# An Exact Algorithm for Knot-Free Vertex Deletion

**M. S. Ramanujan**
Department of Computer Science, University of Warwick, UK

**Abhishek Sahu**
Institute of Mathematical Sciences, Chennai, India

**Saket Saurabh**
Institute of Mathematical Sciences, Chennai, India
University of Bergen, Norway

**Shaily Verma**
Institute of Mathematical Sciences, Chennai, India

## Abstract

The study of the Knot-Free Vertex Deletion problem emerges from its application in the resolution of deadlocks called knots, detected in a classical distributed computation model, that is, the OR-model. A strongly connected subgraph $Q$ of a digraph $D$ with at least two vertices is said to be a knot if there is no arc $(u, v)$ of $D$ with $u \in V(Q)$ and $v \notin V(Q)$ (no-out neighbors of the vertices in $Q$). Given a directed graph $D$, the Knot-Free Vertex Deletion (KFVD) problem asks to compute a minimum-size subset $S \subset V(D)$ such that $D[V \setminus S]$ contains no knots. There is no exact algorithm known for the KFVD problem in the literature that is faster than the trivial $\mathcal{O}^\star(2^n)$ brute-force algorithm. In this paper, we obtain the first non-trivial upper bound for KFVD by designing an exact algorithm running in time $\mathcal{O}^\star(1.576^n)$, where $n$ is the size of the vertex set in $D$.

**2012 ACM Subject Classification** Theory of computation → Fixed parameter tractability

**Keywords and phrases** exact algorithm, knot-free graphs, branching algorithm

**Digital Object Identifier** 10.4230/LIPIcs.MFCS.2022.78

**Funding** *M. S. Ramanujan*: Supported by Engineering and Physical Sciences Research Council (EPSRC) grants EP/V007793/1 and EP/V044621/1.
*Saket Saurabh*: Supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819416); and he also acknowledges the support of Swarnajayanti Fellowship grant DST/SJF/MSA-01/2017-18.

## 1 Introduction

In concurrent computing, a deadlock [6] is a state in which each group member waits for another member, including itself, to take action such as sending a message or, more commonly, releasing a lock. Deadlocks are a common problem in multiprocessing systems, parallel computing, and distributed systems. Resolving these deadlocks is a fundamental problem in distributed settings with no efficient (known) algorithms. Note that while distributed systems are dynamic, a deadlock is a stable property. In other words, once a deadlock occurs in the system, it would remain there until it is resolved. Two of the main classic deadlock models are the AND-model and the OR-model [1, 2, 14, 17]. Deadlocks are characterized by the existence of cycles in the AND-model and by the existence of knots in the OR-model. Hence the problem of preventing deadlocks in the AND-model is equivalent to the Directed Feedback Vertex Set (DFVS) problem and in the OR-model, it is equivalent to the Knot-Free Vertex Deletion (KFVD) problem [13].

A *wait-for* graph is useful to analyze deadlock situations. In a wait-for graph, $D = (V; E)$, the vertex set $V$ represents processes, and the set $E$ of directed arcs represents wait-conditions.


A knot in a directed graph $D$ is a strongly connected subgraph $Q$ of $D$ with at least two vertices, such that no vertex in $V(Q)$ has an out-neighbor in $V(D) \setminus V(Q)$. The number of processes needed to be preempted to prevent a deadlock in the OR-model is essentially the same as solving the KFVD problem in wait-for graphs. Formally, the problem of finding a vertex subset $S \subseteq V$ of minimum cardinality such that $D[V \setminus S]$ does not contain any knots is known as the KFVD problem.

Recently, KFVD has been studied extensively from algorithmic perspectives. In particular, the problem has been studied on graph classes satisfying some structural properties and from the viewpoint of parameterized complexity [4]. In this paper, we study KFVD from yet another algorithmic paradigm, that is, exact exponential-time algorithms. There has been immense progress in this area in the last two decades, resulting in non-trivial exact exponential algorithms for numerous problems, including Chromatic Number, Hamiltonian Cycle and Satisfiability [3, 9, 12, 16] We refer to the monograph of Fomin and Kratsch for a detailed exposition of the field [11].

## 1.1 Our contribution

In this paper, using the powerful method of *Measure & Conquer*, pioneered by Fomin, Kratsch, and Woeginger [10], we design the first non-trivial exact algorithm for the KFVD problem. In particular, we obtain the following result.

▶ **Theorem 1.** *There exists an algorithm for* Knot-free vertex deletion *running in* $\mathcal{O}^*(1.576^n)$ *time.*

The starting point of our algorithm is the following simple observation: *a graph is knot-free if and only if every vertex has a path to a sink.* Moreover, finding a minimum size knot-free deletion set is equivalent to finding a subset $Z$ of *sinks (vertices with no out-neighbors)* such that $N^+(Z)$ is exactly the deletion set [2], that is $N^+(Z) = S$. Our algorithm utilizes this observation, and rather than obtaining a minimum deletion set $S$; it constructs a sink set $Z$ such that every vertex in $D - N^+(Z)$ has a path in $D - N^+(Z)$ to some vertex in $Z$. Note that $Z$ does not complement $S$ The measure associated with the instance $I = (D, V_1, V_2)$, is given by $\phi(I) = |V_1| + \frac{|V_2|}{2}$. We select a vertex $u \in V_1$ to branch, which means that either $u$ is a sink, or it is not. If $u$ is going to be a sink, then observe that all its out-neighbors, $N^+(u)$, must be in the deletion set and be safely deleted. Further, every vertex that can reach $u$ in $D - N^+(u)$, say in-reachable vertices, must not go to the deletion set. However, we can remove them and work on a smaller graph. Thus removing in-reachable vertices is a prepossessing step that reduces the size of the graph. In the branch where we decide that $u$ is not part of the sink set, we cannot delete this vertex, and thus, to capture our progress, we move the vertex $u$ from $V_1$ to $V_2$, resulting in $\phi$ decreasing by 0.5. To show the desired running time, we first do "potential sensitive" branching and then do an extensive case analysis based on the degree of the vertex in $V_1$.

## 1.2 Previous Work

Observe that any directed feedback vertex set (a set of vertices that intersect every cycle in the digraph) is a knot-free deletion set as it removes any strongly connected component of size at least 2. Hence, the size of the smallest directed feedback vertex set gives an upper bound on the size of the smallest knot-free vertex deletion set. One way of eliminating knots in a graph could be to use known algorithms for the Directed Feedback Vertex Set (DFVS) problem. While DFVS has been extensively studied [7, 15], the KFVD problem is yet

to receive the same scale of attention. Notice that the optimal solution for KFVD could be significantly smaller than the solution returned by invoking an existing algorithm for DFVS, which motivated a closer look at the KFVD problem itself.

Carneiro, Souza, and Protti [5] first studied the problem and showed that KFVD is NP-complete even when restricted to planar bipartite graphs with the maximum degree 4. They also presented a polynomial-time algorithm for graphs with the maximum degree 3. Recently, KFVD was studied in the parameterized framework. Carneiro et al. [4] showed that KFVD is W[1]-hard when parameterized by the size of the solution, but it can be solved in $2^{k \log \phi} n^{\mathcal{O}(1)}$ time, where $\phi$ is the size of the largest strongly connected subgraph. KFVD parameterized by the size of the solution $k$, is W[1]-hard even when the length of a longest directed path of the input graph, as well as its Kenny-width, are bounded by constants [2]. However, the problem is known to be in FPT parameterized by either clique-width or treewidth of the underlying undirected graph. Moreover, the KFVD problem is known to admit FPT algorithms when parameterized by *dfvs+maximum path length* or *dfvs+Kenny-width* where, *dfvs* denotes the size of the smallest directed feedback vertex set in the graph.

*Organization of the paper:* In Section 3, we start by providing our algorithm, and we verify the correctness of each of its steps in Section 4. The running time is analyzed in Section 5, while Section 6 provides a lower bound on the worst-case performance of our algorithm.

## 2 Preliminaries and Auxiliary Results

In this section, we first state some notations, definitions, and useful auxiliary results. We also formalize an appropriate potential function based on which we later design our algorithm. A few reduction rules are proved towards the end of this section which are used throughout the paper.

**Standard Notation.** For a digraph $D$, $V(D)$ and $E(D)$ denote the set of vertices and arcs, respectively. We denote an arc from $u$ to $v$ by the ordered pair $(u, v)$. For a vertex $v$ of $D$, the out-neighborhood of $v$ is denoted by $N^+(v) = \{u \mid (v, u) \in E(D)\}$. Similarly, we denote its in-neighborhood by $N^-(v) = \{u \mid (u, v) \in E(D)\}$ and $N(v) = N^+(v) \cup N^-(v)$. A vertex $v$ is called a *source vertex*, if $N^-(v) = \emptyset$. Similarly it is called a *sink vertex*, if $N^+(v) = \emptyset$. We define the *in-reachability set* of a vertex $v$ as the set of vertices that can reach the vertex $v$ via some directed path in $D - N^+(v)$ and we denote it by $R^-(v)$. Notice that $v \in R^-(v)$. We define, $R(v) = N^+(v) \cup R^-(v)$. For a set $S \subseteq V(D)$, $D - S$ denotes the digraph obtained by deleting the vertices $S$ and the edges incident on the vertices of $S$ and $D[S]$ denotes the subgraph of $D$ induced on $S$. A *path* $P$ of length $\ell$ is a sequence of distinct vertices $v_1, v_2, \ldots, v_\ell$ such that $(v_i, v_{i+1})$ is an arc, for each $i$, $i \in [\ell - 1]$. In our algorithm, branching vector $(a, b)$ means a branching where in the first branch, the potential (measure) drops by $a$ while in the second branch, the potential drops by $b$. It is a measure of the effectiveness of any branching step. Later we provide a detailed description of such a potential function and how it relates to the hardness of our instance. For graph-theoretic terms and definitions not stated explicitly here, we refer to [8].

### 2.1 Auxiliary Results

In this subsection, we first state some of the known reduction rules and some new reduction rules for the problem that we use in our branching algorithm.

▶ **Reduction Rule 2** ( [2]). *Let $v \in D$ be such that $N^-(v) = \emptyset$, Then $(D, k)$ is a yes-instance if and only if $(D - v, k)$ is a yes-instance.*

▶ **Reduction Rule 3** ( [2]). *Let $v \in D$ be such that $N^+(v) = \emptyset$, Then $(D, k)$ is a yes-instance if and only if $(D - R^-(v), k)$ is a yes-instance.*

▶ **Proposition 4** ( [2]). *A digraph $D$ is knot-free if and only if for every vertex $v$ of $D$, $v$ has a path to a sink.*

▶ **Corollary 5** ( [2]). *For any optimal solution $S \subseteq V(D)$ with the set of sink vertices $Z$ in $D - S$, we have $N^+(Z) = S$.*

Proposition 4 and Corollary 5 imply that given a digraph $D$, the problem of finding a set of sink vertices $Z$ such that every vertex in $V(D) \setminus N^+(Z)$ has a directed path to a vertex in $Z$ and $|N^+(Z)|$ is minimum; is equivalent to the Knot-free Vertex Deletion (KFVD) problem. Therefore, our algorithm aims to identify a set of (eventually, to be) sink vertices $Z$ while minimizing $|N^+(Z)|$ instead of directly looking for the deletion set.

**Strategy of our Algorithm.** We design a branching algorithm for KFVD in general digraphs. At any iteration in our algorithm, we branch on a potential vertex $v \in V(D)$ based on the two possibilities that either $v$ is a sink vertex or a non-sink vertex in an optimal solution. Observe that even if a vertex becomes a non-sink vertex corresponding to an optimal solution, there are two possibilities: it belongs to the deletion set or does not. So we can not simply forget about this vertex, which means its behavior in the final knot-free graph remains inconclusive. To track the possible vertices that become the sink or non-sink, we use a potential function ($\phi$) for $V(D)$ defined as follows.

▶ **Definition 6** (Potential function). *Given a digraph $D = (V, E)$, we define a* potential function *on $V(D)$, $\phi : V(D) \to \{0.5, 1\}$ such that $\phi(v) = 1$, if $v$ is a potential vertex to become a sink in an optimal solution and $\phi(v) = 0.5$, if $v$ is a non-sink vertex. For any subset $V' \subseteq V(D)$, $\phi(V') = \sum_{x \in V'} \phi(x)$. We call a vertex $v$ as an* undecided *vertex if $\phi(v) = 1$ and a* semi-decided *vertex if $\phi(v) = 0.5$.*

To solve the Knot-free Vertex Deletion problem on a digraph $D$, we initialize the potential values of all vertices to 1. As soon as we decide a vertex to be a non-sink vertex, we drop its potential by 0.5. Any vertex whose potential is 0.5 can not become a sink in the final knot-free graph resulting from removing an optimal vertex deletion set from $D$.

▶ **Definition 7** (Feasible solution). *A set $S \subseteq V(D)$ is called a feasible solution for $(D, \phi)$ if $D - S$ is knot-free and for any sink vertex $s$ in $D - S$, $\phi(s) = 1$. $\mathsf{KFVD}(D, \phi)$ is the size of an optimal solution for $(D, \phi)$.*

▶ **Reduction Rule 8.** *If all the vertices in $D$ are semi-decided and $D$ has no source or sink vertices, then $V(D)$ is contained inside any feasible solution for $(D, \phi)$.*

**Proof.** If $S$ is a solution for $(D, \phi)$, then $D - S$ is knot-free. Since all vertices of $D$ are semi-decided, $D - S$ has no sink vertices. Therefore, $D - S$ is an empty graph. In other words, $S = V(D)$. ◀

Let $S_{opt}$ and $Z_{opt}$ be the set of deleted vertices and the set of sink vertices with respect to some optimal solution $\mathsf{KFVD}(D, \phi)$.

▷ Claim 9. If $x \in Z_{opt}$, then $N^+(x)$ is in $S_{opt}$ and $S_{opt} \cap R^-(x) = \emptyset$.

Proof. By the definition of a sink vertex, if $x \in Z_{opt}$ then $N^+(x)$ is in $S_{opt}$. Let $Y = S_{opt} \cap R^-(x)$. We claim $S' = S_{opt} \setminus Y$ is also a solution which will contradict the fact that $S_{opt}$ is an optimal solution. Suppose $S'$ is not a solution, then there exists a vertex $v$ in $G \setminus S'$, that do not reach to a sink $s$ in $G \setminus S'$ but $v$ reaches to some sink $s$ in $G \setminus S_{opt}$. Since every vertex in $Y$ reaches to some sink $x$ in $G \setminus S'$, $v \notin Y$. If $s$ is not a sink vertex in $G \setminus S'$, then some vertex $y \in Y$ is an out-neighbor of $s$. But then $v$ can reach the sink $x$ in $G \setminus S'$ via $y$. Hence, $S'$ is a solution of strictly smaller size than $S_{opt}$, which is a contradiction. ◁

▷ Claim 10. If $x \in Z_{opt}$, then $|S_{opt}| = |N^+(x)| + \mathsf{KFVD}(D - R(x), \phi)$.

Proof. First, we prove that $S' = S_{opt} \setminus N^+(x)$ is a solution to $(D - R(x), \phi)$. If this is not true, then there exists a vertex $v$ in $D - (R(x) \cup S')$ that do not reach any sink. Since $S_{opt} \subseteq R(x) \cup S'$, $v \notin S_{opt}$. But $S_{opt}$ is a solution to $(D, \phi)$ and $v$ can reach some sink $s \in Z_{opt}$ in $D - S_{opt}$. First we claim that $s \neq x$. Note that $v$ is not in $R(x)$ and it can only reach $x$ via some vertex in $N^+(x)$. So $v$ can not reach $x$ in $D - S_{opt}$ as $N^+(x) \subseteq S_{opt}$. Hence $s \neq x$. Then $v$ has a path to $s$ which is disjoint from $S_{opt} \supseteq N^+(x)$. Moreover this path is also disjoint from the set $R(x) \setminus N^+(x)$, since $v \notin (R(x)$. Hence this path is disjoint from $R(x)$ and $S_{opt}$. Therefore, $v$ can reach $s$ via the same path in $D - (R(x) \cup S')$. If $s$ is a sink in $D - S_{opt}$, then $s$ is also a sink in $(D - R(x)) \setminus S'$. Hence, $v$ has a path to a sink in $D$, which is a contradiction. It implies that $S' = S_{opt} \setminus N^+(x)$ is a solution to $(D - R(x))$ and therefore, $|S_{opt}| - |N^+(x)| \geq \mathsf{KFVD}(D - R(x), \phi)$.

To prove the other direction, assume that $S''$ is an optimal solution to $(D - R(x), \phi)$. We claim that $S' = S'' \cup N^+(x)$ is a solution for $(D, \phi)$. Suppose not, then there exists a vertex $v$ that does not reach a sink in $D - S'$. Note that $v \notin R(x)$ as all vertices in $R(x)$ can reach sink $x$ in $D - S'$. Then $v \notin R(x) \cup S'$ and hence it is also in $D - (R(x) \cup S'')$ and it reaches a sink $s$. Since this path is disjoint from $S'' \cup R(x)$, $v$ still can reach $s$ via the same path in $D - S'$. Note that $s$ is not a sink in $D - S'$ only if it has an out-neighbor in $R(x) \setminus N^+(x)$. But then $s$ and $v$ are in $R(x)$ which is not possible. Hence, $v$ can still reach the same sink $s$ and $S' = S'' \cup N^+(x)$ is a solution to $D$. Thus, $S_{\mathsf{opt}} \leq |N^+(x)| + \mathsf{KFVD}(D - R(x))$. ◁

## 3 An algorithm to compute minimum knot-free vertex deletion set

In this section, we design an exact algorithm to compute a minimum knot-free vertex deletion set. As mentioned earlier, we start by initializing the potential values of all vertices to 1. As soon as we decide a vertex is a non-sink vertex, we drop its potential by 0.5. In the following algorithm, at any step, if there are vertices with no out-neighbors (sinks) or no in-neighbors (sources), we remove such vertices with the help of Reduction Rules 2 and 3. If all the vertices are semi-decided, we apply Reduction Rule 8 to solve the instance in polynomial time. At any step, if there is an undecided vertex $x$ with $\phi(R(x)) \geq 3.5)$, we branch on the possibility of $x$ being a sink or non-sink in the optimal solution. Here we use the large potential drop in the branch where $x$ is chosen to be a sink to our advantage and obtain a $(3.5, 0.5)$ branching. If there are no such vertices, we look for an undecided vertex $x$ such that all its neighbors are semi-decided, and we branch on $x$. If there are no such vertices, we branch on undecided vertices of degrees 2 and 3. Notice that in these branching steps, even if there is not a large potential drop when $x$ is a sink, in the other branch when $x$ is a non-sink vertex, we still find an undecided vertex $s$ close to $x$, which is forced to be a sink. Hence the potential drop together in both the branches ensure *good* running time for our algorithm. If all undecided vertices have only one neighbor each, then we remove such vertices with the help of Reduction Rules 2 and 3.

**Algorithm 1** KFVD $(D, \phi)$.

**Input:** A directed graph $D$ and a potential function $\phi$
**Output:** Size of a minimum knot-free vertex deletion set

1 **if** $\exists$ $x$ *such that* $N^-(x) = \emptyset$ **then**
2 | **return** KFVD$(D - \{x\}, \phi)$;
3 **if** $\exists$ $x$ *such that* $N^+(x) = \emptyset$ **then**
4 | **return** KFVD$(D - R(x), \phi)$;
5 **if** $\forall v \in D$, $\phi(v) = 0.5$ **then**
6 | **return** $|V(D)|$;
7 **if** $\exists$ $x \in D$ *such that* $\phi(x) = 1$ & $\phi(R(x)) \geq 3.5$ **then**
8 | $D_1 = D - R(x)$;
9 | $D_2 = D$;
10 | $\phi_1 = \phi$;
11 | $\phi_2 = \phi$;
12 | $\phi_2(x) = 0.5$;
13 | **return** $\min\{$KFVD$(D_1, \phi_1) + |N^+(x)|$, KFVD$(D_2, \phi_2)\}$;
14 **if** $\exists$ $x \in D$ *such that* $\phi(x) = 1$ & $\forall v \in N(x)$, $\phi(v) = 0.5$ **then**
15 | **for** $y \in N^-(x)$ & $z \in N^+(x)$ **do**
16 | | **if** $\exists$ $s$ *such that* $s \in N^-(y) \cap N^+(z)$ & $\phi(s) = 1$ **then**
17 | | | $D_1 = D - R(x)$;
18 | | | $\phi_1 = \phi$;
19 | | | $D_2 = D - R(s)$;
20 | | | $\phi_2 = \phi$;
21 | | | **return** $\min\{$KFVD$(D_1, \phi_1) + |N^+(x)|$, KFVD$(D_2, \phi_2) + |N^+(s)|\}$;
22 | | **else**
23 | | | $D_1 = D - R(x)$;
24 | | | $\phi_1 = \phi$;
25 | | | **return** KFVD$(D_1, \phi_1) + |N^+(x)|$;
26 **if** $\exists x \in D$ *such that* $|N(x)| \in \{2, 3\}$, $\phi(x) = 1$ & $\exists$ *a unique* $s \in N(x)$ *with* $\phi(s) = 1$ **then**
27 | $D_1 = D - R(x)$;
28 | $\phi_1 = \phi$;
29 | $D_2 = D - R(s)$;
30 | $\phi_2 = \phi$;
31 | **return** $\min\{$KFVD$(D_1, \phi_1) + |N^+(x)|$, KFVD$(D_2, \phi_2) + |N^+(s)|\}$;
32 **if** $\exists$ $x \in D$ *with* $N^-(x) = \{y\}$, $N^+(x) = \{z\}$ *and* $\phi(x) = \phi(y) = \phi(z) = 1$ **then**
33 | $D_1 = D - R(x)$;
34 | $\phi_1 = \phi$;
35 | $D_2 = D - R(y)$;
36 | $\phi_2 = \phi$;
37 | $D_3 = D - R(z)$;
38 | $\phi_3 = \phi$;
39 | **return**
| $\min\{$KFVD$(D_1, \phi_1) + |N^+(x)|$, KFVD$(D_2, \phi_2) + |N^+(y)|$, KFVD$(D_3, \phi_3) + |N^+(z)|\}$;

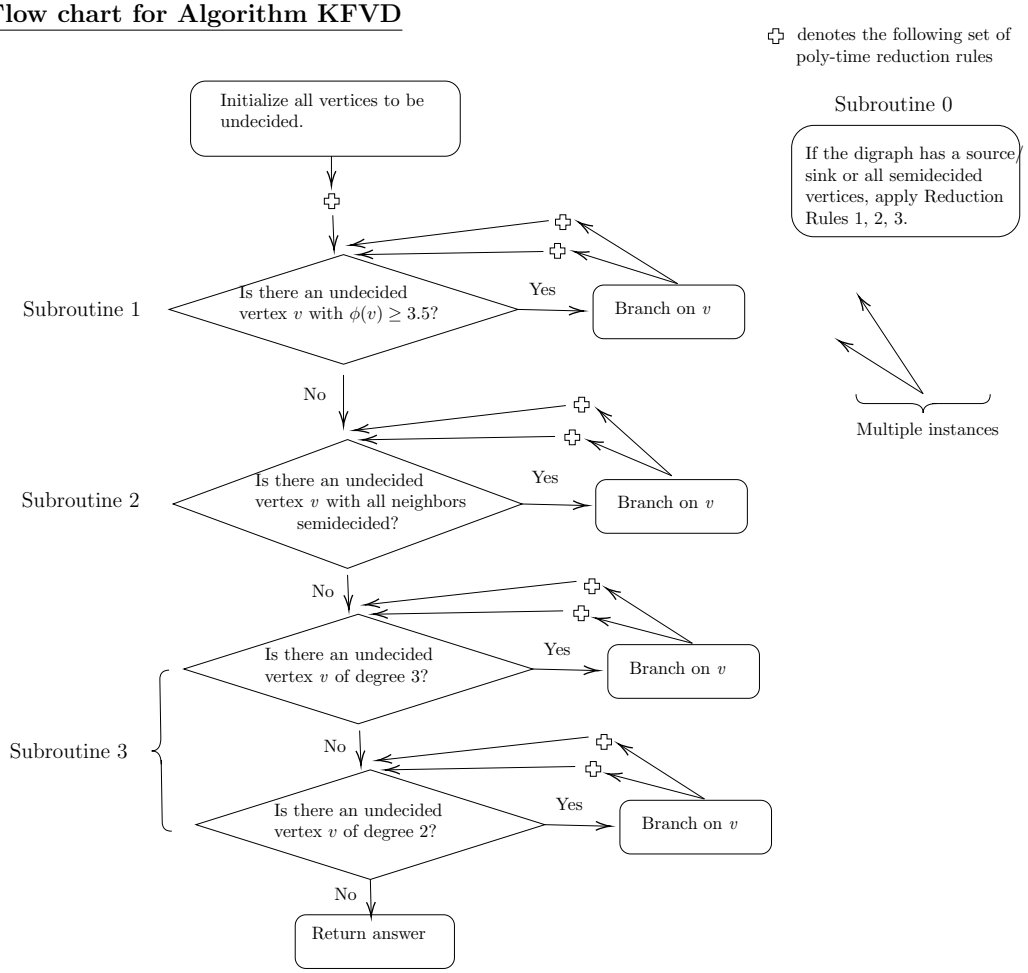
## 4 Correctness of the Algorithm

In this section, we prove that the Algorithm KFVD returns a knot-free vertex deletion set of minimum size. We denote the steps from the lines 1-6 of the algorithm as *Subroutine* 0. The correctness of the lines 1-4 follows from the Reduction Rules 2, and 3 and the correctness of the lines 5-6 follows from the Reduction Rule 8. We call the steps from the lines 7-13, 14-25 and 26-39 of Algorithm KFVD as *Subroutine* 1, *Subroutine* 2, and *Subroutine* 3, respectively.

In the subsequent subsections, we will prove the correctness of Subroutines 1, 2, and 3.

Below we provide a flowchart for Algorithm KFVD.

**Flow chart for Algorithm KFVD**




## 4.1 Correctness of Subroutine 1

Subroutine 1 branches on an undecided vertex $x \in V(D)$, if $\phi(R(x)) \geq 3.5$. Notice that for any solution $S$ in polynomial time we can determine the set of sink vertices, say $Z_S$, corresponding to set $S$.

▶ **Lemma 11.** *If $x \in V(D)$ such that $\phi(x) = 1$ and $\phi(R(x)) \geq 3.5$, then $\mathsf{KFVD}(D, \phi) = \min\{\mathsf{KFVD}(D_1, \phi_1) + |N^+(x)|,\ \mathsf{KFVD}(D_2, \phi_2)\}$ where $D_1 = D - R(x)$, $D_2 = D$ and $\phi_1 = \phi$, $\phi_2(v) = \phi(v)$, for all $v \in V(D) \setminus \{x\}$ and $\phi_2(x) = 0.5$.*

**Proof.** We use induction on the total potential $\phi(V(D))$ of the digraph $D$ to prove the lemma. For the base case, assume that $\phi(V(D)) = 1$ and there exists exactly one undecided vertex $x$. Note that the given recurrence relation holds in this case. Next assume that $\mathsf{KFVD}(D_1, \phi_1)$ and $\mathsf{KFVD}(D_2, \phi_2)$ compute the optimal solution correctly, say $S_1$ and $S_2$, respectively. Let $S$ be an optimal solution for $\mathsf{KFVD}(D, \phi)$. We consider the following two cases:

**Case 1: $x \in Z_S$.**

Here we claim that $S_1 \cup N^+(x)$ is an optimal solution for $\mathsf{KFVD}(D, \phi)$ if and only if $S_1$ is an optimal solution for $\mathsf{KFVD}(D_1, \phi_1)$. The arguments are exactly the same as that in Claim 6. And by construction, $Z_S \in \phi^{-1}(1)$ if and only if $Z_S \in \phi_1^{-1}(1)$ where $S' = S \setminus N^+(x)$. We also claim that $\mathsf{KFVD}(D_2, \phi_2) \geq \mathsf{KFVD}(D_1, \phi_1) + |N^+(x)|$. For contradiction suppose $\mathsf{KFVD}\ (D_2, \phi_2) < \mathsf{KFVD}(D_1, \phi_1) + |N^+(x)|$. Then any optimal solution $S_2$ for $(D_2, \phi_2)$ is also a feasible solution for $\mathsf{KFVD}(D, \phi)$. But $S_2$ has size strictly smaller than $S$, which is a contradiction. Hence, $\mathsf{KFVD}(D, \phi) = \min\{\mathsf{KFVD}\ (D_1, \phi) + |N^+(x)|,\ \mathsf{KFVD}(D, \phi_2)\}$.

**Case 2:** $x \notin Z_S$.

In this case, $\mathsf{KFVD}(D_2, \phi_2) = \mathsf{KFVD}(D, \phi)$, by definition. Also $\mathsf{KFVD}(D_2, \phi_2) \leq \mathsf{KFVD}(D_1, \phi_1) + |N^+(x)|$, otherwise we have $S' = S_1 \cup N^+(x)$ as a solution with size strictly smaller than $S$ for $\mathsf{KFVD}(D, \phi)$, where $x \in Z_{S'}$, and that is a contradiction. Hence, $\mathsf{KFVD}(D, \phi) = \min\{\mathsf{KFVD}(D_1, \phi) + |N^+(x)|,\ \mathsf{KFVD}(D, \phi_2)\}$. ◀

In Subroutine 1, we get a $(3.5, 0.5)$ branching. If Subroutine 1 is no longer applicable on the instance, the digraph has no undecided vertices with $\phi(R(x)) \geq 3.5$. This fact is crucial to obtaining desirable branching vectors for Subroutine 2 and Subroutine 3.

▶ Remark 12. After Subroutine 1 completion, there are no undecided vertices with degree more than 4.

## 4.2 Correctness of Subroutine 2

Subroutine 2 branches on any undecided vertex $x$ who has semi-decided neighbors only. Note that $x$ has at least one out-neighbor and at least one in-neighbor; otherwise, reduction rules 2 or 3 would have been applied.

▷ Claim 13. Let $D$ be a digraph such that $\phi(R(v)) < 3.5$, for every vertex $v \in V(D)$ and $x$ be an undecided vertex in $D$ such that for all $v \in N(x)$, $\phi(v) = 0.5$. If $x$ is a non-sink vertex in an optimal solution $S$ for $(D, \phi)$, then there exist vertices $y \in N^-(x)$ and $z \in N^+(x)$ such that there exists a unique $s \in N^-(y) \cap N^+(z)$ with $\phi(s) = 1$ and $s$ is a sink vertex in $D - S$.

Proof. If $x$ is not a sink vertex, then there exists a vertex $z \in N^+(x)$ such that it does not belong to $S$. Therefore, in digraph $D - S$ the vertex $z$ must reach to a sink $s$ i.e., there exists a path between $z$ and the undecided vertex $s$. It implies that the vertices $x,\ z,\ s$ belong to $R(s)$. Therefore, $\phi(R(s)) \geq 3.5$ as $\phi(x) = \phi(s) = 1$, which is not possible. It implies that $s \in N^+(z)$ with $\phi(s) = 1$. Since vertex $x$ must have one in-neighbor, say $y \in N^-(x)$. Next, $s$ can not have an out-neighbor outside the set $\{x, y, z\}$, otherwise $\phi(R(x)) \geq 3.5$ which is not possible. Moreover, $x$ or $z$ cannot be an out-neighbor of $s$ as $x$ has no undecided out-neighbor and $z$ does not belong to $S_{opt}$. Hence, $y \in N^+(s)$ which implies that $s \in R(x)$. Since $\phi(R(x)) < 3.5$, there does not exist any vertex $u \in N^-(y)$ except $s$ such that $\phi(u) = 1$. Hence, $s$ is a unique vertex such that $s \in N^-(y) \cap N^+(z)$ with $\phi(s) = 1$. ◁

Next, we prove the correctness of Subroutine 2.

▶ **Lemma 14.** *Let $D$ be a digraph such that $\phi(R(v)) < 3.5$, for every vertex $v \in V(D)$ and $x \in V(D)$ be such that $\phi(x) = 1$ and for all $v \in N(x)$, $\phi(v) = 0.5$. If $y \in N^-(x)$ and $z \in N^+(x)$, then*

$$\mathsf{KFVD}(D, \phi) = \begin{cases} \min\Big\{\mathsf{KFVD}(D_1, \phi_1) + |N^+(x)|, \\ \qquad \mathsf{KFVD}(D_2, \phi_2) + |N^+(s)|\Big\} & \text{if } \exists\ s \in N^-(y) \cap N^+(z) \text{ with } \phi(s) = 1, \\ \mathsf{KFVD}(D_1, \phi_1) + |N^+(x)| & \text{otherwise,} \end{cases}$$

*where* $D_1 = D - R(x)$, $D_2 = D - R(s)$ *and* $\phi_1 = \phi_2 = \phi$.

**Proof.** Let $S$, $S_1$ ans $S_2$ be the optimal solutions for $\mathsf{KFVD}(D, \phi)$, $\mathsf{KFVD}(D_1, \phi_1)$ and $\mathsf{KFVD}(D_2, \phi_2)$, respectively. We use induction on the total potential $\phi(V(D))$ of the digraph to prove the correctness of the above claim. Let $y \in N^-(x)$ and $z \in N^(x)$. Now, we consider the following two cases:

**Case 1:** $\exists s \in N^-(y) \cap N^+(z)$ such that $\phi(s) = 1$.

Assume that there exists $s \in N^-(y) \cap N^+(z)$ with $\phi(s) = 1$. From Claim 13, we know that either $x$ or $s$ will be a sink in an optimal solution. Let $x \in Z_S$, then from Claim 10, $|S| = \mathsf{KFVD}(D_1, \phi_1) + |N^+(x)|$. And $S_2 \cup N^+(s)$ is a feasible solution to $\mathsf{KFVD}(D, \phi)$ and hence $\mathsf{KFVD}(D_2, \phi_2) + |N^+(s)| \geq |S| = \mathsf{KFVD}(D_1, \phi_1) + |N^+(x)|$. Similarly, if $s \in Z(S)$ then from Claim 10, $|S| = \mathsf{KFVD}(D_2, \phi_2) + |N^+(s)|$. Note that $S_1 \cup N^+(x)$ is also a feasible solution to $\mathsf{KFVD}(D, \phi)$ and therefore, $\mathsf{KFVD}(D_1, \phi_1) + |N^+(x)| \geq |S| = \mathsf{KFVD}(D_2, \phi_2) + |N^+(s)|$. Hence, we have $\mathsf{KFVD}(D, \phi) = \min\{\mathsf{KFVD}(D_1, \phi_1) + |N^+(x)|, \mathsf{KFVD}(D_2, \phi_2) + |N^+(s)|\}$.

**Case 2:** $\nexists s \in N^-(y) \cap N^+(z)$ such that $\phi(s) = 1$.

Suppose there does not exist any vertex $s \in N^-(y) \cap N^+(z)$ such that $\phi(s) = 1$. Observe that in this case, the vertex $x$ has to be a sink; otherwise, by Claim 13 there exists a vertex $s \in N^-(y) \cap N^+(z)$ with $\phi(s) = 1$, which is a contradiction. Therefore, the vertex $x$ has to be a sink. Hence, $\mathsf{KFVD}(D, \phi) = \mathsf{KFVD}(D_1, \phi_1) + N^+(x)$ by Claim 10, where $D_1 = D - R(x)$. ◀

▶ Remark 15. If Subroutine 2 is no longer applicable, the instance has no undecided vertices with a degree more than 3.

## 4.3 Correctness of Subroutine 3

Subroutine 3 branches on undecided vertices with degrees 2 and 3. When Subroutine 0, 1, and 2 are no longer applicable, there are no vertices with all semi-decided neighbors. Also there is no undecided vertex $x$ with degree 4 as $\phi(R(x)) \geq 3.5$. So any undecided vertex has degree at most 3. Moreover, any undecided vertex $x$ of degree 3 has exactly one undecided vertex. We will prove the correctness of branching in Subroutine 3 in two parts. First, we analyze the branching vector for degree 3 undecided vertices and then for degree 2 undecided vertices.

### Branching on a degree 3 vertex

Given an undecided vertex $x$ of degree 3, there are two cases; either $x$ has one in-neighbor and two out-neighbors or $x$ has two in-neighbors and one out-neighbor.

#### Case 1: $x$ has one in-neighbor $y_1$ and two out-neighbors $z_1, z_2$

Out of three neighbors, $x$ has exactly one neighbor which is undecided. So we have the following 3 three subcases.

**Subcase 1:** $\phi(y_1) = 1$ and $\phi(z_1) = \phi(z_2) = 0.5$.

- If $x$ is a sink, there is a potential drop of at least 3 since $\{y_1, z_1, z_2, x\} \in R(x)$ .
- If $x$ is not a sink, then $x$ and some $z \in N^+(x)$ are not in the deletion set.Without loss of generality, let $z = z_1$. Now $z_1$ reaches a sink $s \in Z_{opt}$ in $D - S_{opt}$. If $z_1$ can reach a sink $s$ ($\neq y_1$), then $\{y_1, x, z_1, s\} \subseteq R(s)$. This is true since $s$ can not have $x$ or $z_1$ as its out-neighbor and either $y_1$ is an out-neighbor of $s$ or $y_1 \in R^-(x)$. It implies that

$\phi(R(s)) \geq 3.5$, which is a contradiction. Hence $y_1$ is the only sink that $z_1$ must reach. Note that $\phi(R(y_1)) \geq 2.5$.

This gives us a $(3, 2.5)$ branching vector.

**Subcase 2:** $\phi(y_1) = 0.5$, $\phi(z_1) = 1$ and $\phi(z_2) = 0.5$.

- If $x$ is a sink, the potential drops by at least 3.
- If $x$ is not a sink and $z_1$ is not in the deletion set, then $z_1$ reaches a sink say, $s$ in $Z_{opt}$ in $D - S_{opt}$. If $z_1$ can reach a sink $s$ ($\neq z_1$), then $\{y_1, x, z_1, s\} \subseteq R(s)$ as $s$ can not have $x$ , $z_1$ or $y_1$ as its out-neighbors. Therefore, $\phi(R(s)) \geq 3.5$ which is a contradiction. Therefore $z_1$ has to be a sink when $x$ is not a sink. Note that $\phi(R(z_1)) \geq 2.5$.

This gives us a $(3, 2.5)$ branching vector.

**Subcase 3:** $\phi(y_1) = \phi(z_1) = 0.5$ and $\phi(z_2) = 1$.

- if $x$ is a sink, again the potential drops by at least 3 in $(D_1, \phi_1)$.
- If $x$ is not a sink and $z_1$ is not in the deletion set, then $z_1$ reaches s in $Z_{opt}$ in $D - S_{opt}$. Suppose $z_1$ can reach a sink, say $s$ ($\neq z_2$). If $z_1$ has an out-neighbor outside $\{y_1, x, z_1\}$ then $\phi(R(s)) \geq 3.5$, which is a contradiction. Otherwise $y_1$ is its only out-neighbor. But then $\{s, y_1, x, z_1, x, z_2\} \subseteq R(x)$ and $\phi(R(s)) \geq 4$, which is not possible. Therefore, the only possibility for $z_2$ is to be the sink for $z_1$ when $x$ is not a sink. Note that $\phi(R(z_2)) \geq 2.5$.

Hence we have a $(3, 2.5)$ branching vector.

### Case 2: $x$ has two in-neighbors $y_1$, $y_2$ and one out-neighbor $z_1$

**Subcase 1:** $\phi(y_1) = \phi(y_2) = 0.5$ and $\phi(z) = 1$.

- If $x$ is a sink, there is a potential drop of at least 3.
- If $x$ is not a sink, then $z_1$ is not in the deletion set. Note that $z_1$ reaches a sink, say $s \in Z_{opt}$ in $D - S_{opt}$. If $z_1$ can reach a sink $s$ ($\neq z_1$), then $\{y_1, x, z_1, s\} \subseteq R(s)$. But then $\phi(R(s)) \geq 3.5$ which is a contradiction. The only possibility is for $z_1$ itself to be a sink when $x$ is not a sink. Note that $\phi(R(z_1)) \geq 2.5$.

Hence we have a $(3, 2.5)$ branching vector.

**Subcase 2:** $\phi(y_1) = 1$ and $\phi(y_2) = \phi(z_1) = 0.5$.

- If $x$ is a sink, the potential drops by 3.
- If $x$ is not a sink, then $z_1$ is not in the deletion set. Therefore, $z_1$ reaches a sink $s$. If there is a path from $z_1$ to $s$ not using $y_1, y_2$, then $\phi(R(s)) \geq 4$, which is not possible. Otherwise if there is a path from $z_1$ to $s$ via $y_1$ or $y_2$, then $\phi(R(s)) \geq 4$, which is again not possible. Hence, the only possibility is that $y_1$ becomes a sink such that $z_1$ reaches $y_1$. Note that in this case, the potential drops by at least 2.5.

Hence we have a $(3, 2.5)$ branching vector in this case.

## Branching on a degree 2 vertex

This subsection analyzes the potential drop while branching on a degree 2 undecided vertex $x$, which has an undecided neighbor. An undecided vertex $x$ of degree 2 has exactly one in-neighbor and one out-neighbor. Let $y \in N^-(x)$ and $z \in N^+(x)$. We consider the following three cases.

**Case 1:** $\phi(y) = 1$ and $\phi(z) = 0.5$.

- If $x$ is a sink, the potential drops by 2.5.
- If $x$ is not a sink, then $z$ is not in the deletion set and it reaches a sink $s$. If $s \neq y$, then $\{y, x, z, s\} \subseteq R(s)$ and $\phi(R(s)) \geq 3.5$, which is not possible. Hence, the only possibility is that $y$ is a sink that $z$ reaches. Note that here the potential drops by 2.5.

Hence, we get a $(2.5, 2.5)$ branching vector.

**Case 2:** $\phi(y) = 0.5$ and $\phi(z) = 1$.

- If $x$ is a sink, the potential drops by 2.5.
- If $x$ is not a sink, then $z$ is not in the deletion set. Therefore, $z$ reaches a sink, say $s$. If $s \neq y$ then $\{y, x, z, s\} \subseteq R(s)$ and $\phi(R(s)) \geq 3.5$, which is not possible. Hence, the only possibility is for $z$ to be a sink itself. Note that in this case, the potential drops by 2.5

Hence we have a $(2.5, 2.5)$ branching vector.

**Case 3:** $\phi(y) = 1$ and $\phi(z) = 1$.

- If $x$ is a sink, potential drops by 3.
- If $x$ is not a sink, then $z$ is not in the deletion set. Therefore, $z$ reaches a sink, say $s$. If $s \neq y$ and $s \neq z$, then $\{y, x, z, s\} \subseteq R(s)$ and $\phi(R(s)) \geq 3.5$, which is not possible. Hence, the only possibility is for $z$ to be a sink itself or to reach sink $y$. Note both in these cases, the potential drops by at least 3.

Hence we have a (3, 3, 3) branching vector.

Next, we give a formal proof of the correctness for Subroutine 3.

▶ **Lemma 16.** *Let $D$ be a digraph such that $\phi(R(v)) < 3.5$, for every vertex $v \in V(D)$ and $x \in V(D)$ be such that $d(x) \in \{2, 3\}$, $\phi(x) = 1$ and there exists some vertex $s \in N(x)$ such that $\phi(s) = 1$. Then* $\mathsf{KFVD}(D, \phi) = \min\{\mathsf{KFVD}(D_1, \phi) + |N^+(x)|,\ \mathsf{KFVD}(D_2, \phi) + N^+(s)\}$ *where $D_1 = D - R(x)$, $D_2 = D - R(s)$ and $\phi_1 = \phi_2 = \phi$.*

**Proof.** Notice that there is no vertex $v$ with $\phi(v) \geq 3.5$ and no vertex $v$ that has all its neighbors with potential 0.5. We use induction on the potential function to prove the correctness of the claim. Let $\mathsf{KFVD}(D_1, \phi_1)$ and $\mathsf{KFVD}(D_2, \phi_2)$ correctly compute optimal solution $(S_1, Z_1)$ and $(S_2, Z_2)$, where $Z_i \subseteq \phi^{-1}(1)$. Let $(S, Z)$ be an optimal solution for $\mathsf{KFVD}(D, \phi)$.

**Case 1:** $x \in Z$.

In this case, we know that $S_1 \cup N^+(x)$ is an optimal solution for $\mathsf{KFVD}(D, \phi)$ if and only if $S_1$ is an optimal solution for $\mathsf{KFVD}(D_1, \phi_1)$ and $|S_2 \cup N^+(s)| \geq |S_1 \cup N^+(x)|$, since it is also a solution to $\mathsf{KFVD}(D, \phi)$ that does not contain $x \in z$. Hence $\mathsf{KFVD}(D, \phi)$=$\min\{\mathsf{KFVD}(D_2, \phi_2) + |N^+(x)|,\ \mathsf{KFVD}(D_1, \phi_1) + |N^+(s)|\}$.

**Case 2:** $x \notin Z$.

By arguments made in the branching steps for degree 3 and 2 vertices, we can always find an $s$ that must be a sink in $Z$. And hence $S_2 \cup N^+(s)$ is an optimal solution for for $\mathsf{KFVD}(D, \phi)$ if and only if $S_1$ is an optimal solution for $\mathsf{KFVD}(D_2, \phi_2)$. $|S_1 \cup N^+(x)| \geq |S_2 \cup N^+(s)|$ since it is also a solution to $\mathsf{KFVD}(D, \phi)$ that does contain $x \in z$. Hence $\mathsf{KFVD}(D, \phi)$=$\min\{\mathsf{KFVD}(D_2, \phi_2) + |N^+(x)|,\ \mathsf{KFVD}(D_1, \phi_1) + |N^+(s)|\}$. ◀

## 5 Running time analysis

Reduction Rules 1, 2 and 3 are applied on the instance $(D, \phi)$ in $n^{\mathcal{O}(1)}$ time. In Subroutine 1, while branching we get a potential drop of at least 3.5 in one branch, while in the other branch the potential drop is at least 0.5. Hence, we get the recurrence $f(\mu) \leq f(\mu - 3.5) + f(\mu - .5)$, which solves to $f(\mu) = \mathcal{O}(1.576^{\mu})$. Similarly for Subroutine 2, we have a branching vector $(2, 3)$. Therefore, we have the recurrence $f(\mu) \leq f(\mu - 2) + f(\mu - 3)$, which solves to $f(\mu) = \mathcal{O}(1.324^{\mu})$. For Subroutine 3, we have branching vectors $(3, 2.5), (2.5, 2.5)$, and $(3, 3, 3)$, out of which $(3, 3, 3)$ gives the worst running time. This solves to $f(\mu) = \mathcal{O}(1.442^{\mu})$.

To solve the Knot-free Vertex Deletion, when we call KFVD$(D, \phi)$, potentials of all the vertices of $D$ are initialized to 1 and $\phi(V(D)) = |V(D)| = n$. Moreover, in the recursive calls to the algorithm, the potential of the input graph never drops below 0. Hence we can upper bound the running time of Algorithm KFVD by the worst-case running subroutine (Subroutine 1). Thus we obtain the following theorem.

▶ **Theorem 17.** *Algorithm* KFVD *solves* Knot-free vertex deletion *in* $\mathcal{O}^*(1.576^n)$ *time.*

## 6 A lower bound on the worst case running time of our algorithm

In this section, we give a lower bound on the worst-case running time of our algorithm.

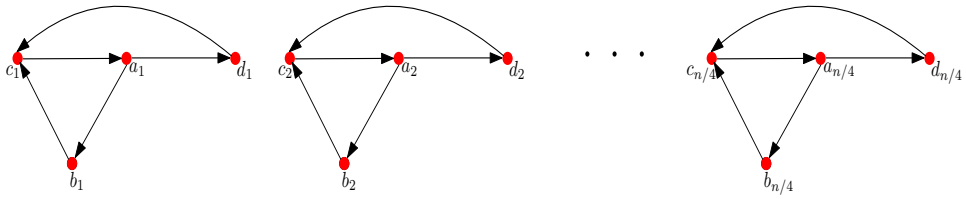



**Figure 1** Illustration of a worst-case instance for our algorithm.

We run our algorithm KFVD on the graph $D$ (shown in Figure 1), where $V(D) = \cup_{i=1}^{n/4}\{a_i, b_i, c_i, d_i\}$ and $E(D) = \cup_{i=1}^{n/4}\{(a_i, b_i), (b_i, c_i), (c_i, a_i), (a_i, d_i), (d_i, c_i)\}$. We claim that in the worst case, our algorithm takes $\mathcal{O} * (2^{n/2})$ time to solve KFVD on $D$. We will give this lower bound via adversarial arguments.

Let $V_i = \{a_i, b_i, c_i, d_i\}$ where $i \in [1, n/4]$. Since the potentials of all the vertices are initialized to 1, we have $\phi(R(a_i)) = 4$, $\phi(R(b_i)) = 3$, $\phi(R(c_i)) = 4$, $\phi(R(d_i)) = 3$. The adversary chooses the vertex $a_i$ for KFVD to branch on. If $a_i$ is a sink, all the four vertices are deleted. If $a_i$ is a non-sink vertex, its potential drops by 0.5 while all other vertices' potentials remain unchanged. In the next iteration, $\phi(R(c_i)) = 3.5$, so we branch on $c_i$. Again, if $c_i$ is a sink then all four vertices get deleted. If $c_i$ is not a sink, then its potential drops by 0.5. In the next step, $b_i$ and $d_i$ are vertices where all their neighbors are semi-decided. The adversary chooses $b_i$ to branch on. If $b_i$ becomes a sink all the vertices are again removed. Else, when $b_i$ is a non-sink $d_i$ has to be a sink and all vertices are removed. There are four leaves of the branching tree while branching on the set of vertices $\{a_i, b_i, c_i, d_i\}$. This gives us a recurrence equation: $T(n) = 4T(n-4)$ which solves to $4^{n/4} = 1.414^n$.

▶ **Theorem 18.** *Algorithm* KFVD *runs in time* $\Omega(1.414^n)$.

## 7 Conclusion

We obtain a $\mathcal{O}(1.576^n)$ time and polynomial space algorithm for KFVD problem. Notice that our algorithm is not optimal and its improvement is a suggested direction for future work. Also exact algorithms for KFVD with dependency on number of edges instead of vertices can be an interesting research topic.

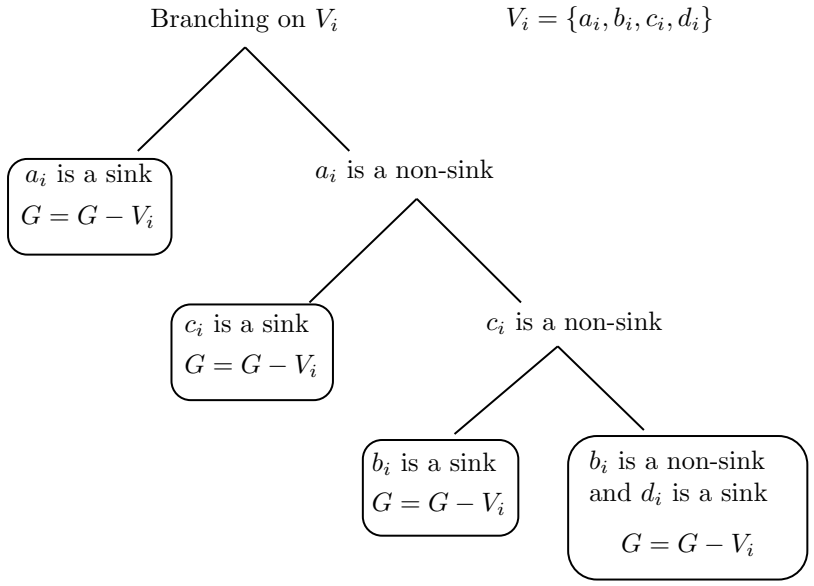



**Figure 2** Branching steps on $V_i$.

## References


1. Valmir Carneiro Barbosa, Alan Diêgo A. Carneiro, Fábio Protti, and Uéverton S. Souza. Deadlock models in distributed computation: foundations, design, and computational complexity. In Sascha Ossowski, editor, *Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, April 4-8, 2016*, pages 538–541. ACM, 2016. doi:10.1145/2851613.2851880.
2. Stéphane Bessy, Marin Bougeret, Alan Diêgo A. Carneiro, Fábio Protti, and Uéverton S. Souza. Width parameterizations for knot-free vertex deletion on digraphs. In Bart M. P. Jansen and Jan Arne Telle, editors, *14th International Symposium on Parameterized and Exact Computation, IPEC 2019, September 11-13, 2019, Munich, Germany*, volume 148 of *LIPIcs*, pages 2:1–2:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.IPEC.2019.2.
3. Andreas Björklund. Determinant sums for hamiltonicity (invited talk). In Jiong Guo and Danny Hermelin, editors, *11th International Symposium on Parameterized and Exact Computation, IPEC 2016, August 24-26, 2016, Aarhus, Denmark*, volume 63 of *LIPIcs*, pages 1:1–1:1. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.IPEC.2016.1.
4. Alan Diêgo Aurélio Carneiro, Fábio Protti, and Uéverton dos Santos Souza. On knot-free vertex deletion: Fine-grained parameterized complexity analysis of a deadlock resolution graph problem. *Theoretical Computer Science*, 909:97–109, 2022.
5. Alan Diêgo Aurélio Carneiro, Fábio Protti, and Uéverton S. Souza. Deletion graph problems based on deadlock resolution. In Yixin Cao and Jianer Chen, editors, *Computing and Combinatorics – 23rd International Conference, COCOON 2017, Hong Kong, China, August 3-5, 2017, Proceedings*, volume 10392 of *Lecture Notes in Computer Science*, pages 75–86. Springer, 2017. doi:10.1007/978-3-319-62389-4_7.
6. Alan Diêgo Aurélio Carneiro, Fábio Protti, and Uéverton S. Souza. Deadlock resolution in wait-for graphs by vertex/arc deletion. *J. Comb. Optim.*, 37(2):546–562, 2019. doi:10.1007/s10878-018-0279-5.
7. Jianer Chen, Yang Liu, Songjian Lu, Barry O'Sullivan, and Igor Razgon. A fixed-parameter algorithm for the directed feedback vertex set problem. *J. ACM*, 55(5):21:1–21:19, 2008. doi:10.1145/1411509.1411511.

**8** Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

**9** Fedor V. Fomin, Serge Gaspers, Daniel Lokshtanov, and Saket Saurabh. Exact algorithms via monotone local search. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 764–775. ACM, 2016. doi:10.1145/2897518.2897551.

**10** Fedor V. Fomin, Dieter Kratsch, and Gerhard J. Woeginger. Exact (exponential) algorithms for the dominating set problem. In *Proceedings of the 30th International Conference on Graph-Theoretic Concepts in Computer Science*, WG'04, pages 245–256, Berlin, Heidelberg, 2004. Springer-Verlag. doi:10.1007/978-3-540-30559-0_21.

**11** F.V. Fomin and D. Kratsch. *Exact Exponential Algorithms*. Texts in Theoretical Computer Science. An EATCS Series. Springer Berlin Heidelberg, 2010. URL: https://books.google.co.in/books?id=LXDe1XHJCYIC.

**12** Gordon Hoi. An improved exact algorithm for the exact satisfiability problem. *CoRR*, abs/2010.03850, 2020. arXiv:2010.03850.

**13** Kamal Jain, Mohammad Taghi Hajiaghayi, and Kunal Talwar. The generalized deadlock resolution problem. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings*, volume 3580 of *Lecture Notes in Computer Science*, pages 853–865. Springer, 2005. doi:10.1007/11523468_69.

**14** Carlos VGC Lima, Fábio Protti, Dieter Rautenbach, Uéverton S Souza, and Jayme L Szwarcfiter. And/or-convexity: a graph convexity based on processes and deadlock models. *Annals of Operations Research*, 264(1):267–286, 2018.

**15** Daniel Lokshtanov, M. S. Ramanujan, and Saket Saurabh. When recursion is better than iteration: A linear-time algorithm for acyclicity with few error vertices. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1916–1933. SIAM, 2018. doi:10.1137/1.9781611975031.125.

**16** Alane Marie de Lima and Renato Carmo. Exact algorithms for the graph coloring problem. *Revista de Informática Teórica e Aplicada*, 25:57, November 2018. doi:10.22456/2175-2745.80721.

**17** Fabiano de S Oliveira and Valmir C Barbosa. Revisiting deadlock prevention: A probabilistic approach. *Networks*, 63(2):203–210, 2014.