Parameterized Approximation via Fidelity Preserving Transformations[☆]

Michael R. Fellows[1], Ariel Kulik[2], Frances Rosamond[1], Hadas Shachnai[2,*]

Abstract

We motivate and describe a new parameterized approximation paradigm which studies the interaction between approximation ratio and running time for any parametrization of a given optimization problem. As a key tool, we introduce the concept of an α-*shrinking transformation*, for $\alpha \geq 1$. Applying such transformation to a parameterized problem instance decreases the parameter value, while preserving the approximation ratio of α (or α-*fidelity*). Our algorithms define a continuous tradeoff between running times and approximation ratios, allowing practitioners to appropriately allocate computational resources.

Moving even beyond the approximation ratio, we call for a new type of *approximative kernelization race*. Our α-shrinking transformations can be used to obtain *approximative* kernels which are smaller than the best known for a given problem. The smaller "α-fidelity" kernels allow us to obtain an exact solution for the *reduced instance* more efficiently, while obtaining an approximate solution for the original instance. We show that such fidelity preserving transformations exist for several fundamental problems, including *Vertex Cover*, *d-Hitting Set*, *Connected Vertex Cover* and *Steiner Tree*.

Keywords: Fidelity preserving transformation, fixed parameter tractability, kernelization, parameterized complexity, approximation algorithms.

---

1. Introduction

Given the common belief that most NP-hard problems cannot be solved, or even well-approximated, in polynomial time, it is natural for us to turn to a generalization

---

[☆] A preliminary version of this paper appeared in Proc. of the 39th International Colloquium on Automata, Languages and Programming (ICALP), Warwick, July 2012.

[*] Corresponding author. Work partially supported by the Loewengart Research Fund, and by the Ministry of Trade and Industry MAGNET program through the NEGEV Consortium (www.negev-initiative.org).

Email addresses: `michael.fellows@cdu.edu.au` (Michael R. Fellows), `ariel.kulik@gmail.com.` (Ariel Kulik), `frances.rosamond@cdu.edu.au` (Frances Rosamond), `hadas@cs.technion.ac.il` (Hadas Shachnai)

[1] School of Engineering and IT, Charles Darwin Univ., Darwin, NT Australia 0909.

[2] Department of Computer Science, Technion, Haifa 3200003, Israel.

of polynomial time, *fixed-parameter tractability*, to develop a paradigm of *parameterized approximation.*

Parameterized complexity approaches hard computational problems through a multivariate analysis of the running time. Instead of expressing the running time as a function of the input size n only, the running time is expressed as a function of n and k, where k is a well-defined parameter of the input instance (that can be an aggregate of several measurements). We say that a problem (with a particular parameter k) is *fixed-parameter tractable (FPT)* if it can be solved in time $f(k) \cdot \text{poly}(n)$, where f is an arbitrary function depending only on k. Thus we relax polynomial time by committing the exponential explosion to the parameter k. For further background on parameterized complexity we refer the reader to the textbooks [29, 41, 20], and the surveys in [21, 23].

Extensive research since the beginning of the 70's has led to results exhibiting limits to the approximability of NP-hard problems. Comprehensive surveys of works on classical approximation algorithms can be found, e.g., in [32, 45, 47]. Formally, given a maximization (minimization) problem Π, we say that $\mathcal{A}$ is an r-approximation algorithm for some $r \geq 1$, if for any instance I of Π $\mathcal{A}$ yields a solution that satisfies $OPT(I)/\mathcal{A}(I) \leq r$ $(\mathcal{A}(I)/OPT(I) \leq r)$, where $OPT(I)$ is the value of an optimal solution for I. Thus, for instance, *Maximum Independent Set* on a graph $G = (V, E)$, with $|V| = n$, is inapproximable within a ratio better than $n^{1-\epsilon}$, for some $\epsilon > 0$, unless P = NP [48]. Assuming the Unique Games Conjecture (UGC), the *Vertex Cover* problem cannot be approximated in polynomial time to within any constant factor better than 2, and the best constant-factor approximation for *d-Hitting Set* is d [35]. These results lead to the question that is at the heart of our study.

> "Given an optimization problem, Π, that is hard to approximate within factor ρ, for some $\rho > 1$: can we devise a family of α-approximation algorithms, A_α, such that A_ρ is polynomial, A_1 has the running time of the best FPT algorithm for Π, and A_α defines a continuous tradeoff between approximation ratios and running times?"

We show below that our parameterized approximation algorithms have approximation ratios better than the best possible polynomial-time approximation algorithms (under the common assumption that $P \neq NP$, and assuming that UGC holds). Our technique enables us to obtain any ratio $\alpha \in [1, \rho(\Pi)]$ for a given problem Π, where $\rho(\Pi)$ is the best known polynomial-time approximation ratio for the problem, and α is the approximation ratio achieved, depending on the desired running-time of the algorithm. In developing a general paradigm for parameterized approximation, we combine tools used in approximation algorithms with the framework of parameterized complexity. We move now to an overview of our results, after which will follow an in-depth presentation of α-shrinking transformations.

1.1. Our Results

In this paper, we describe a new parameterized approximation paradigm which relates parameterized complexity and polynomial-time approximation. While many earlier studies refer to parametrization by solution size, or, more generally, by the value of the objective function, our approximation approach can be applied for *any parametrization* of a given problem. We demonstrate our techniques with several fundamental problems, including *Vertex Cover*, *d-Hitting Set*, *Connected Vertex Cover*, and *Steiner Tree*.

We summarize our results in Table 1. For each of the studied problems, we specify the kernel size obtained by our algorithms (when applicable), as well as the running time of the algorithm as function of the approximation ratio, $\alpha \geq 1$, and the best known running time of an exact FPT algorithm for the problem.

Problem	**Parameter**	**Kernel size**	**Running time**	**Best FPT algorithm**
Vertex cover	solution size	$2(2-\alpha)k$	$O^*(1.273^{(2-\alpha)k})$	$O^*(1.273^k)$ [13]
Connected vertex cover	solution size	No $k^{O(1)}$	$O^*(2^{k(2-\alpha)})$	$O^*(2^k)$ [17]
3-Hitting set	solution size	$\frac{5(3-\alpha)^2}{4}k^2 + \frac{3-\alpha}{2}k$	$O^*(2.076^{k(3-\alpha)/2})$	$O^*(2.076^k)$ [46]
Steiner tree	size of terminal set	No $k^{O(1)}$	$O^*(2^{(3-\alpha)k/2})$	$O^*(2^k)$ [6]

Table 1: Approximations via α-fidelity Shrinking: Four Examples

One of the most important practical techniques in parameterized complexity is *kernelization*. Here one takes a problem instance specified by $(x, k) \in \Sigma^* \times N$ and produces, typically in polynomial time, a small instance of the problem: (x', k') such that (x, k) is a 'yes' instance iff (x', k') is a 'yes' instance; moreover, $|x'| \leq g(k)$ and usually $k' \leq k$. This technique is widely used in practice, as it usually relies on a number of easily implementable reduction rules.

There are two types of *races* in parameterized complexity research: the race for the smallest possible function $f(k)$ in the running time of an exact algorithm, and the race for the smallest possible function $g(k)$ to bound the size of a kernel. These races are well-established, and the current leader boards are exhibited on the FPT community wiki [42]. Our parameterized approximation paradigm gives rise to a new kind of race, *approximative kernelization*.

As a key tool in our study, we introduce (in Section 2) the concept of *α-shrinking transformation*, for $\alpha \geq 1$. We shall see that applying such transformation to a parameterized problem instance decreases the parameter value, while preserving α-fidelity in the approximation ratio. We show that α-shrinking transformations can be used also as a tool for *approximative kernelization*, to obtain kernels which are smaller than the best known for a given problem. Thus, we define the notion of *α-fidelity kernel*, for $\alpha \geq 1$, where the special case of $\alpha = 1$ is a standard kernel. Such smaller α-fidelity kernels will allow us to solve exactly problem instances more efficiently, while obtaining an α-approximate solution for the original instance.

Our technique yields a continuous tradeoff between the approximation ratios achieved by an algorithm and the running times. This positive feature will allow practitioners to obtain as much accuracy as they can computationally afford. We note that most of our algorithms are easy to implement, and could therefore be practical in use.

In Section 5 we show lower bounds which suggest that the α-shrinking transformations used in solving *Vertex Cover*, *Connected Vertex Cover*, and *d-Hitting Set* cannot be improved. Specifically, polynomial time transformations achieving the same approximation ratios and better running times are unlikely to exist (assuming $P \neq NP$).

In developing our approximation algorithm for Steiner Tree (in Section 4), we make non-standard use of a result of Björklund et al. [6] for solving efficiently the Steiner Tree problem for a subset of the terminals in a given instance.

1.2. Related Work

Recently, it has been proposed that the notion of approximability can be investigated in the framework of fixed-parameter tractability, and various models have been suggested (see, e.g., [11, 14, 22]. These models seek, for example, an FPT-algorithm which on input k either delivers "a no size k dominating set is possible" or produces one of size $2k$. Marx and Razgon [39] follow this approach and present an algorithm with running time $f(k)n^{O(1)}$ that, given an instance of the Edge Multicut problem and an integer $k \geq 1$, either finds a solution of size $2k$ or correctly concludes that no solution of size k exists. This approach was adopted also in developing parameterized approximations for the treewidth of an input graph (see [7] and the references therein). The general subject of parameterized complexity and approximation is well-surveyed by Marx in [38].

A different but very interesting kind of trade-off between exact computation and polynomial approximation has been studied by [44], which proposes to cope with hardness through the usage of *hybrid* algorithms.

Other research has studied the FPT approximability of W-hard problems. The algorithms developed for such problems yield a solution of value $g(k)$ for a problem parameterized by k, where k is the solution size (see, e.g., [31, 22, 24]).

Parameterized approximations for NP-hard problems by "moderately exponential time" algorithms have been studied with the goal of devising algorithms with exponential running time $O(2^{n/r})$ and r large enough. For *Vertex Coloring* the first $O^*(2^{n/0.77})$-time[3] algorithm by Lawler was then improved in a series of papers culminating in a breakthrough $O^*(2^n)$ bound by Björklund et al. [5]. Bourgeois et al. [8] used such algorithms to improve the best known approximation ratios for subgraph maximization and minimum covering problems. The paper [8] also gives results similar to ours for Vertex Cover, however, the technique used seems to be specialized for Vertex Cover and cannot be applied to other problems. A similar approach was developed by Cygan et

---

[3] O^* hides factors polynomial in the input size.

al. [16] (see also [18]). Fernau [28] applied a related approach in deriving parameterized approximation schemes for a class of graph minimization problems. Moderately exponential approximation has been investigated by [11, 14, 22], though with objectives oriented towards development of fixed-parameter algorithms.

Recent works by Brankovic and Fernau [9, 10] present algorithms that achieve approximation ratios of the form $\frac{2\ell+1}{\ell+1}$, where $\ell \geq 1$ is an integer, for Vertex Cover and for 3-Hitting Set. The choice of the value of ℓ defines an interesting tradeoff between performance guarantee and the running time of the algorithm. The main technique used is accelerated branching, applied also in [27], to obtain parameterized approximation algorithms for Total Vertex Cover. The algorithms in [9, 10] outperform our algorithms for Vertex Cover and 3-Hitting Set in terms of running times, however, they can be used to obtain only a restricted set of approximation ratios and are significantly more complicated. We note that our approach for obtaining parameterized α-approximations for these problems can be combined with the techniques used in [9] and [10] to obtain improved running times for some values of $\alpha > 1$. Indeed, let $\mathcal{A}$ be a β-approximation algorithm for Π, as given in [9], or [10], for some $\beta > 1$. Then for $\alpha' = \alpha/\beta$, applying α'-shrinking on the instance, followed by $\mathcal{A}$, we obtain an α-approximation algorithm for Π with better running time than the algorithms presented in this paper.

A notable study in approximation has been *polynomial time approximation schemes (PTAS)*. These provide solutions with cost within $(1+\epsilon)$ of optimal, where the error $\epsilon > 0$ can be chosen arbitrarily close to 0. The running time for these algorithms is typically of the form $n^{O(1/\epsilon)}$. Many problems do not admit a PTAS unless $P = NP$. Many that do admit a PTAS have impractically high running times because of the $1/\epsilon$ in the exponent. This motivates the search for *efficient* polynomial-time approximation schemes (EPTAS) in the parameterized framework, where the $1/\epsilon$ is considered a parameter. Early work was done by Bazgan [3]. The connection between EPTAS and the M-hierarchy was first explained by Cai and Juedes [12]. For recent results relating to the existence of EPTASs for some classical problems, see, e.g., [33, 34, 36].

To our knowledge, there have been few studies that link approximation and kernelization. One of these studies is the work of van Bevern et al. [43]. In a method for kernelizing vertex deletion problems whose goal graphs can be characterized by forbidden induced subgraphs, the paper shows how polynomial time approximation results can be exploited in kernelization.

Recently, our paradigm, as described in an earlier version of this paper [26], inspired the development of a general framework that yields parameterized approximation algorithms also for problems that are known to be W-hard [37].

2. Main Technique: Fidelity Preserving Transformations

We consider languages that consist of words in $U = \{0,1\}^* \times \mathbb{N}$. Define a language to be $\mathcal{L} \subseteq U$, such that $(x, k) \in \mathcal{L}$ implies that $(x, k+1) \in \mathcal{L}$. Such a language can represent any minimization problem in which $k \geq 0$ is the objective value.

For some $\alpha \geq 1$, we say that an algorithm $\mathcal{A}$ is α-approximation for a language $\mathcal{L}$ if the following conditions hold. For any $(x, k) \in U$: (i) if $(x, k) \in \mathcal{L}$ then $\mathcal{A}(x, k)$ returns *true*, and (ii) if $\mathcal{A}(x, k)$ returns *true* then $(x, \alpha k) \in \mathcal{L}$.

While commonly, an approximation algorithm is expected to output an explicit solution, our definition fits well for decision problems. Moreover, it coincides with the standard definition of FPT algorithms, in the special case where $\alpha = 1$. Thus, our theorems do not yield approximation algorithms in the classical meaning of the term. Nevertheless, we note that the resulting algorithms can be easily modified to output solutions.

We consider problems that also have a *parametrization*, that is a function $\kappa : U \to N$. Often, the parametrization of the problem is $\kappa(x, k) = k$. For $\alpha > 1$, our objective is to find an α-approximation algorithm (or, a family of algorithms with varying α values) for a given problem $\mathcal{L}$, whose running time is of the form $f(\kappa(x, k)) \cdot |(x, k)|^{O(1)}$. Such an algorithm is called *fixed parameter α-approximation*. We may view a family of such algorithms, which yield α-approximation for any $\alpha > 1$, as the parameterized analog of an *efficient polynomial time approximation scheme (EPTAS)*, since the running times are polynomial in $|x|$, but may depend arbitrarily on $\kappa(x, k)$. When $\alpha = 1$, we get a fixed-parameter algorithm for the problem. If there exists such an algorithm for a language $\mathcal{L}$, we say that $\mathcal{L}$ is *fixed-parameter tractable* ($\mathcal{L} \in FPT$).

To obtain such an algorithm, we first define the notion of fidelity preserving transformations.

Definition 1. *Given a language $\mathcal{L}$, a transformation $t : U \to U$ is α-fidelity preserving, for a given $\alpha \geq 1$, if the following hold: For any $(x, k) \in U$, (i) if $(x, k) \in \mathcal{L}$ then $t(x, k) \in \mathcal{L}$, and (ii) if $t(x, k) \in \mathcal{L}$ then $(x, \alpha k) \in \mathcal{L}$.*

Indeed, a kernelization of a problem is a 1-fidelity preserving transformation which guarantees that, for any $(x', k') = t(x, k)$, $|x'| \leq f(\kappa(x, k))$ for some function f, and $\kappa(x', k') \leq \kappa(x, k)$.

We now introduce the notion of α-shrinking transformation, an α-fidelity transformation which reduces the magnitude of the parameter κ.

Definition 2. *Given a language $\mathcal{L}$ with parametrization κ, a transformation $t : U \to U$ is α-shrinking of order f if*

(i) t is α-fidelity preserving transformation with respect to $\mathcal{L}$.
(ii) For any $(x, k) \in U$ and $(x', k') = t(x, k)$, it holds that $\kappa(x', k') \leq f(\kappa(x, k))$.

If the transformation t can also be evaluated in polynomial time in $|(x, k)|$, we refer to t as a polynomial α-shrinking *of order f.*

2.1. Approximation via Shrinking

We now show that with α-shrinking transformations, we can significantly improve the running time, if we are willing to settle for an approximation. Given an α-shrinking transformation t of order f, and a parameterized algorithm $\mathcal{A}$ for a problem $\mathcal{L}$, a parameterized approximation algorithm for $\mathcal{L}$ can be obtained as follows. For any $(x, k) \in U$ we simply run $\mathcal{A}(t(x, k))$. If the output of the algorithm is *true* then $t(x, k) \in \mathcal{L}$, and since t is α-fidelity preserving, we have that $(x, \alpha k) \in \mathcal{L}$. Also, if $(x, k) \in \mathcal{L}$ we get that $t(x, k) \in \mathcal{L}$, therefore $\mathcal{A}(t(x, k))$ returns *true*. It follows, that $\mathcal{A}(t(x, k))$ is an α-approximation algorithm for $\mathcal{L}$. The running time of $\mathcal{A}$ is of the form $g(\kappa(x, k)) \cdot \text{poly}(|x|)$, and therefore the running time of $\mathcal{A}(t(x, k))$ is $g(f(\kappa(x, k))) \cdot \text{poly}(|x|)$ plus the time for applying the transformation. When the transformation is polynomial, we get a parameterized α-approximation algorithm. We note that the function g is commonly exponential in κ, thus any reduction of the value of $f(\kappa(x, k))$ yields a significant improvement in the running time of the algorithm.

For example, in Section 3.2, we present an α-shrinking transformation for *Vertex Cover (VC)* of order $(2 - \alpha)k$, for any $1 \leq \alpha \leq 2$. The best known running time of an FPT algorithm for VC is $O^*(1.273^k)$, due to [13]. By combining the two, we obtain a parameterized α-approximation for VC, whose running time is $O^*(1.273^{(2-\alpha)k})$. For $k = 160$, if we are willing to settle for a 1.25-approximation, we reduce the running time by a factor of 2^{14}! Recall that the *klam value* of a parameterized algorithm is a number that bounds the parameter values for which the algorithm might reasonably be expected to be practical [19]. Using the common bound of 10^{20} steps on the running time (see, e.g., [40, 19]), we have that by allowing to obtain a 1.25-approximation, the clam value increases by factor of 1.33, to 254.

2.2. α-Fidelity Kernels

We can also use α-shrinking to generate α-fidelity kernels, defined as follows.

Definition 3. *Given a language $\mathcal{L}$ with parametrization κ, a transformation $t : U \to U$ is an α-fidelity kernel of size f if*

(i) t is an α-fidelity preserving transformation with respect to $\mathcal{L}$.
(ii) There is a function f such that, for any $(x, k) \in U$ and $(x', k') = t(x, k)$, it holds that $\kappa(x', k') \leq \kappa(x, k)$, and $|(x', k')| \leq f(\kappa(x, k))$.
(iii) t can be evaluated in polynomial time in $|(x, k)|$.

We see that α-fidelity kernels generalize the standard notion of kernels. We note that, often, algorithms that apply kernelization on a given input, followed by enumeration over the kernel, are faster than algorithms that use branch and bound on the *original* input. Thus, it makes sense to find an α-fidelity kernel for a problem (whose size is smaller than the 1-fidelity kernel) and then use enumeration to find an approximate solution. Throughout the paper we measure the size of a kernel by the number of vertices in the corresponding graph.

Given a kernelization algorithm that yields a kernel of size $g(k)$ for a problem $\mathcal{L}$, and a *polynomial* α-shrinking t of order f for the problem, we can generate an α-fidelity kernel similar to the way we used shrinking to obtain an approximation algorithm. For any $(x, k) \in U$, we run the kernelization algorithm over $t(x, k)$. We see that the resulting transformation is an α-fidelity kernel of size $g(f(k))$. For *Vertex Cover*, using the α-shrinking of Section 3.2, this leads to an α-fidelity kernel with at most $2(2-\alpha)k$ vertices, for any $1 \leq \alpha \leq 2$.

3. Parametrization by Problem Objective

The reduction steps we use to obtain the α-shrinking are quite simple. We describe them here, and in the following section show how they are applied. Throughout this section, we consider problems for which the parametrization is $\kappa(x, k) = k$. For simplicity, we ignore the κ notation and simply use k.

3.1. Obtaining α-Shrinking by Simple Reduction Steps

To efficiently obtain polynomial α-shrinking, we use as a key building block the following reduction step.

Definition 4. *Given a language $\mathcal{L}$, a transformation $r : U \to U$ is an (a, b)-reduction step if, for any $(x, k) \in U$ and $(x', k') = r(x, k)$,*

(i) $k' = k - a$

(ii) If $(x, k) \in \mathcal{L}$ then $(x', k') \in \mathcal{L}$.

(iii) For any integer $n \geq 0$, if $(x', k' + n) \in \mathcal{L}$ then $(x, k + b + n) \in \mathcal{L}$.

The above reduction step can be applied on the following class of languages. We say that $\mathcal{L}$ is a *polynomial objective language* if there is a polynomial p such that, for any $(x, k) \in U$ with $k \geq p(|x|)$, it holds that $(x, k) \in \mathcal{L}$, and there is an instance $(x, 0) \in \mathcal{L}$. Intuitively, this captures all languages for which, when k is large relative to $|x|$, the problem becomes easy to solve. For example, in the case of Vertex Cover, for any graph $G = (V, E)$, if $k \geq |V|$ then the pair (G, k) is in the language, as the set of all vertices V is a cover. Also, the empty set is a cover for a singleton graph (i.e., a single vertex with no edges). Thus, Vertex Cover is a polynomial objective language.

The reduction step in Definition 4 is useful due to the next lemma.

Lemma 5. *Given a polynomial objective language $\mathcal{L}$, an (a, b)-reduction step r and $\alpha \leq \frac{a+b}{a}$ such that r can be evaluated in polynomial time, there is polynomial α-shrinking of order $\left(k \cdot \frac{b+a-\alpha a}{b}\right)$ for $\mathcal{L}$.*[4]

Proof: Let p be a polynomial such that if $(x, k) \in U$ with $k \geq p(|x|)$ then $(x, k) \in \mathcal{L}$. We note that if r is an (a, b)-reduction step, then r^ℓ is $(a\ell, b\ell)$-reduction step. We use this property as follows. Given $(x, k) \in U$, if $k < p(|x|)$ we select $\ell = k \cdot \frac{(\alpha-1)}{b}$ and apply r^ℓ on (x, k), otherwise we select a fixed instance $(x', 0) \in \mathcal{L}$. Let t denote the resulting transformation. Now notice, if $t(x, k) \in \mathcal{L}$ then $(x, k + b\ell) = (x, \alpha k) \in \mathcal{L}$. Also,

---

[4] For $\alpha > \frac{a+b}{a}$ an α-approximation for the problem can be obtained by iteratively applying the reduction step.

if $(x,k) \in \mathcal{L}$ then $t(x,k) \in \mathcal{L}$, and as r can be evaluated in polynomial time, t can be evaluated in polynomial time as well. This means that t is α-shrinking, and its order is $k' = k - a\ell = k \cdot \frac{(b+a-\alpha a)}{b}$. ∎

For many problems, finding such a reduction step is easy, as described in Section 3.2. In all cases, we rely heavily on ideas used in local-ratio algorithms for the problems. While these algorithms inspired the reduction steps used within our framework for the studied problems, we could not establish a general scheme for applying the local-ratio technique to obtain fidelity preserving transformations. For more details on the local ratio technique, see, e.g., [2].

3.2. Applications of the Technique

In this section we apply our α-shrinking technique to obtain parameterized approximations for Vertex Cover, d-Hitting Set and Connected Vertex Cover.

3.2.1. Vertex Cover

The *Vertex Cover (VC)* problem is defined as follows. Given a graph $G = (V,E)$, a subset of vertices $S \subseteq V$ is a cover of G if, for any edge $(v,u) \in E$, either $v \in S$ or $u \in S$. The VC problem is to find a cover of G of minimum cardinality. As a language, Vertex Cover can be defined by

$$VC = \{(G,k) \mid \text{there is a cover of } G \text{ of size at most } k\}.$$

Given an instance (G,k), we use the following reduction step. For an arbitrarily selected edge (u,v), let $G' = G \setminus \{u,v\}$ (G' is derived from G by removing the vertices u and v along with all adjacent edges). We take $r(G,k) = (G', k-1)$. Let $(G,k) \in U$, denote $(G',k') = r(G,k)$, and let (u,v) be the edge selected by the transformation r. Then, if $(G,k) \in \mathcal{L}$, there is a vertex cover C of G with $|C| \leq k$; either $u \in C$ or $v \in C$. Therefore, $C' = C \setminus \{v,u\}$ is of size at most $k-1$, and C' is a cover of G'. Hence, $(G',k') \in \mathcal{L}$. We also note that if $(G', k'+n) \in \mathcal{L}$, then there is a cover C' of G' of size at most $k'+n = k-1+n$. Let $C = C' \cup \{u,v\}$, then we see that C is a vertex cover of G of size at most $k'+n+2 \leq k+1+n$.

This implies that r is a $(1,1)$-reduction for VC. The reduction r can be evaluated in polynomial time and Vertex Cover is a polynomial objective language.Therefore, by Lemma 5, there is a polynomial α-shrinking for VC of order $k \cdot (2-\alpha)$, for any $1 \leq \alpha \leq 2$. As mentioned above, such shrinking can be used to obtain a parameterized α-approximation algorithm for VC, with running time $O^*(1.273^{(2-\alpha)k})$ and an α-fidelity kernel of size $2(2-\alpha)k$, for any $1 \leq \alpha \leq 2$.

3.2.2. d Hitting-Set

The *d-Hitting Set (d-HS)* problem is the following extension of Vertex Cover to hypergraphs. Given a hypergraph $G = (V,E)$ with edge sizes bounded by d, a set $S \subseteq V$ is a cover of G if, for any $e \in E$, it holds that $e \cap S \neq \emptyset$. The d-HS problem is to find a cover of G of minimum cardinality. As a language, d-Hitting-Set can be defined by $d\text{-HS} = \{(G,k) \mid \text{there is a cover of } G \text{ of size } k\}$.

For any fixed $d \geq 2$, we show a $(1, d-1)$-reduction step for d-HS, which extends the reduction used for VC. Given an instance (G, k), arbitrarily select an edge $e \in E$ and let $G' = (V', E')$, where $V' = V \setminus e$ and $E' = \{e' \in E \mid e' \cap V' = e'\}$. Consider $r(G, k) = (G', k-1)$. It can be easily shown that r is indeed a $(1, d-1)$-reduction step for d-HS, which can be evaluated in polynomial time. Also, it can be easily verified that Hitting Set is a polynomial objective language. By Lemma 5, there is a polynomial α-shrinking for d-HS of order $k \cdot \frac{d-\alpha}{d-1}$, for any $1 \leq \alpha \leq d$.

The best known parameterized algorithm for 3-HS, due to Wahlström [46], has running time $O^*(2.076^k)$. Combining α-shrinking with this algorithm, we obtain a parameterized α-approximation algorithm with running time $O^*(2.076^{k \cdot \frac{3-\alpha}{2}})$, for any $1 \leq \alpha \leq 3$. Abu-Khzam showed a kernelization for d-HS of size $(2d-1)k^{d-1} + k$ [1]. Combining this kernelization with our α-shrinking, we have an α-fidelity kernel for d-HS of size $(2d-1)\left(k \cdot \frac{d-\alpha}{d-1}\right)^{d-1} + k \cdot \frac{d-\alpha}{d-1}$, for any $1 \leq \alpha \leq d$.

3.2.3. Connected Vertex Cover

The *Connected Vertex Cover (CVC)* problem is a variant of VC in which the cover S must be a set of connected vertices in the input graph G. Formally,

$$CVC = \{(G, k) \mid \text{there is a connected cover of } G \text{ of size at most } k\}.$$

In order to define α-shrinking for CVC, we use an ad-hoc variant of the problem to which we refer as *Blue Vertices Connected Vertex Cover (BCVC)*. The input for BCVC is a graph $G = (V, E)$, a set of vertices $B \subseteq V$ which are connected in G (B is the set of blue vertices), and a parameter k. The problem is to determine if there is a set of vertices $S \subseteq V$ of size k or less, such that $S \cup B$ is a connected vertex cover for G. Formally,

$$\text{BCVC} = \{(G, B, k) \mid S \cup B \text{ is a connected vertex cover of } G \text{ for some } S,\ |S| \leq k\ \}.$$

There is a strong connection between the two problems. Clearly, $(G, k) \in CVC$ iff $(G, \emptyset, k) \in BCVC$. Also, consider the following transformation. Given the graph $G = (V, E)$ and the set $B \neq \emptyset$, we define a new graph $G' = (V', E')$ by $V' = \{s, s'\} \cup V \setminus B$ (s and s' are new vertices), and

$$E' = (E \cap (V' \times V')) \cup \{(s, s')\} \cup E^*,$$

where

$$E^* = \{(s, v) \mid \text{ there is } u \in B \text{ such that } (u, v) \in E\}.$$

Intuitively, we map all the vertices in B into a single vertex s and add a new vertex s' that is connected to s. We denote this transformation by $h(G, B) = G'$. Then, we have the following.

Lemma 6. *Given a graph $G = (V, E)$ and a connected set of vertices $\emptyset \subset B \subseteq V$, for any $k \geq 1$, it holds that $(G, B, k) \in BCVC$ iff $(h(G, B), k+1) \in CVC$.*

Proof: Let $G' = h(G, B)$. We show the two directions.

(i) If $(G, B, k) \in$ BCVC then there is a set S, $|S| \leq k$, such that $B \cup S$ is a connected vertex cover of G. Consider the set $S' = S \cup \{s\} \subseteq V'$. All the edges in E^* are covered by S', as well as the edge (s, s'), and also all the edges in $E \cap (V' \times V')$ as they were covered by vertices in S. We note that the set S' is also connected in G', since for any $v \in S'$, $v \neq s$, there is a path in G from v to a vertex in B, which contains only vertices in S. This path can be easily converted to a path in G' connecting v to s, which contains only vertices in S. Hence, we have that S' is a connected vertex cover for G' of size $(k + 1)$, and $(G', k + 1) \in CVC$.

(ii) In the other direction, suppose that $(G', k + 1) \in CVC$, then there is a connected cover S' of G' of size $k + 1$ or less. Note that if B is a cover of G then $(G, B, \ell) \in$ BCVC for any $\ell \geq 0$, and in particular for $\ell = k$. If B is not a cover of G then $s \in S'$ (otherwise the edge (s, s') cannot be covered while keeping S' connected). Consider the set $S = S' \cap V$. The size of S is at most k, as $s \in S'$ and $s \notin V$. Also, $S \cup B$ is a cover of G. Indeed, if $e \in E$ is an edge not covered by B then $e \in E'$; therefore, e is covered by some vertex $v \in S$. Finally, it can be easily shown that $S \cup B$ is connected. Hence, we get that $(G, B, k) \in CVC$ as desired.

■

We now show a $(1, 1)$-reduction step for BCVC. Given (G, B, k), if $B \neq \emptyset$ we select

$$e = (v, u) \mid v \notin B,\ u \notin B, \text{ and there is } b \in B \text{ such that } (b, v) \in E. \tag{1}$$

If $B = \emptyset$ we select an arbitrary edge in G. First, we show that if B is not a cover of G then such an edge exists. For the caes in which $B = \emptyset$ this is trivial. Otherwise, since B is not a cover, there is an edge $e' = (u', v')$ that is not covered by B, and since G is connected (otherwise, there is no connected cover for G), there is a path in G from u' to some vertex $b' \in B$. Let $e_1, e_2, \ldots, e_m$ denote the edges on this path. Let $e_0 = e'$, and denote by $e_0, e_1, \ldots, e_m$ a path from v' to b'. Let e_i be the first edge on the path having an endpoint in B; clearly, $i \geq 1$. Let $e_i = (y, z)$, and $e_{i-1} = (x, y)$. We note that $x \notin B$ and $y \notin B$, due to the minimality of i, and since $z \in B$, the edge e_{i-1} can be selected. Now, given the selection of an edge ($e = (u, v)$ satisfying (1), or arbitrary if $B = \emptyset$), define the transformation t by $t(G, B, k) = (G, B \cup \{u, v\}, k - 1)$.

Lemma 7. *The transformation t is a $(1, 1)$-reduction step for BCVC.*

Proof: For any (G, B, k), let $t(G, B, k) = (G, B', k')$, and let $e = (v, u)$ be the edge selected by t. Clearly, $k' = k - 1$, and the first condition in Definition 4 is satisfied. If $(G, B, k) \in$ BCVC then there is a vertex set $S \subseteq V$ such that $B \cup S$ is a connected vertex cover and $|S| \leq k$. Since e is not covered by B, either $u \in S$ or $v \in S$. Let $S' = S \setminus \{u, v\}$, then $|S'| \leq k - 1$ and $S' \cup B'$ is a connected cover of G (note that $S \cup B \subseteq S' \cup B'$), therefore $(G, B', k') \in$ BCVC.

If $(G, B', k' + n) \in$ BCVC then there is a vertex set $S' \subseteq V$ of size $(k' + n)$ or less such that $B' \cup S'$ is a connected vertex cover of G'. Let $S = S' \cup \{u, v\}$, then

$|S| \leq k' + n + 2 = k + n + 1$, and $S \cup B = B' \cup S'$ is a connected cover of G; therefore, $(G, B, k + n + 1) \in$ BCVC. Hence, t is a $(1, 1)$-reduction step as desired. ∎

By Lemma 5, for any $\alpha \in [1, 2]$, there is a polynomial α-shrinking of order $k(2 - \alpha)$ for BCVC (we note that BCVC is also a polynomial objective language). Denote this transformation by t_α. Given (G, k), consider $r_\alpha(G, k) = h(t_\alpha(G, \emptyset, k))$, where h is the transformation defined above. The transformation r_α converts an instance of CVC to another instance of CVC. Let $(G', k') = r_\alpha(G, k)$. If $(G, k) \in CVC$ then $(G, \emptyset, k) \in$ BCVC, therefore $t_\alpha(G, \emptyset, k) \in$ BCVC and $(G', k') = h(t_\alpha(G, \emptyset, k)) \in CVC$. Also, if $(G', k') \in CVC$ then, by Lemma 6, $t_\alpha(G, \emptyset, k) \in$ BCVC, therefore $(G, \emptyset, \alpha k) \in$ BCVC and, finally, $(G, \alpha k) \in CVC$. Also, $k' = k(2 - \alpha) + 1$, and we get that r_α is a polynomial α-fidelity shrinking of order $k(2 - \alpha) + 1$. We summarize in the next result.

Theorem 8. *For any $\alpha \in [1, 2]$, there is a polynomial α-fidelity shrinking of order $k(2 - \alpha) + 1$ for Connected Vertex Cover.*

The best known (deterministic) parameterized algorithm for CVC, due to [4], has running time 2.488^k (ignoring polynomial factors). Combining this algorithm with the above α-fidelity shrinking for CVC leads to a parameterized α-approximation algorithm for CVC with running time of $O^*(2.488^{k(2-\alpha)})$. Cygan et al. [17] proposed a Monte-Carlo algorithm whose running time is $O^*(2^k)$. The algorithm of Cygan et al. never gives false positive and gives a false negative with probability at most 1/2. Combining this algorithm with the above α-fidelity shrinking gives an approximation algorithm $\mathcal{A}$ which satisfies the following. If $(G, k) \in CVC$ then $\mathcal{A}(G, k)$ returns true with probability at least 1/2, and whenever $\mathcal{A}(G, k)$ returns true, it holds that $(G, \alpha k) \in CVC$. The running time of the algorithm is $O^*(2^{k(2-\alpha)})$.

4. The Parametrized Steiner Tree Problem

The *Steiner Tree (ST)* problem is defined as follows. We are given an undirected graph $G = (V, E)$, a set of terminals $T \subseteq V$, and a value $k \geq 1$. We say that a subset of edges E' is a *Steiner tree*, if E' forms a tree, and for any $v \in T$ there is $(u, v) \in E'$. Our objective is to determine if T has a Steiner tree in G of size k or less. Formally, denote by $ST_G(T)$ a Steiner tree of T of minimum size in G, and let $\text{ST} = \{(G, T, k) \mid |ST_G(T)| \leq k\}$.

We consider ST with its standard parametrization, by the number of terminals, that is, $\kappa(G, T, k) = |T|$. We define below an α-shrinking transformation. While the running time of our shrinking procedure is non-polynomial, it still yields a significant improvement over the running time of the best known exact algorithm.

4.1. The Shrinking Technique for Parameterized Steiner Tree

4.1.1. Overview

Our shrinking technique is based on the following observations, whose proofs are given below.

(1) Given a subset $S \subseteq T$, the graph G and the set of terminals T can be reduced to G' and T', respectively, such that *(i)* $|T'| = |T| - |S| + 1$, *(ii)* if $(G', T', k) \in \mathrm{ST}$ then $(G, T, k + |\mathrm{ST}_G(S)|) \in \mathrm{ST}$, and *(iii)* if $(G, T, k) \in \mathrm{ST}$ then $(G', T', k) \in \mathrm{ST}$.

(2) For any $\ell \geq 1$, there is $S \subseteq T$ of size ℓ, such that $|\mathrm{ST}_G(S)| \leq |ST_G(T)| \cdot \frac{2 \cdot \ell}{|T|}$.

(3) For any $\ell \geq 1$, a subset $S \subseteq T$ of size ℓ for which $|\mathrm{ST}_G(S)|$ is minimal can be found in $O^*(h(|T|, \ell))$-time, where $h(|T|, \ell)$ is the number of subsets of T of size at most ℓ.

Using the above observations, we define our shrinking procedure as follows. Given a value of $\alpha \geq 1$, we select $\ell = \frac{(\alpha-1)}{2}|T|$ and find a subset $S \subseteq T$ of size ℓ for which $|\mathrm{ST}_G(S)|$ is minimal. By (2) we have that $|ST_G(S)| \leq (\alpha - 1)|ST_G(T)|$; therefore, by Definition 2 and (1), we have that the graph G' and the set T' are α-shrinking of (G, T, k) of order $f(|T|) = \frac{3-\alpha}{2}|T| + 1$.

4.1.2. Reducing the graph

For any $S \subseteq T$, we define $G_S = (V_S, E_S)$, which is basically the graph G after merging all vertices in S into a single vertex, as follows. The set of vertices is $V_S = V \cup \{s\} \setminus S$, where s is a new vertex. The set of edges is

$$E_S = (E \cap (V_S \times V_S)) \cup \{(v, s) | \text{there is } u \in S \text{ such that } (v, u) \in E\}.$$

The set of terminals is $T_S = T \cup \{s\} \setminus S$. Notice, $|T_S| = |T| - |S| + 1$. Given a Steiner tree H of T in G, let its projection on G_S be $H_S = (H \cap E_S) \cup \{(u, s) | (u, v) \in H, v \in S\}$. We note that H_S is a connected component in G_S, which spans the vertices in T_S. Thus, $|ST_{G_S}(T_S)| \leq |H_S| \leq |H|$.

Now, given a Steiner tree H_S of T_S in G, let H consist of all edges of H_S which are in G, an edge (u, v) for each $(u, s) \in H$, where $v \in S$ is arbitrarily chosen, and also $ST_G(S)$. It is not difficult to see that H is a connected component in G which spans T; therefore, we have that $|ST_G(T)| \leq |H_S| + |ST_G(S)|$. The next lemma shows the existence of a good subset of size ℓ.

Lemma 9. *For any $\ell \geq 1$ satisfying $(|T| \bmod \ell) = 0$, there is $S \subseteq T$ of size ℓ, such that $|ST_G(S)| \leq |ST_G(T)| \cdot \frac{2\ell}{|T|}$.*

Proof: Let H be a Steiner Tree for G, T of minimal size. Consider a *Depth-First-Search (DFS)* on the tree H (see, e.g., [15]). Starting with $i = 1$ and $T_1 = \emptyset$, whenever we reach a vertex $v \in T$ for the first time we add it to the set T_i. When we get $|T_i| = \ell$ we increase i by one and set $T_i = \emptyset$. Since ℓ divides $|T|$, at the end of the process we get exactly $m = |T|/\ell$ sets T_i, each of size ℓ.

For all $1 \leq i \leq m$ let H_i be the set of edges DFS traversed between the discovery of the first vertex in T_i and the discovery of the last vertex in T_i. The set H_i forms a connected component which spans all the vertices in T_i, therefore $|ST_G(T_i)| \leq |H_i|$. Algorithm DFS crosses each edge at most twice, and therefore $\sum_{i=1}^{m} |H_i| \leq 2|H|$, and by the pigeon hole principle, we get that there is i such that $|H_i| \leq 2|H|/m$ ∎

4.1.3. Finding a good subset

To find a subset $S \subseteq T$ of size ℓ, such that $|\mathrm{ST}_G(S)|$ is minimal, we use a slight adaptation of the algorithm of [6] for the (parametrized) Steiner Tree problem. The algorithm uses the recursive formula of Dreyfus and Wagner [25] . For any $q \in V$ and $X \subseteq T \setminus \{q\}$,

$$|ST_G(\{q\} \cup X)| = \min_{p \in V} \{|\mathrm{ST}_G(\{p, q\})| + g_p(X)\},$$

where

$$g_p(X) = \min_{\emptyset \subset D \subset X} \{|\mathrm{ST}_G(\{p\} \cup D)| + |\mathrm{ST}_G(\{p\} \cup (X \setminus D)|\}.$$

While a simple bottom up evaluation of the formula has running time $3^{|T|}$, the algorithm of [6] is based on evaluating $g_p(X), |ST_G(\{q\} \cup X)|$ for sets $X \subseteq T$ of increasing size, by using a subset convolution algorithm, with running time $2^{|T|}$. This results in a total running time of $O^*(2^{|T|})$[5]

To find the desired set S, we need to evaluate $g_p(X), |ST_G(\{q\} \cup X)|$ for all sets $X \subseteq T$ satisfying $|X| \leq \ell$. While not explicitly mentioned in [6], we note that, given the values of $|ST_G(\{q\} \cup X)|$ for any $X \subseteq T$ of size at most r, we can evaluate $g_p(X)$, for any $X \subseteq T$ of size at most $r+1$, in time $O^*(h(|T|, r+1))$. This is done by using the convolution algorithm only over sets of size at most $r+1$. Therefore, we can evaluate $|ST_G(\{q\} \cup X)|$, for all sets $X \subseteq T$ such that $|X| \leq \ell$, in time $O^*(h(|T|, \ell))$. Now, we can find the set S for which $|\mathrm{ST}_G(S)|$ is minimal, by going over all subsets. Thus, we have

Lemma 10. *For any $\ell \geq 1$, a subset $S \subseteq T$ of size ℓ for which $|ST_G(S)|$ is minimized can be found in time $O^*(h(|T|, \ell))$, where $h(|T|, \ell)$ is the number of subsets of T of size at most ℓ.*

Combining the previous results, and using Stirling's approximation to evaluate the running time, we summarize in the following theorem.

Theorem 11. *For any $\alpha \in [1, 3/2]$, there is an α-shrinking of order $f_\alpha(|T|) = \frac{3-\alpha}{2}|T|+1$ for parameterized Steiner Tree, which can be evaluated in time $O^*(((\frac{1}{\beta})^\beta \cdot (\frac{1}{1-\beta})^{1-\beta})^{|T|})$, where $\beta = (\alpha - 1)/2$.*

4.2. Applicability

Define $g(\beta) = \left(\frac{1}{\beta}\right)^{\frac{\beta}{1-\beta}} \cdot \frac{1}{1-\beta}$, and note that the running time of the shrinking procedure can be written as $g(\beta)^{f_\alpha(|T|)-1} = g\left(\frac{\alpha-1}{2}\right)^{f_\alpha(|T|)-1}$ (f_α and β are defined as in Theorem 11). Apply the α-shrinking for the given input, and run the algorithm of [6] on the reduced

---

[5] For a comprehensive exposition of the subset convolution tool and known techniques we refer the reader to [30].

instance. We obtain an α-approximation algorithm for the Steiner Tree problem. The running time of the algorithm is $g\left(\frac{3-\alpha}{2}\right)^{f_\alpha(|T|)-1} + 2^{f_\alpha(|T|)}$ (ignoring polynomial factors). For any $1 \leq \alpha \leq 1.4$, we have $g\left(\frac{3-\alpha}{2}\right) \leq 2$; therefore, the running time of the algorithm is $2^{f_\alpha(|T|)} = 2^{\frac{(3-\alpha)}{2}|T|}$ (ignoring polynomial factors).

Theorem 12. *There is a parametrized α-approximation algorithm for the Steiner Tree problem parametrized by the number of terminals, whose running time is $O^*(2^{\frac{(3-\alpha)\kappa}{2}})$, for any $\alpha \in [1, 1.4]$.*

5. Lower Bounds for Linear Shrinking

In this section we show a lower bound for the order of a *linear* polynomial α-shrinking transformation. We say the an α-shrinking transformation t is *linear* if $(x', k') = t(x, k)$, and $(x', k' + n) \in \mathcal{L}$ implies $(x, \alpha k + n) \in \mathcal{L}$. Note that this property holds for each of the transformations in Section 3.2. The assumption for the lower bound is that t is not expected to yield an approximation ratio better than the best known for the problem at hand (else, it is the best approximation no more).

For a given language $\mathcal{L}$, assume that the best polynomial-time approximation algorithm, $\mathcal{A}$, has approximation ratio of $\beta > 1$. Now, let t be a linear polynomial α-shrinking transformation of order f.

Let $\mathcal{B}$ be the algorithm $\mathcal{A}(t(x, k))$. If $(x, k) \in \mathcal{L}$ then $t(x, k) \in \mathcal{L}$, therefore $\mathcal{B}(x, k) = \mathcal{A}(t(x, k)) = \textit{true}$. If $\mathcal{B}(x, k) = \mathcal{A}(t(x, k)) = \textit{true}$, then let $(x', k') = t(x, k)$. Since $\mathcal{A}$ yields a β-approximation, we have that $(x', \beta k') \in \mathcal{L}$. By the linearity of t, we have $(x, \alpha k + (\beta - 1)k') \in \mathcal{L}$. This implies that $\mathcal{B}$ is a polynomial time $\left(\alpha + (\beta - 1)\frac{f(k)}{k}\right)$-approximation algorithm for $\mathcal{L}$. Unless we improve the ratio of the best known algorithm for $\mathcal{L}$, this implies that $\left(\alpha + (\beta - 1)\frac{f(k)}{k}\right) \geq \beta$. Consequently, $f(k) \geq \frac{\beta-\alpha}{\beta-1}k$.

Thus, assuming there is no polynomial-time ρ-approximation algorithm for VC and CVC, for any $\rho < 2$, the above implies that a linear α-shrinking for these problems cannot have order smaller than $f(k) = (2 - \alpha)k$. And, assuming there is no polynomial-time ρ-approximation algorithm for d-HS, for $\rho < d$, a linear α-shrinking for d-HS cannot have order smaller than $f(k) = \frac{d-\alpha}{d-1}k$.

We remark that the above results do not rule out the existence of *non-linear* transformations having better running times, or the existence of better approximation ratios for VC, CVC and d-HS. Thus, in attempting to improve the order of any of the transformations presented in Section 3.2, one should either seek non-linear transformations, or expect the task to be at least as hard as improving the best known approximation ratios for these problems.

6. Discussion

We introduced a new parameterized approximation paradigm with important and general features. Our algorithms, which obtain any approximation ratio between 1 and the best known P-time ratio for a given problem, yield a continuous trade-off between approximation and running times.

We showed how our key tool of α-shrinking transformations can be applied to obtain parameterized approximation algorithms for several fundamental problems. We further showed that, even when the running time of our shrinking procedure is non-polynomial (as in the Steiner Tree problem), it can still yield significant improvement over the running time of an exact algorithm. Finally, we note that in applying our technique, the problem parameter is not restricted to be the solution size.

We point to a few of the many avenues for future work.

- Further explore the generic approach of approximations based on α-fidelity shrinking and seek efficient application to other problems, such as Feedback Vertex Set, Edge Dominating Set, and others.
- As noted above, our approximation algorithms rely heavily on ideas used in local-ratio algorithms for the studied problems. It would be interesting to study the applicability of other approximation techniques, such as the primal-dual schema, in which our α-fidelity shrinking will be used, e.g., to reduce the size of the linear programming formulation for the problem.
- Further explore approximative kernelization. For example, can non-linear reduction (kernelization) rules be used to obtain decreased running times?
- Extend the approach to maximization problems (parameterized by a value other than the solution size, k).

References

[1] F. N. Abu-Khzam. Kernelization algorithms for d-hitting set problems. In *WADS*, pages 434–445, 2007.

[2] R. Bar-Yehuda. One for the price of two: a unified approach for approximating covering problems. *Algorithmica*, 27(2):131–144, 2000.

[3] C. Bazgan. *Approximation schemes and parameterized complexity.* PhD Thesis, INRIA, Orsay, France, 1995.

[4] D. Binkele-Raible and H. Fernau. Parameterized measure and conquer for problems with no small kernels. *Algorithmica*, pages 1–24. 10.1007/s00453-011-9566-6.

[5] A. Björklund and T. Husfeldt. Inclusion–exclusion algorithms for counting set partitions. In *FOCS*, pages 575–582, 2006.

[6] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. Fourier meets Möbius: fast subset convolution. In *STOC*, pages 67–74, 2007.

[7] H. L. Bodlaender, P. G. Drange, M. S. Dregi, F. V. Fomin, D. Lokshtanov, and M. Pilipczuk. A $O(c^k n)$ 5-approximation algorithm for treewidth. In *FOCS*, 2013.

[8] N. Bourgeois, B. Escoffier, and V. T. Paschos. Efficient approximation of combinatorial problems by moderately exponential algorithms. In *WADS*, pages 507–518, 2009.

[9] L. Brankovic and H. Fernau. Parameterized approximation algorithms for hitting set. In *WAOA*, 2011.

[10] L. Brankovic and H. Fernau. A novel parameterised approximation algorithm for minimum vertex cover. *Theor. Comput. Sci.*, 511:85–108, 2013.

[11] L. Cai and X. Huang. Fixed-parameter approximation: Conceptual framework and approximability results. *Algorithmica*, 57(2):398–412, 2010.

[12] L. Cai and D. W. Juedes. Subexponential parameterized algorithms collapse the W-hierarchy. In *ICALP*, pages 273–284, 2001.

[13] J. Chen, I. A. Kanj, and G. Xia. Improved parameterized upper bounds for vertex cover. In *MFCS*, pages 238–249, 2006.

[14] Y. Chen, M. Grohe, and M. Grüber. On parameterized approximability. In *IWPEC*, pages 109–120, 2006.

[15] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* The MIT Press, 3rd edition, 2009.

[16] M. Cygan, L. Kowalik, M. Pilipczuk, and M. Wykurz. Exponential-time approximation of hard problems. *CoRR*, abs/0810.4934, 2008.

[17] 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. In *FOCS*, 2011.

[18] M. Cygan and M. Pilipczuk. Exact and approximate bandwidth. *Theor. Comput. Sci.*, 411(40-42):3701–3713, 2010.

[19] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer, 1999.

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

[21] R. G. Downey, M. R. Fellows, and M. A. Langston. The computer journal special issue on parameterized complexity: Foreword by the guest editors. *Comput. J.*, 51(1):1–6, 2008.

[22] R. G. Downey, M. R. Fellows, C. McCartin, and F. A. Rosamond. Parameterized approximation of dominating set problems. *Inf. Process. Lett.*, 109(1):68–70, 2008.

[23] R. G. Downey and D. M. Thilikos. Confronting intractability via parameters. *Computer Science Review*, 5(4):279–317, 2011.

[24] M. Drescher and A. Vetta. An approximation algorithm for the maximum leaf spanning arborescence problem. *ACM Transactions on Algorithms*, 6(3), 2010.

[25] S. E. Dreyfus and R. A. Wagner. The steiner problem in graphs. *Networks*, 1(3):195–207, 1971.

[26] M. R. Fellows, A. Kulik, F. A. Rosamond, and H. Shachnai. Parameterized approximation via fidelity preserving transformations. In *ICALP*, pages 351–362, 2012.

[27] H. Fernau. Saving on phases: Parametrized approximation for total vertex cover. In *IWOCA*, 2012.

[28] H. Fernau. A systematic approach to moderately exponential-time approximation schemes. Manusctript, 2012.

[29] J. Flum and M. Grohe. *Parameterized Complexity Theory*. An EATCS Series: Texts in Theoretical computer Science. Springer, 1998.

[30] F. V. Fomin and D. Kratsch. *Exact Exponential Algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

[31] M. Grohe and M. Grüber. Parameterized approximability of the disjoint cycle problem. In *ICALP*, pages 363–374, 2007.

[32] D. S. Hochbaum. *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, 1997.

[33] K. Jansen. Parameterized approximation scheme for the multiple knapsack problem. In *SODA*, pages 665–674, 2009.

[34] K. Jansen. An EPTAS for scheduling jobs on uniform processors: Using an MILP relaxation with a constant number of integral variables. *SIAM J. Discrete Math.*, 24(2):457–485, 2010.

[35] S. Khot and O. Regev. Vertex cover might be hard to approximate to within 2-ϵ. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008.

[36] A. Kulik and H. Shachnai. There is no EPTAS for two-dimensional knapsack. *Inf. Process. Lett.*, 110(16):707–710, 2010.

[37] D. Lokshtanov, F. Panolan, M. S. Ramanujan, and S. Saurabh. Lossy kernelization. Manuscript, 2015.

[38] D. Marx. Parameterized complexity and approximation algorithms. *Comput. J.*, 51(1):60–78, 2008.

[39] D. Marx and I. Razgon. Constant ratio fixed-parameter approximation of the edge multicut problem. *Information Processing Letters*, 109(20):1161 – 1166, 2009.

[40] R. Niedermeier. Some prospects for efficient fixed parameter algorithms. In *SOFSEM '98: Theory and Practice of Informatics, 25th Conference on Current Trends in Theory and Practice of Informatics, Jasná, Slovakia, November 21-27, 1998, Proceedings*, pages 168–185, 1998.

[41] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford Lecture Series in Mathematics and Its Applications. Oxford Univerity Press, 2006.

[42] Parameterized Complexity community Wiki. `http://fpt.wikidot.com/`.

[43] R. van Bevern, H. Moser, and R. Niedermeier. Kernelization through tidying. In *LATIN*, pages 527–538, 2010.

[44] V. Vassilevska, R. Williams, and S. L. M. Woo. Confronting hardness using a hybrid approach. In *SODA*, pages 1–10, 2006.

[45] V. V. Vazirani. *Approximation Algorithms.* Springer, 2001.

[46] M. Wahlström. *Algorithms, Measures and Upper Bounds for Satisfiability and Related Problems.* PhD thesis, Department of Computer and Information Science, Linkopings University, Sweden, 2007.

[47] D. P. Williamson and D. B. Shmoys. *The Design of Approximation Algorithms.* Cambridge University Press, 2011.

[48] D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *STOC*, pages 681–690, 2006.