Solving connectivity problems parameterized by treewidth in single exponential time (Extended abstract)

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

Jesper Nederlof
Department of Informatics
University of Bergen
Bergen, Norway
Jesper.Nederlof@ii.uib.no

Marcin Pilipczuk
Institute of Informatics
University of Warsaw
Warsaw, Poland
malcin@mimuw.edu.pl

Michał Pilipczuk
Faculty of Mathematics, Informatics and Mechanics, University of Warsaw
Warsaw, Poland
michal.pilipczuk@students.mimuw.edu.pl

Johan M. M. van Rooij
Department of Information and Computing Sciences, Utrecht University
Utrecht, The Netherlands
jmmrooij@cs.uu.nl

Jakub Onufry Wojtaszczyk
Google Inc.
Cracow, Poland
onufry@google.com

Abstract— For the vast majority of local problems on graphs of small treewidth (where by local we mean that a solution can be verified by checking separately the neighbourhood of each vertex), standard dynamic programming techniques give $c^{\mathbf{tw}}|V|^{O(1)}$ time algorithms, where $\mathbf{tw}$ is the treewidth of the input graph $G = (V, E)$ and c is a constant. On the other hand, for problems with a global requirement (usually connectivity) the best–known algorithms were naive dynamic programming schemes running in at least $\mathbf{tw}^{\mathbf{tw}}$ time.

We breach this gap by introducing a technique we named Cut&Count that allows to produce $c^{\mathbf{tw}}|V|^{O(1)}$ time Monte Carlo algorithms for most connectivity-type problems, including HAMILTONIAN PATH, STEINER TREE, FEEDBACK VERTEX SET and CONNECTED DOMINATING SET. These results have numerous consequences in various fields, like parameterized complexity, exact and approximate algorithms on planar and H-minor-free graphs and exact algorithms on graphs of bounded degree. The constant c in our algorithms is in all cases small, and in several cases we are able to show that improving those constants would cause the Strong Exponential Time Hypothesis to fail.

In contrast to the problems aiming to *minimize* the number of connected components that we solve using Cut&Count as mentioned above, we show that, assuming the Exponential Time Hypothesis, the aforementioned gap cannot be breached for some problems that aim to *maximize* the number of connected components like CYCLE PACKING.

Keywords-treewidth; fixed parameter tractability; randomized algorithms; exact algorithms;

1. INTRODUCTION AND NOTATION

The notion of *treewidth* was introduced independently by Rose in 1974 [40] (under the name of partial k-tree) and in 1984 by Robertson and Seymour [39], and in many cases proved to be a good measure of the intrinsic difficulty of various NP-hard problems on graphs, and a useful tool for attacking those problems. Many of them can be efficiently solved through dynamic programming if we assume the input graph to have bounded treewidth. For example, an expository algorithm to solve VERTEX COVER and INDEPENDENT SET running in time $4^{\mathbf{tw}(G)}|V|^{O(1)}$ is described in the algorithms textbook by Kleinberg and Tardos [30], while the book of Niedermeier [37] on fixed-parameter algorithms presents an algorithm with running time $2^{\mathbf{tw}(G)}|V|^{O(1)}$.

The interest in algorithms for graphs of bounded treewidth stems from their utility: such algorithms are used as subroutines in a variety of settings. Amongst them prominent are approximation algorithms [2], [7], [12], [18] and parametrized algorithms [16], [20] for a vast number of problems on planar, bounded-genus and H-minor-free graphs, including VERTEX COVER, DOMINATING SET and INDEPENDENT SET; there are applications for parametrized algorithms in general graphs [35], [42] for problems like CONNECTED VERTEX COVER and CUTWIDTH; and exact algorithms [20], [44] such as MINIMUM MAXIMAL MATCHING and DOMINATING SET.

In many cases, where the problem to be solved is "local" (loosely speaking this means that the property of the object to be found can be verified by checking separately the neighbourhood of each vertex), matching upper and lower bounds for the runtime of the optimal solution are known. For instance for the aforementioned $2^{\mathbf{tw}(G)}|V|^{O(1)}$ algorithm for VERTEX COVER there is a matching lower bound — unless the Strong Exponential Time Hypothesis (see [26]) fails, there is no algorithm for VERTEX COVER running faster than $(2-\varepsilon)^{\mathbf{tw}(G)}$ for any $\varepsilon > 0$ (see [32]).

On the other hand, when the problem involves some sort of a "global" constraint — e.g., connectivity — the best known algorithms usually have a runtime on the order

The full version of this work can be found at arXiv [11]. The second author was supported by the Norwegian Research Council. The authors from the University of Warsaw were partially supported by Polish Ministry of Science grant no. N206 567140 and Foundation for Polish Science.

of $2^{O(\mathbf{tw}(G)\log \mathbf{tw}(G))}|V|^{O(1)}$. In these cases the typical dynamic programming routine has to keep track of all the ways in which the solution can traverse the corresponding separator of the tree decomposition, that is $\Omega(l^l)$ on the size l of the separator, and therefore of treewidth. This obviously implies weaker results in the applications mentioned above. This problem was observed, for instance, by Dorn, Fomin and Thilikos [16], [15] and by Dorn et al. in [17], and the question whether the known $2^{O(\mathbf{tw}(G)\log \mathbf{tw}(G))}|V|^{O(1)}$ parametrized algorithms for HAMILTONIAN PATH, CONNECTED VERTEX COVER and CONNECTED DOMINATING SET are optimal was explicitly asked by Lokshtanov, Marx and Saurabh [33].

The $2^{O(\mathbf{tw}(G)\log \mathbf{tw}(G))}$ dynamic programming routines for connectivity problems were thought to be optimal, because in these routines the dynamic programming table reflects the whole information that needs to be memoized in order to continue the computation. For every two distinct tables at some bag of the tree decomposition there exists a possible future on which the algorithm should behave differently. This resembles the notion of Myhill-Nerode equivalence classes [25], which, in a variety of settings, define the minimal automaton for a given problem. Hence, shrinking the size of the dynamic programming table would be, in some sense, trying to reduce the size of the minimal automaton. From this point of view the results of this paper come as a significant surprise.

1.1. Our results

In this paper we introduce a technique we named "Cut&Count". Briefly stated, we first reduce the original problem to the task of counting possibly disconnected "cut solutions" modulo 2 by (i) making sure that the number of disconnected cut solutions is always even and (ii) using randomization to make sure that the number of connected cut solutions is odd iff there is a solution. The reduction is performed in such a way that counting cut solutions is a local problem and can be done sufficiently fast by standard dynamic programming.

For most problems involving a global constraint our technique gives a randomized algorithm with runtime $c^{\mathbf{tw}(G)}|V|^{O(1)}$. In particular we are able to give such algorithms for the three problems mentioned in [33], as well as for all the other sample problems mentioned in [15]. Moreover, the constant c is in all cases well defined and small. The randomization we mention comes from the usage of the Isolation Lemma [36]. This gives us Monte Carlo algorithms with a one-sided error. The formal statement of a typical result is as follows:

Theorem 1.1. *There exists a randomized algorithm, which given a graph $G = (V, E)$, a tree decomposition of G of width t and a number k in $3^t|V|^{O(1)}$ time either states that there exists a connected vertex cover of size at most k in G, or that it could not verify this hypothesis. If there indeed exists such a cover, the algorithm will return "unable to verify" with probability at most $1/2$.*

We call an algorithm as in Theorem 1.1 an algorithm with *false negatives*. We see similar results for a number of other global problems. As the exact value of c in the $c^{\mathbf{tw}(G)}$ expression is often important and highly non-trivial to obtain, we gather the results in the second column of Table I.

For a number of these results we have matching lower bounds, such as the following one:

Theorem 1.2. *Unless the Strong Exponential Time Hypothesis is false, there do not exist a constant $\varepsilon > 0$ and an algorithm that given an instance $(G = (V, E), T, k)$ together with a path decomposition of the graph G of width p solves the* STEINER TREE *problem in $(3 - \varepsilon)^p|V|^{O(1)}$ time.*

Since each path decomposition is also a tree decomposition a lower bound for pathwidth is at least as strong as for treewidth. We have such matching lower bounds for several other problems presented in the third column of Table I. We feel that the results for CONNECTED VERTEX COVER, CONNECTED DOMINATING SET, CONNECTED FEEDBACK VERTEX SET and CONNECTED ODD CYCLE TRANSVERSAL are of particular interest here and should be compared to the algorithms and lower bounds for the analogous problems without the connectivity requirement.

We have found Cut&Count to fail for two maximization problems: CYCLE PACKING and MAX CYCLE COVER. We believe this is an example of a more general phenomenon — problems that ask to maximize (instead of minimizing) the number of connected components in the solution seem more difficult to solve than the problems that ask to minimize (including problems where we demand that the solution forms a single connected component). As evidence we present lower bounds for the time complexity of solutions to such problems, proving that $c^{\mathbf{tw}(G)}$ solutions of these problems are unlikely:

Theorem 1.3. *Unless the Exponential Time Hypothesis is false, there does not exist a $2^{o(p\log p)}|V|^{O(1)}$ algorithm solving* CYCLE PACKING *or* MAX CYCLE COVER *(either in the directed and undirected setting). The parameter p denotes the width of a given path decomposition of the input graph.*

To further verify this intuition, we investigated an artificial problem (the MAXIMALLY DISCONNECTED DOMINATING SET), in which we ask for a dominating set with the largest possible number of connected components, and indeed we found a similar phenomenon.

1.2. Previous work

The Cut&Count technique has two main ingredients. The first is an algebraic approach, where we assure that objects

Problem name	algorithms param. by treewidth	lower bounds	algorithms param. by solution size	previous best algorithms param. by solution size
STEINER TREE	$3^{\mathbf{tw}(G)}$	$3^{\mathbf{pw}(G)}$		
FEEDBACK VERTEX SET	$3^{\mathbf{tw}(G)}$	$3^{\mathbf{pw}(G)}$	3^k	3.83^k [10]
CONNECTED VERTEX COVER	$3^{\mathbf{tw}(G)}$	$3^{\mathbf{pw}(G)}$	2^k	2.4882^k [3]
CONNECTED DOMINATING SET	$4^{\mathbf{tw}(G)}$	$4^{\mathbf{pw}(G)}$		
CONNECTED FEEDBACK VERTEX SET	$4^{\mathbf{tw}(G)}$	$4^{\mathbf{pw}(G)}$	3^k	46.2^k [34]
CONNECTED ODD CYCLE TRANSVERSAL	$4^{\mathbf{tw}(G)}$	$4^{\mathbf{pw}(G)}$		
UNDIRECTED/DIRECTED MIN CYCLE COVER	$4^{\mathbf{tw}(G)}/6^{\mathbf{tw}(G)}$			
UNDIRECTED/DIRECTED LONGEST PATH (CYCLE)	$4^{\mathbf{tw}(G)}/6^{\mathbf{tw}(G)}$			
EXACT k-LEAF SPANNING TREE	$4^{\mathbf{tw}(G)}$	$4^{\mathbf{pw}(G)}$		
EXACT k-LEAF OUTBRANCHING	$6^{\mathbf{tw}(G)}$			
MAXIMUM FULL DEGREE SPANNING TREE	$4^{\mathbf{tw}(G)}$			
GRAPH METRIC TRAVELLING SALESMAN PROBLEM	$4^{\mathbf{tw}(G)}$			
(DIRECTED) CYCLE PACKING		$2^{\Omega(\mathbf{pw}(G)\log\mathbf{pw}(G))}$		
(DIRECTED) MAX CYCLE COVER		$2^{\Omega(\mathbf{pw}(G)\log\mathbf{pw}(G))}$		
MAXIMALLY DISCONNECTED DOMINATING SET		$2^{\Omega(\mathbf{pw}(G)\log\mathbf{pw}(G))}$		

Table I
SUMMARY OF OUR RESULTS. FOR THE SAKE OF PRESENTATION IN EACH ENTRY WE SKIP THE $|V|^{O(1)}$ MULTIPLICATIVE TERM.

we are not interested in are counted an even number of times, and then do the calculations in $\mathbb{Z}_2$ (or for example any other field of characteristic 2), which causes them to disappear. This line of reasoning goes back to Tutte [43], and was recently used by Björklund [4] and Björklund et. al [6].

The second is the idea of defining the connectivity requirement through cuts, which is frequently used in approximation algorithms via linear programming relaxations. In particular cut based constraints were used in the Held and Karp relaxation for the TRAVELLING SALESMAN PROBLEM problem from 1970 [23], [24] and appear up to now in the best known approximation algorithms, for example in the recent algorithm for the STEINER TREE problem by Byrka et al. [9]. To the best of our knowledge the idea of defining problems through cuts was never used in the exact and parameterized settings.

A number of papers circumvent the problems stemming from the lack of single exponential algorithms parametrized by treewidth for connectivity–type problems. For instance in the case of parametrized algorithms, sphere cuts [16], [17] (for planar and bounded genus graphs) and Catalan structures [15] (for H-minor-free graphs) were used to obtain $2^{O(\sqrt{k})}|V|^{O(1)}$ algorithms for a number of problems with connectivity requirements. To the best of our knowledge, however, no attempt to attack the problem directly was published before; indeed the non-existence of $2^{o(\mathbf{tw}(G)\log\mathbf{tw}(G))}|V|^{O(1)}$ algorithms was deemed to be more likely.

1.3. Consequences of the Cut&Count technique

As already mentioned, algorithms for graphs with a bounded treewidth have a number of applications in various branches of algorithmics. Thus, it is not a surprise that the results obtained by our technique give a large number of corollaries. In this extended abstract we do not explore all possible applications, but only give sample applications in various directions.

We would like to emphasize that the strength of the Cut&Count technique shows not only in the quality of the results obtained in various fields, which are frequently better than the previously best known ones, achieved through a plethora of techniques and approaches, but also in the ease in which new strong results can be obtained.

1.3.1. Consequences for FPT algorithms: Let us recall the definition of the FEEDBACK VERTEX SET problem:

FEEDBACK VERTEX SET **Parameter:** k
Input: An undirected graph G and an integer k
Question: Is it possible to remove k vertices from G so that the remaining vertices induce a forest?

This problem is on Karp's original list of 21 NP-complete problems [29]. It has also been extensively studied from the parametrized complexity point of view. Let us recall that in the fixed-parameter setting (FPT) the problem comes with a parameter k, and we are looking for a solution with time complexity $f(k)|V|^{O(1)}$, where n is the input size and f is some function (usually exponential in k). Thus, we seek to move the intractability of the problem from the input size to the parameter.

There is a long sequence of FPT algorithms for FEEDBACK VERTEX SET. The best — so far — result in this series is the $3.83^k k|V|^2$ result of Cao, Chen and Liu [10]. Our technique gives an improvement of their result:

Theorem 1.4. *There exists a Monte Carlo algorithm with constant one-sided error probability that solves the* FEEDBACK VERTEX SET *problem in a graph $G = (V, E)$ in $3^k|V|^{O(1)}$ time and polynomial space.*

We give similar improvements for CONNECTED VERTEX COVER (from the $2.4882^k|V|^{O(1)}$ of [3] to $2^k|V|^{O(1)}$) and CONNECTED FEEDBACK VERTEX SET (from the

$46.2^k|V|^{O(1)}$ of [34] to $3^k|V|^{O(1)}$).

1.3.2. Parametrized algorithms for H-minor-free graphs: A large branch of applications of algorithms parametrized by treewidth is the bidimensionality theory, used to find subexponential algorithms for various problems in H-minor-free graphs. In this theory we use the theorem of Demaine et al. [13], which ensures that any H-minor-free graph either has treewidth bounded by $C\sqrt{k}$, or a $2\sqrt{k} \times 2\sqrt{k}$ grid as a minor. In the latter case we are assumed to be able to answer the problem in question (for instance a $2\sqrt{k} \times 2\sqrt{k}$ grid as a minor guarantees that the graph does not have a VERTEX COVER or CONNECTED VERTEX COVER smaller than k). Thus, we are left with solving the problem with the assumption of bounded treewidth. In the case of for instance VERTEX COVER, a standard dynamic programming algorithm suffices, thus giving us a $2^{O(\sqrt{k})}$ algorithm to check whether a graph has a vertex cover no larger than k. In the case of CONNECTED VERTEX COVER, however, the standard dynamic programming routine gives a $2^{O(\sqrt{k}\log k)}$ complexity — thus, we lose a logarithmic factor in the exponent.

There were a number of attempts to deal with this problem, taking into account the structure of the graph, and using it to deduce some properties of the tree decomposition under consideration. The latest and most efficient of those approaches is due to Dorn, Fomin and Thilikos [15], and exploits the so-called Catalan structures. The approach deals with most of the problems mentioned in our paper, and is probably applicable to the remaining ones. Thus, the gain here is not in improving the running times (though our approach does improve the constants hidden in the big-O notation these are rarely considered to be important in the bidimensionality theory), but rather in simplifying the proof — instead of delving into the combinatorial structure of each particular problem, we are back to a simple framework of applying the Robertson-Seymour theorem and then following up with a dynamic programming algorithm on the obtained tree decomposition.

1.3.3. Consequences for Exact Algorithms for graphs of bounded degree: Another application of our methods can be found in the field of solving problems with a global constraint in graphs of bounded degree. The problems that have been studied in this setting are mostly local in nature (such as VERTEX COVER, see, e.g., [8]); however global problems such as the TRAVELLING SALESMAN PROBLEM (TSP) and HAMILTONIAN CYCLE have also received considerable attention [5], [19], [22], [27].

Throughout the following we let n denote the number of vertices of the given graph. The starting point is the following theorem by Fomin et al. [20]:

Theorem 1.5 ([20]). *For any $\varepsilon > 0$ there exists an integer n_ε such that for any graph G with $n > n_\varepsilon$ vertices,*

$$\mathbf{pw}(G) \leq \frac{1}{6}n_3 + \frac{1}{3}n_4 + \frac{13}{30}n_5 + n_{\geq 6} + \varepsilon n,$$

where n_i is the number of vertices of degree i in G for any $i \in \{3, \ldots, 5\}$ and $n_{\geq 6}$ is the number of vertices of degree at least 6.

This theorem is constructive, and the corresponding path decomposition (and, consequently, tree decomposition) can be found in polynomial time. Combining this theorem with our results gives algorithms running in faster than 2^n time for graphs of maximum degree 3, 4 and (in the case of the $3^{\mathbf{tw}(G)}$ and $4^{\mathbf{tw}(G)}$ algorithms) 5. Furthermore in the full version of the paper we improve the general $4^{\mathbf{tw}(G)}$ algorithm for HAMILTONIAN CYCLE to $3^{\mathbf{pw}(G)}$ in case of a path decomposition of cubic graphs. Consequently we prove the following theorem which improves over previously best results for maximum degree three $O(1.251^n)$ algorithm of Iwama and Nakashima [27] and for degree four $O(1.657^n)$ algorithm of Björklund [4].

Corollary 1.6. *There exists a Monte Carlo algorithm with constant one-sided error probability that solves the* HAMILTONIAN CYCLE *problem in $O(1.201^n)$ time for cubic graphs and $O(1.588^n)$ for graphs of maximum degree 4.*

1.3.4. Consequences for exact algorithms on planar graphs: Recall from the previous section that n denotes the number of vertices of the given graph. Here we begin with a consequence of the work of Fomin and Thilikos [21]:

Proposition 1.7. *For any planar graph G, $\mathbf{tw}(G) + 1 \leq \frac{3}{2}\sqrt{4.5n} \leq 3.183\sqrt{n}$. Moreover a tree decomposition of such width can be found in polynomial time.*

Using this we immediately obtain $O(c^{\sqrt{n}})$ algorithms for solving problems with a global constraint on planar graphs with good constants. For the HAMILTONIAN CYCLE problem on planar graphs we obtain the following result:

Corollary 1.8. *There exists a Monte Carlo algorithm with constant one-sided error probability that solves the* HAMILTONIAN CYCLE *problem on planar graphs in $O(4^{3.183\sqrt{n}}) = O(2^{6.366\sqrt{n}})$ time.*

To the best of our knowledge the best algorithm known so far was the $O(2^{6.903\sqrt{n}})$ of Bodlaender et al. [17].

Similarly, we obtain an $O(2^{6.366\sqrt{n}})$ algorithm for LONGEST CYCLE on planar graphs (compare to the $O(2^{7.223\sqrt{n}})$ of [17]), and — as in the previous subsections — well-behaved $c^{\sqrt{n}}$ algorithms for all mentioned problems.

1.4. Organization of the paper

Section 2 is devoted to presenting the background material for our algorithms; in particular we recall the notion of treewidth and in Subsection 2.2 we introduce the Isolation Lemma. In Section 3 we present the Cut&Count technique

on two examples: the STEINER TREE problem and the DIRECTED MIN CYCLE COVER problem. In Section 4 we give the $3^k|V|^{O(1)}$ algorithm for FEEDBACK VERTEX SET when parameterized by the solution size, whereas in Section 5 we move to lower bounds. We finish the paper with a number of conclusions and open problems in Section 6.

As the reader might have already noticed, there is a quite a large amount of material covered in this paper and due to space limitations a considerable number of proofs and analyses was postponed to the full version of the paper which is available online [11].

2. PRELIMINARIES AND NOTATION

2.1. Notation

Just as $G[X]$ for $X \subseteq V(G)$ denotes a subgraph induced by the set X of vertices, we use $G[X]$ for $X \subseteq E(G)$ for the graph (V, X), where $G = (V, E)$. Note that in the graph $G[X]$ for an edge set X the set of vertices remains the same as in the graph G.

By a *cut* of a set X we mean a pair (X_1, X_2), with $X_1 \cap X_2 = \emptyset$, $X_1 \cup X_2 = X$. We refer to X_1 and X_2 as to the (left and right) *sides* of the cut.

In a directed graph G by weakly connected components we mean the connected components of the underlying undirected graph. For a (directed) graph G, we let $\mathtt{cc}(G)$ denote the number of (weakly) connected components of G.

We denote the symmetric difference of two sets A and B by $A \triangle B$. For two integers a, b we use $a \equiv b$ to indicate that a is even if and only if b is even. If $\omega : U \to \{1, \ldots, N\}$, we shorthand $\omega(S) := \sum_{e \in S} \omega(e)$ for $S \subseteq U$.

Definition 2.1 (Tree Decomposition, [39])**.** *A* tree decomposition *of a (undirected or directed) graph G is a tree $\mathbb{T}$ in which each vertex $x \in \mathbb{T}$ has an assigned set of vertices $B_x \subseteq V$ (called a* bag*) with the following properties: (i) $\bigcup_{x \in \mathbb{T}} B_x = V$, (ii) for any $uv \in E$, there exists an $x \in \mathbb{T}$ such that $u, v \in B_x$, and (iii) if $v \in B_x$ and $v \in B_y$, then $v \in B_z$ for all z on the path from x to y in $\mathbb{T}$.*

The *treewidth* $tw(\mathbb{T})$ of a tree decomposition $\mathbb{T}$ is the size of the largest bag of $\mathbb{T}$ minus one, and the treewidth of a graph G is the minimum treewidth over all possible tree decompositions of G. A *path decomposition* is a tree decomposition that is a path. The pathwidth of a graph is the minimum width of all path decompositions. Since a path decomposition is a special tree decomposition, the pathwidth is at least the treewidth of a graph.

2.2. Isolation lemma

An important ingredient of our algorithms is the Isolation Lemma:

Definition 2.2. *A function $\omega : U \to \mathbb{Z}$* isolates *a set family $\mathcal{F} \subseteq 2^U$ if there is a unique $S' \in \mathcal{F}$ with $\omega(S') = \min_{S \in \mathcal{F}} \omega(S)$.*

Lemma 2.3 (Isolation Lemma, [36])**.** *Let $\mathcal{F} \subseteq 2^U$ be a set family over a universe U with $|\mathcal{F}| > 0$. For each $u \in U$, choose a weight $\omega(u) \in \{1, 2, \ldots, N\}$ uniformly and independently at random. Then* $\mathtt{prob}[\omega$ isolates $\mathcal{F}] \geq 1 - |U|/N$.

The Isolation Lemma allows us to count objects modulo 2, since with a large probability it reduces a possibly large number of solutions to some problem to a unique one (with an additional weight constraint imposed).

An alternative method to a similar end is obtained by using Polynomial Identity Testing [14], [41], [45] over a field of characteristic two. This second method has been already used in the field of exact and parameterized algorithms [4], [31]. The two methods do not differ much in their consequences: Both use the same number of random bits, and the challenge of giving a full derandomization seems to be equally difficult for both methods [1], [28]. The usage of the Isolation Lemma gives greater polynomial overheads, however we choose to use it because it requires less preliminary knowledge and it simplifies the presentation.

3. CUT&COUNT: ILLUSTRATION OF THE TECHNIQUE

In this section we present the Cut&Count technique by demonstrating how it applies to the STEINER TREE and DIRECTED MIN CYCLE COVER problems. We go through all the important details in an expository manner, as we aim not only to show the solutions to these particular problems, but also to show the general workings.

The Cut&Count technique applies to problems with certain connectivity requirements. Let $\mathcal{S} \subseteq 2^U$ be a set of solutions; we aim to decide whether it is empty. Conceptually, Cut&Count can naturally be split in two parts:

- **The Cut part**: Relax the connectivity requirement by considering the set $\mathcal{R} \supseteq \mathcal{S}$ of possibly connected candidate solutions. Furthermore, consider the set $\mathcal{C}$ of pairs (X, C) where $X \in \mathcal{R}$ and C is a consistent cut (to be defined later) of X.
- **The Count part**: Compute $|\mathcal{C}|$ modulo 2 using a subprocedure. Non-connected candidate solutions $X \in \mathcal{R} \setminus \mathcal{S}$ cancel since they are consistent with an even number of cuts. Connected candidates $x \in \mathcal{S}$ remain.

Note that we need the number of solutions to be odd in order to make the counting part work. For this we use the Isolation Lemma (Lemma 2.3): We introduce uniformly and independently chosen weights $\omega(v)$ for every $v \in U$ and compute $|\mathcal{C}_W|$ modulo 2 for every W, where $\mathcal{C}_W = \{(X, C) \in \mathcal{C} | \omega(X) = W\}$. Let us recall that for two integers a, b we use $a \equiv b$ to indicate that a is even if and only if b is even. The general setup can thus be summarized as in Algorithm 1.

The following corollary that we use throughout the paper follows from Lemma 2.3 by setting $\mathcal{F} = \mathcal{S}$ and $N = 2|U|$:

Function `cutandcount`$(U, \mathbb{T}, \texttt{CountC})$
Input Set U; tree decomposition $\mathbb{T}$; Procedure `CountC` accepting a $\omega : U \to \{1, \ldots, N\}$, $W \in \mathbb{Z}$ and $\mathbb{T}$.
1: **for** every $v \in U$ **do**
2: Choose $\omega(v) \in \{1, \ldots, 2|U|\}$ uniformly at random.
3: **for** every $0 \leq W \leq 2|U|^2$ **do**
4: **if** $\texttt{CountC}(\omega, W, \mathbb{T}) \equiv 1$ **then return yes**
5: **return no**

Algorithm 1: `cutandcount`$(U, \mathbb{T}, \texttt{CountC})$

Corollary 3.1. *Let $\mathcal{S} \subseteq 2^U$ and $\mathcal{C} \subseteq 2^U \times (2^V \times 2^V)$. Suppose that for every $W \in \mathbb{Z}$:*

1) $|\{(X, C) \in \mathcal{C} | \omega(X) = W\}| \equiv |\{X \in \mathcal{S} | \omega(X) = W\}|$,
2) $\texttt{CountC}(\omega, W, \mathbb{T}) \equiv |\{(X, C) \in \mathcal{C} | \omega(X) = W\}|$.

*Then Algorithm 1 returns **no** if $\mathcal{S}$ is empty and **yes** with probability at least $\frac{1}{2}$ otherwise.*

When applying the technique, both the Cut and the Count part are non-trivial: In the Cut part one has to find the proper relaxation of the solution set, and in the Count part one has to show that the number of non-solutions counted is even for each W and provide an algorithm `CountC`. In the next two subsections, we illustrate both parts by giving two specific applications.

3.1. Steiner Tree

STEINER TREE
Input: An undirected graph $G = (V, E)$, a set of terminals $T \subseteq V$ and an integer k.
Question: Is there a set $X \subseteq V$ of cardinality k such that $T \subseteq X$ and $G[X]$ is connected?

The Cut part. Let us first consider the Cut part of the Cut&Count technique, and start by defining the objects we are going to count. Suppose we are given a weight function $\omega : V \to \{1, \ldots, N\}$. For any integer W, let $\mathcal{R}_W$ be the set of all such subsets X of V that $T \subseteq X$, $\omega(X) = W$ and $|X| = k$. Also, define $\mathcal{S}_W = \{X \in \mathcal{R}_W \mid G[X] \text{ is connected}\}$. The set $\bigcup_W \mathcal{S}_W$ is our set of solutions — if for any W this set is nonempty, our problem has a positive answer. The set $\mathcal{R}_W$ is the set of candidate solutions, where we relax the connectivity requirement. In this easy application the only requirement that remains is that the set of terminals is contained in the candidate solution.

Definition 3.2. *A cut (V_1, V_2) of an undirected graph $G = (V, E)$ is* consistent *if $u \in V_1$ and $v \in V_2$ implies $uv \notin E$. A* consistently cut subgraph *of G is a pair $(X, (X_1, X_2))$ such that (X_1, X_2) is a consistent cut of $G[X]$.*

Similarly for a directed graph $D = (V, A)$ a cut (V_1, V_2) is consistent if (V_1, V_2) is a consistent cut in the underlying undirected graph. A consistently cut subgraph of D is a pair $(X, (X_1, X_2))$ such that (X_1, X_2) is a consistent cut of the underlying undirected graph of $D[X]$.

Let v_1 be an arbitrary terminal. Define $\mathcal{C}_W$ to be the set of all consistently cut subgraphs $(X, (X_1, X_2))$ such that $X \in \mathcal{R}_W$ and $v_1 \in X_1$. Before we proceed with the Count part, let us state the following easy combinatorial identity:

Lemma 3.3. *Let $G = (V, E)$ be a graph and let X be a subset of vertices such that $v_1 \in X \subseteq V$. The number of consistently cut subgraphs $(X, (X_1, X_2))$ such that $v_1 \in X_1$ is equal to $2^{\texttt{cc}(G[X])-1}$.*

Proof: By definition, we know for every consistently cut subgraph $(X, (X_1, X_2))$ and connected component C of $G[X]$ that either $C \subseteq X_1$ or $C \subseteq X_2$. For the connected component containing v_1, the choice is fixed, and for all $\texttt{cc}(G[X]) - 1$ other connected components we are free to choose a side of a cut, which gives $2^{\texttt{cc}(G[X])-1}$ possibilities leading to different consistently cut subgraphs. ■

The Count part. The following lemma shows that the first condition of Corollary 3.1 is indeed met:

Lemma 3.4. *Let $G, \omega, \mathcal{C}_W$, $\mathcal{S}_W$ and $\mathcal{R}_W$ be as defined above. Then for every W, $|\mathcal{S}_W| \equiv |\mathcal{C}_W|$.*

Proof: By Lemma 3.3, we know that $|\mathcal{C}_W| = \sum_{X \in \mathcal{R}_W} 2^{\texttt{cc}(G[X])-1}$. Thus $|\mathcal{C}_W| \equiv \left|\{X \in \mathcal{R}_W | \texttt{cc}(G[X]) = 1\}\right| = |\mathcal{S}_W|$. ■

Now the only missing ingredient left is the sub-procedure `CountC`. This sub-procedure, which counts the cardinality of $\mathcal{C}_W$ modulo 2, is a standard application of dynamic programming and the details are omitted.

Lemma 3.5. *Given $G = (V, E)$, $T \subseteq V$, an integer k, $\omega : V \to \{1, \ldots, N\}$ and a tree decomposition of G, there exists an algorithm that can determine $|\mathcal{C}_W|$ modulo 2 for every $0 \leq W \leq kN$ in $3^t N^2 |V|^{O(1)}$ time.*

We conclude this section with the following theorem.

Theorem 3.6. *There exists a Monte-Carlo algorithm that given a tree decomposition of width t solves* STEINER TREE *in $3^t |V|^{O(1)}$ time. The algorithm cannot give false positives and may give false negatives with probability at most $1/2$.*

Proof: Run Algorithm 1 by setting $U = V$, and `CountC` to be the algorithm implied by Lemma 3.5. The correctness follows from Corollary 3.1 by setting $\mathcal{S} = \bigcup_W \mathcal{S}_W$ and $\mathcal{C} = \bigcup_W \mathcal{C}_W$ and Lemma 3.4. It is easy to see that the timebound follows from Lemma 3.5. ■

3.2. Directed Min Cycle Cover

DIRECTED MIN CYCLE COVER
Input: A directed graph $D = (V, A)$, an integer k.
Question: Can the vertices of D be covered with at most k vertex disjoint directed cycles?

This problem is significantly different from the one considered in the previous section, since the aim is to maximize connectivity in a more flexible way: in the previous section the solution induced one connected component, while it may induce at most k weakly connected components in the context of the current section. Note that with the Cut&Count technique as introduced above, the solutions we are looking for cancel modulo 2.

We introduce a concept called *markers*. A set of solutions consists of pairs (X, M), where $X \subseteq A$ is a cycle cover and $M \subseteq X, |M| = k$ is a set of *marked* arcs, such that each cycle in X contains at least one marked arc. Since $|M| = k$, this ensures that for every solution (X, M) the cycle cover X consists of at most k cycles. Note that distinguishing two different sets of marked arcs of a single cycle cover is considered to induce two *different solutions*. For this reason, with each arc of the graph we associate *two* random weights: the first contributes to the weight of a solution, when an arc belongs to X, while the second contributes additionally, when it belongs to M as well. When we relax the requirement that in the pair (X, M) each cycle in X contains at least one vertex from M, we obtain a set of candidate solutions. The objects we count are pairs consisting of (i) a pair (X, M), where $X \subseteq A$ is a cycle cover and $M \subseteq X$ is a set of k markers, (ii) a cut consistent with $D[X]$, where all the marked arcs from M have both endpoints on the left side of the cut. We will see that candidate solutions that contain a cycle without any marked arc cancel modulo 2. Formal definition follows.

The Cut part. As said before, we assume that we are given a weight function $\omega : U = A \times \{\mathbf{X}\} \cup A \times \{\mathbf{M}\} \to \{1, \ldots, N\}$, where $N = 2|U| = 4|A|$. The arguments $A \times \{\mathbf{X}\}$ correspond to the contribution of choosing an arc to belong to X, while $A \times \{\mathbf{M}\}$ correspond to additional contribution of choosing it to M as well.

Definition 3.7. *For an integer W we define:*

1) *$\mathcal{R}_W$ to be the family of candidate solutions, that is, $\mathcal{R}_W$ is the family of all pairs (X, M), such that $X \subseteq A$ is a cycle cover, i.e., $\mathrm{outdeg}_X(v) = \mathrm{indeg}_X(v) = 1$ for every vertex $v \in V$; $M \subseteq X$, $|M| = k$ and $\omega(X \times \{\mathbf{X}\} \cup M \times \{\mathbf{M}\}) = W$;*
2) *$\mathcal{S}_W$ to be the family of solutions, that is, $\mathcal{S}_W$ is the family of all pairs (X, M), where $(X, M) \in \mathcal{R}_W$ and every cycle in X contains at least one arc from the set M;*
3) *$\mathcal{C}_W$ as all pairs $((X, M), (V_1, V_2))$ such that $(X, M) \in \mathcal{R}_W$, (V_1, V_2) is a consistent cut of $D[X]$ and $V(M) \subseteq V_1$.*

Observe that the graph D admits a cycle cover with at most k cycles if and only if there exists W such that $\mathcal{S}_W$ is nonempty.

The Count part. We proceed to the Count part by showing that candidate solutions that contain an unmarked cycle cancel modulo 2.

Lemma 3.8. *Let $D, \omega, \mathcal{C}_W$ and $\mathcal{S}_W$ be defined as above. Then, for every W, $|\mathcal{S}_W| \equiv |\mathcal{C}_W|$.*

Proof: For subsets $M \subseteq X \subseteq A$, let $\mathtt{cc}(M, X)$ denote the number of weakly connected components of $D[X]$ not containing any arc from M. Then,

$$|\mathcal{C}_W| = \sum_{(X,M) \in \mathcal{R}_W} 2^{\mathtt{cc}(M,X)}.$$

To see this, note that for any $((X, M), (V_1, V_2)) \in \mathcal{C}_W$ and any vertex set C of a cycle from X not containing arcs from M, we have $((X, M), (V_1 \triangle C, V_2 \triangle C)) \in \mathcal{C}_W$ — we can move all the vertices of C to the other side of the cut, also obtaining a consistent cut. Thus, for any set of choices of a side of the cut for every cycle not containing a marker, there is an object in $\mathcal{C}_W$. Hence (analogously to Lemma 3.3) for any W and $(M, X) \in \mathcal{R}_W$ there are $2^{\mathtt{cc}(M,X)}$ cuts (V_1, V_2) such that $((X, M), (V_1, V_2)) \in \mathcal{C}_W$ and the lemma follows, because:

$$|\mathcal{C}_W| \equiv |\{((X, M), (V_1, V_2)) \in \mathcal{C}_W : \mathtt{cc}(M, X) = 0\}|,$$

■

which equals $|\mathcal{S}_W|$ by the definition. Now, it suffices to present a dynamic programming routine counting $|\mathcal{C}_W|$ modulo 2 in a bottom-up fashion. The details are postponed to the full version of the paper. Combining all the observations in the same way as in the proof of Theorem 3.6, we can conclude the following:

Theorem 3.9. *There exists a Monte-Carlo algorithm that, given a tree decomposition of width t, solves* DIRECTED MIN CYCLE COVER *in $6^t |V|^{O(1)}$ time. The algorithm cannot give false positives and may give false negatives with probability at most $1/2$.*

4. FEEDBACK VERTEX SET PARAMETERIZED BY SOLUTION SIZE

In this section we sketch a proof of Theorem 1.4. Recall the definition of FEEDBACK VERTEX SET from Section 1.3.1. Defining a solution candidate with a relaxed connectivity condition to work with our technique is somewhat more tricky here, as there is no explicit connectivity requirement in the problem. To overcome this, we reformulate the problem using the following simple lemma:

Lemma 4.1. *A graph $G = (V, E)$ with n vertices and m edges is a forest iff G has at most $n - m$ connected components.*

Now we use the Cut&Count technique. Instead of looking for the feedback vertex set we look for its complement, being a forest. Let $U = V \times \{\mathbf{F}, \mathbf{M}\}$, where $V \times \{\mathbf{F}\}$ is used to assign weights to vertices from the chosen forest and $V \times \{\mathbf{M}\}$ for markers. We also assume that we are given

a weight function $\omega : U \to \{1, ..., N\}$, where $N = 2|U| = 4|V|$.

The Cut part. For integers A, B, W, we define:

1) $\mathcal{R}_W^{A,B}$ to be the family of pairs (X, M), where $X \subseteq V$, $|X| = A$, $G[X]$ contains exactly B edges, $M \subseteq X$, $|M| = n-k-B$, and $\omega(X \times \{\mathbf{F}\}) + \omega(M \times \{\mathbf{M}\}) = W$;
2) $\mathcal{S}_W^{A,B}$ to be the family of pairs (X, M), where $(X, M) \in \mathcal{R}_W^{A,B}$, and $G[X]$ is a forest containing at least one marker from the set M in each connected component;
3) $\mathcal{C}_W^{A,B}$ to be the family of pairs $((X, M), (X_1, X_2))$, where $(X, M) \in \mathcal{R}_W^{A,B}$, $M \subseteq X_1$, and (X_1, X_2) is a consistent cut of $G[X]$.

Observe that by Lemma 4.1 the graph G admits a feedback vertex set of size k if and only if there exist integers B, W such that the set $\mathcal{S}_W^{n-k,B}$ is nonempty.

The Count part. Similarly as in the case of MIN CYCLE COVER (analogously to Lemma 3.8), note that, for any A, B, $(X, M) \in \mathcal{R}_W^{A,B}$, there are $2^{\mathtt{cc}(M,G[X])}$ cuts (X_1, X_2) such that $((X, M), (X_1, X_2)) \in \mathcal{C}_W^{A,B}$, where $\mathtt{cc}(M, G[X])$ denotes the number of connected components of $G[X]$ which do not contain any marker from the set M. Hence, we have $|\mathcal{S}_W^{A,B}| \equiv |\mathcal{C}_W^{A,B}|$ for every A, B and W by Lemma 4.1.

It remains to show how to count $|\mathcal{C}_W^{n-k,B}|$ modulo 2 for every W and B in $3^k|V|^{O(1)}$ time and polynomial space. For this we will combine the Cut & Count approach with *iterative compression* [38]. The idea of *iterative compression* is to break the given task down in *compression steps*: we assume we are given a solution of size at most $k+1$ and use it to find a solution of size at most k, or conclude that none exists. Here, we apply iterative compression by showing that, given a feedback vertex set of size $k+1$, we can compute $|\mathcal{C}_W^{n-k,B}|$ modulo 2 in $3^k|V|^{O(1)}$ time and polynomial space. Let $V = \{v_1, \ldots, v_n\}$ and $G_i = G[\{v_1, \ldots, v_i\}]$. Given a feedback vertex set S_i of G_i of size at most k, observe that $S' = S_i \cup \{v_{i+1}\}$ is a feedback vertex set of G_{i+1} of size at most $k+1$. The compression step can be used to find a feedback vertex set S_{i+1} of G_{i+1} of size at most k or to conclude that none exists in $3^k|V|^{O(1)}$ time and polynomial space. In the latter case, it is easy to see we can return no; otherwise the original problem can be solved using n compression steps. In each compression step, $|\mathcal{C}_W^{n-k,B}|$ modulo 2 is counted by guessing, for every element in S_i, whether it is in X_1, X_2, or neither of them, counting in how many ways these vertices can be marked, and using dynamic programming on the remaining forest. If $|\mathcal{C}_W^{n-k,B}| \equiv 1$, then the corresponding feedback vertex set is constructed by using a standard self-reduction. The full algorithm and its analysis can be found in the full version of the paper [11].

5. LOWER BOUNDS

In this section we briefly describe a bunch of negative results concerning the possible time complexities for algorithms for connectivity problems parameterized by treewidth or pathwidth. Due to space limitations we postpone the proofs to the full version [11]. Our goal is to complement our positive results by showing that in some situations the known algorithms (including ours) probably cannot be further improved.

First, let us introduce the complexity assumptions made in this section. Let c_k be the infimum of the set of the positive reals c that satisfy the following condition: there exists an algorithm that solves k-SAT in $O(2^{cn})$ time, where n denotes the number of variables in the input formula. The Exponential Time Hypothesis (ETH for short) asserts that $c_3 > 0$, whereas the Strong Exponential Time Hypothesis (SETH) asserts that $\lim_{k\to\infty} c_k = 1$. It is well known that SETH implies ETH [26].

The lower bounds presented below are of two different types. In Section 5.1 we discuss several problems that, assuming ETH, do not admit an algorithm running in $2^{o(p \log p)}|V|^{O(1)}$ time, where p denotes the pathwidth of the input graph. In Section 5.2 we state that, assuming SETH, the base of the exponent in some of our algorithms cannot be improved further.

5.1. Lower bounds assuming ETH

We have shown that a lot of well-known algorithms running in $2^{O(t)}|V|^{O(1)}$ time can be turned into algorithms that keep track of the connectivity issues, with only small loss in the base of the exponent. The problems solved in that manner include CONNECTED VERTEX COVER, CONNECTED DOMINATING SET, CONNECTED FEEDBACK VERTEX SET and CONNECTED ODD CYCLE TRANSVERSAL. Note that using the markers technique introduced in Section 3.2 we can solve similarly the following artificial generalizations: given a graph G and an integer r, what is the minimum size of a vertex cover (dominating set, feedback vertex set, odd cycle transversal) that induces **at most** r connected components?

We provide evidence that problems in which we would ask to maximize (instead of minimizing) the number of connected components are harder: they probably do not admit algorithms running in $2^{o(p \log p)}|V|^{O(1)}$ time, where p denotes the pathwidth of the input graph.

In the CYCLE PACKING problem we are asked whether a given graph contains at least ℓ vertex-disjoint cycles, whereas in the MAX CYCLE COVER we additionally want those cycles to cover all the vertices of the graph. The third problem is an artificial problem defined by us that should be compared to CONNECTED DOMINATING SET. An instance of MAXIMALLY DISCONNECTED DOMINATING SET consists of an undirected graph G and integers ℓ, r. The question is whether G contains a dominating set of size at most ℓ that induces **at least** r connected components?

Using the framework introduced by Lokshtanov et al. [33] we prove the following theorem:

Theorem 5.1. *Assuming ETH, there is no $2^{o(p \log p)}|V|^{O(1)}$ time algorithm for* CYCLE PACKING*,* MAX CYCLE COVER *(both in the directed and undirected setting) nor for* MAXIMALLY DISCONNECTED DOMINATING SET*. The parameter p denotes the width of a given path decomposition of the input graph.*

5.2. Lower bounds assuming SETH

Following the framework introduced by Lokshtanov et al. [32], we prove that an improvement in the base of the exponent in a number of our algorithms would contradict SETH. Formally, we prove the following type of a theorem for problems marked in the third column of Table I.

Theorem 5.2. *Unless the Strong Exponential Time Hypothesis is false, there do not exist a constant $\varepsilon > 0$ and an algorithm that given an instance $(G = (V, E), k)$ together with a path decomposition of the graph G of width p solves* CONNECTED VERTEX COVER *in $(3 - \varepsilon)^p |V|^{O(1)}$ time.*

Note that VERTEX COVER (without a connectivity requirement) admits a $2^t|V|^{O(1)}$ algorithm whereas DOMINATING SET, FEEDBACK VERTEX SET and ODD CYCLE TRANSVERSAL admit $3^t|V|^{O(1)}$ algorithms and those algorithms are optimal (assuming SETH) [32]. To use the Cut&Count technique for the connected versions of these problems we need to increase the base of the exponent by one to keep the side of the cut for vertices in the solution. Our results show that this is not an artifact of the Cut&Count technique, but rather an intrinsic characteristic of these problems.

6. CONCLUDING REMARKS

For several years it was known that most of the local problems (where by local we mean that a solution can be verified by checking separately the neighbourhood of each vertex), standard dynamic programming techniques give $c^{\mathbf{tw}}|V|^{O(1)}$ time algorithms for a constant c. The main consequence of the Cut&Count technique as presented in this work is that problems which can be formulated as a local constraint with an additional upper bound on the number of connected components also admit $c^{\mathbf{tw}}|V|^{O(1)}$ time algorithms. Moreover, many problems cannot be solved faster unless the Strong Exponential Time Hypothesis fails. We have chosen not to pursue a general theorem in the above spirit, as the techniques required to get optimal constants seem varied and depend on the particular problem.

We have also shown that several problems in which one aims to maximize the number of connected components are not solvable in $2^{o(p \log p)}|V|^{O(1)}$ unless the Exponential Time Hypothesis fails. Hence, assuming the Exponential Time Hypothesis, there is a marked difference between the minimization and maximization of the number of connected components in this context.

Finally, we leave the reader with some interesting open questions:

- Can Cut&Count be derandomized? For example, can CONNECTED VERTEX COVER be solved deterministically in $c^t|V|^{O(1)}$ on graphs of treewidth t for some constant c?
- Since general derandomization seems hard, we ask whether it is possible to derandomize the presented FPT algorithms parameterized by the solution size for FEEDBACK VERTEX SET, CONNECTED VERTEX COVER or CONNECTED FEEDBACK VERTEX SET? Note that the tree decomposition considered in these algorithms is of a very specific type, which could potentially make this problem easier than the previous one.
- Do there exist algorithms running in time $c^t|V|^{O(1)}$ on graphs of treewidth t that solve counting or weighted variants? For example can the number of Hamiltonian paths be determined, or the Traveling Salesman Problem solved in $c^t|V|^{O(1)}$ on graphs of treewidth t?
- Can exact exponential time algorithms be improved using Cut&Count (for example for CONNECTED DOMINATING SET, STEINER TREE and FEEDBACK VERTEX SET)?
- All our algorithms for directed graphs run in time $6^t|V|^{O(1)}$. Can the constant 6 be improved? Or maybe it is optimal (again, assuming SETH)?

Acknowledgements: We are grateful to Fedor Fomin, Daniel Lokshtanov, Dániel Marx for discussions and useful suggestions. The second authors thanks Geevarghese Philip for useful discussions in a very early stage.

Furthermore we would like to thank Andreas Björklund for sharing with us an observation allowing to improve the HAMILTONIAN CYCLE exact algorithm for cubic graphs.

REFERENCES

[1] V. Arvind and P. Mukhopadhyay, "Derandomizing the isolation lemma and lower bounds for circuit size," in *Proc. of APPROX/RANDOM 2008*, ser. LNCS, vol. 5171, pp. 276–289.

[2] M. Bateni, M. Hajiaghayi, and D. Marx, "Approximation schemes for steiner forest on planar graphs and graphs of bounded treewidth," in *Proc. of STOC 2010*, pp. 211–220.

[3] D. Binkele-Raible, "Amortized analysis of exponential time and parameterized algorithms: Measure and conquer and reference search trees," Ph.D. dissertation, University of Trier, Trier, Germany, 2010.

[4] A. Björklund, "Determinant sums for undirected Hamiltonicity," in *Proc. of FOCS 2010*, pp. 173–182.

[5] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto, "The travelling salesman problem in bounded degree graphs," in *Proc. of ICALP 2008*, ser. LNCS, vol. 5125, pp. 198–209.

[6] ——, "Narrow sieves for parameterized paths and packings," *arXiv.org*, vol. abs/1007.1161, 2010.

[7] G. Borradaile, C. Kenyon-Mathieu, and P. N. Klein, "A polynomial-time approximation scheme for Steiner tree in planar graphs," in *Proc. of SODA 2007*, pp. 1285–1294.

[8] N. Bourgeois, B. Escoffier, V. T. Paschos, and J. M. M. van Rooij, "A bottom-up method and fast algorithms for max independent set," in *Proc. of SWAT 2010*, ser. LNCS, vol. 6139, pp. 62–73.

[9] J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità, "An improved LP-based approximation for Steiner tree," in *Proc. of STOC 2010*, pp. 583–592.

[10] Y. Cao, J. Chen, and Y. Liu, "On feedback vertex set new measure and new structures," in *Proc. of SWAT 2010*, ser. LNCS, vol. 6139, pp. 93–104.

[11] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. van Rooij, and J. O. Wojtaszczyk, "Solving connectivity problems parameterized by treewidth in single exponential time," *arXiv.org*, vol. abs/1103.0534, 2011.

[12] E. D. Demaine and M. T. Hajiaghayi, "The bidimensionality theory and its algorithmic applications," *The Computer Journal*, vol. 51, no. 3, pp. 292–302, 2008.

[13] E. D. Demaine, M. T. Hajiaghayi, and K. ichi Kawarabayashi, "Algorithmic graph minor theory: Decomposition, approximation, and coloring," in *Proc. of FOCS 2005*, pp. 637–646.

[14] R. A. DeMillo and R. J. Lipton, "A probabilistic remark on algebraic program testing," *Information Processing Letters*, vol. 7, no. 4, pp. 193–195, 1978.

[15] F. Dorn, F. V. Fomin, and D. M. Thilikos, "Catalan structures and dynamic programming in H-minor-free graphs," in *Proc. of SODA 2008*, pp. 631–640.

[16] ——, "Fast subexponential algorithm for non-local problems on graphs of bounded genus," in *Proc. of SWAT 2006*, ser. LNCS, vol. 4059, pp. 172–183.

[17] F. Dorn, E. Penninkx, H. L. Bodlaender, and F. V. Fomin, "Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions," *Algorithmica*, vol. 58, no. 3, pp. 790–810, 2010.

[18] D. Eppstein, "Diameter and treewidth in minor-closed graph families," *Algorithmica*, vol. 27, no. 3, pp. 275–291, 2000.

[19] ——, "The traveling salesman problem for cubic graphs," *Journal of Graph Algorithms and Applications*, vol. 11, no. 1, pp. 61–81, 2007.

[20] F. V. Fomin, S. Gaspers, S. Saurabh, and A. A. Stepanov, "On two techniques of combining branching and treewidth," *Algorithmica*, vol. 54, no. 2, pp. 181–207, 2009.

[21] F. V. Fomin and D. M. Thilikos, "A simple and fast approach for solving problems on planar graphs," in *Proc. of STACS 2004*, ser. LNCS, vol. 2996, pp. 56–67.

[22] H. Gebauer, "On the number of Hamilton cycles in bounded degree graphs," in *Proc. of ANALCO 2008*, pp. 241–248.

[23] M. Held and R. M. Karp, "The traveling-salesman problem and minimum spanning trees," *Operations Research*, vol. 18, no. 6, pp. 1138–1162, 1970.

[24] ——, "The traveling-salesman problem and minimum spanning trees: Part II," *Mathematical Programming*, vol. 1, no. 1, pp. 6–25, 1971.

[25] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 2nd ed. Addison-Wesley-Longman, 2001.

[26] R. Impagliazzo and R. Paturi, "On the complexity of k-SAT," *Journal of Computer and System Sciences*, vol. 62, no. 2, pp. 367–375, 2001.

[27] K. Iwama and T. Nakashima, "An improved exact algorithm for cubic graph TSP," in *Proc. of COCOON 2007*, ser. LNCS, vol. 4598, pp. 108–117.

[28] V. Kabanets and R. Impagliazzo, "Derandomizing polynomial identity tests means proving circuit lower bounds," *Computational Complexity*, vol. 13, no. 1-2, pp. 1–46, 2004.

[29] R. M. Karp, "Reducibility among combinatorial problems," in *A Symposium on the Complexity of Computer Computations*, 1972, pp. 85–103.

[30] J. Kleinberg and É. Tardos, *Algorithm Design*. Addison-Wesley, 2005.

[31] I. Koutis, "Faster algebraic algorithms for path and packing problems," in *Proc. of ICALP 2008*, ser. LNCS, vol. 5125, pp. 575–586.

[32] D. Lokshtanov, D. Marx, and S. Saurabh, "Known algorithms on graphs of bounded treewidth are probably optimal," in *Proc. of SODA 2011*, pp. 777–789.

[33] ——, "Slightly superexponential parameterized problems," in *Proc. of SODA 2011*, pp. 760–776.

[34] N. Misra, G. Philip, V. Raman, S. Saurabh, and S. Sikdar, "FPT algorithms for connected feedback vertex set," in *Proc. of WALCOM 2010*, ser. LNCS, vol. 5942, pp. 269–280.

[35] D. Mölle, S. Richter, and P. Rossmanith, "Enumerate and expand: Improved algorithms for connected vertex cover and tree cover," *Theory of Computing Systems*, vol. 43, no. 2, pp. 234–253, 2008.

[36] K. Mulmuley, U. V. Vazirani, and V. V. Vazirani, "Matching is as easy as matrix inversion," *Combinatorica*, vol. 7, no. 1, pp. 105–113, 1987.

[37] R. Niedermeier, *Invitation to Fixed-Parameter Algorithms*, ser. Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, 2006, vol. 31.

[38] B. A. Reed, K. Smith, and A. Vetta, "Finding odd cycle transversals," *Operations Research Letters*, vol. 32, no. 4, pp. 299–301, 2004.

[39] N. Robertson and P. D. Seymour, "Graph minors. III. Planar tree-width," *Journal of Combinatorial Theory, Series B*, vol. 36, no. 1, pp. 49–64, 1984.

[40] D. J. Rose, "On simple characterizations of k-trees," *Discrete Mathematics*, vol. 7, no. 3-4, pp. 317–322, 1974.

[41] J. T. Schwartz, "Fast probabilistic algorithms for verification of polynomial identities," *Journal of the ACM*, vol. 27, no. 4, pp. 701–717, 1980.

[42] D. M. Thilikos, M. J. Serna, and H. L. Bodlaender, "Cutwidth I: A linear time fixed parameter algorithm," *Journal of Algorithms*, vol. 56, no. 1, pp. 1–24, 2005.

[43] W. T. Tutte, "The factorization of linear graphs," *Journal of the London Mathematical Society*, vol. 22, no. 2, pp. 107–111, 1947.

[44] J. M. M. van Rooij, J. Nederlof, and T. C. van Dijk, "Inclusion/exclusion meets measure and conquer," in *Proc. of ESA 2009*, ser. LNCS, vol. 5757, pp. 554–565.

[45] R. Zippel, "Probabilistic algorithms for sparse polynomials," in *Proc. of EUROSAM 1979*, ser. LNCS, vol. 72, pp. 216–226.