$ is a solution with $y_{i,F} \notin S$ there should be $(k + 1)m + 1$ disjoint paths from $y_{i,F}$ to $S$. Thus, $S$ must contain some $x_u$ with $u \in F$, and then $T \cap F \neq \emptyset$.

Now, assume that $(U, F, k)$ is yes for hitting set$(m)$ and let $T$ a hitting set of size at most $k$ for $F$. We create a vector connectivity set $S$ by selecting all $x_u$ with $u \in T$ as well as all $y'$-vertices; thus $|S| \leq k' = (k + 1)m + k$. Clearly, this satisfies all $y'$-vertices. Consider any vertex $y_{i,F}$ and recall that its demand is $\phi(y_{i,F}) = (k + 1)m + 1$. We know that $S$ contains at least one vertex $x_u$ with $u \in F$ that is adjacent to $y_{i,F}$. Thus, we can find the required $(k + 1)m + 1$ disjoint paths from $y_{i,F}$ to $S$:

- We have one path $(y_{i,F}, y'_{i,F})$ and one path $(y_{i,F}, x_u)$.
- For all $(j, F') \neq (i, F)$ we get one path $(y_{i,F}, y_{j,F}, y'_{j,F})$; we get $(k + 1)m - 1$ paths total.

It follows that $(G, \phi, k')$ is yes for vector connectivity$(k)$.

We have given a polynomial parameter transformation from hitting set$(m)$, which is known not to admit a polynomial kernelization unless $\text{NP} \subseteq \text{coNP}/\text{poly}$ [10] (see also [14]). This is known to imply the same lower bound for vector connectivity$(k)$ [3].

6 Conclusion

We have presented kernelization and approximation results for vector connectivity and vector $d$-connectivity. The centerpiece of our results is a reduction rule that reduces the number of vertices with nonzero demand to at most $d^2\text{opt}$ (or, similarly, to at most $\text{opt}^3 + \text{opt}$ or $k^3 + k$). Since the set of nonzero demand vertices always constitutes a feasible solution, the approximation results follow directly. For vector $d$-connectivity we get a constant factor of $d^2$. For the special case of all demands being either 0 or $d$ the approximation factor improves to $(2d - 1)$; this is still NP-hard when $d \geq 4$ from vertex cover (Milanič [20]). It would be interesting whether a ratio of $O(d)$ can also be achieved for demands in $\{0, \ldots, d\}$.

On the kernelization side we show that vector connectivity$(k)$ does not admit a polynomial kernelization unless $\text{NP} \subseteq \text{coNP}/\text{poly}$. Since demands greater than $k + 1$ can be safely replaced by demand $k + 1$ (because they cannot be fulfilled without putting the vertex into the solution) the
lower bound extends also to parameter $k + d$. For \textsc{vector $d$-connectivity}($k$), where $d$ is a problem-specific constant, we get a vertex-linear kernelization with at most $f(d) \cdot k = \mathcal{O}(k)$ vertices; the computable function $f(d)$ is superpolynomial in $d$, which is necessary (unless $\text{NP} \subseteq \text{coNP/poly}$) due to the lower bound for $d + k$. We do not optimize the dependence on $d$ and ask whether single exponential dependence on $d$ is possible.

Finally, the reduction to $k^3 + k$ nonzero demand vertices allows an alternative proof for fixed-parameter tractability. We have sketched how to use an algorithm of Marx [13] that permits us to find a set of size $k$ that is independent in each of $\ell$ linear matroids in time FPT with respect to $k + \ell$ to find a vector connectivity set for $(G, \phi, k)$ if one exists.

References

[1] F. N. Abu-Khzam. A kernelization algorithm for d-hitting set. \textit{J. Comput. Syst. Sci.}, 76(7):524–531, 2010.

[2] H. L. Bodlaender, F. V. Fomin, D. Lokshtanov, E. Penninkx, S. Saurabh, and D. M. Thilikos. (Meta) Kernelization. In \textit{Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)}, pages 629–638. IEEE Computer Society, 2009.

[3] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Kernelization lower bounds by cross-composition. \textit{SIAM J. Discrete Math.}, 28(1):277–305, 2014.

[4] E. Boros, P. Heggernes, P. van ’t Hof, and M. Milanic. Vector connectivity in graphs. In T.-H. H. Chan, L. C. Lau, and L. Trevisan, editors, \textit{Proceedings of the 10th International Conference on Theory and Applications of Models of Computation (TAMC)}, volume 7876 of \textit{Lecture Notes in Computer Science}, pages 331–342. Springer, 2013.

[5] E. Boros, P. Heggernes, P. van ’t Hof, and M. Milanic. Vector connectivity in graphs. \textit{Networks}, 63(4):277–285, 2014.

[6] N. Bousquet, J. Daligault, and S. Thomassé. Multicut is FPT. In L. Fortnow and S. P. Vadhan, editors, \textit{Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)}, pages 459–468. ACM, 2011.

[7] F. Cicalese, M. Milanič, and R. Rizzi. On the complexity of the vector connectivity problem. Manuscript.

[8] M. Cygan, S. Kratsch, M. Pilipczuk, M. Pilipczuk, and M. Wahlström. Clique cover and graph separation: New incompressibility results. \textit{Transactions on Computation Theory}, 6(2):6, 2014.
[9] H. Dell and D. Marx. Kernelization of packing problems. In Y. Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 68–81. SIAM, 2012.

[10] M. Dom, D. Lokshtanov, and S. Saurabh. Incompressibility through colors and IDs. In S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. E. Nikoletseas, and W. Thomas, editors, *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 5555 of *Lecture Notes in Computer Science*, pages 378–389. Springer, 2009.

[11] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013.

[12] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[13] F. V. Fomin, D. Lokshtanov, S. Saurabh, and D. M. Thilikos. Bidiimensionality and kernels. In M. Charikar, editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 503–510. SIAM, 2010.

[14] D. Hermelin, S. Kratsch, K. Soltys, M. Wahlström, and X. Wu. A completeness theory for polynomial (Turing) kernelization. In G. Gutin and S. Szeider, editors, *Proceedings of the 8th International Symposium on Parameterized and Exact Computation (IPEC)*, volume 8246 of *Lecture Notes in Computer Science*, pages 202–215. Springer, 2013.

[15] S. Kratsch and M. Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 450–459. IEEE Computer Society, 2012.

[16] D. Lokshtanov. Private communication, 2014.

[17] D. Marx. Parameterized graph separation problems. *Theor. Comput. Sci.*, 351(3):394–406, 2006.

[18] D. Marx. A parameterized view on matroid optimization problems. *Theor. Comput. Sci.*, 410(44):4471–4479, 2009.

[19] D. Marx and I. Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM J. Comput.*, 43(2):355–388, 2014.

[20] M. Milanic. Private communication, 2014.

[21] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.
[22] J. Oxley. *Matroid Theory*. Oxford University Press, 2011.

[23] N. Robertson and P. Seymour. Graph minors XIII. The disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.