

Original citation:

Cygan , Marek, Lokshtanov , Daniel , Pilipczuk, Marcin, Pilipczuk , Michał and Saurabh, Saket (2014) Minimum bisection is fixed parameter tractable. In: STOC '14 : 46th Annual ACM Symposium on Theory of Computing, New York, 31 May- 3 Jun 2014. Published in: STOC '14 Proceedings of the 46th Annual ACM Symposium on Theory of Computing pp. 323-332.

Permanent WRAP url:

<http://wrap.warwick.ac.uk/66072>

Copyright and reuse:

The Warwick Research Archive Portal (WRAP) makes this work of researchers of the University of Warwick available open access under the following conditions. Copyright © and all moral rights to the version of the paper presented here belong to the individual author(s) and/or other copyright owners. To the extent reasonable and practicable the material made available in WRAP has been checked for eligibility before being made available.

Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.


Publisher statement:

© ACM. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive version was published in STOC '14 Proceedings
<http://dx.doi.org/10.1145/2591796.2591852>

A note on versions:

The version presented here may differ from the published version or, version of record, if you wish to cite this item you are advised to consult the publisher's version. Please see the 'permanent WRAP url' above for details on accessing the published version and note that access may require a subscription.

For more information, please contact the WRAP Team at: publications@warwick.ac.uk

warwick**publications**wrap

highlight your research

<http://wrap.warwick.ac.uk/>

Minimum Bisection is Fixed Parameter Tractable

[Extended Abstract] *

Marek Cygan
Institute of Informatics
University of Warsaw, Poland
cygan@mimuw.edu.pl

Daniel Lokshtanov
Department of Informatics
University of Bergen, Norway
daniello@ii.uib.no

Marcin Pilipczuk
Department of Informatics
University of Bergen, Norway
Marcin.Pilipczuk@ii.uib.no

Michał Pilipczuk
Department of Informatics
University of Bergen, Norway
michal.pilipczuk@ii.uib.no

Saket Saurabh
Institute of Mathematical
Sciences, India &
Department of Informatics
University of Bergen, Norway
Saket.Saurabh@ii.uib.no

ABSTRACT

In the classic MINIMUM BISECTION problem we are given as input a graph G and an integer k . The task is to determine whether there is a partition of $V(G)$ into two parts A and B such that $||A| - |B|| \leq 1$ and there are at most k edges with one endpoint in A and the other in B . In this paper we give an algorithm for MINIMUM BISECTION with running time $\mathcal{O}(2^{\mathcal{O}(k^3)} n^3 \log^3 n)$. This is the first fixed parameter tractable algorithm for MINIMUM BISECTION. At the core of our algorithm lies a new decomposition theorem that states that every graph G can be decomposed by small separators into parts where each part is “highly connected” in the following sense: any cut of bounded size can separate only a limited number of vertices from each part of the decomposition.

Our techniques generalize to the weighted setting, where we seek for a bisection of minimum weight among solutions that contain at most k edges.

Categories and Subject Descriptors

F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems—*Computations on discrete structures*

General Terms

Algorithms, Theory

Keywords

minimum bisection, fixed-parameter tractability, tree de-

*A full version of this paper is available at arXiv [8]

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
STOC '14, May 31 - June 03 2014, New York, NY, USA
Copyright 2014 ACM 978-1-4503-2710-7/14/05 ...\$15.00.
<http://dx.doi.org/10.1145/2591796.2591852>.

composition, randomized contractions

1. INTRODUCTION

In the MINIMUM BISECTION problem the input is a graph G on n vertices together with an integer k , and the objective is to find a partition of the vertex set into two parts A and B such that $|A| = \lfloor \frac{n}{2} \rfloor$, $|B| = \lceil \frac{n}{2} \rceil$, and there are at most k edges with one endpoint in A and the other endpoint in B . The problem can be seen as a variant of MINIMUM CUT, and is one of the classic NP-complete problems [13]. MINIMUM BISECTION has been studied extensively from the perspective of approximation algorithms [11, 10, 18, 23], heuristics [2, 4] and average case complexity [1].

In this paper we consider the complexity of MINIMUM BISECTION when the solution size k is small relative to the input size n . A naïve brute-force algorithm solves the problem in time $n^{\mathcal{O}(k)}$. Until this work, it was unknown whether there exists a *fixed parameter tractable* algorithm, that is an algorithm with running time $f(k)n^{\mathcal{O}(1)}$, for the MINIMUM BISECTION problem. In fact MINIMUM BISECTION was one of very few remaining classic NP-hard graph problems whose parameterized complexity status was unresolved. Our main result is the first fixed parameter tractable algorithm for MINIMUM BISECTION.

THEOREM 1.1. MINIMUM BISECTION *admits an algorithm running in time* $\mathcal{O}(2^{\mathcal{O}(k^3)} n^3 \log^3 n)$.

Theorem 1.1 implies that MINIMUM BISECTION can be solved in polynomial time for $k = \mathcal{O}(\sqrt[3]{\log n})$. In fact, our techniques can be generalized to solve the more general problem where the target $|A|$ is given as input, the edges have non-negative weights, and the objective is to find, among all partitions of $V(G)$ into A and B such that A has the prescribed size and there are at most k edges between A and B , such a partition where the total weight of the edges between A and B is minimized.

1.1 Our methods

The crucial technical component of our result is a new graph decomposition theorem. Roughly speaking, the theorem states that for any k , every graph G may be decomposed in a tree-like fashion by separators of size $2^{\mathcal{O}(k)}$ such

that each part of the decomposition is “highly connected”. To properly define what we mean by “highly connected” we need a few definitions. A *separation* of a graph G is a pair $A, B \subseteq V(G)$ such that $A \cup B = V(G)$ and there are no edges between $A \setminus B$ and $B \setminus A$. The *order* of the separation (A, B) is $|A \cap B|$. A vertex set $X \subseteq V(G)$ is called (q, k) -*unbreakable* if every separation (A, B) of order at most k satisfies $|(A \setminus B) \cap X| \leq q$ or $|(B \setminus A) \cap X| \leq q$. The parts of our decomposition will be “highly connected” in the sense that they are $(2^{\mathcal{O}(k)}, k)$ -unbreakable. We can now state the decomposition theorem as follows.

THEOREM 1.2. *There is an algorithm that given G and k runs in time $\mathcal{O}(2^{\mathcal{O}(k^2)} n^2 m)$ and outputs a tree-decomposition (T, β) of G such that (i) for each $a \in V(T)$, $\beta(a)$ is $(2^{\mathcal{O}(k)}, k)$ -unbreakable in G , (ii) for each $ab \in E(T)$ we have that $|\beta(a) \cap \beta(b)| \leq 2^{\mathcal{O}(k)}$, and $\beta(a) \cap \beta(b)$ is $(2k, k)$ -unbreakable in G .*

Here $\beta(a)$ denotes the bag at node $a \in V(T)$; the completely formal definition of tree-decompositions may be found in the preliminaries. It is not immediately obvious that a set X which is (q, k) -unbreakable is “highly connected”. To get some intuition it is helpful to observe that if a set X of size at least $3q$ is (q, k) -unbreakable then removing any k vertices from G leaves almost all of X , except for at most q vertices, in the same connected component. In other words, one cannot separate two large chunks of X with a small separator. From this perspective Theorem 1.2 can be seen as an approximate way to “decompose a graph by k vertex-cuts into its $k + 1$ -connected components” [5], which is considered an important quest in structural graph theory. The proof strategy of Theorem 1.2 is inspired by the recent decomposition theorem of Marx and Grohe [15] for graphs excluding a topological subgraph. Contrary to the approach of Marx and Grohe [15], however, the crucial technical tool we use to decompose the graph are the important separators of Marx [19].

Our algorithm for MINIMUM BISECTION applies Theorem 1.2 and then proceeds by performing bottom up dynamic programming on the tree-decomposition. The states in the dynamic program are similar to the states in the dynamic programming algorithm for MINIMUM BISECTION on graphs of bounded treewidth [16]. Property (ii) of Theorem 1.2 ensures that the size of the dynamic programming table is upper bounded by $2^{\mathcal{O}(k^2)} n^{\mathcal{O}(1)}$. For graphs of bounded treewidth all bags have small size, making it easy to compute the dynamic programming table at a node b of the decomposition tree, if the tables for the children of b have already been computed. In our setting we do not have any control over the size of the bags, we only know that they are $(2^{\mathcal{O}(k)}, k)$ -unbreakable. We show that the sole assumption that the bag at b is $(2^{\mathcal{O}(k)}, k)$ -unbreakable is already sufficient to efficiently compute the table at b from the tables of its children, despite no a priori guarantee on the bag’s size. The essence of this step is an application of the “randomized contractions” technique [7].

We remark here that the last property of the decomposition of Theorem 1.2 — the one that asserts that adhesions $\beta(a) \cap \beta(b)$ are $(2k, k)$ -unbreakable in G — is not essential to establish the fixed-parameter tractability of MINIMUM BISECTION. This high unbreakability of adhesions is used to further limit the number of states of the dynamic programming, decreasing the dependency on k in the algorithm of

Theorem 1.1 from double- to single-exponential.

1.2 Related work on balanced separations

There are several interesting results concerning the parameterized complexity of finding balanced separators in graphs. Marx [19] showed that the vertex-deletion variant of the bisection problem is W[1]-hard. In MINIMUM VERTEX BISECTION the task is to partition the vertex set into three parts A, S and B such that $|S| \leq k$ and $|A| = |B|$, and there are no edges between A and B . It is worth mentioning that the hardness result of Marx [19] applies to the more general problem where $|A|$ is given as input, however the hardness of MINIMUM VERTEX BISECTION easily follows from the results presented in [19].

As the vertex-deletion variant of the bisection problem is W[1]-hard, we should not expect that our approach would work also in this case. Observe that one can compute the decomposition of Theorem 1.2 and define the states of the dynamic programming over the tree decomposition, as it is done for graphs of bounded treewidth. However, we are unable to perform the computations needed for one bag of the decomposition. Moreover, it is not only the artifact of the “randomized contractions” technique, but the hard instances obtained from the reduction of [19] are in fact highly unbreakable by our definition, and Theorem 1.2 would return a trivial decomposition.

Feige and Mahdian [12] studied cut problems that may be considered as approximation variants of MINIMUM BISECTION and MINIMUM VERTEX BISECTION. We say that a vertex (edge) set S is an α -(edge)-separator if every connected component of $G \setminus S$ has at most αn vertices. The main result of Feige and Mahdian [12] is a randomized algorithm that given an integer k , $\frac{2}{3} \leq \alpha < 1$ and $\epsilon > 0$ together with a graph G which has an α -separator of size at most k , outputs in time $2^{f(\epsilon)k} n^{\mathcal{O}(1)}$ either an α -separator of size at most k or an $(\alpha + \epsilon)$ -separator of size strictly less than k . They also give a deterministic algorithm with similar running time for the edge variant of this problem. To complement this result they show that, at least for the vertex variant, the exponential running time dependence on $1/\epsilon$ is unavoidable. Specifically, they prove that for any $\alpha > \frac{1}{2}$ finding an α -separator of size k is W[1]-hard, and therefore unlikely to admit an algorithm with running time $f(k)n^{\mathcal{O}(1)}$, for any function f . On the other hand, our methods imply a $2^{\mathcal{O}(k^3)} n^{\mathcal{O}(1/\alpha)}$ time algorithm for finding an α -edge-separator of size at most k , for any $\alpha > 0$.

MINIMUM BISECTION on planar graphs was shown to be fixed parameter tractable by Bui and Peck [3]. It is interesting to note that MINIMUM BISECTION is not known to be NP-hard on planar graphs, and the complexity of MINIMUM BISECTION on planar graphs remains a challenging open problem. More recently, van Bevern et al. [25] used the treewidth reduction technique of Marx et al. [20] to give a fixed parameter tractable algorithm for MINIMUM BISECTION for the special case when removing the cut edges leaves a constant number of connected components. Their algorithm also works for the vertex-deletion variant the same restrictions. Since MINIMUM VERTEX BISECTION is known to be W[1]-hard, it looks difficult to extend their methods to give a fixed parameter tractable algorithm for MINIMUM BISECTION without any restrictions. Thus, Theorem 1.1 resolves an open problem of van Bevern et al. [25] on the existence of such an algorithm.

1.3 Related work on graph decompositions

The starting point of our decomposition theorem is the “recursive understanding” technique pioneered by Grohe et al. [14], and later used by Kawarabayashi and Thorup [17] and by Chitnis et al. [7] to design a number of interesting parameterized algorithms for cut problems. Recursive understanding can be seen as a reduction from a parameterized problem on general graphs to the same problem on graphs with a particular structure. Grohe et al. [14] essentially use recursive understanding to reduce the problem of deciding whether G contains H as a topological subgraph to the case where G either excludes a clique on $f(|H|)$ vertices as a *minor* or contains at most $f(|H|)$ vertices of degree more than $f(|H|)$, for some function f . Marx and Grohe [15] subsequently showed that any graph which excludes H as a topological subgraph can be decomposed by small separators, in a tree-like fashion, into parts such that each part either excludes a clique on $f(|H|)$ vertices as a minor or contains at most $f(|H|)$ vertices of degree more than $f(|H|)$, for some function f . Thus, the decomposition theorem of Marx and Grohe [15] can be seen as a “structural” analogue of the recursive understanding technique for topological subgraph containment.

Both Kawarabayashi and Thorup [17] and Chitnis et al. [7] apply recursive understanding to reduce certain parameterized cut problems on general graphs to essentially the same problem on a graph G where $V(G)$ is $(f(k), k)$ -unbreakable for some function f . Then they proceed to show that the considered problem becomes fixed parameter tractable on $(f(k), k)$ -unbreakable graphs. Observe that MINIMUM BISECTION on $(f(k), k)$ -unbreakable graphs is trivially fixed parameter tractable - if the number of vertices is more than $2f(k)$ we can immediately say no, while if the number of vertices is at most $2f(k)$, then a brute force algorithm is already fixed parameter tractable. More importantly, it turns out that even the more general problem where $|A|$ is given on the input can be solved in fixed parameter tractable time on $(f(k), k)$ -unbreakable graphs via an application of the “randomized contractions” technique of Chitnis et al [7]. It is therefore very natural to try to use recursive understanding in order to reduce MINIMUM BISECTION on general graphs to MINIMUM BISECTION on $(f(k), k)$ -unbreakable graphs.

Unfortunately, it seems very difficult to pursue this route. In particular, recursive understanding works by cutting the graph into two parts by a small separator, “understanding” the easier of the two parts recursively, and then replacing the “understood” part by a constant size gadget. For MINIMUM BISECTION it seems unlikely that the understood part can be emulated by any constant size gadget because of the balance constraint in the problem definition. Intuitively, we would need to encode the behaviour of the understood part for every possible cardinality of A , which gives us amount of information that is not bounded by a function of k . The issue has strong connections to the fact that the best known algorithm for MINIMUM BISECTION on graphs of bounded treewidth is at least quadratic [16] rather than linear.

At this point our decomposition theorem comes to the rescue. It precisely allows us to *structurally* decompose the graph in a tree-like fashion into $(f(k), k)$ -unbreakable parts, which provides much more robust foundations for further algorithmic applications. Thus, essentially our decomposition theorem does the same for cut problems as the decomposition theorem of Marx and Grohe [15] does for topologi-

cal subgraph containment. Notably, the “recursive understanding” step used by Kawarabayashi and Thorup [17] and Chitnis et al. [7] for their problems could be replaced by dynamic programming over the tree-decomposition given by Theorem 1.2.

We remark here that it has been essentially known, and observed earlier by Chitnis, Cygan and Hajiaghayi (private communication), that MINIMUM BISECTION can be solved in FPT time on sufficiently unbreakable graphs via the “randomized contractions” technique. Furthermore, although our application of this framework to handle one bag of the decomposition is more technical than in [7], due to the presence of the information for children bags, it uses no novel tools compared to [7]. Hence, we emphasize that our main technical contribution is the decomposition theorem (Theorem 1.2), with the fixed-parameter algorithm for MINIMUM BISECTION being its corollary via an involved application of known techniques.

In this extended abstract we prove the main technical contribution of the paper, namely Theorem 1.2. This is done in Section 3, after setting up notation and preliminary results in Section 2. We briefly sketch the dynamic programming algorithm for MINIMUM BISECTION in Section 4. For full exposition of this algorithm, as well as discussion on weighted extension and α -edge-separators we refer to the full version of the paper [8].

2. PRELIMINARIES

We use standard graph notation, see e.g. [9]. We use n and m to denote cardinalities of the vertex and edge sets, respectively, of a given graph provided it is clear from the context. We begin with some definitions and known results on separators and separations in graphs.

DEFINITION 2.1 (separator). *For two sets $X, Y \subseteq V(G)$ a set $W \subseteq V(G)$ is called an $X - Y$ separator if in $G \setminus W$ no connected component contains a vertex of X and a vertex of Y at the same time.*

DEFINITION 2.2 (separation). *A pair (A, B) where $A \cup B = V(G)$ is called a separation if $E(A \setminus B, B \setminus A) = \emptyset$. The order of a separation (A, B) is defined as $|A \cap B|$.*

DEFINITION 2.3 (important separator). *An inclusion-wise minimal $X - Y$ separator W is called an important $X - Y$ separator if there is no $X - Y$ separator W' with $|W'| \leq |W|$ and $R_{G \setminus W}(X \setminus W) \subsetneq R_{G \setminus W'}(X \setminus W')$, where $R_H(A)$ is the set of vertices reachable from A in the graph H .*

LEMMA 2.4 ([6, 21]). *For any two sets $S, T \subseteq V(G)$ there are at most 4^k important $S - T$ separators of size at most k and one can list all of them in $\mathcal{O}(4^k k(n + m))$ time.*

We proceed to define tree-decompositions. For a rooted tree T and a non-root node $t \in V(T)$, by $\text{parent}(t)$ we denote the parent of t in the tree T . For two nodes $u, t \in T$, we say that u is a *descendant* of t , denoted $u \preceq t$, if t lies on the unique path connecting u to the root. Note that every node is thus its own descendant.

DEFINITION 2.5 (tree decomposition). *A tree decomposition of a graph G is a pair (T, β) , where T is a rooted tree and $\beta : V(T) \rightarrow 2^{V(G)}$ is a mapping such that:*

- for each node $v \in V(G)$ the set $\{t \in V(G) | v \in \beta(t)\}$ induces a nonempty and connected subtree of T ,
- for each edge $e \in E(G)$ there exists $t \in V(T)$ such that $e \subseteq \beta(t)$.

The set $\beta(t)$ is called the *bag at t* , while sets $\beta(u) \cap \beta(v)$ for $uv \in E(T)$ are called *adhesions*. Following the notation from [15], for a tree decomposition (T, β) of a graph G we define auxiliary mappings $\sigma, \gamma : V(T) \rightarrow 2^{V(G)}$ as

$$\gamma(t) = \bigcup_{u \leq t} \beta(u),$$

$$\sigma(t) = \begin{cases} \emptyset & \text{if } t \text{ is the root of } T \\ \beta(t) \cap \beta(\text{parent}(t)) & \text{otherwise} \end{cases}$$

Finally, we proceed to the definition of unbreakability.

DEFINITION 2.6 (((q, k) -unbreakable set). *We say that a set A is (q, k) -unbreakable, if for any separation (X, Y) of order at most k we have $|(X \setminus Y) \cap A| \leq q$ or $|(Y \setminus X) \cap A| \leq q$. Otherwise A is (q, k) -breakable, and any separation (X, Y) certifying this is called a witnessing separation.*

Let us repeat the intuition on unbreakable sets from the introduction. If a set X of size at least $3q$ is (q, k) -unbreakable then removing any k vertices from G leaves almost all of X , except for at most q vertices, in the same connected component. In other words, one cannot separate two large chunks of X with a small separator.

Observe that if a set A is (q, k) -unbreakable in G , then any of its subset $A' \subseteq A$ is also (q, k) -unbreakable in G . Moreover, if A is (q, k) -unbreakable in G , then A is also (q, k) -unbreakable in any supergraph of G . For a small set A it is easy to efficiently verify whether A is (q, k) -unbreakable in G , or to find a witnessing separation, by checking, for all possible pairs (A_1, A_2) of disjoint subsets of A of size $q + 1$, whether A_1 and A_2 can be separated in G by a cut of size at most k .

LEMMA 2.7. *Given a graph G , a set $A \subseteq V(G)$ and an integer q one can check in $\mathcal{O}(|A|^{2q+2}k(n+m))$ time whether A is (q, k) -unbreakable in G , and if not, then find a separation (X, Y) of order at most k such that $|(X \setminus Y) \cap A| > q$ and $|(Y \setminus X) \cap A| > q$.*

3. DECOMPOSITION

We now restate our decomposition theorem in a slightly stronger form that will emerge from the proof.

THEOREM 3.1. *There is an $\mathcal{O}(2^{\mathcal{O}(k^2)}n^2m)$ time algorithm that, given a connected graph G together with an integer k , computes a tree decomposition (T, β) of G with at most n nodes such that the following conditions hold:*

- for each $t \in V(T)$, the graph $G[\gamma(t)] \setminus \sigma(t)$ is connected and $N(\gamma(t) \setminus \sigma(t)) = \sigma(t)$,
- for each $t \in V(T)$, the set $\beta(t)$ is $(2^{\mathcal{O}(k)}, k)$ -unbreakable in $G[\gamma(t)]$,
- for each non-root $t \in V(T)$, we have that $|\sigma(t)| \leq 2^{\mathcal{O}(k)}$ and $\sigma(t)$ is $(2k, k)$ -unbreakable in $G[\gamma(\text{parent}(t))]$.

3.1 Proof overview

We first give an overview of the proof of Theorem 3.1, ignoring the requirement that each adhesion is supposed to be $(2k, k)$ -unbreakable. As discussed in the introduction, this property is only used to improve the running time of the algorithm, and is not essential to establish fixed-parameter tractability.

We prove the decomposition theorem using a recursive approach, similar to the standard framework used for instance by Robertson and Seymour [24] or by Marx and Grohe [15]. That is, in the recursive step we are given a graph G together with a relatively small set $S \subseteq V(G)$ (i.e., of size bounded by $2^{\mathcal{O}(k)}$), and our goal is to construct a decomposition of G satisfying the requirements of Theorem 3.1 with an additional property that S is contained in the root bag of the decomposition. The intention is that the recursive step is invoked on some subgraph of the input graph, and the set S is the adhesion towards the decomposition of the rest of the graph.

Henceforth we focus on one recursive step, and consider three cases. In the base case, if $|S| \leq 3k$, we add an arbitrary vertex to S and repeat. In what follows, we assume $|S| > 3k$.

First, assume that S is $(2k, k)$ -breakable in G , and let (X, Y) be the witnessing separation. We proceed in a standard manner (cf. [24]): we create a root bag $A := S \cup (X \cap Y)$, for each connected component C of $G \setminus A$ recurse on $(N_G[C], N_G(C))$, and glue the obtained trees as children of the root bag. It is straightforward from the definition of the witnessing separation that in every recursive call we have $|N_G(C)| \leq |S|$. Moreover, clearly $|A| \leq |S| + k$ and hence A is appropriately unbreakable.

In the last, much more interesting case the adhesion S turns out to be $(2k, k)$ -unbreakable. Hence, any separation (X, Y) in G partitions S very unevenly: almost the entire set S , up to $\mathcal{O}(k)$ elements, lies on only one side of the separation. Let us call this side the “big” side, and the second side the “small” one.

The main idea now is as follows: if, for each $v \in V(G)$, we mark all important separators of size $\mathcal{O}(k)$ between v and S , then the marked vertices will separate all “small” sides of separations from the set S . Let B be the set of marked vertices and let A be the set of all vertices of G that are either in $B \cup S$, or are not separated from S by any of the considered important separator. We observe that the strong structure of important separators — in particular, the single-exponential bound on the number of important separators for one vertex v — allows us to argue that each connected component C of $G \setminus A$ that is separated by some important separator from S has only bounded number of neighbours in A . Moreover, the fact that we cut all “small” sides of separations implies that A is appropriately unbreakable in G . Hence, we may recurse, for each connected component C of $G \setminus A$ that is separated by some important separator from S , on $(N_G[C], N_G(C))$, and take A as a root bag.

The section is organised as follows. In Section 3.2 we define formally the notion of *chips*, that are parts of the graph cut out by important separators, and provide all the properties that play crucial role in Section 3.3. In Section 3.3 we also show how to proceed with the case S being unbreakable, that is, how extract the root bag containing S by cutting away all the chips. In Section 3.4 we perform some technical augmentation to ensure that the adhesions are $(2k, k)$ -unbreakable. Finally in Section 3.5 we combine

the obtained results and construct the main decomposition of Theorem 3.1.

3.2 Chips

In this subsection we define fragments of the graph which are easy to chip (i.e. cut out of the graph) from some given set of vertices S , and show their basic properties.

DEFINITION 3.2 (chips). *For a fixed set of vertices $S \subseteq V$, a subset $C \subseteq V$ is called a chip, if $G[C]$ is connected, $|N(C)| \leq 3k$, and $N(C)$ is an important $C - S$ separator.*

Let \mathcal{C} be the set of all inclusion-wise maximal chips.

The following lemma is straightforward from the definition of important separators.

LEMMA 3.3. *For any nonempty set $C \subseteq V(G)$ such that $G[C]$ is connected, the following conditions are equivalent.*

- (i) $N(C)$ is an important $C - S$ separator;
- (ii) for any $v \in C$, $N(C)$ is an important $v - S$ separator;
- (iii) there exists $v \in C$ such that $N(C)$ is an important $v - S$ separator.

Note also that for a connected set of vertices D and any important $D - S$ separator Z of size at most $3k$ that is disjoint with D , the set of vertices reachable from D in $G \setminus Z$ forms a chip. Lemmata 2.4 and 3.3 show how to enumerate inclusion-wise maximal chips.

LEMMA 3.4. *Given a set $S \subseteq V(G)$ one can compute the set \mathcal{C} of all inclusion-wise maximal chips in $\mathcal{O}(2^{\mathcal{O}(k)}n(n+m))$ time. In particular, $|\mathcal{C}| \leq 4^{3k}n$.*

PROOF. For any $v \in V$, we use Lemma 2.4 to enumerate the set \mathcal{Z}_v of all important $v - S$ separators of size at most $3k$. Recall that for any $Z \in \mathcal{Z}_v$, the set $R_{G \setminus Z}(v)$ is the vertex set of the connected component of $G \setminus Z$ containing v . Define $\mathcal{A}_v = \{R_{G \setminus Z}(v) : Z \in \mathcal{Z}_v\}$ and let \mathcal{C}_v be the set of inclusion-wise maximal elements of \mathcal{A}_v . By Lemma 3.3 we infer that if some chip $C \in \mathcal{A}_v$ is not inclusion-wise maximal, then there exists $C' \in \mathcal{A}_v$ such that $C \subsetneq C'$. Therefore, we have that $\mathcal{C} = \bigcup_{v \in V(G)} \mathcal{C}_v$.

As $|\mathcal{Z}_v| \leq 4^{3k}$ for any $v \in V(G)$, the bound on $|\mathcal{C}|$ follows. For each $v \in V(G)$, the sets \mathcal{Z}_v , \mathcal{A}_v and \mathcal{C}_v can be computed in $\mathcal{O}(2^{\mathcal{O}(k)}(n+m))$ time in a straightforward manner. The computation of $\mathcal{C} = \bigcup_{v \in V(G)} \mathcal{C}_v$ in $\mathcal{O}(2^{\mathcal{O}(k)}n(n+m))$ time can be done by inserting all the elements of $\bigcup_{v \in V(G)} \mathcal{C}_v$ into a prefix tree (trie), each in $\mathcal{O}(n)$ time, and ignoring encountered duplicates. \square

DEFINITION 3.5 (chips touching). *We say that two chips $C_1, C_2 \in \mathcal{C}$, $C_1 \neq C_2$, touch each other, denoted $C_1 \sim C_2$, if $C_1 \cap C_2 \neq \emptyset$ or $E(C_1, C_2) \neq \emptyset$.*

The following lemma provides an alternative definition of touching that we will find useful.

LEMMA 3.6. $C_1 \in \mathcal{C}$ touches $C_2 \in \mathcal{C}$ if and only if $N(C_1) \cap C_2 \neq \emptyset$.

PROOF. From right to left, if $v \in N(C_1) \cap C_2$ then there exists a neighbour u of v that belongs to C_1 , and consequently $uv \in E(C_1, C_2)$.

From left to right, first assume $C_1 \cap C_2 \neq \emptyset$. Since \mathcal{C} contains only inclusion-wise maximal chips, we have that $C_2 \setminus C_1 \neq \emptyset$. By the properties of chips, the graph $G[C_2]$ is connected, hence there is an edge between $C_2 \setminus C_1$ and $C_1 \cap C_2$ inside $G[C_2]$. This proves $N(C_1) \cap C_2 \neq \emptyset$.

In the other case, assume that $C_1 \cap C_2 = \emptyset$ but there exists $uv \in E(C_1, C_2)$ such that $u \in C_1$ and $v \in C_2$. Since $C_1 \cap C_2 = \emptyset$, it follows that $v \notin C_1$, and hence $v \in N(C_1) \cap C_2$. \square

The next result provides the most important tool for bounding the size of adhesions in the constructed decomposition.

LEMMA 3.7. *Any chip $C \in \mathcal{C}$ touches at most $3k \cdot 4^{3k}$ other chips of \mathcal{C} .*

PROOF. Assume that C touches some $C' \in \mathcal{C}$. By Lemma 3.6 there exists a vertex $v \in N(C) \cap C'$. Observe that since $N(C')$ is an important $C' - S$ separator, then $N(C')$ is also an important $v - S$ separator. By Lemma 2.4 there are at most 4^{3k} important $v - S$ separators of size at most $3k$. Since $|N(C)| \leq 3k$ (by the properties of chips), we infer that C touches at most $3k \cdot 4^{3k}$ chips from \mathcal{C} . \square

3.3 Local decomposition

Equipped with basic properties of chips we are ready to prove the main step of the decomposition part of the paper. In what follows we show that given a $(2k, k)$ -unbreakable set S of size bounded in k one can find a (potentially large) unbreakable part $A \subseteq V$ of the graph, such that $S \subseteq A$ and each connected component of $G \setminus A$ is adjacent to a small number of vertices of A . In what follows, let us define

$$\eta = 3k \cdot (3k \cdot 4^{3k} + 1), \quad \tau = (3k)^2 \cdot 8^{3k} + 2k.$$

THEOREM 3.8. *There is an $\mathcal{O}(2^{\mathcal{O}(k)}nm)$ time algorithm that, given a connected graph G together with an integer k and a $(2k, k)$ -unbreakable set $S \subseteq V(G)$, computes a set $A \subseteq V(G)$ such that:*

- (a) $S \subseteq A$,
- (b) for each connected component D of $G \setminus A$ we have $|N_G(D)| \leq \eta$,
- (c) A is (τ, k) -unbreakable in G , and
- (d) if $|S| > 3k$, $G \setminus S$ is connected and $N(V(G) \setminus S) = S$, then $S \neq A$.

PROOF. Let \mathcal{C} be the set of inclusion-wise maximal chips, enumerated by Lemma 3.4. We define

$$A = \left(\bigcap_{C \in \mathcal{C}} V(G) \setminus N[C] \right) \cup \bigcup_{C \in \mathcal{C}} N(C).$$

In the definition we assume that when \mathcal{C} is empty, then $A = V(G)$. The claimed running time of the algorithm follows directly from Lemma 3.4.

For property (a), note that no vertex of S is contained in a chip of \mathcal{C} , hence $S \subseteq A$. We now show property (d). Note that $N(V(G) \setminus S) = S$ and $|S| > 3k$ implies $S \neq V(G)$. Consequently, if $\mathcal{C} = \emptyset$, property (d) is straightforward. Otherwise, let $C \in \mathcal{C}$. Note that $|S| > 3k$ implies that $S \setminus N(C) \neq \emptyset$ and the connectivity of $G \setminus S$ together with $N(V(G) \setminus S) = S$ further implies that $N(C) \setminus S \neq \emptyset$. Consequently, $A \setminus S \neq \emptyset$ and property (d) is proven.

We now move to the remaining two properties.

CLAIM 3.9. For any connected component D of $G \setminus A$ there exists a chip $C_1 \in \mathcal{C}$ such that $D \subseteq C_1$.

PROOF. Observe that a vertex which is not contained in any chip belongs to the set A , as it is either contained in $N(C)$ for some $C \in \mathcal{C}$ or it belongs to $V(G) \setminus N[\mathcal{C}]$ for every $C \in \mathcal{C}$. Let D be an arbitrary connected component of $G \setminus A$ and let $v \in D$ be its arbitrary vertex. As $v \notin A$, there is a chip $C_v \in \mathcal{C}$ such that $v \in C_v$. Recall that by its definition the set A contains all the neighbours of all the chips in \mathcal{C} , hence $N(C_v) \cap D = \emptyset$ and by the connectivity of $G[D]$ we have $D \subseteq C_v$. \lrcorner

In the following claim we show that the set A satisfies property (b) of Theorem 3.8.

LEMMA 3.10. For any connected component D of $G \setminus A$ it holds that $|N(D)| \leq \eta$.

PROOF. Let D be an arbitrary connected component of $G \setminus A$. By Claim 3.9 there exists $C \in \mathcal{C}$ such that $D \subseteq C$. Intuitively each vertex of $N(D)$ belongs to the set A for one of two reasons: (i) it belongs to $N(C)$, or (ii) it is adjacent to a vertex of some other chip, which touches C . In both cases we show that there is only a bounded number of such vertices, which is formalized as follows.

Let v be any vertex of $N(D)$. Clearly $v \in N[\mathcal{C}]$, hence we either have $v \in N(C)$ or $v \in C$. Observe that if $v \in C$, then since $v \in A$, by the definition of the set A we have $v \in N(C')$ for some $C' \in \mathcal{C}$, $C' \neq C$. Since $v \in N(C') \cap C$, then C' touches C by Lemma 3.6. We infer that $N(D) \subseteq N(C) \cup \bigcup_{C' \in \mathcal{C}, C' \sim C} N(C')$. The claimed upper bound on $|N(D)|$ follows from Lemma 3.7. \square

Next, we show that the set A is unbreakable. A short an informal rationale behind this property is that everything what could be easily cut out of the graph was already excluded in the definition of A .

LEMMA 3.11. The set A is (τ, k) -unbreakable.

PROOF. Assume the contrary, and let (X, Y) be a witnessing separation, i.e. we have that $|X \cap Y| \leq k$, $|X \setminus Y \cap A| > \tau$ and $|Y \setminus X \cap A| > \tau$. Since S is $(2k, k)$ -unbreakable, then either $|X \setminus Y \cap S| \leq 2k$ or $|Y \setminus X \cap S| \leq 2k$. Without loss of generality we assume that $|X \setminus Y \cap S| \leq 2k$. Let us define a set $Q = (X \cap Y) \cup (X \cap S)$ and observe that $|Q| \leq 3k$.

Note that each connected component of $G \setminus Q$ is either entirely contained in $X \setminus Y$ or in $Y \setminus X$ (see Fig. 1a). Consider connected components of the graph $G \setminus Q$ that are contained in $X \setminus Y$ and observe that they contain at least $|((X \setminus Y) \cap A) \setminus S| > \tau - 2k$ vertices of A in total. Therefore, by grouping the connected components of $G \setminus Q$ contained in $X \setminus Y$ by their neighbourhoods in Q , we infer that there exists a set of connected components $\mathcal{D} = \{D_1, \dots, D_r\}$, such that $\forall_{1 \leq i, j \leq r} N_G(D_i) = N_G(D_j)$ and

$$\left| \bigcup_{i=1}^r D_i \cap A \right| > \frac{\tau - 2k}{2^{3k}} = (3k)^2 \cdot 4^{3k}. \quad (1)$$

We now need the following claim.

CLAIM 3.12. There is a subset $\mathcal{C}_0 \subseteq \mathcal{C}$, such that each $v \in \bigcup_{i=1}^r D_i \cap A$ belongs to some chip of \mathcal{C}_0 , and there are at most $3k \cdot 4^{3k}$ chips in \mathcal{C} that touch some chip of \mathcal{C}_0 .

PROOF. Observe that, for each $1 \leq i \leq r$, Q is a $D_i - S$ separator (see Fig. 1a) of size at most $3k$. Therefore, for each D_i there is an important $D_i - S$ separator of size at most $3k$ disjoint with D_i , hence each D_i is contained in some chip of \mathcal{C} . Consider two cases.

First, assume that for each $1 \leq i \leq r$ we have $D_i \in \mathcal{C}$. As \mathcal{C}_0 take $\{D_1, \dots, D_r\}$. Observe that as components D_i have the same neighbourhoods in G , then by Lemma 3.6 each chip of \mathcal{C} that touches some chip D_i touches also D_1 . Therefore, by Lemma 3.7 there are at most $3k \cdot 4^{3k}$ chips in \mathcal{C} that touch some chip of \mathcal{C}_0 .

In the second case assume that there exist $1 \leq i_0 \leq r$ and a chip $C \in \mathcal{C}$ such that $D_{i_0} \subsetneq C$. We shall prove that for each $1 \leq i \leq r$ we have $D_i \subseteq C$. Since C is connected and $C \setminus D_{i_0}$ is non-empty, we have that $C \cap N(D_{i_0}) \neq \emptyset$. Let $C' = C \cup \bigcup_{1 \leq i \leq r} D_i$. Clearly $C' \cap S = \emptyset$, and C' is connected since each component D_i is adjacent to every vertex of $C \cap N(D_{i_0})$. Moreover, as each D_i has the same neighbourhood in Q we have $|N(C')| \leq |N(C)| \leq 3k$ (see Fig. 1b). As \mathcal{C} contains only maximal chips we have $C' = C$ and hence $\bigcup_{1 \leq i \leq r} D_i \subseteq C$. Define \mathcal{C}_0 as $\{C\}$. By Lemma 3.7 a single chip touches at most $3k \cdot 4^{3k}$ other chips, which finishes the proof of Claim 3.12. \lrcorner

Let $v \in A \cap D_i$ for some $1 \leq i \leq r$. Since v is contained in some $C' \in \mathcal{C}_0$, we have $v \notin V(G) \setminus N[\mathcal{C}']$. Consequently, by the definition of the set A there exists a chip $C_v \in \mathcal{C}$ such that $v \in N(C_v)$. Note that $C' \neq C_v$ and $N(C_v) \cap C' \neq \emptyset$, hence by Lemma 3.6 C' touches C_v . By Claim 3.12 there are at most $3k \cdot 4^{3k}$ chips touching a chip of \mathcal{C}_0 . As each C_v satisfies $|N(C_v)| \leq 3k$, we infer that the number of vertices of A in $\bigcup_{1 \leq i \leq r} D_i$ is at most $(3k)^2 4^{3k}$, which contradicts (1) and finishes the proof of Lemma 3.11. \square

Lemma 3.10 and Lemma 3.11 ensure properties (b) and (c) of the set A , respectively. This concludes the proof of Theorem 3.8.

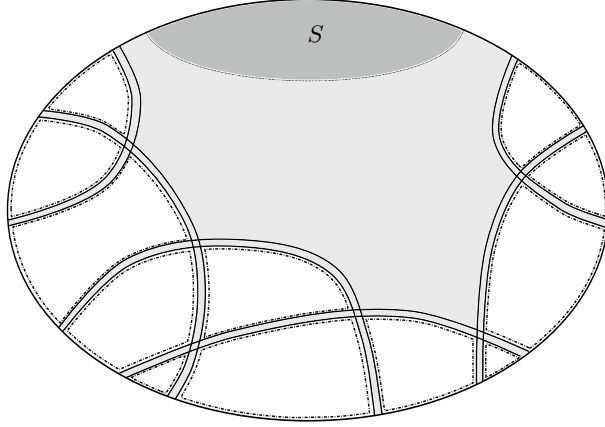
3.4 Strengthening unbreakability of adhesions

So far Theorem 3.8 provides us with a construction of the bag that meets almost all the requirements, apart from $(2k, k)$ -unbreakability of adhesions. For this reason, in this section we want to show that the set A from Theorem 3.8 can be extended to a set A' in such a way that for each connected component D of $G \setminus A'$ the set $N_G(D)$ is even $(2k, k)$ -unbreakable. During this extension we may weaken unbreakability of A' , but if we are careful enough then this loss will be limited to a single-exponential function of k . In the following we let

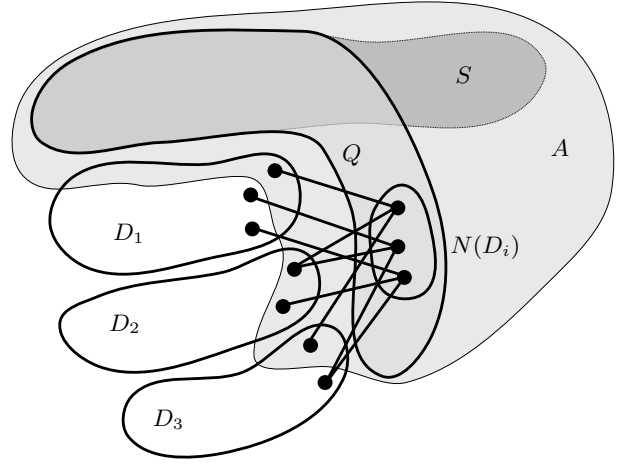
$$\tau' = \tau + \left(\binom{\tau + k}{2} \cdot k + k \right) \cdot k\eta.$$

LEMMA 3.13. Let G be a graph, and $L \subseteq V(G)$ be a subset of vertices of size at least $2k + 1$. Then one can in $\mathcal{O}(|L|^{4k+3} kn(n+m))$ time find a set L' , $L \subseteq L'$, such that $|L' \setminus L| \leq (|L| - 2k - 1) \cdot k$ and for each connected component D of $G \setminus L'$, we have that $|N_G(D)| \leq |L|$ and $N_G(D)$ is $(2k, k)$ -unbreakable in G .

PROOF. We prove the lemma by induction on $|L|$, with the following two base cases. If $L = V(G)$, clearly we may return $L' = L$. In the second base case we assume that L is $(2k, k)$ -unbreakable in G , which can be checked in



(a) Construction of the bag A



(b) Situation in the proof of Claim 3.12.

Figure 1: Illustrations of the proof of Theorem 3.8

$\mathcal{O}(|L|^{4k+2}k(n+m))$ time using Lemma 2.7. Then for each connected component D of $G \setminus L$ we have that $N_G(D) \subseteq L$, and thus $|N_G(D)| \leq |L|$ and $N_G(D)$ is also $(2k, k)$ -unbreakable in G . Hence we can set $L' = L$, and since $|L| \geq 2k + 1$, we have that $|L' \setminus L| \leq (|L| - 2k - 1) \cdot k$.

Now let us assume that L is $(2k, k)$ -breakable in G , and hence there exists a separation (X, Y) of G such that $|X \cap Y| \leq k$, $|(X \setminus Y) \cap L| > 2k$ and $|(Y \setminus X) \cap L| > 2k$, found by the algorithm of Lemma 2.7. We use inductively Lemma 3.13 for the pair $(G_1 = G[X], L_1 = (X \cap L) \cup (X \cap Y))$ and for the pair $(G_2 = G[Y], L_2 = (Y \cap L) \cup (X \cap Y))$, to obtain sets L'_1 and L'_2 , respectively. Note here that $|L_1|, |L_2| \geq 2k + 1$ and $|L_1|, |L_2| < |L|$. Define $L' = L'_1 \cup L'_2$. Each connected component D of $G \setminus L'$ is either a connected component of $G_1 \setminus L'_1$ and is adjacent only to L'_1 , or is a connected component of $G_2 \setminus L'_2$ and is adjacent only to L'_2 . Assume without loss of generality the first case. By inductive assumption we infer that $|N_{G_1}(D)| \leq |L_1|$ and $N_{G_1}(D)$ is $(2k, k)$ -unbreakable in G_1 , and since $N_{G_1}(D) = N_G(D)$, $|L_1| < |L|$, and G_1 is a subgraph of G , then it follows that $|N_G(D)| \leq |L|$ and $N_G(D)$ is $(2k, k)$ -unbreakable in G . It remains to argue that the cardinality of $L' \setminus L$ is not too large. Observe that

$$L' \setminus L \subseteq (L'_1 \setminus L_1) \cup (L'_2 \setminus L_2) \cup (X \cap Y);$$

therefore, by induction we have

$$\begin{aligned} |L' \setminus L| &\leq (|L_1| - 2k - 1) \cdot k + (|L_2| - 2k - 1) \cdot k + k \\ &\leq (|L_1| + |L_2| - 4k - 1) \cdot k \\ &\leq (|L| + 2|X \cap Y| - 4k - 1) \cdot k \\ &\leq (|L| - 2k - 1) \cdot k. \end{aligned}$$

Let us now bound the running time of the recursion. Clearly, as the size of the set L decreases in the recursive calls, the depth of the recursion is at most $|L|$. Moreover, note that any vertex may appear in $V(G) \setminus L$ in at most one recursive call (G, L) at any fixed level of the recursion tree. Hence, there are at most $|L|n$ recursive calls that do not correspond to the first base case, and, consequently, at most $2|L|n + 1$ recursive calls in total. As each recursive call takes $\mathcal{O}(|L|^{4k+2}k(n+m))$ time, the promised running time bound follows. \square

THEOREM 3.14. *There is an $\mathcal{O}(2^{\mathcal{O}(k^2)}nm)$ time algorithm that, given a connected graph G together with an integer k and a $(2k, k)$ -unbreakable set S , computes a set $A' \subseteq V(G)$ such that:*

- (a) $S \subseteq A'$,
- (b) for each connected component D of $G \setminus A'$ the set $N_G(D)$ is $(2k, k)$ -unbreakable, and $|N_G(D)| \leq \eta$,
- (c) A' is (τ', k) -unbreakable in G ,
- (d) moreover, if $|S| > 3k$, $G \setminus S$ is connected and $N(V(G) \setminus S) = S$, then $S \neq A'$.

PROOF. We start by finding the set A by running the algorithm Theorem 3.8. Next, for each connected component D of $G \setminus A$ using Lemma 2.7 we check whether $N(D)$ is $(2k, k)$ -breakable in G . By Theorem 3.8, the cardinality of $N(D)$ is bounded by η , hence all tests take total time $\mathcal{O}(\eta^{4k+2}knm) = \mathcal{O}(2^{\mathcal{O}(k^2)}nm)$ time. Note that if $N(D)$ is $(2k, k)$ -breakable in G , then in particular $|N(D)| > 2k$, hence we can use Lemma 3.13 for the pair $(G[N[D]], L_D = N(D))$; let L'_D be the obtained set. As $|L_D| \leq \eta$, the algorithm of Lemma 3.13 runs in $\mathcal{O}(\eta^{4k+3}k|N[D]|m)$ time for a fixed component D , and total time taken by calls to Lemma 3.13 is:

$$\begin{aligned} &\sum_D \mathcal{O}(\eta^{4k+3}k(|D| + |N(D)|)m) \\ &\leq \mathcal{O}(\eta^{4k+3}km) \cdot \left(\sum_D |D| + \sum_D \eta \right) \\ &= \mathcal{O}(\eta^{4k+4}knm) = \mathcal{O}(2^{\mathcal{O}(k^2)}nm). \end{aligned}$$

In the case when $N(D)$ is $(2k, k)$ -unbreakable, let $L_D = L'_D = N(D)$. Define $A' = A \cup (\bigcup_D L'_D)$, where the union is taken over all the connected components D of $G \setminus A$.

Since $S \subseteq A \subseteq A'$, we have that $S \subseteq A'$, and, moreover, the property (d) follows directly from property (d) of Theorem 3.8. Moreover, as $|L_D| \leq \eta$ for each connected component D of $G \setminus A$, then by Lemma 3.13 for each connected component D' of $G \setminus A'$ we also have $|N_G(D')| \leq \eta$. The fact that $N_G(D')$ is $(2k, k)$ -unbreakable in G follows

directly from Lemma 3.13. It remains to show that A' is (τ', k) -unbreakable in G .

Consider any separation (X, Y) of G of order at most k . By Theorem 3.8 the set A is (τ, k) -unbreakable, hence either $|(X \setminus Y) \cap A| \leq \tau$ or $|(Y \setminus X) \cap A| \leq \tau$, and without loss of generality assume the former. As (X, Y) is an arbitrary separation of order at most k , to show that A' is (τ', k) -unbreakable it suffices to prove that $|(X \setminus Y) \cap (A' \setminus A)| \leq \binom{\tau+k}{2} \cdot k + k \cdot k\eta$.

Note that $A' \setminus A \subseteq \bigcup_D L'_D \setminus L_D$. As for each D we have $|L'_D \setminus L_D| \leq k\eta$ by Lemma 3.13, to finish the proof of Theorem 3.14 we are going to show that there are at most $\binom{\tau+k}{2} \cdot k + k$ connected components D of $G \setminus A$ such that $D \cap (X \setminus Y) \neq \emptyset$ and $L'_D \neq L_D$. As (X, Y) is of order at most k , there are at most k connected components D of $G \setminus A$ intersecting $X \cap Y$. Hence we restrict our attention to connected components D of $G \setminus A$, such that $D \subseteq X \setminus Y$, which in turn implies $N(D) \subseteq A \cap X$. Recall that if $L'_D \neq L_D$ for such a connected component D , then $N(D)$ is $(2k, k)$ -breakable in G , and hence there exist two vertices $v_a, v_b \in N(D) \subseteq A \cap X$, such that the minimum vertex cut separating v_a and v_b in G is at most k . However, such a pair of vertices v_a, v_b may be simultaneously contained in neighbourhoods of at most k connected components D , since each component D adjacent both to v_a and to v_b contributes with at least one path between them. As $|A \cap X| \leq \tau + k$, the theorem follows. \square

3.5 Constructing a decomposition

In this subsection we prove our main decomposition theorem, i.e. Theorem 3.1. However, for the inductive approach to work we need a bit stronger statement, where additionally we have a set $S \subseteq V(G)$ that has to be contained in the top bag of the tree decomposition. Note that Theorem 3.1 follows from the following by setting $S = \emptyset$.

THEOREM 3.15. *There is an $\mathcal{O}(2^{\mathcal{O}(k^2)} n^2 m)$ time algorithm that, given a connected graph G together with an integer k and a set $S \subseteq V(G)$ of size at most η such that $G \setminus S$ is connected and $N(V(G) \setminus S) = S$, computes a tree decomposition (T, β) such that S is contained in the top bag of the tree decomposition, and the following conditions are satisfied:*

- (i) for each $t \in V(T)$, the graph $G[\gamma(t)] \setminus \sigma(t)$ is connected and $N(\gamma(t) \setminus \sigma(t)) = \sigma(t)$,
- (ii) for each $t \in V(T)$, the set $\beta(t)$ is (τ', k) -unbreakable in $G[\gamma(t)]$,
- (iii) for each non-root $t \in V(T)$, we have that $|\sigma(t)| \leq \eta$ and $\sigma(t)$ is $(2k, k)$ -unbreakable in $G[\gamma(\text{parent}(t))]$.
- (iv) $|V(T)| \leq |V(G) \setminus S|$.

PROOF. If $|V(G)| \leq \tau'$, the algorithm creates a single bag containing the entire $V(G)$. It is straightforward to verify that such a decomposition satisfies all the required properties. Thus, in the rest of the proof we assume that $|V(G)| > \tau'$, in particular, $|V(G)| > 3k$.

Define $S' = S$ and, if $|S| \leq 3k$, add $3k + 1 - |S|$ arbitrary vertices of $V(G) \setminus S$ to S' . Note that, as $\eta > 3k$, we have $3k \leq |S'| \leq \eta$.

We now define a set A' as follows. First, we verify, using Lemma 2.7, whether S' is $(2k, k)$ -breakable in G or not. If it turns out to be $(2k, k)$ -breakable in G , we apply Lemma 3.13 to the pair (G, S') , obtaining a set which we denote by A' .

Otherwise, we can use Theorem 3.14 on the pair (G, S') to obtain a set A' . Note that in both cases $S \subseteq S' \subseteq A'$ and all computations so far take $\mathcal{O}(2^{\mathcal{O}(k^2)} nm)$ time in total.

Regardless of the way the set A' was obtained, we proceed with it as follows. For each connected component D of $G \setminus A'$, we use Theorem 3.15 inductively for the graph $G[N[D]]$ and $S_D = N(D)$. Let us now verify that (a) each S_D is $(2k, k)$ -unbreakable in G , (b) that the assumptions of the theorem are satisfied, and (c) that the recursive call is applied to a strictly smaller instance in the sense defined in the following.

For the first two claims, if S is $(2k, k)$ -breakable, Lemma 3.13 asserts that $|S_D| \leq |S| \leq \eta$ and S_D is $(2k, k)$ -unbreakable in G . Otherwise, property (b) of Theorem 3.14 ensures that $|S_D| \leq \eta$ and S_D is $(2k, k)$ -unbreakable in G . The other assumptions on the set S_D in the recursive calls follow directly from the definitions of these calls.

For the last claim, we show that either $|N[D]| < |V(G)|$ or $N[D] = V(G)$ and $|D| < |V(G) \setminus S|$. Assume the contrary, that is, $D = V(G) \setminus S$ and $N(D) = S_D = S = S' = A'$. In particular, as S_D is $(2k, k)$ -unbreakable in G , the set A' was obtained using Theorem 3.14. However, as $|S'| > 3k$, property (d) of Theorem 3.14 ensures that $S' \subsetneq A'$, a contradiction.

Let (T_D, β_D) be the tree decomposition obtained in the recursive call for the pair $(G[N[D]], S_D)$. Construct a tree decomposition (T, β) , by creating an auxiliary node r , which will be the root of T , and attach T_D to r , by making the root r_D of T_D a child of r in T . Finally, define $\beta = \bigcup_D \beta_D$ and set $\beta(r) = A'$. A straightforward check shows that (T, β) is indeed a valid tree decomposition. We now proceed to verify its promised properties.

Clearly, $S \subseteq S' \subseteq A'$. For any connected component D of $G \setminus A'$, note that $\gamma(r_D) = N[D]$ and $\sigma(r_D) = N(D) = S_D$. This, together with inductive assumptions on recursive calls, proves properties (i) and (iii).

If A' is obtained using Lemma 3.13, then $|A'| \leq k|S'| \leq k\eta < \tau'$, hence clearly $A' = \beta(r)$ is (τ', k) -unbreakable. In the other case, property (c) of Theorem 3.14 ensures the unbreakability promised in property (ii).

It remains to bound the number of bags of (T, β) ; as each bag is processed in $\mathcal{O}(2^{\mathcal{O}(k^2)} nm)$ time this would also prove the promised running time bound. Note that by property (iv) for the recursive calls we have that $|V(T_D)| \leq |D|$ and, consequently, $|V(T)| \leq |V(G) \setminus A'| + 1 = |V(G) \setminus S| + 1 - |A' \setminus S|$. To finish the proof of property (iv) it suffices to show that $S \subsetneq A'$. If $S \subsetneq S'$, the claim is straightforward. Otherwise, if $S = S'$ is $(2k, k)$ -breakable, then Lemma 3.13 cannot return $A' = S'$ as $G \setminus S'$ is connected and $N(V(G) \setminus S') = S'$ is not $(2k, k)$ -unbreakable. Consequently, $S' \subsetneq A'$ in this case. In the remaining case, when $S = S'$ is $(2k, k)$ -unbreakable, property (d) of Theorem 3.14 ensures that $S' \subsetneq A'$. This finishes the proof of Theorem 3.15. \square

4. BISECTION

In this section we give a very brief overview on how to prove Theorem 3.1, that is, how to solve MINIMUM BISECTION by dynamic programming on the tree decomposition obtained from Theorem 1.2.

The information we need to compute for each bag t of the tree decomposition is quite natural. In what follows, we interpret a partition of the vertex set of G into two sides as a colouring of vertices into colours **B** and **W**; any edge

with endpoints of different colours is called a *cut edge*. In the dynamic programming algorithm, for each node t of the tree decomposition, for each colouring $\text{col} : \sigma(t) \rightarrow \{\mathbf{B}, \mathbf{W}\}$ and for each integer μ we would like to find a colouring of $\gamma(t)$ that extends col , colours exactly μ vertices with \mathbf{B} and minimizes the number of cut edges that are not present in $G[\sigma(t)]$. Moreover, we can discard any such colouring that has more than k such cut edges.

The essence of computation task for one bag of the decomposition is captured by the following definition.

HYPERGRAPH PAINTING^a

Input: Positive integers k, b, d, q , a multihypergraph H with hyperedges of size at most d , a partial function $\text{col}_0 : V(H) \rightarrow \{\mathbf{B}, \mathbf{W}\}$, and a function $f_F : \{\mathbf{B}, \mathbf{W}\}^F \times \{0, \dots, b\} \rightarrow \{0, 1, \dots, k, \infty\}$ for each $F \in E(H)$.

Goal: For each $0 \leq \mu \leq b$, compute the value w_μ ,

$$w_\mu = \min_{\text{col} \supseteq \text{col}_0, (a_F)_{F \in E(H)}} \sum_{F \in E(H)} f_F(\text{col}|_F, a_F),$$

where the minimum is taken over colourings $\text{col} : V(H) \rightarrow \{\mathbf{B}, \mathbf{W}\}$ extending col_0 and partitions of μ into non-negative integers $\mu = \sum_{F \in E(H)} a_F$, and the sum attains value ∞ whenever its value exceeds k .

^aWe are intentionally not using the name HYPERGRAPH COLOURING, as it has an established, and different, meaning.

In each node t of the decomposition, we create auxiliary HYPERGRAPH PAINTING instances as follows. We set $b = n$. The partial colouring col_0 iterates through all colourings of the adhesion $\sigma(t)$. Each edge F of the multihypergraph H corresponds either to a vertex, an edge of $G[\beta(t)]$, or an adhesion $\sigma(t')$ for some child t' of t .

In the case of F corresponding to an adhesion $\sigma(t')$, the function f_F represents the already computed information for the subtree rooted at t' . That is, $f_F(\text{col}_F, a_F)$ is the minimum number of cut edges not present in $G[\beta(t)]$ that can be attained by a colouring of $\gamma(t')$ extending col_F that colours exactly a_F vertices of $\gamma(t') \setminus \sigma(t')$ with \mathbf{B} . If this minimum exceeds k , we set $f_F(\text{col}_F, a_F) = \infty$.

In the case of F corresponding to an edge uv of $G[\beta(t)]$, we define f_F in a straightforward manner to fit into the same interpretation and count the number of cut edges inside $G[\beta(t)]$. That is, $f_F((\mathbf{B}, \mathbf{B}), 0) = f_F((\mathbf{W}, \mathbf{W}), 0) = 0$, $f_F((\mathbf{B}, \mathbf{W}), 0) = f_F((\mathbf{W}, \mathbf{B}), 0) = 1$ and the value ∞ is attained otherwise.

Finally, the edges F for single vertices are designed to count the number of vertices coloured \mathbf{B} in $\beta(t)$: $f_F(\mathbf{B}, 1) = f_F(\mathbf{W}, 0) = 1$ and the value ∞ is attained otherwise.

By the properties of the decomposition obtained from Theorem 1.2, the instances $(k, b, d, q, H, \text{col}_0, (f_F)_{F \in E(H)})$ of HYPERGRAPH PAINTING encountered by the algorithm satisfy the following properties (henceforth called *proper instances*):

- **(local unbreakability)**, for each $F \in E(H)$, each $\text{col} : F \rightarrow \{\mathbf{B}, \mathbf{W}\}$ marking more than $3k$ vertices of each colour, i.e. $|\text{col}^{-1}(\mathbf{B})|, |\text{col}^{-1}(\mathbf{W})| > 3k$, and each $0 \leq \mu \leq b$ the value $f_F(\text{col}, \mu)$ equals ∞ ,
- **(connectivity)**, for each $F \in E(H)$, each $\text{col} : F \rightarrow \{\mathbf{B}, \mathbf{W}\}$ marking at least one vertex with each colour,

i.e. $|\text{col}^{-1}(\mathbf{B})|, |\text{col}^{-1}(\mathbf{W})| > 0$, and each $0 \leq \mu \leq b$ the value $f_F(\text{col}, \mu)$ is non-zero,

- **(global unbreakability)** for each $0 \leq \mu \leq b$ such that $w_\mu < \infty$ there is a witnessing colouring $\text{col} : V(H) \rightarrow \{\mathbf{B}, \mathbf{W}\}$, which colours at most q vertices with one of the colours, i.e. $\min(|\text{col}^{-1}(\mathbf{B})|, |\text{col}^{-1}(\mathbf{W})|) \leq q$.

Note that, by local unbreakability, for proper instances each function f_F can be represented by at most $(2 \sum_{i=0}^{3k} d^i) \cdot (b+1) \leq 4(b+1)d^{3k}$ values which are smaller than ∞ .

By a quite involved application of the ‘‘randomized contractions’’ framework we prove the following (the \mathcal{O}^* notation suppresses factors polynomial in the input size):

LEMMA 4.1. *There is an $\mathcal{O}^*(q^{\mathcal{O}(k)} \cdot d^{\mathcal{O}(k^2)})$ time algorithm solving the HYPERGRAPH PAINTING problem for proper instances.*

Hence, Lemma 4.1 allows us to perform all necessary computations in a bottom-to-top manner, eventually resolving the input MINIMUM BISECTION instance. A detailed analysis of the polynomial factor of Lemma 4.1, together with the sparsification technique of Nagamochi and Ibaraki [22], gives the running time bound promised by Theorem 1.1.

5. CONCLUSIONS

In this paper we have settled the parameterized complexity of MINIMUM BISECTION. Our algorithm also works in the more general setting when the edges are weighted, when the vertex set is to be partitioned into a constant number of parts rather than only two, and when the cardinality of each of the parts is given as input.

The core component of our algorithm is a new decomposition theorem for general graphs. Intuitively, we show that it is possible to partition any graph in a tree-like manner using small separators so that each of the resulting pieces cannot be broken any further. This uncovered structure is very natural in the context of cut-problems, and we strongly believe that our decomposition theorem will find many further algorithmic applications.

Having settled the parameterized complexity of MINIMUM BISECTION it is natural to ask whether the problem also admits a *polynomial kernel*, i.e. a polynomial-time preprocessing algorithm that would reduce the size of the input graph to some polynomial of the budget k . This question, however, has been already resolved by van Bevern et al. [25], who showed that MINIMUM BISECTION does not admit a polynomial kernel unless $\text{coNP} \subseteq \text{NP/poly}$. We conclude with a few intriguing open questions.

- Can the running time of our algorithm be improved? In particular, does there exist an algorithm for MINIMUM BISECTION with running time $2^{\mathcal{O}(k)} n^{\mathcal{O}(1)}$, that is with linear dependence on the parameter in the exponent?
- The running time dependence of our algorithm on the input size is roughly cubic. Is it possible to obtain a fixed-parameter tractable algorithm with quadratic, or even nearly-linear running time dependence on input size? Note that the best known algorithm for graphs of bounded treewidth has quadratic dependence on the input size [16].

- (c) Are the parameters in the decomposition theorem tight? For example, is it possible to lower the adhesion size from $2^{\mathcal{O}(k)}$ to polynomial in k ? Similarly, can one make the bags $(k^{\mathcal{O}(1)}, k)$ -unbreakable rather than $(2^{\mathcal{O}(k)}, k)$ -unbreakable? Is it possible to achieve both simultaneously? We remark that if the latter question has a positive answer, this would improve the parameter dependence in the running time of our algorithm for MINIMUM BISECTION to $k^{\mathcal{O}(k)}$.
- (d) Is it possible to compute our decomposition faster, say in $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$ or even in $2^{\mathcal{O}(k)} n^{\mathcal{O}(1)}$ time? Currently the main bottleneck is the very simple Lemma 2.7, which we are unable to speed up.

6. ACKNOWLEDGMENTS

We would like to thank Rajesh Chitnis, Fedor Fomin, MohammadTaghi Hajiaghayi and M. S. Ramanujan for earlier discussions on this subject.

M. Cygan is supported by the Polish National Science Centre, grant n. UMO-2013/09/B/ST6/03136. D. Lokshтанov is supported by the BeHard grant under the recruitment programme of the of Bergen Research Foundation. The research of M. Pilipczuk and M. Pilipczuk leading to these results has received funding from the European Research Council under the European Union’s Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 267959. S. Saurabh is supported by PARAPPROX, ERC starting grant no. 306992.

We also acknowledge the very inspiring atmosphere of the Dagstuhl seminar 13121, where the authors discussed the core ideas leading to this work.

7. REFERENCES

- [1] T. N. Bui, S. Chaudhuri, F. T. Leighton, and M. Sipser. Graph bisection algorithms with good average case behavior. *Combinatorica*, 7(2):171–191, 1987.
- [2] T. N. Bui, C. Heigham, C. Jones, and F. T. Leighton. Improving the performance of the Kernighan-Lin and simulated annealing graph bisection algorithms. In *DAC*, pages 775–778, 1989.
- [3] T. N. Bui and A. Peck. Partitioning planar graphs. *SIAM J. Comput.*, 21(2):203–215, 1992.
- [4] T. N. Bui and L. C. Strite. An ant system algorithm for graph bisection. In *GECCO*, pages 43–51, 2002.
- [5] J. Carmesin, R. Diestel, F. Hundertmark, and M. Stein. Connectivity and tree structure in finite graphs. *CoRR*, abs/1105.1611, 2011.
- [6] J. Chen, Y. Liu, and S. Lu. An improved parameterized algorithm for the minimum node multiway cut problem. *Algorithmica*, 55(1):1–13, 2009.
- [7] R. H. Chitnis, M. Cygan, M. Hajiaghayi, M. Pilipczuk, and M. Pilipczuk. Designing FPT algorithms for cut problems using randomized contractions. In *FOCS*, pages 460–469, 2012.
- [8] M. Cygan, D. Lokshтанov, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. Minimum bisection is fixed parameter tractable. *CoRR*, abs/1311.2563, 2013.
- [9] R. Diestel. *Graph Theory*. Springer, 2005.
- [10] U. Feige and R. Krauthgamer. A polylogarithmic approximation of the minimum bisection. *SIAM J. Comput.*, 31(4):1090–1118, 2002.
- [11] U. Feige, R. Krauthgamer, and K. Nissim. Approximating the minimum bisection size (extended abstract). In *STOC*, pages 530–536, 2000.
- [12] U. Feige and M. Mahdian. Finding small balanced separators. In *STOC*, pages 375–384, 2006.
- [13] M. R. Garey and D. S. Johnson. *Computers and intractability*, volume 174. Freeman New York, 1979.
- [14] M. Grohe, K. Kawarabayashi, D. Marx, and P. Wollan. Finding topological subgraphs is fixed-parameter tractable. In *STOC*, pages 479–488, 2011.
- [15] M. Grohe and D. Marx. Structure theorem and isomorphism test for graphs with excluded topological subgraphs. In *STOC*, pages 173–192, 2012.
- [16] K. Jansen, M. Karpinski, A. Lingas, and E. Seidel. Polynomial time approximation schemes for max-bisection on planar and geometric graphs. *SIAM J. Comput.*, 35(1):110–119, 2005.
- [17] K. Kawarabayashi and M. Thorup. The minimum k -way cut of bounded size is fixed-parameter tractable. In *FOCS*, pages 160–169, 2011.
- [18] S. Khot and N. K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into l_1 . In *FOCS*, pages 53–62, 2005.
- [19] D. Marx. Parameterized graph separation problems. *Theor. Comput. Sci.*, 351(3):394–406, 2006.
- [20] D. Marx, B. O’Sullivan, and I. Razgon. Finding small separators in linear time via treewidth reduction. *ACM Transactions on Algorithms*, 9(4):30, 2013.
- [21] D. Marx and I. Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. In L. Fortnow and S. P. Vadhan, editors, *STOC*, pages 469–478. ACM, 2011.
- [22] H. Nagamochi and T. Ibaraki. A Linear-Time Algorithm for Finding a Sparse k -Connected Spanning Subgraph of a k -Connected Graph. *Algorithmica*, 7(5&6):583–596, 1992.
- [23] H. Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *STOC*, pages 255–264, 2008.
- [24] N. Robertson and P. D. Seymour. Graph minors XIII. The disjoint paths problem. *J. Comb. Theory, Ser. B*, 63(1):65–110, 1995.
- [25] R. van Bevern, A. E. Feldmann, M. Sorge, and O. Suchý. On the parameterized complexity of computing graph bisections. In A. Brandstädt, K. Jansen, and R. Reischuk, editors, *WG*, volume 8165 of *Lecture Notes in Computer Science*, pages 76–87. Springer, 2013.