Single-Exponential FPT Algorithms for Enumerating Secluded $\mathcal{F}$-Free Subgraphs and Deleting to Scattered Graph Classes

Bart M. P. Jansen
Eindhoven University of Technology, The Netherlands

Jari J. H. de Kroon
Eindhoven University of Technology, The Netherlands

Michał Włodarczyk
University of Warsaw, Poland

Abstract

The celebrated notion of important separators bounds the number of small (S, T)-separators in a graph which are "farthest from S" in a technical sense. In this paper, we introduce a generalization of this powerful algorithmic primitive, tailored to undirected graphs, that is phrased in terms of *k-secluded* vertex sets: sets with an open neighborhood of size at most k.

In this terminology, the bound on important separators says that there are at most 4^k maximal k-secluded connected vertex sets C containing S but disjoint from T. We generalize this statement significantly: even when we demand that $G[C]$ avoids a finite set $\mathcal{F}$ of forbidden induced subgraphs, the number of such maximal subgraphs is $2^{\mathcal{O}(k)}$ and they can be enumerated efficiently. This enumeration algorithm allows us to make significant improvements for two problems from the literature.

Our first application concerns the Connected k-Secluded $\mathcal{F}$-free subgraph problem, where $\mathcal{F}$ is a finite set of forbidden induced subgraphs. Given a graph in which each vertex has a positive integer weight, the problem asks to find a maximum-weight connected k-secluded vertex set $C \subseteq V(G)$ such that $G[C]$ does not contain an induced subgraph isomorphic to any $F \in \mathcal{F}$. The parameterization by k is known to be solvable in triple-exponential time via the technique of recursive understanding, which we improve to single-exponential.

Our second application concerns the deletion problem to *scattered graph classes*. A scattered graph class is defined by demanding that every connected component is contained in at least one of the prescribed graph classes $\Pi_1, \ldots, \Pi_d$. The deletion problem to a scattered graph class is to find a vertex set of size at most k whose removal yields a graph from the class. We obtain a single-exponential algorithm whenever each class Π_i is characterized by a finite number of forbidden induced subgraphs. This generalizes and improves upon earlier results in the literature.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of computation → Graph algorithms analysis; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases fixed-parameter tractability, important separators, secluded subgraphs

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2023.42

Related Version *Full Version*: https://arxiv.org/abs/2309.11366 [24]

Funding This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 803421, ReduceSearch).

34th International Symposium on Algorithms and Computation (ISAAC 2023).
Editors: Satoru Iwata and Naonori Kakimura; Article No. 42; pp. 42:1–42:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Graph separations have played a central role in algorithmics since the discovery of min-cut/max-flow duality and the polynomial-time algorithm to compute a maximum flow [15]. Nowadays, more complex separation properties are crucial in the study of parameterized complexity, where the goal is to design algorithms for NP-hard problems whose running time can be bounded as $f(k) \cdot n^{\mathcal{O}(1)}$ for some function f that depends only on the *parameter* k of the input. There are numerous graph problems which either explicitly involve finding separations of a certain kind (such as Multiway Cut [33], Multicut [4, 36], k-Way Cut [25], and Minimum Bisection [11]) or in which separation techniques turn out to be instrumental for an efficient solution (such as Directed Feedback Vertex Set [7] and Almost 2-SAT [39]).

The field of parameterized complexity has developed a robust toolbox of techniques based on graph separators, e.g., treewidth reduction [35], important separators [34], shadow removal [36], discrete relaxations [12, 18, 19, 20], protrusion replacement [37], randomized contractions and recursive understanding [8, 10, 31], and flow augmentation [26, 27]. These powerful techniques allowed a large variety of graph separation problems to be classified as fixed-parameter tractable. However, this power comes at a cost. The running times for many applications of these techniques are superexponential: of the form $2^{p(k)} \cdot n^{\mathcal{O}(1)}$ for a high-degree polynomial p, double-exponential, or even worse. Discrete relaxations form a notable exception, which we discuss in Section 4.

The new algorithmic primitive we develop can be seen as an extension of important separators [34] [9, §8]. The study of important separators was pioneered by Marx [33, 34] and refined by follow-up work by several authors [6, 29], which was recognized by the EATCS-IPEC Nerode Prize 2020 [3]. The technique is used to bound the number of extremal (S,T)-separators in an n-vertex graph G with vertex sets S and T. The main idea is that, even though the number of distinct inclusion-minimal (S,T)-separators (which are vertex sets potentially intersecting $S \cup T$) of size at most k can be as large as $n^{\Omega(k)}$, the number of *important* separators which leave a maximal vertex set reachable from S, is bounded by 4^k. For Multiway Cut, a pushing lemma [33, Lem. 6] shows that there is always an optimal solution that contains an important separator, which leads to an algorithm solving the problem in time $2^{\mathcal{O}(k)} \cdot n^{\mathcal{O}(1)}$. Important separators also form a key ingredient for solving many other problems such as Multicut [4, 36] and Directed Feedback Vertex Set [7].

For our purposes, it will be convenient to view the bound on the number of important separators through the lens of *secluded subgraphs*.

▶ **Definition 1.** *A vertex set $S \subseteq V(G)$ or induced subgraph $G[S]$ of an undirected graph G is said to be k-*secluded *if $|N_G(S)| \leq k$, that is, the number of vertices outside S which are adjacent to a vertex of S is bounded by k.*

A vertex set S in a graph G is called seclusion-maximal *with respect to a certain property Π if S satisfies Π and for all sets $S' \supsetneq S$ that satisfy Π we have $|N_G(S')| > |N_G(S)|$.*

Hence a seclusion-maximal set with property Π is inclusion-maximal among all subsets with the same size neighborhood. Consequently, the number of inclusion-maximal k-secluded sets satisfying Π is at most the number of seclusion-maximal k-secluded sets with that property.

Using the terminology of seclusion-maximal subgraphs, the bound on the number of important (S,T)-separators of size at most k in a graph G is equivalent to the following statement: in the graph G' obtained from G by inserting a new source r adjacent to S, the number of *seclusion-maximal* k-secluded connected subgraphs C containing r but no vertex of T is bounded by 4^k. The neighborhoods of such subgraphs C correspond exactly to the important (S,T)-separators in G.

While a number of previously studied cut problems [30, 35] place further restrictions on the vertex set that forms the separator (for example, requiring it to induce a connected graph or independent set) our generalization instead targets the structure of the k-secluded connected subgraph C. We will show that, for any fixed finite family $\mathcal{F}$ of graphs, the number of k-secluded connected subgraphs C as above which are seclusion-maximal with respect to satisfying the additional requirement that $G[C]$ contains no induced subgraph isomorphic to a member of $\mathcal{F}$ is still bounded by $2^{\mathcal{O}(k)}$. Observe that the case $\mathcal{F} = \emptyset$ corresponds to the original setting of important separators. Note that a priori, it is not even clear that the number of seclusion-maximal graphs of this form can be bounded by any function $f(k)$, let alone a single-exponential one.

Our contribution

Having introduced the background of secluded subgraphs, we continue by stating our result exactly. This will be followed by a discussion on its applications.

For a finite set $\mathcal{F}$ of graphs we define $||\mathcal{F}|| := \max_{F \in \mathcal{F}} |V(F)|$, the maximum order of any graph in $\mathcal{F}$. We say that a graph is $\mathcal{F}$-free if it does not contain an *induced* subgraph isomorphic to a graph in $\mathcal{F}$. Our generalization of important separators is captured by the following theorem, in which we use $\mathcal{O}_{\mathcal{F}}(\dots)$ to indicate that the hidden constant depends on $\mathcal{F}$.

▶ **Theorem 2.** *Let $\mathcal{F}$ be a finite set of graphs. For any n-vertex graph G, non-empty vertex set $S \subseteq V(G)$, potentially empty $T \subseteq V(G) \setminus S$, and integer k, the number of k-secluded induced subgraphs $G[C]$ which are seclusion-maximal with respect to being connected, $\mathcal{F}$-free, and satisfying $S \subseteq C \subseteq V(G) \setminus T$, is bounded by $2^{\mathcal{O}_{\mathcal{F}}(k)}$. A superset of size $2^{\mathcal{O}_{\mathcal{F}}(k)}$ of these subgraphs can be enumerated in time $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n^{||\mathcal{F}||+\mathcal{O}(1)}$ and polynomial space.*

The single-exponential bound given by the theorem is best-possible in several ways. Existing lower bounds on the number of important separators [9, Fig. 8.5] imply that even when $\mathcal{F} = \emptyset$ the bound cannot be improved to $2^{o(k)}$. The term $n^{||\mathcal{F}||}$ in the running time is unlikely to be avoidable, since even testing whether a single graph is $\mathcal{F}$-free is equivalent to Induced Subgraph Isomorphism and cannot be done in time $n^{o(||\mathcal{F}||)}$ [9, Thm. 14.21] assuming the Exponential Time Hypothesis (ETH) due to lower bounds for k-Clique.

The polynomial space bound applies to the internal space usage of the algorithm, as the output size may be exponential in k. More precisely, we consider polynomial-space algorithms equipped with a command that outputs an element and we require that for each element in the enumerated set, this command is called at least once. The algorithm could also enumerate just the set in question (rather than its superset) by postprocessing the output and comparing each pair of enumerated subgraphs. However, storing the entire output requires exponential space.

By executing the enumeration algorithm for every singleton set S of the form $\{v\}$, $v \in V(G)$, and $T = \emptyset$, we immediately obtain the following.

▶ **Corollary 3.** *Let $\mathcal{F}$ be a finite set of graphs. For any n-vertex graph G and integer k, the number of k-secluded induced subgraphs $G[C]$ which are seclusion-maximal with respect to being connected and $\mathcal{F}$-free is $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n$. A superset of size $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n$ of these subgraphs can be enumerated in time $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n^{||\mathcal{F}||+\mathcal{O}(1)}$ and polynomial space.*

Note that we require that the set $\mathcal{F}$ of forbidden induced subgraphs is finite. This is necessary in order to obtain a bound independent of n in Theorem 2. For example, the number of seclusion-maximal $(k = 1)$-secluded connected subgraphs C containing a prescribed vertex r for which C induces an acyclic graph is already as large as $n-1$ in a graph consisting of a single cycle, since each way of omitting a vertex other than r gives such a subgraph. For this case, the forbidden induced subgraph characterization $\mathcal{F}$ consists of all cycles. Extending this example to a flower structure of k cycles of length n/k pairwise intersecting only in r shows that the number of seclusion-maximal k-secluded $\mathcal{F}$-free connected subgraphs containing r is $\Omega(n^k/k^k)$ and cannot be bounded by $f(k) \cdot n^{\mathcal{O}(1)}$ for any function f.

We give two applications of Theorem 2 to improve the running time of existing super-exponential (or even triple-exponential) parameterized algorithms to single-exponential, which is optimal under ETH. For each application, we start by presenting some context.

Application I: Optimization over connected k-secluded $\mathcal{F}$-free subgraphs

The computation of secluded versions of graph-theoretic objects such as paths [2, 5, 32], trees [13], Steiner trees [14], or feedback vertex sets [1], has attracted significant attention over recent years. This task becomes hard already for detecting k-secluded disconnected sets satisfying very simple properties. In particular, detecting a k-secluded independent set of size s is W[1]-hard when parameterized by $k + s$ [1].

Golovach, Heggernes, Lima, and Montealegre [17] suggested then to focus on *connected* k-secluded subgraphs and studied the problem of finding one, which belongs to a graph class $\mathcal{H}$, of maximum total weight. They therefore studied the Connected k-secluded $\mathcal{F}$-free subgraph problem for a finite family $\mathcal{F}$ of forbidden induced subgraphs. Given an undirected graph G in which each vertex v has a positive integer weight $w(v)$, and an integer k, the problem is to find a maximum-weight connected k-secluded vertex set C for which $G[C]$ is $\mathcal{F}$-free. They presented an algorithm based on recursive understanding to solve the problem in time $2^{2^{2^{\mathcal{O}_{\mathcal{F}}(k \log k)}}} \cdot n^{\mathcal{O}_{\mathcal{F}}(1)}$. We improve the dependency on k to single-exponential.

▶ **Corollary 4.** *For each fixed finite family $\mathcal{F}$, Connected k-secluded $\mathcal{F}$-free subgraph can be solved in time $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n^{||\mathcal{F}||+\mathcal{O}(1)}$ and polynomial space.*

This result follows directly from Corollary 3 since a maximum-weight k-secluded $\mathcal{F}$-free subgraph must be seclusion-maximal. Hence it suffices to check for each enumerated subgraph whether it is $\mathcal{F}$-free, and remember the heaviest one for which this is the case.

The parameter dependence of our algorithm for Connected k-secluded $\mathcal{F}$-free subgraph is optimal under ETH. This follows from an easy reduction from Maximum Independent Set, which cannot be solved in time $2^{o(n)}$ under ETH [9, Thm. 14.6]. Finding a maximum independent set in an n-vertex graph G is equivalent to finding a maximum-weight triangle-free connected induced $(k = n)$-secluded subgraph in the graph G' that is obtained from G by inserting a universal vertex of weight n and setting the weights of all other vertices to 1. Consequently, an algorithm with running time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ for Connected k-secluded triangle-free induced subgraph would violate ETH and our parameter dependence is already optimal for $\mathcal{F} = \{K_3\}$.

Application II: Deletion to scattered graph classes

When there are several distinct graph classes (e.g., split graphs and claw-free graphs) on which a problem of interest (e.g. Vertex Cover) becomes tractable, it becomes relevant to compute a minimum vertex set whose removal ensures that each resulting component belongs to one such tractable class. This can lead to fixed-parameter tractable algorithms for solving the original problem on inputs which are *close* to such so-called *islands of tractability* [16]. The corresponding optimization problem has been coined the deletion problem to *scattered* graph classes [21, 23]. Jacob, Majumdar, and Raman [22] (later joined by de Kroon for the journal version [21]) consider the $(\Pi_1, \ldots, \Pi_d)$-deletion problem; given hereditary graph classes $\Pi_1, \ldots, \Pi_d$, find a set $X \subseteq V(G)$ of at most k vertices such that each connected component of $G - X$ belongs to Π_i for some $i \in [d]$. Here d is seen as a constant. When the set of forbidden induced subgraphs $\mathcal{F}_i$ of Π_i is finite for each $i \in [d]$, they show [21, Lem. 12] that the problem is solvable in time $2^{q(k)+1} \cdot n^{\mathcal{O}_\Pi(1)}$, where $q(k) = 4k^{10(pd)^2+4} + 1$. Here p is the maximum number of vertices of any forbidden induced subgraph.

Using Theorem 2 as a black box, we obtain a single-exponential algorithm for this problem.

▶ **Theorem 5.** *$(\Pi_1, \ldots, \Pi_d)$-DELETION can be solved in time $2^{\mathcal{O}_\Pi(k)} \cdot n^{\mathcal{O}_\Pi(1)}$ and polynomial space when each graph class Π_i is characterized by a finite set $\mathcal{F}_i$ of (not necessarily connected) forbidden induced subgraphs.*

The main idea behind the algorithm is the following. For an arbitrary vertex v, either it belongs to the solution, or we may assume that in the graph that results by removing the solution, the vertex v belongs to a connected component that forms a seclusion-maximal connected k-secluded $\mathcal{F}_i$-free induced subgraph of G for some $i \in [d]$. Branching on each of the $2^{\mathcal{O}_\Pi(k)}$ options gives the desired running time by exploiting the fact that in most recursive calls, the parameter decreases by more than a constant (cf. [9, Thm. 8.19]). Prior to our work, single-exponential algorithms were only known for a handful of ad-hoc cases where $d = 2$, such as deleting to a graph in which each component is a tree or a clique [21], or when one of the sets of forbidden induced subgraphs $\mathcal{F}_i$ contains a path.

Similarly as our first application, the resulting algorithm for $(\Pi_1, \ldots, \Pi_d)$-DELETION is ETH-tight: the problem is a strict generalization of k-VERTEX COVER, which is known not to admit an algorithm with running time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ unless ETH fails.

Techniques

The proof of Theorem 2 is based on a bounded-depth search tree algorithm with a nontrivial progress measure. By adding vertices to S or T in branching steps of the enumeration algorithm, the sets grow and the size of a minimum (S,T)-separator increases accordingly. The size of a minimum (S,T)-separator disjoint from S is an important progress measure for the algorithm: if it ever exceeds k, there can be no k-secluded set containing all of S and none of T and therefore the enumeration is finished.

The branching steps are informed by the farthest minimum (S,T)-separator (see Lemma 9), similarly as the enumeration algorithm for important separators, but are significantly more involved because we have to handle the forbidden induced subgraphs. A distinctive feature of our algorithm is that the decision made by branching can be to add certain vertices to the set T, while the important-separator enumeration only branches by enriching S. A key step is to use submodularity to infer that a certain vertex set is contained in *all* seclusion-maximal secluded subgraphs under consideration when other branching steps are inapplicable.

As an illustrative example consider the case $\mathcal{F} = \{K_3\}$, that is, we want to enumerate seclusion-maximal vertex sets $C \subseteq V(G) \setminus T$, $C \supseteq S$, which induce connected triangle-free subgraphs with at most k neighbors. Let $\lambda^{\mathrm{L}}(S,T)$ denote the size of a minimum vertex set disjoint from S that separates T from S – we will refer to such separators as *left-restricted*. Then $\lambda^{\mathrm{L}}(S,T)$ corresponds to the minimum possible size of $N(C)$. Similarly to the enumeration algorithm for important separators, we keep track of two measures: (M1) the value of k, and (M2) the gap between k and $\lambda^{\mathrm{L}}(S,T)$. We combine them into a single progress measure which is bounded by $2k$ and decreases during branching.

The first branching scenario occurs when there is some triangle in the graph G which intersects or is adjacent to S; then we guess which of its vertices should belong to $N(C)$, remove it from the graph, and decrease k by one. Otherwise, let $\mathcal{U} = \{U_1, \ldots U_d\}$ be the collection of all vertex sets of triangles in G (which are now disjoint from S). When there exists a triangle U_i whose addition to T increases the value $\lambda^{\mathrm{L}}(S,T)$, we branch into two

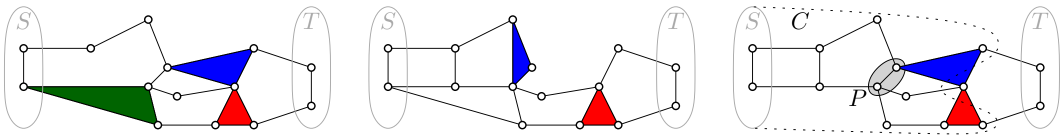

Figure 1 Illustration of the branching steps for enumerating triangle-free k-secluded subgraphs for $k = 3$. Left: the green triangle intersects S; we branch to guess which vertex belongs to $N(C)$. Middle: setting where $2 = \lambda^{\mathrm{L}}(S,T) < \lambda^{\mathrm{L}}(S,T \cup V(\mathcal{U})) = 3$; adding the top triangle to T increases λ^{L}. The set $\mathcal{U}$ consists of the colored triangles. Right: setting where $\lambda^{\mathrm{L}}(S,T) = \lambda^{\mathrm{L}}(S,T \cup V(\mathcal{U})) = 2$, with a corresponding farthest separator P. In this case every seclusion-maximal triangle-free set $C \supseteq S$ must be a superset of the reachability set of S in $G - P$.

possibilities: either U_i is disjoint from $N[C]$ – then we set $T \leftarrow T \cup U_i$ so the measure (M2) decreases – or U_i intersects $N(C)$ – then we perform branching as above. We show that in the remaining case all the triangles are separated from S by the minimum left-restricted (S,T)-separator closest to S; hence the value of $\lambda^{\mathrm{L}}(S,T)$ equals the value of $\lambda^{\mathrm{L}}(S,T \cup V(\mathcal{U}))$. Next, let P be the farthest minimum left-restricted $(S,T \cup V(\mathcal{U}))$-separator; we use submodularity to justify that we can now safely add to S all the vertices reachable from S in $G - P$. This allows us to assume that when $u \in P$ then either $u \in N(C)$ or $u \in C$, which leads to the last branching strategy. We either delete u (so k drops) or add u to S; note that in this case the progress measure may not change directly. The key observation is that adding u to S invalidates the farthest $(S,T \cup V(\mathcal{U}))$-separator P and now we are promised to make progress in the very next branching step. The different branching scenarios are illustrated in Figure 1.

The only property of K_3 that we have relied on is connectivity: if a triangle intersects a triangle-free set C then it must intersect $N(C)$ as well. This is no longer true when $\mathcal{F}$ contains a disconnected graph. For example, the forbidden family for the class of split graphs includes $2K_2$. A subgraph of $F \in \mathcal{F}$ that can be obtained by removing some components from F is called a *partial forbidden graph*. We introduce a third measure to keep track of how many different partial forbidden graphs appear as induced subgraph in $G[S]$. The main difficulty in generalizing the previous approach lies in justification of the greedy argument: when P is a farthest minimum separator between S and a certain set then we want to replace S with the set S' of vertices reachable from S in $G - P$. In the setting of connected obstacles this fact could be proven easily because S' was disjoint from all the obstacles. The problem is now it may contain some partial forbidden subgraphs. We handle this issue by defining P in such a way that the sets of partial forbidden graphs appearing in $G[S]$ and $G[S']$ are the same and giving a rearrangement argument about subgraph isomorphisms. This allows us to extend the analysis to any family $\mathcal{F}$ of forbidden subgraphs.

Organization. We begin with formal preliminaries in Section 2, including proofs of several properties of extremal separators. Next, we present the algorithm for enumerating secluded $\mathcal{F}$-free subgraphs in Section 3 and conclude in Section 4. The proofs of claims indicated with (★) can be found in the full version of the article [24]. The applications of the main theorem are discussed in the full version as well.

2 Preliminaries

Graphs and separators

We consider finite, simple, undirected graphs. We denote the vertex and edge sets of a graph G by $V(G)$ and $E(G)$ respectively, with $|V(G)| = n$ and $|E(G)| = m$. For a set of vertices $S \subseteq V(G)$, by $G[S]$ we denote the graph induced by S. We use shorthand $G - v$

and $G - S$ for $G[V(G) \setminus \{v\}]$ and $G[V(G) \setminus S]$, respectively. The open neighborhood $N_G(v)$ of $v \in V(G)$ is defined as $\{u \in V(G) \mid \{u, v\} \in E(G)\}$. The closed neighborhood of v is $N_G[v] = N_G(v) \cup \{v\}$. For $S \subseteq V(G)$, we have $N_G[S] = \bigcup_{v \in S} N_G[v]$ and $N_G(S) = N_G[S] \setminus S$. The set C is called connected if the graph $G[C]$ is connected.

We proceed by introducing notions concerning separators which are crucial for the branching steps of our algorithms. For two sets $S, T \subseteq V(G)$ in a graph G, a set $P \subseteq V(G)$ is an unrestricted (S,T)-separator if no connected component of $G - P$ contains a vertex from both $S \setminus P$ and $T \setminus P$. Note that such a separator may intersect $S \cup T$. Equivalently, P is an (S,T)-separator if each (S,T)-path contains a vertex of P. A restricted (S,T)-separator is an unrestricted (S,T)-separator P which satisfies $P \cap (S \cup T) = \emptyset$. A left-restricted (S,T)-separator is an unrestricted (S,T)-separator P which satisfies $P \cap S = \emptyset$. Let $\lambda^{\mathrm{L}}_G(S,T)$ denote the minimum size of a left-restricted (S,T)-separator, or $+\infty$ if no such separator exists (which happens when $S \cap T \neq \emptyset$).

▶ **Theorem 6** (Ford-Fulkerson). *There is an algorithm that, given an n-vertex m-edge graph $G = (V, E)$, disjoint sets $S, T \subseteq V(G)$, and an integer k, runs in time $\mathcal{O}(k(n + m))$ and determines whether there exists a restricted (S,T)-separator of size at most k. If so, then the algorithm returns a separator of minimum size.*

By the following observation we can translate properties of restricted separators into properties of left-restricted separators.

▶ **Observation 7.** *Let G be a graph and $S, T \subseteq V(G)$. Consider the graph G' obtained from G by adding a new vertex t adjacent to each $v \in T$. Then $P \subseteq V(G)$ is a left-restricted (S,T)-separator in G if and only if P is a restricted (S,t)-separator in G'.*

Extremal separators and submodularity

The following submodularity property of the cardinality of the open neighborhood is well-known; cf. [40, §44.12] and [28, Fn. 3].

▶ **Lemma 8** (Submodularity). *Let G be a graph and $A, B \subseteq V(G)$. Then the following holds:*

$$|N_G(A)| + |N_G(B)| \geq |N_G(A \cap B)| + |N_G(A \cup B)|.$$

For a graph G and vertex sets $S, P \subseteq V(G)$, we denote by $R_G(S, P)$ the set of vertices which can be reached in $G - P$ from at least one vertex in the set $S \setminus P$.

▶ **Lemma 9.** *Let G be a graph and $S, T \subseteq V(G)$ be two disjoint non-adjacent vertex sets. There exist minimum restricted (S,T)-separators P^- (closest) and P^+ (farthest), such that for each minimum restricted (S,T)-separator P, it holds that $R_G(S, P^-) \subseteq R_G(S, P) \subseteq R_G(S, P^+)$. Moreover, if a minimum restricted (S,T)-separator has size k, then P^- and P^+ can be identified in $\mathcal{O}(k(n + m))$ time.*

Proof. It is well-known (cf. [9, Thm. 8.5] for the edge-based variant of this statement, or [28, §3.2] for the same concept with slightly different terminology) that the existence of these separators follows from submodularity (Lemma 8), while they can be computed by analyzing the residual network when applying the Ford-Fulkerson algorithm to compute a minimum separator. We sketch the main ideas for completeness.

By merging S into a single vertex s^+ and merging T into a single vertex t^-, which is harmless because a restricted separator is disjoint from $S \cup T$, we may assume that S and T are singletons. Transform G into an edge-capacitated directed flow network D in which s^+ is the source and t^- is the sink. All remaining vertices $v \in V(G) \setminus (S \cup T)$ are split into two representatives v^-, v^+ connected by an arc (v^-, v^+) of capacity 1. For each edge $uv \in E(G)$ with $u, v \in V(G) \setminus \{s^+, t^-\}$ we add arcs $(u^+, v^-), (u^-, v^+)$ of capacity 2. For edges of the form s^+v we add an arc (s^+, v^-) of capacity 2 to D. Similarly, for edges of the form t^-v we add an arc (v^+, t^-) of capacity 2. Then the minimum size k of a restricted (S, T)-separator in G equals the maximum flow value in the constructed network, which can be computed by k rounds of the Ford-Fulkerson algorithm. Each round can be implemented to run in time $\mathcal{O}(n + m)$. From the state of the residual network when Ford-Fulkerson terminates we can extract P^- and P^+ as follows: the set P^- contains all vertices $v \in V(G) \setminus (S \cup T)$ for which the source can reach v^- but not v^+ in the final residual network. Similarly, P^+ contains all vertices $v \in V(G) \setminus (S \cup T)$ for which v^+ can reach the sink but v^- cannot. ◀

By Observation 7, we can apply the lemma above for left-restricted separators too; when the sets S, T are disjoint, then S is non-adjacent to t in the graph obtained by adding a vertex t adjacent to every vertex in T.

The extremal separators identified in Lemma 9 explain when adding a vertex to S or T increases the separator size. The following statement is not symmetric because we work with the non-symmetric notion of a left-restricted separator.

▶ **Lemma 10.** *Let G be a graph, let S, T be disjoint vertex sets, and let P^- and P^+ be the closest and farthest minimum* left-restricted *(S, T)-separators. Then for any vertex $v \in V(G)$, the following holds:*

1. *$\lambda^{\mathrm{L}}_G(S \cup \{v\}, T) > \lambda^{\mathrm{L}}_G(S, T)$ if and only if $v \in R_G(T, P^+) \cup P^+$.*
2. *$\lambda^{\mathrm{L}}_G(S, T \cup \{v\}) > \lambda^{\mathrm{L}}_G(S, T)$ if and only if $v \in R_G(S, P^-)$.*

Proof. Adding a vertex to S or T can never decrease the separator size, so for both cases, the left-hand side is either equal to or strictly greater than the right-hand side.

(1). Observe that if $v \notin R_G(T, P^+) \cup P^+$, then P^+ is also a left-restricted $(S \cup \{v\}, T)$-separator which implies $\lambda^{\mathrm{L}}_G(S \cup \{v\}, T) = \lambda^{\mathrm{L}}_G(S, T)$. If $v \in T$, then (1) holds as $\lambda^{\mathrm{L}}_G(S \cup \{v\}, T) = +\infty$. Consider now $v \in (R_G(T, P^+) \cup P^+) \setminus T$; we argue that adding it to S increases the separator size. Assume for a contradiction that there exists a minimum left-restricted $(S \cup \{v\}, T)$-separator P of size at most $\lambda^{\mathrm{L}}_G(S, T) = |P^+|$. Note that since P is left-restricted, we have $v \notin P$. Observe that P is also a left-restricted (S, T)-separator. By Lemma 9 we have $R_G(S, P) \subseteq R_G(S, P^+)$. Since $v \in (R_G(T, P^+) \cup P^+) \setminus T$, it follows that $v \notin R_G(S, P)$. We do a case distinction on v to construct a path Q from v to T.

- In the case that $v \in P^+ \setminus T$, then since P^+ is a minimum separator it must be inclusion-minimal. Therefore, since $P^+ \setminus \{v\}$ is not an (S, T)-separator, it follows that v has a neighbor in $R_G(T, P^+)$ and so there is a path Q from v to T in the graph induced by $R_G(T, P^+) \cup \{v\}$ such that $V(Q) \cap P^+ = \{v\}$.
- In the case that $v \in R_G(T, P^+) \setminus T$, then by definition there is a path from v to T in the graph induced by $R_G(T, P^+)$.

Since P is a left-restricted $(S \cup \{v\}, T)$-separator and therefore $v \notin P$, it follows that P contains at least one vertex $u \in V(Q)$ that is not in $R_G(S, P^+) \cup P^+$. Let P' be the set of vertices adjacent to $R_G(S, P)$. Since all vertices of P' belong to P while $u \notin P'$, it follows that P' is a left-restricted (S, T)-separator that is strictly smaller than P, a contradiction to $|P| \leq \lambda^{\mathrm{L}}_G(S, T)$.

(2). If $v \notin R_G(S, P^-)$, then P^- is a left-restricted $(S, T \cup \{v\})$-separator as well which implies $\lambda^{\mathrm{L}}_G(S, T \cup \{v\}) = \lambda^{\mathrm{L}}_G(S, T)$. If $v \in R_G(S, P^-)$, suppose that there exists a minimum left-restricted $(S, T \cup \{v\})$-separator P of size $|P^-|$. Note that $v \notin S$, as otherwise no such separator exists. Furthermore P is also a left-restricted (S, T)-separator. By Lemma 9 we have $R_G(S, P^-) \subseteq R_G(S, P)$. But since $v \notin R_G(S, P)$ we reach a contradiction as $R_G(S, P) \not\supseteq R_G(S, P^-)$. ◀

The following lemma captures the idea that if $\lambda^{\mathrm{L}}_G(S, T \cup Z) > \lambda^{\mathrm{L}}_G(S, T)$, then there is a single vertex from Z whose addition to T already increases the size of a minimum left-restricted (S, T)-separator. We will use it to argue that when it is cheaper to separate S from T than to separate S from T together with all obstacles of a certain form, then there is already a single vertex from one such obstacle which causes this increase.

▶ **Lemma 11.** *Let G be a graph, $S \subseteq V(G)$, and $T, Z \subseteq V(G) \setminus S$. If there is no vertex $v \in Z$ such that $\lambda^{\mathrm{L}}_G(S, T \cup \{v\}) > \lambda^{\mathrm{L}}_G(S, T)$, then $\lambda^{\mathrm{L}}_G(S, T) = \lambda^{\mathrm{L}}_G(S, T \cup Z)$. Furthermore if $\lambda^{\mathrm{L}}_G(S, T) \leq k$, then in $\mathcal{O}(k(n + m))$ time we can either find such a vertex v or determine that no such vertex exists.*

Proof. Let P^- be the minimum left-restricted (S, T)-separator which is closest to S. If for every $v \in Z$ the value of $\lambda^{\mathrm{L}}_G(S, T \cup \{v\})$ equals $\lambda^{\mathrm{L}}_G(S, T)$ then Lemma 10 implies that each $v \in Z$ lies outside $R_G(S, P^-)$ so $Z \cap R_G(S, P^-) = \emptyset$. Then P^- is a left-restricted $(S, T \cup Z)$-separator of size $\lambda^{\mathrm{L}}_G(S, T)$.

On the other hand, if there is a vertex $v \in Z$ for which $\lambda^{\mathrm{L}}_G(S, T \cup \{v\}) > \lambda^{\mathrm{L}}_G(S, T)$ then $v \in R_G(S, P^-)$. Hence, in order to detect such a vertex it suffices to compute the closest minimum left-restricted (S, T)-separator P^-, which can be done in time $\mathcal{O}(k(n + m))$ via Lemma 9. ◀

Finally, the last lemma of this section uses submodularity to argue that the neighborhood size of a vertex set C with $S \subseteq C \subseteq V(G) \setminus T$ does not increase when taking its union with the reachable set $R_G(S, P)$ with respect to a minimum left-restricted (S, T)-separator P.

▶ **Lemma 12.** *If $P \subseteq V(G)$ is a minimum left-restricted (S, T)-separator in a graph G and $S' = R_G(S, P)$, then for any set C with $S \subseteq C \subseteq V(G) \setminus T$ we have $|N_G(C \cup S')| \leq |N_G(C)|$.*

Proof. Observe that since P is a minimum left-restricted (S, T)-separator, we have $|P| = \lambda^{\mathrm{L}}_G(S, T)$ and $P = N_G(S')$. We apply the submodular inequality to the sets C and S'.

$$|N_G(C)| + |N_G(S')| \geq |N_G(C \cup S')| + |N_G(C \cap S')| \geq |N_G(C \cup S')| + \lambda^{\mathrm{L}}_G(S, T).$$

Here the last step comes from the fact that $S \subseteq S' \subseteq V(G) \setminus T$ since it is the set reachable from S with respect to a left-restricted (S, T)-separator, so that $C \cap S'$ contains all of S and is disjoint from T. This implies that $N_G(C \cap S')$ is a left-restricted (S, T)-separator, so that $|N_G(C \cap S')| \geq \lambda^{\mathrm{L}}_G(S, T)$.

As $|N_G(S')| = |P| = \lambda^{\mathrm{L}}_G(S, T)$, canceling these terms from both sides gives $|N_G(C)| \geq |N_G(C \cup S')|$ which completes the proof. ◀

3 The enumeration algorithm

We need the following concept to deal with forbidden subgraphs which may be disconnected.

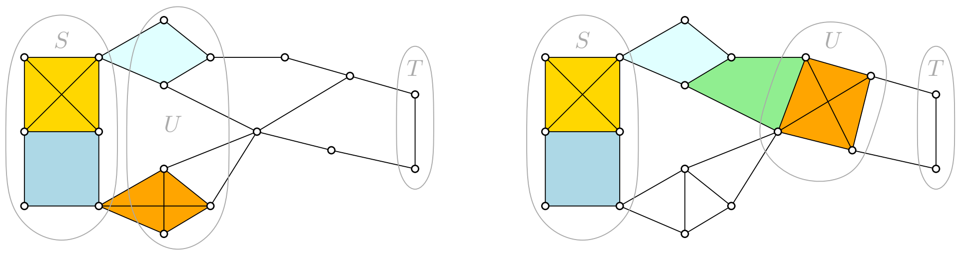

Figure 2 Illustration of the idea of enrichment and the branching steps in the proof of Theorem 2. Here $F = C_4 \uplus K_4$. Left: The graph $G[S]$ contains C_4 and K_4, but not F. The set U enriches S since $G[S \cup U]$ contains a new partial forbidden graph F. Every component of $G[U]$ is adjacent to S, so Step 3 applies. Right: The two top copies of C_4 do not enrich S. One of them intersects the only copy of K_4 in $G[S]$; the other one is adjacent to the only copy of K_4, while F has to appear as an induced subgraph. However the connected set U enriches S and it gets detected in Step 4. In both cases the enrichments are tight.

▶ **Definition 13.** *A partial forbidden graph F' is a graph obtained from some $F \in \mathcal{F}$ by deleting zero or more connected components. (So each $F \in \mathcal{F}$ itself is also considered a partial forbidden graph.)*

We use the following notation to work with induced subgraph isomorphisms. An induced subgraph isomorphism from H to G is an injection $\phi\colon V(H) \to V(G)$ such that for all distinct $u, v \in V(H)$ we have $\{u, v\} \in E(H)$ if and only if $\{\phi(u), \phi(v)\} \in E(G)$. For a vertex set $U \subseteq V(H)$ we let $\phi(U) := \{\phi(u) \mid u \in U\}$. For a subgraph H' of H we write $\phi(H')$ instead of $\phi(V(H'))$.

The following definition will be important to capture the progress of the recursive algorithm. See Figure 2 for an illustration.

▶ **Definition 14.** *We say that a vertex set $U \subseteq V(G)$ enriches a vertex set $S \subseteq V(G)$ with respect to $\mathcal{F}$ if there exists a partial forbidden graph F' such that $G[S \cup U]$ contains an induced subgraph isomorphic to F' but $G[S]$ does not. We call such a set U an enrichment.*

An enrichment U is called tight if $U = \phi(F') \setminus S$ for some induced subgraph isomorphism $\phi\colon V(F') \to V(G)$ from some partial forbidden graph F' for which $G[S]$ does not contain an induced subgraph isomorphic to F'.

The following observation will be used to argue for the correctness of the recursive scheme. Note that we get an implication only in one way (being seclusion-maximal in G implies being seclusion-maximal in $G - v$, not the other way around), which is the reason why we output a superset of the sought set in Theorem 2.

▶ **Observation 15.** *Let G be a graph containing disjoint sets $S, T \subseteq V(G)$ and let $C \subseteq V(G)$ be seclusion-maximal with respect to being connected, $\mathcal{F}$-free, k-secluded and satisfying $S \subseteq C \subseteq V(G) \setminus T$. For each $v \in N_G(C)$ it holds that C is seclusion-maximal in $G - v$ with respect to being connected, $\mathcal{F}$-free, $(k-1)$-secluded and satisfying $S \subseteq C \subseteq V(G - v) \setminus T$.*

With these ingredients, we present the enumeration algorithm. Recall that $||\mathcal{F}|| = \max_{F \in \mathcal{F}} |V(F)|$ denotes the maximum order of any graph in $\mathcal{F}$.

▶ **Theorem 2.** *Let $\mathcal{F}$ be a finite set of graphs. For any n-vertex graph G, non-empty vertex set $S \subseteq V(G)$, potentially empty $T \subseteq V(G) \setminus S$, and integer k, the number of k-secluded induced subgraphs $G[C]$ which are seclusion-maximal with respect to being connected, $\mathcal{F}$-free, and satisfying $S \subseteq C \subseteq V(G) \setminus T$, is bounded by $2^{\mathcal{O}_{\mathcal{F}}(k)}$. A superset of size $2^{\mathcal{O}_{\mathcal{F}}(k)}$ of these subgraphs can be enumerated in time $2^{\mathcal{O}_{\mathcal{F}}(k)} \cdot n^{||\mathcal{F}||+\mathcal{O}(1)}$ and polynomial space.*

Proof. Algorithm $\mathsf{Enum}_{\mathcal{F}}(G, S, T, k)$ solves the enumeration task as follows.

1. Stop the algorithm if one of the following holds:
   a. $\lambda^{\mathrm{L}}_G(S, T) > k$,
   b. the vertices of S are not contained in a single connected component of G, or
   c. the graph $G[S]$ contains an induced subgraph isomorphic to some $F \in \mathcal{F}$.
   There are no secluded subgraphs satisfying all imposed conditions.
2. If the connected component C of G which contains S is $\mathcal{F}$-free and includes no vertex of T: output C and stop.
   Component C is the unique seclusion-maximal one satisfying the imposed conditions.
3. If there is a vertex set $U \subseteq V(G) \setminus (S \cup T)$ such that:
   - each connected component of $G[U]$ is adjacent to a vertex of S, and
   - the set U is a tight enrichment of S with respect to $\mathcal{F}$ (so $G[S \cup U]$ contains a new partial forbidden graph)

   then execute the following calls and stop:
   a. For each $u \in U$ call $\mathsf{Enum}_{\mathcal{F}}(G - u, S, T, k - 1)$.
   b. Call $\mathsf{Enum}_{\mathcal{F}}(G, S \cup U, T, k)$.
   A tight enrichment can have at most $||\mathcal{F}||$ vertices which bounds the branching factor in Step 3a. Note that these are exhaustive even though we do not consider adding U to T: since each component of $G[U]$ is adjacent to a vertex of S, if a relevant secluded subgraph does not contain all of U then it contains some vertex of U in its neighborhood and we find it in Step 3a.
4. For the rest of the algorithm, let $\mathcal{U}$ denote the collection of all connected vertex sets $U \subseteq V(G) \setminus (S \cup T)$ which form tight enrichments of S with respect to $\mathcal{F}$. Let $V(\mathcal{U}) := \bigcup_{U \in \mathcal{U}} U$.
   a. If $\lambda^{\mathrm{L}}_G(S, T) < \lambda^{\mathrm{L}}_G(S, T \cup V(\mathcal{U}))$: then (using Lemma 11) there exists $U \in \mathcal{U}$ such that $\lambda^{\mathrm{L}}_G(S, T \cup U) > \lambda^{\mathrm{L}}_G(S, T)$, execute the following calls and stop:
      i. For each $u \in U$ call $\mathsf{Enum}_{\mathcal{F}}(G - u, S, T, k - 1)$. (The value of k decreases.)
      ii. Call $\mathsf{Enum}_{\mathcal{F}}(G, S \cup U, T, k)$. (We absorb a new partial forbidden graph.)
      iii. Call $\mathsf{Enum}_{\mathcal{F}}(G, S, T \cup U, k)$. (The separator size increases.)
   b. If $\lambda^{\mathrm{L}}_G(S, T) = \lambda^{\mathrm{L}}_G(S, T \cup V(\mathcal{U}))$, then let P be the farthest left-restricted minimum $(S, T \cup V(\mathcal{U}))$-separator in G, and let $S' = R_G(S, P) \supseteq S$. Pick an arbitrary $p \in P$ (which may be contained in T but not in S).
      i. Call $\mathsf{Enum}_{\mathcal{F}}(G - p, S', T \setminus \{p\}, k - 1)$. (The value of k decreases.)
      ii. If $p \notin T$, then call $\mathsf{Enum}_{\mathcal{F}}(G, S' \cup \{p\}, T, k)$.
      (Either here or in the next iteration we will be able to make progress.)

   It might happen that $\mathcal{U}$ is empty; in this case the algorithm will execute Step 4b. Also note that P is non-empty because the algorithm did not stop in Step 2; hence it is always possible to choose a vertex $p \in P$.

Before providing an in-depth analysis of the algorithm, we establish that it always terminates. For each recursive call, either a vertex outside S is deleted, or one of S or T grows in size while the two remain disjoint. Since S and T are vertex subsets of a finite graph, this process terminates. The key argument in the correctness of the algorithm is formalized in the following claim.

▷ Claim 16. If the algorithm reaches Step 4b, then every seclusion-maximal k-secluded subgraph satisfying the conditions of the theorem statement contains S'.

Proof. We prove the claim by showing that for an arbitrary k-secluded $\mathcal{F}$-free connected induced subgraph $G[C]$ satisfying $S \subseteq C \subseteq V(G) \setminus T$, the subgraph induced by $C \cup S'$ also satisfies these properties while $|N_G(C \cup S')| \leq |N_G(C)|$. Hence any seclusion-maximal subgraph satisfying the conditions contains S'.

Under the conditions of Step 4b, we have $\lambda^{\mathrm{L}}_G(S,T) = \lambda^{\mathrm{L}}_G(S, T \cup V(\mathcal{U}))$, so that the set P is a left-restricted minimum (S,T)-separator. Next, we have $S' = R_G(S,P)$. By exploiting submodularity of the size of the open neighborhood, we prove in Lemma 12 that $|N_G(C \cup S')| \leq |N_G(C)|$. The key part of the argument is to prove that $C \cup S'$ induces an $\mathcal{F}$-free subgraph. Assume for a contradiction that $G[C \cup S']$ contains an induced subgraph isomorphic to $F \in \mathcal{F}$ and let $\phi\colon V(F) \to C \cup S'$ denote an induced subgraph isomorphism. Out of all ways to choose ϕ, fix a choice that minimizes the number of vertices $|\phi(F) \setminus S|$ the subgraph uses from outside S. We distinguish two cases.

Neighborhood of S intersects $\phi(F)$

If $\phi(F) \cap N_G(S) \neq \emptyset$, then we will use the assumption that Step 3 of the algorithm was not applicable to derive a contradiction. Let F' be the graph consisting of those connected components F_i of F for which $\phi(F_i) \cap N_G[S] \neq \emptyset$; let $U = \phi(F') \setminus S$. Observe that each connected component of $G[U]$ is adjacent to a vertex of S. By construction U is disjoint from S, and U is disjoint from T since $\phi(F) \subseteq C \cup S'$ while both these sets are disjoint from T. Hence U satisfies all but one of the conditions for applying Step 3. Since the algorithm reached Step 4b, it follows that U failed the last criterion which means that the partial forbidden graph F' also exists as an induced subgraph in $G[S]$. Let $\phi_{F'}\colon V(F') \to S$ be an induced subgraph isomorphism from F' to $G[S]$. Since all vertices $v \in V(F)$ for which $\phi(v) \in N_G[S]$ satisfy $v \in V(F')$, we can define a new subgraph isomorphism ϕ' of F in $G[C \cup S']$ as follows for each $v \in V(F)$:

$$\phi'(v) = \begin{cases} \phi_{F'}(v) & \text{if } v \in F' \\ \phi(v) & \text{otherwise.} \end{cases} \tag{1}$$

Observe that this is a valid induced subgraph isomorphism since F' consists of some connected components of F, and we effectively replace the model of F' by $\phi_{F'}$. Since the model of the remaining graph $\overline{F'} = F - F'$ does not use any vertex of $N_G[S]$ by definition of F', there are no edges between vertices of $\phi_{F'}(F')$ and vertices of $\phi(\overline{F'})$, which validates the induced subgraph isomorphism.

Since $\phi(F)$ contains at least one vertex from $N_G(S)$ while $\phi'(F)$ does not, and the only vertices of $\phi'(F) \setminus \phi(F)$ belong to S, we conclude that $\phi'(F)$ contains strictly fewer vertices outside S than $\phi(F)$; a contradiction to minimality of ϕ.

Neighborhood of S does not intersect $\phi(F)$

Now suppose that $\phi(F) \cap N_G(S) = \emptyset$. If $\phi(F) \subseteq C$, then $\phi(F)$ is an induced F-subgraph in $G[C]$, a contradiction to the assumption that C is $\mathcal{F}$-free. Hence $\phi(F)$ must contain a vertex $v \in S' \setminus C \subseteq S' \setminus S$. Since the previous case was not applicable, $v \notin N_G(S)$ and therefore $v \in S' \setminus N_G[S]$.

Fix an arbitrary connected component F_i of F for which $\phi(F_i)$ contains a vertex of $S' \setminus N_G[S]$. We derive several properties of $\phi(F_i)$.

1. Since F_i is a connected component of F, the graph $G[\phi(F_i)]$ is connected.
2. We claim that $\phi(F_i) \cap S = \emptyset$. Note that a connected subgraph cannot both contain a vertex from S and a vertex outside $N_G[S]$ without intersecting $N_G(S)$. Since $\phi(F) \cap N_G(S) = \emptyset$ by the case distinction, the graph $G[\phi(F_i)]$ is connected since F_i is connected, and $\phi(F_i)$ contains a vertex of $S' \setminus N_G[S]$, we find $\phi(F_i) \cap S = \emptyset$.
3. $\phi(F_i) \cap T = \emptyset$, since $\phi(F) \subseteq C \cup S'$ while both C and S' are disjoint from T.

4. We claim that $\phi(F_i) \notin \mathcal{U}$. To see that, recall that $S' = R_G(S, P)$ is the set of vertices reachable from S when removing the $(S, T \cup V(\mathcal{U}))$-separator P. The definition of separator therefore ensures that no vertex of S' belongs to $V(\mathcal{U})$. Since $\phi(F_i)$ contains a vertex of $S' \setminus N_G[S]$ by construction, some vertex of $\phi(F_i)$ does not belong to $V(\mathcal{U})$ and therefore $\phi(F_i) \notin \mathcal{U}$.

Now note that $\phi(F_i)$ satisfies almost all requirements for being contained in the set $\mathcal{U}$ defined in Step 4: it induces a connected subgraph and it is disjoint from $S \cup T$. From the fact that $\phi(F_i) \notin \mathcal{U}$ we therefore conclude that it fails the last criterion: the set $\phi(F_i)$ is not a tight enrichment of S.

Let F' be the graph formed by F_i together with all components F_j of F for which $\phi(F_j) \subseteq S$; then $\phi(F_i) = \phi(F') \setminus S$. Since $\phi(F_i)$ is not a tight enrichment of S, the partial forbidden graph F' is also contained in $G[S]$. Let $\phi_{F'} \colon F' \to S$ denote an induced subgraph isomorphism of F' to $G[S]$. Since $\phi(F)$ contains no vertex of $N_G(S)$, we can define a new subgraph isomorphism ϕ' of F in $G[C \cup S']$ exactly as in (1).

Since the graph F' consists of some connected components of F, while $\phi_{F'}(F') \subseteq S$ and $\phi(\overline{F'}) \cap N_G[S] = \emptyset$, it follows that ϕ' is an induced subgraph isomorphism of F in $G[C \cup S']$. But $|\phi'(F) \setminus S|$ is strictly smaller than $|\phi(F) \setminus S|$ since $\phi(F_i)$ intersects $S' \setminus N_G[S]$ while $\phi'(F_i) \subseteq \phi'(F') \subseteq S$ and ϕ and ϕ' coincide on $\overline{F'}$. This contradicts the minimality of the choice of ϕ.

Since the case distinction is exhaustive, this proves the claim. ◁

Using the previous claim, we can establish the correctness of the algorithm.

▷ Claim 17. If $G[C]$ is an induced subgraph of G that is seclusion-maximal with respect to being connected, $\mathcal{F}$-free, k-secluded and satisfying $S \subseteq C \subseteq V(G) \setminus T$, then C occurs in the output of $\mathsf{Enum}_{\mathcal{F}}(G, S, T, k)$.

Proof. We prove this claim by induction on the recursion depth of the $\mathsf{Enum}_{\mathcal{F}}$ algorithm, which is valid as we argued above it is finite. In the base case, the algorithm does not recurse. In other words, the algorithm either stopped in Step 1 or 2. If the algorithm stops in Step 1, then there can be no induced subgraph satisfying the conditions and so there is nothing to show. If the algorithm stops in Step 2, then the only seclusion-maximal induced subgraph is the $\mathcal{F}$-free connected component containing S. Note that this component is k-secluded since $k \geq 0$ as $\lambda^{\mathrm{L}}_G(S, T) \geq 0$ and the algorithm did not stop in Step 1a.

For the induction step, we may assume that each recursive call made by the algorithm correctly enumerates a superset of the seclusion-maximal subgraphs satisfying the conditions imposed by the parameters of the recursive call, as the recursion depth of the execution of those calls is strictly smaller than the recursion depth for the current arguments (G, S, T, k). Consider a connected $\mathcal{F}$-free k-secluded induced subgraph $G[C]$ of G with $S \subseteq C \subseteq V(G) \setminus T$ that is seclusion-maximal with respect to satisfying all these conditions. Suppose there is a vertex set $U \subseteq V(G) \setminus (S \cup T)$ that satisfies the conditions of Step 3. If $U \subseteq C$, then by induction C is part of the enumerated output of Step 3b. Otherwise, since each connected component of $G[U]$ is adjacent to a vertex in S, there is at least one vertex $u \in U$ such that $u \in N_G(C)$. By Observation 15, the output of the corresponding call in Step 3a contains C. Note that since $U \cap T = \emptyset$, we have $T \subseteq V(G) \setminus (S \cup U)$ and therefore the recursive calls satisfy the input requirements.

Next we consider the correctness in case such a set U does not exist so the algorithm reaches Step 4. Let $\mathcal{U}$ be the set of tight enrichments as defined in Step 4. First suppose that $\lambda^{\mathrm{L}}_G(S, T) < \lambda^{\mathrm{L}}_G(S, T \cup V(\mathcal{U}))$. Then by the contrapositive of the first part of Lemma 11 with $Z = V(\mathcal{U})$, there is a vertex $v \in V(\mathcal{U}) \setminus T$ such that $\lambda^{\mathrm{L}}_G(S, T \cup \{v\}) > \lambda^{\mathrm{L}}_G(S, T)$. By picking an enrichment $U \in \mathcal{U}$ such that $v \in U$, this implies $\lambda^{L}_{G}(S, T \cup U) > \lambda^{L}_{G}(S, T)$. Now if there is a vertex $u \in U$ such that $u \in N_G(C)$, then by induction and Observation 15 we get that C is output by the corresponding call in Step 4(a)i. Otherwise, either $U \subseteq C$ or $U \cap C = \emptyset$ (since U is connected) and C is found in Step 4(a)ii or Step 4(a)iii respectively. Again observe that these recursive calls satisfy the input requirements as $U \cap (S \cup T) = \emptyset$.

Finally suppose that $\lambda^{L}_{G}(S, T) = \lambda^{L}_{G}(S, T \cup V(\mathcal{U}))$. By Claim 16 we get that $S' \subseteq C$. We first argue that $P = N_G(S')$ is non-empty. Note that since the algorithm did not stop in Step 1, the graph $G[S]$ is $\mathcal{F}$-free and S is contained in a single connected component of G. Furthermore since it did not stop in Step 2, the connected component containing S either has a vertex of T or is not $\mathcal{F}$-free. Note that the former case already implies $\lambda^{L}_{G}(S, T) > 0$. If the component has no vertex of T and is not $\mathcal{F}$-free, then it contains a vertex set J for which $G[J]$ is isomorphic to some $F \in \mathcal{F}$. Observe that $J \setminus (S \cup T) = J \setminus S$ is a tight enrichment of S. We have established that it is possible to enrich S but we need an enrichment that meets the conditions of Step 4. Let $U \subseteq V(G) \setminus (S \cup T)$ be a tight enrichment of minimum size and let $\phi\colon V(F') \to V(G)$ be the corresponding subgraph isomorphism from some partial forbidden graph F'; we have $U = \phi(F') \setminus S$. We argue that $G[U]$ is connected. If each connected component of $G[U]$ is adjacent to a vertex of S, then Step 3 would have applied, contradicting the fact that the algorithm reaches Step 4. Hence, there exists a connected component of $G[U]$ that is non-adjacent to S; let U' be the vertex set of such a component. Since U is chosen to be minimum, we get that $U \setminus U'$ is not a tight enrichment, and so there is an induced subgraph of $G[S]$ isomorphic to the partial forbidden graph $F'' = G[\phi(F') \setminus U']$. This subgraph of $G[S]$ combines with the graph $G[U']$ to form an induced subgraph isomorphic to F' (we exploit that U' is not adjacent to S), which shows that U' is a tight enrichment. By minimality of U we obtain $U = U'$. Hence U is not adjacent to S and the graph $G[U]$ is connected so $U \in \mathcal{U}$. Since U and S are contained in the same connected component we get that $\lambda^{L}_{G}(S, T \cup V(\mathcal{U})) > 0$. This implies there exists some vertex $p \in P = N_G(S')$. Since $S' \subseteq C$, we either get $p \in N_G(C)$, or (if $p \notin T$) $p \in C$. By induction (and Observation 15) we conclude that C is part of the output of Step 4(b)i or Step 4(b)ii. The condition $p \notin T$ ensures that the input requirements of the latter recursive call are satisfied. ◁

As the previous claim shows that the algorithm enumerates a superset of the relevant seclusion-maximal induced subgraphs, to prove Theorem 2 it suffices to bound the size of the search tree generated by the algorithm, and thereby the running time and total number of induced subgraphs which are given as output. To that end, we argue that for any two successive recursive calls in the recursion tree, at least one of them makes strict progress on a relevant measure. Since no call can increase the measure, this will imply a bound on the depth of the recursion tree. Since it is easy to see that the branching factor is a constant depending on $||\mathcal{F}||$, this will lead to the desired bound.

▷ Claim 18 (★). The search tree generated by the call $\mathsf{Enum}_{\mathcal{F}}(G, S, T, k)$ has depth $\mathcal{O}_{\mathcal{F}}(k)$ and $2^{\mathcal{O}_{\mathcal{F}}(k)}$ leaves.

The previous claim implies that the number of seclusion-maximal connected $\mathcal{F}$-free k-secluded induced subgraphs containing all of S and none of T is $2^{\mathcal{O}_{\mathcal{F}}(k)}$, since the algorithm outputs at most one subgraph per call and only does so in leaf nodes of the recursion tree. As Claim 18 bounds the size of the search tree generated by the algorithm, the desired bound on the total running time follows from the claim below.

▷ Claim 19 (★). A single iteration of $\mathsf{Enum}_{\mathcal{F}}(G, S, T, k)$ can be implemented to run in time $|\mathcal{F}| \cdot 2^{||\mathcal{F}||} \cdot n^{||\mathcal{F}||+\mathcal{O}(1)}$ and polynomial space.

This concludes the proof of Theorem 2. ◀

4 Conclusion

We have introduced a new algorithmic primitive based on secluded connected subgraphs which generalizes important separators. The high-level idea behind the algorithm is *enumeration via separation*: by introducing an artificial set T and considering the more general problem of enumerating secluded subgraphs containing S but disjoint from T, we can analyze the progress of the recursion in terms of the size of a minimum (left-restricted) (S,T)-separator. We expect this idea to be useful in scenarios beyond the one studied here.

We presented a single-exponential, polynomial-space FPT algorithm to enumerate the family of seclusion-maximal connected $\mathcal{F}$-free subgraphs for finite $\mathcal{F}$, making it potentially viable for practical use [38]. The combination of single-exponential running time and polynomial space usage sets our approach apart from others such as recursive understanding [8, 10, 31] and treewidth reduction [36]. Algorithms exploiting half-integrality of the linear-programming relaxation or other discrete relaxations also have these desirable properties, though [12, 18, 19, 20, 41]. Using this approach, Iwata, Yamaguchi, and Yoshida [20] even obtained a *linear-time* algorithm in terms of the number of vertices n, solving (vertex) Multiway Cut in time $2^k \cdot k \cdot (n+m)$. At a high level, there is some resemblance between their approach and ours. They work on a discrete relaxation of deletion problems in graphs which are not standard LP-relaxations, but are based on relaxations of a *rooted* problem in which only constraints involving a prescribed set S are active. This is reminiscent of the fact that we enumerate secluded subgraphs containing a prescribed set S. Their branching algorithms are based on the notion of an extremal optimal solution to the LP relaxation, which resembles our use of the farthest minimum left-restricted (S,T)-separator. However, the two approaches diverge there. To handle problems via their approach, they should be expressible as a 0/1/ALL CSP. Problems for which the validity of a solution can be verified by unit propagation (such as Node Unique Label Cover, Node Multiway Cut, Subset and Group Feedback Vertex Set) belong to this category, but it seems impossible to express the property of being $\mathcal{F}$-free for arbitrary finite sets $\mathcal{F}$ in this framework.

The branching steps underlying our algorithm were informed by the structure of the subgraphs induced by certain vertex sets. In the considered setting, where certain possibly disconnected structures are not allowed to appear inside C, it is necessary to characterize the forbidden sets in terms of the graph structure they induce. But when the forbidden sets are connected, we believe our proof technique can be used in a more general setting to establish the following. For any n-vertex graph G, non-empty vertex set $S \subseteq V(G)$, potentially empty $T \subseteq V(G) \setminus S$, integer k, and collection $F_1, \ldots, F_m \subseteq V(G)$ of vertex sets of size at most ℓ which are connected in G, the number of k-secluded induced subgraphs $G[C]$ which are seclusion-maximal with respect to being connected, not containing any set F_i, and satisfying $S \subseteq C \subseteq V(G) \setminus T$, is bounded by $(2+\ell)^{\mathcal{O}(k)}$, and a superset of them can be enumerated in time $(2+\ell)^{\mathcal{O}(k)} \cdot m \cdot n^{\mathcal{O}(1)}$ and polynomial space. The reason why dealing with general connected obstacles is feasible is that whenever $F_i \cap C \neq \emptyset$ then also $F_i \cap N(C) \neq \emptyset$; this allows us to always make progress using the simpler branching strategy without keeping track of partial forbidden graphs. The corresponding generalization for *disconnected* vertex sets F_i is false, even for $|F_i| = 2$. To see this, consider a graph consisting of a cycle on $2m+1$ vertices consecutively labeled $s, a_1, \ldots, a_m, b_1, \ldots, b_m$ with $F_i = \{a_i, b_i\}$ for each $i \in [m]$, in which the number of relevant seclusion-maximal 2-secluded sets containing s is $\Omega(m)$.

We leave it to future work to consider generalizations of our ideas to *directed graphs*. Since important separators also apply in that setting, we expect the branching step in terms of left-restricted minimum separators to be applicable in directed graphs as well. However, there are multiple ways to generalize the notion of a connected secluded induced subgraph to the directed setting: one can consider weak connectivity, strong connectivity, or a rooted variant where we consider all vertices reachable from a source vertex x. Similarly, one can define seclusion in terms of the number of in-neighbors, out-neighbors, or both.

References

1. René van Bevern, Till Fluschnik, George B. Mertzios, Hendrik Molter, Manuel Sorge, and Ondrej Suchý. The parameterized complexity of finding secluded solutions to some classical optimization problems on graphs. *Discret. Optim.*, 30:20–50, 2018. doi:10.1016/j.disopt.2018.05.002.
2. René van Bevern, Till Fluschnik, and Oxana Yu. Tsidulko. Parameterized algorithms and data reduction for the short secluded s-t-path problem. *Networks*, 75(1):34–63, 2020. doi:10.1002/net.21904.
3. Hans L. Bodlaender, Anuj Dawar, and Virginia V. Williams. EATCS-IPEC Nerode Prize 2020, 2020. URL: https://eatcs.org/index.php/component/content/article/1-news/2861-eatcs-ipec-nerode-prize-2020-.
4. Nicolas Bousquet, Jean Daligault, and Stéphan Thomassé. Multicut is FPT. *SIAM J. Comput.*, 47(1):166–207, 2018. doi:10.1137/140961808.
5. Shiri Chechik, Matthew P. Johnson, Merav Parter, and David Peleg. Secluded connectivity problems. *Algorithmica*, 79(3):708–741, 2017. doi:10.1007/s00453-016-0222-z.
6. Jianer Chen, Yang Liu, and Songjian Lu. An improved parameterized algorithm for the minimum node multiway cut problem. *Algorithmica*, 55(1):1–13, 2009. doi:10.1007/s00453-007-9130-6.
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. Rajesh Chitnis, Marek Cygan, MohammadTaghi Hajiaghayi, Marcin Pilipczuk, and Michal Pilipczuk. Designing FPT algorithms for cut problems using randomized contractions. *SIAM J. Comput.*, 45(4):1171–1229, 2016. doi:10.1137/15M1032077.
9. Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
10. Marek Cygan, Pawel Komosa, Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, Saket Saurabh, and Magnus Wahlström. Randomized contractions meet lean decompositions. *ACM Trans. Algorithms*, 17(1):6:1–6:30, 2021. doi:10.1145/3426738.
11. Marek Cygan, Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. Minimum bisection is fixed-parameter tractable. *SIAM J. Comput.*, 48(2):417–450, 2019. doi:10.1137/140988553.
12. Marek Cygan, Marcin Pilipczuk, Michal Pilipczuk, and Jakub Onufry Wojtaszczyk. On multiway cut parameterized above lower bounds. *ACM Trans. Comput. Theory*, 5(1):3:1–3:11, 2013. doi:10.1145/2462896.2462899.
13. Huib Donkers, Bart M. P. Jansen, and Jari J. H. de Kroon. Finding k-secluded trees faster. In *Proceeding of the 48th International Workshop on Graph-Theoretic Concepts in Computer Science, WG 2022*, volume 13453 of *Lecture Notes in Computer Science*, pages 173–186. Springer, 2022. doi:10.1007/978-3-031-15914-5_13.
14. Fedor V. Fomin, Petr A. Golovach, Nikolay Karpov, and Alexander S. Kulikov. Parameterized complexity of secluded connectivity problems. *Theory Comput. Syst.*, 61(3):795–819, 2017. doi:10.1007/s00224-016-9717-x.
15. L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956. doi:10.4153/CJM-1956-045-5.

16 Robert Ganian, M. S. Ramanujan, and Stefan Szeider. Discovering archipelagos of tractability for constraint satisfaction and counting. *ACM Trans. Algorithms*, 13(2):29:1–29:32, 2017. `doi:10.1145/3014587`.

17 Petr A. Golovach, Pinar Heggernes, Paloma T. Lima, and Pedro Montealegre. Finding connected secluded subgraphs. *J. Comput. Syst. Sci.*, 113:101–124, 2020. `doi:10.1016/j.jcss.2020.05.006`.

18 Sylvain Guillemot. FPT algorithms for path-transversal and cycle-transversal problems. *Discret. Optim.*, 8(1):61–71, 2011. `doi:10.1016/j.disopt.2010.05.003`.

19 Yoichi Iwata, Magnus Wahlström, and Yuichi Yoshida. Half-integrality, LP-branching, and FPT algorithms. *SIAM J. Comput.*, 45(4):1377–1411, 2016. `doi:10.1137/140962838`.

20 Yoichi Iwata, Yutaro Yamaguchi, and Yuichi Yoshida. 0/1/all CSPs, half-integral A-path packing, and linear-time FPT algorithms. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 462–473. IEEE Computer Society, 2018. `doi:10.1109/FOCS.2018.00051`.

21 Ashwin Jacob, Jari J. H. de Kroon, Diptapriyo Majumdar, and Venkatesh Raman. Deletion to scattered graph classes I – case of finite number of graph classes. *J. Comput. Syst. Sci.*, 138:103460, 2023. `doi:10.1016/j.jcss.2023.05.005`.

22 Ashwin Jacob, Diptapriyo Majumdar, and Venkatesh Raman. Parameterized complexity of deletion to scattered graph classes. In Yixin Cao and Marcin Pilipczuk, editors, *15th International Symposium on Parameterized and Exact Computation, IPEC 2020, December 14-18, 2020, Hong Kong, China (Virtual Conference)*, volume 180 of *LIPIcs*, pages 18:1–18:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. `doi:10.4230/LIPIcs.IPEC.2020.18`.

23 Ashwin Jacob, Diptapriyo Majumdar, and Venkatesh Raman. Deletion to scattered graph classes II – improved FPT algorithms for deletion to pairs of graph classes. *J. Comput. Syst. Sci.*, 136:280–301, 2023. `doi:10.1016/j.jcss.2023.03.004`.

24 Bart M. P. Jansen, Jari J. H. de Kroon, and Michał Włodarczyk. Single-exponential fpt algorithms for enumerating secluded $\mathcal{F}$-free subgraphs and deleting to scattered graph classes, 2023. `arXiv:2309.11366`.

25 Ken-ichi Kawarabayashi and Mikkel Thorup. The minimum k-way cut of bounded size is fixed-parameter tractable. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 160–169. IEEE Computer Society, 2011. `doi:10.1109/FOCS.2011.53`.

26 Eun Jung Kim, Stefan Kratsch, Marcin Pilipczuk, and Magnus Wahlström. Solving hard cut problems via flow-augmentation. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10–13, 2021*, pages 149–168. SIAM, 2021. `doi:10.1137/1.9781611976465.11`.

27 Eun Jung Kim, Stefan Kratsch, Marcin Pilipczuk, and Magnus Wahlström. Directed flow-augmentation. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20–24, 2022*, pages 938–947. ACM, 2022. `doi:10.1145/3519935.3520018`.

28 Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. *J. ACM*, 67(3):16:1–16:50, 2020. `doi:10.1145/3390887`.

29 Daniel Lokshtanov and Dániel Marx. Clustering with local restrictions. *Inf. Comput.*, 222:278–292, 2013. `doi:10.1016/j.ic.2012.10.016`.

30 Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, Roohani Sharma, and Meirav Zehavi. Covering small independent sets and separators with applications to parameterized algorithms. *ACM Trans. Algorithms*, 16(3):32:1–32:31, 2020. `doi:10.1145/3379698`.

31 Daniel Lokshtanov, M. S. Ramanujan, Saket Saurabh, and Meirav Zehavi. Reducing CMSO model checking to highly connected graphs. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 135:1–135:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. `doi:10.4230/LIPIcs.ICALP.2018.135`.

32 Max-Jonathan Luckow and Till Fluschnik. On the computational complexity of length- and neighborhood-constrained path problems. *Inf. Process. Lett.*, 156:105913, 2020. doi:10.1016/j.ipl.2019.105913.

33 Dániel Marx. Parameterized graph separation problems. *Theor. Comput. Sci.*, 351(3):394–406, 2006. doi:10.1016/j.tcs.2005.10.007.

34 Dániel Marx. Important separators and parameterized algorithms. In Petr Kolman and Jan Kratochvíl, editors, *Graph-Theoretic Concepts in Computer Science – 37th International Workshop, WG 2011, Teplá Monastery, Czech Republic, June 21-24, 2011. Revised Papers*, volume 6986 of *Lecture Notes in Computer Science*, pages 5–10. Springer, 2011. doi:10.1007/978-3-642-25870-1_2.

35 Dániel Marx, Barry O'Sullivan, and Igor Razgon. Finding small separators in linear time via treewidth reduction. *ACM Trans. Algorithms*, 9(4):30:1–30:35, 2013. doi:10.1145/2500119.

36 Dániel Marx and Igor Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM J. Comput.*, 43(2):355–388, 2014. doi:10.1137/110855247.

37 Neeldhara Misra. Kernelization, Planar F-deletion. In *Encyclopedia of Algorithms*, pages 1033–1036. Springer, 2016. doi:10.1007/978-1-4939-2864-4_527.

38 Marcin Pilipczuk and Michal Ziobro. Experimental evaluation of parameterized algorithms for graph separation problems: Half-integral relaxations and matroid-based kernelization. *CoRR*, abs/1811.07779, 2018. arXiv:1811.07779.

39 Igor Razgon and Barry O'Sullivan. Almost 2-SAT is fixed-parameter tractable. *J. Comput. Syst. Sci.*, 75(8):435–450, 2009. doi:10.1016/j.jcss.2009.04.002.

40 A. Schrijver. *Combinatorial Optimization – Polyhedra and Efficiency*. Springer, 2003.

41 Mingyu Xiao. Simple and improved parameterized algorithms for multiterminal cuts. *Theory Comput. Syst.*, 46(4):723–736, 2010. doi:10.1007/s00224-009-9215-5.