Analysis of Two-variable Recurrence Relations with Application to Parameterized Approximations

Ariel Kulik
Computer Science Department
Technion
Haifa, Israel
kulik@cs.technion.ac.il

Hadas Shachnai
Computer Science Department
Technion
Haifa, Israel
hadas@cs.technion.ac.il

***Abstract*—In this paper we introduce randomized branching as a tool for parameterized approximation and develop the mathematical machinery for its analysis. Our algorithms substantially improve the best known running times of parameterized approximation algorithms for Vertex Cover and 3-Hitting Set for a wide range of approximation ratios.**

The running times of our algorithms are derived from an asymptotic analysis of a broad class of two-variable recurrence relations. Our main theorem gives a simple formula for this asymptotics. The formula can be efficiently calculated by solving a simple numerical optimization problem, and provides the mathematical insight required for the algorithm design. To this end, we show an equivalence between the recurrence and a stochastic process. We analyze this process using the *method of types*, by introducing an adaptation of Sanov's theorem to our setting. We believe our novel analysis of recurrence relations which is of independent interest is a main contribution of this paper.

***Index Terms*—Recurrences and difference equations; Combinatorial algorithms**

I. Introduction

In search of tools for deriving efficient parameterized approximations, we explore the power of randomization in branching algorithms. Recall that a cover of an undirected graph $G = (V, E)$ is a subset $S \subseteq V$ such that for any $(u, v) \in E$ it holds that $S \cap \{u, v\} \neq \emptyset$. The *Vertex Cover* problem is to find a cover of minimum cardinality for G. In Vertex Cover parameterized by the solution size, k, we are given an integer parameter $k \geq 1$, and we wish to determine if G has a vertex cover of size k in time $O^*(f(k))$, for some computable function f.[1]

Consider the following simple algorithm for the problem. Recursively pick a vertex v of degree at least 3, and branch over the following two options: v is in the cover, or three of v's neighbors are in the cover. If the maximal degree is 2 or less then find a minimal vertex cover in polynomial time. The algorithm has a running time $O^*(1.4656^k)$ (see Chapter 3 in [1] for more details).

The randomized branching version of this algorithm, $\mathrm{VC3}_\gamma$, replaces branching by a random selection with some probability $\gamma \in (0, 1)$. In each recursive call the algorithm selects either v or three of its neighbors for the solution, with probabilities γ and $1-\gamma$, respectively (see Figure 2 for a formal description of the algorithm). If v is in a minimal cover then the algorithm has probability γ to decrease the minimal cover size by one, and probability $1-\gamma$ to select three vertices into the solution, possibly with no decrease in the minimal cover size. A similar argument holds in case v is not in a minimal cover. This suggests that the function $p(b, k)$ defined in equation (1) lower bounds the probability the above algorithm returns a cover of size b, given a graph which has a cover of size k.

$$p(b,k) = \min \begin{cases} \gamma \cdot p(b-1,k-1) \\ \quad + \ (1-\gamma) \cdot p(b-3,k) & k \geq 1 \\ \\ \gamma \cdot p(b-1,k) \\ \quad + \ (1-\gamma) \cdot p(b-3,k-3) & k \geq 3 \end{cases} \tag{1}$$

$$p(b,k) = 0 \qquad \forall b < 0, k \in \mathbb{N}$$

$$p(b,0) = 1 \qquad \forall b \geq 0$$

Thus, for any $\alpha > 1$, we can obtain an α-approximation with constant probability by repeating the randomized branching process $\frac{1}{p(\alpha k,k)}$ times. While $p(b, k)$ can be evaluated using dynamic programming, for any $b, k \geq 0$, finding the asymptotic behavior of $\frac{1}{p(\alpha k,k)}$ as $k \to \infty$, which dominates the running time of our algorithm, is less trivial.

II. Our Results

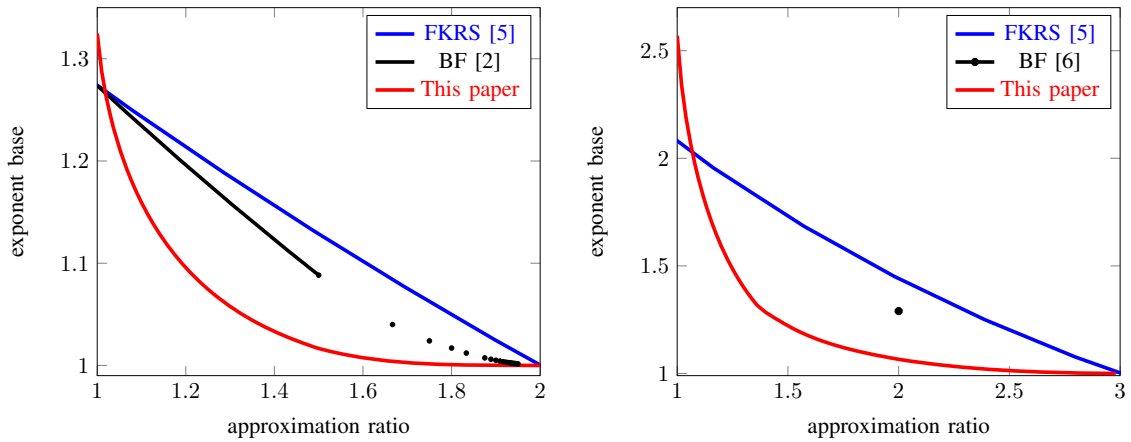
In this paper we show that randomized branching is a highly efficient tool in the development of parameterized approximation algorithms for Vertex Cover and 3-Hitting Set, leading to significant improvements in running times over algorithms developed by using existing tools.[2] One notable example is a simple parameterized random 1.5-approximation algorithm for Vertex Cover, whose running time of $O^*(1.01657^k)$ substantially improves the currently best known $O^*(1.0883^k)$ algorithm for the problem [2].

To evaluate the running times of our algorithms, we develop mathematical tools for analyzing the asymptotic behavior of a wide class of two-variable recurrence relations generalizing the relation in (1). To this end, we introduce an adaptation of Sanov's theorem [3] (see also [4]) to our setting, which

[1]The notation O^* hides factors polynomial in the input size.

[2]See Section II-A for a formal definition of 3-Hitting Set.

2575-8454/20/$31.00 ©2020 IEEE
DOI 10.1109/FOCS46700.2020.00076



(a) Vertex Cover

(b) 3-Hitting Set

Fig. 1. Results for Vertex Cover and 3-Hitting Set. A dot at (α, c) means that the respective algorithm outputs α-approximation in time $O^*(c^k)$ or $O^*\left((c+\varepsilon)^k\right)$ for any $\varepsilon > 0$.

facilitates the use of *method of types* and *information theory* for the first time in the analysis of *branching* algorithms. We believe our novel analysis of recurrence relations which is of independent interest is a main contribution of this paper.

A. Vertex Cover and 3-Hitting Set

We say that an algorithm $\mathcal{A}$ is a *parameterized random α-approximation for Vertex Cover* if, given a graph G and a parameter k, such that G has a vertex cover of size k, $\mathcal{A}$ returns a vertex cover S of G satisfying $|S| \leq \alpha k$ with constant probability $\lambda > 0$, and has running time $O^*(f(k))$. We refer the reader to [5], [6], [7] for similar and more general definitions.

Vertex Cover: Our results for Vertex Cover include two parameterized random α-approximation algorithms, EnhancedVC3* and BetterVC. Algorithm EnhancedVC3* uses a single branching rule (either v or $N(v)$ are in a minimal cover) and has the best running times for approximation ratios greater than 1.4. We note that this simple algorithm outputs a 1.5-approximation in time $O^*(1.01657^k)$.

Algorithm BetterVC is more complex. It is based on a parameterized $O^*(1.33^k)$ algorithm for Vertex Cover presented in [8]. BetterVC achieves the best running times for approximation ratios smaller than 1.4. This algorithm shows that applying randomization in a sophisticated branching algorithm can result in an excellent tradeoff between approximation and time complexity for approximation ratios approaching 1.

Figure 1(a) shows a graphical comparison between our results and the previous best known results [2], [5].[3] The values for several selected approximation ratios are given in Table I. The results for Vertex Cover are presented in Section IV.

3-Hitting Set: The input for 3-Hitting Set is a hypergraph $G = (V, E)$, where each hyperedge e contains at most 3 vertices, i.e., $|e| \leq 3$. We refer to such hypergraph as 3-*hypergraph*. We say that a subset $S \subseteq V$ is a *hitting set* if, for every $e \in E$, $e \cap S \neq \emptyset$. The objective is to find a hitting set of minimum cardinality. In the parameterized version, the goal is to determine if the input graph has a hitting set of at most k vertices, where $k \geq 1$ is the parameter.

We say that an algorithm $\mathcal{A}$ is a *parameterized random α-approximation for 3-Hitting Set* if, given a 3-hypergraph G and a parameter k, such that G has a hitting set of size k, $\mathcal{A}$ returns a hitting set S of G satisfying $|S| \leq \alpha k$ with a constant probability $\lambda > 0$, and has running time $O^*(f(k))$.

In Section V we present a parameterized random α-approximation algorithm for 3-Hitting Set for any $1 < \alpha < 3$. The algorithm, 3HS can be viewed as an adaptation of EnhancedVC3* to hypergraphs, using the following observation. For any $v \in V$ we define the *neighbors graph* of v as the hypergraph in which $\{u, w\}$ (or $\{u\}$) is an edge if $\{u, v, w\}$ ($\{u, v\}$) is an edge in the original hypergraph. It holds that for any hitting set S either $v \in S$ or S contains a hitting set of the neighbors graph of v. The actual branching rules of 3HS were determined via computer-aided search tree generation, using the above observation.

While 3HS may not be the best for approximation ratios close to 1, it yields a significant improvement over previous results for higher approximation ratios. For $\alpha = 2$ the running time is $O^*(1.0659^k)$, substantially improving the best known result of $O^*(1.29^k)$ due to [6]. Figure 1(b) gives a graphical comparison between the running times achieved in this paper and the results of [6] and [5].

We note that while our algorithms yield significant improvements in running times for both Vertex Cover and 3-Hitting Set over the algorithms of [6], [2] and [5], the previous algorithms

[3]The running times presented in Figure 1 are extrapolations of numerically evaluated running times for 99 evenly spaced approximation ratios over the relevant range, with possible additional approximation ratios close to the endpoints of this range.

TABLE I
RUNNING TIMES FOR VERTEX COVER

ratio	1.1	1.2	1.3	1.4	1.5	1.666	1.75	1.8	1.9
BF [2]	1.235	1.197	1.160	1.1232	1.0883	1.0396	1.0243	1.0166	1.0051
This paper	1.160	1.096	1.058	1.0331	1.0166	1.0043	1.0016	1.00073	1.000083

A comparison of the running time of the best algorithm presented in this paper for Vertex Cover for a given approximation ratio to the previous best results due to Brankovic and Fernau [2]. A value of c for ratio α means that the respective algorithm yields an α-approximation with running time $O^*(c^k)$. The set of values selected for α matches the set of approximation ratios listed in [2]. The running times are always rounded up.

are deterministic; our algorithms use randomization as a key tool.

B. Recurrence Relations

The objective of our algorithms is to find a cover of a graph under the restriction that this cover must not exceed a given budget. The algorithms consist of a recursive application of a random branching step. Each time the step is executed it adds vertices to the solution, thereby decreasing the available budget, and possibly reducing the number of vertices required to complete the solution. To analyze the running times of our algorithms, we need to evaluate the probability of obtaining a cover satisfying the budget constraint.

Similar to branching algorithms, this property can be formulated using a recurrence relation. In our case, the recurrence relation defines a function $p : \mathbb{Z} \times \mathbb{N} \to [0, 1]$ satisfying the following equations.[4]

$$\begin{aligned} p(b,k) &= \min_{\{1\le j\le N|\ \bar{k}^j\le k\}} \sum_{i=1}^{r_j} \bar{\gamma}_i^j \cdot p(b-\bar{b}_i^j, k-\bar{k}_i^j) \\ p(b,k) &= 0 \qquad \forall b<0, k\in\mathbb{N} \\ p(b,0) &= 1 \qquad \forall b\ge 0, \end{aligned} \tag{2}$$

where $N \in \mathbb{N}$, and for any $1 \le j \le N$ the following hold: $\bar{b}^j \in \mathbb{N}_+^{r_j}$, $\bar{k}^j \in \mathbb{N}^{r_j}$ and $\bar{\gamma}^j \in \mathbb{R}_+^{r_j}$ with $\sum_{i=1}^{r_j} \bar{\gamma}_i^j = 1$. We say that $\bar{k}^j \le k$ if $\bar{k}_i^j \le k\ \forall 1 \le i \le r_j$. We refer to the recurrence relation in (2) as the **composite recurrence** of $\{(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j) | 1 \le j \le N\}$. Note that for the recurrence to be properly defined, there must be $1 \le j \le N$ such that $\bar{k}^j \le 1$ (otherwise the $\min$ operation in (2) may be taken over an empty set). Throughout this section we use the word *term* when referring to triplets such as $(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j)$.

In the context of our randomized branching algorithms, the number of terms, N, corresponds to the number of possible branching *states* (note, this is different than the number of *branching rules*). For example, in VC3$_\gamma$ (See Section IV and an informal outline at the beginning of Section I) there are two possible states: either v is in an optimal cover, or its neighbors are. Indeed, the analysis of the algorithm utilizes a composite relation with $N = 2$ as appears in (1).

To evaluate the running times of our algorithms we need to analyze the asymptotic behavior of $p(\alpha k, k)$ for a fixed α as k grows to infinity. With some surprise, we did not find an existing analysis of this behavior, even for $N = 1$. The main technical contribution of this paper is Theorem 2 that gives such analysis for any $N \ge 1$. We emphasize that while the recurrence relations we want to solve are derived from coverage problems, our solution is generic and can be used for any composite recurrence.

We say that a vector $\bar{q} \in \mathbb{R}_{\ge 0}^r$ is a *distribution* if $\sum_{i=1}^r \bar{q}_i = 1$, and use $D(\cdot\|\cdot)$ to denote **Kullback-Leibler divergence** [4].[5] To state our main result we need the next definition. For short, associate the term $(\bar{b}, \bar{k}, \bar{\gamma})$ with the expression $\sum_{i=1}^r \bar{\gamma}_i \cdot p(b - \bar{b}_i, k - \bar{k}_i)$.

Definition 1: Let $\bar{b} \in \mathbb{N}_+^r$, $\bar{k} \in \mathbb{N}^r$ and $\bar{\gamma} \in \mathbb{R}_{\ge 0}^r$ with $\sum_{i=1}^r \bar{\gamma}_i = 1$. Then for $\alpha > 0$, the **α-branching number** of the term $(\bar{b}, \bar{k}, \bar{\gamma})$ is the optimal value M^* of the following minimization problem over $\bar{q} \in \mathbb{R}_{\ge 0}^r$:

$$\begin{aligned} M^* \quad &= \min \quad \frac{1}{\sum_{i=1}^r \bar{q}_i \cdot \bar{k}_i} D(\bar{q}\|\bar{\gamma}) \\ &\text{such that} \\ &\sum_{i=1}^r \bar{q}_i \cdot \bar{b}_i \le \alpha \sum_{i=1}^r \bar{q}_i \cdot \bar{k}_i \\ &\bar{q} \text{ is a distribution} \end{aligned} \tag{3}$$

If the above optimization does not have a feasible solution then $M^* = \infty$.

Our main result is the following.

Theorem 2: Let p be the composite recurrence of $\{(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j) |\ 1 \le j \le N\}$, and $\alpha > 0$. Denote by M_j the α-branching number of $(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j)$, and let $M = \max\{M_j | 1 \le j \le N\}$. If $M < \infty$ then[6]

$$\lim_{k\to\infty} \frac{\log p(\lfloor \alpha k \rfloor, k)}{k} = -M.$$

Intuitively, Theorem 2 asserts that $p(\alpha k, k) \approx \exp(-M)^k$. Furthermore, it shows that the asymptotics of $p(\alpha k, k) \approx \exp(-M)^k$ is dominated by the "worst" term in p. We note that the optimization problem (3) is *quasiconvex*. Furthermore, all of the numerical problems in this paper arising as consequences of (3) and Theorem 2 are *quasiconvex*, and as such can be solved efficiently using standard tools (these problems involve the optimization of $\bar{\gamma}^j$ as well). Moreover, most of these problems have a nearly closed form solution.

[4]Throughout the paper we use $\mathbb{N}$ (resp. $\mathbb{N}_+$) to denote the non-negative (resp. positive) integers ($\mathbb{N} = \mathbb{N}_+ \cup \{0\}$).

[5]Formally, for $\bar{c}, \bar{d} \in \mathbb{R}^k$ define $D(\bar{c}\|\bar{d}) = \sum_{i=1}^k \bar{c}_i \log \frac{\bar{c}_i}{\bar{d}_i}$.

[6]Throughout the paper we refer by $\log$ to the natural logarithm.

It is easy to show that for p as defined in (2) and every $b, k, n \in \mathbb{N}_+$ it holds that $p(nb, nk) \geq (p(b,k))^n$. This suggests that p can be lower bounded empirically by $p(\alpha k, k) = \Omega(c^k)$ where $c = (p(\alpha k_0, k_0))^{\frac{1}{k_0}}$ for any fixed k_0. Indeed, this simple approach can be used in practice to derive a fairly good lower bound for p in simple cases such as (1). However, it lacks both the scale and insight required to derive the algorithmic results presented in this paper. Furthermore, Theorem 2 readily gives the desired solution, thus eliminating the need for an empirical approach as described above.

The observation that the asymptotic behavior of $p(b,k)$ is dominated by the highest α-branching number of the terms in p served as a main guiding principle for designing the algorithms in this paper. Most notably, the 1.5-approximation for Vertex Cover was explicitly derived by this insight (see Section IV-B). In addition, Theorem 2 reduces the problem of optimizing the values of $\bar{\gamma}^j$ of the terms of p (e.g., the selection of γ in (1)) to multiple simple continuous quasiconvex optimization problems. In contrast, the empirical approach provides no tools for optimizing the distributions $\bar{\gamma}^j$. This was crucial for deriving all of our algorithmic results, in particular the results for 3-Hitting Set (see Section V) which involve multiple (computer generated) branching rules.

Due to space constraints, the detailed proof of Theorem 2 is omitted in this extended abstract. We give the proof in the full version of the paper [9]. In Section VI-A we present some of the tools and techniques used in the proof of Theorem 2, and motivate the optimization problem (3). Then, in Section VI-B, we provide a proof sketch for a weaker variant of Theorem 2.

We note that the requirement that the minimum operation in (2) is only taken over values of j such that $\bar{k}^j \leq k$ is indeed important: the statement of Theorem 2 may not hold if the requirement is removed, due to several corner cases.[7] Algorithmically, the condition $\bar{k}^j \leq k$ represents the requirement that a branching step cannot lead to a negative size minimum cover for the remaining graph. Therefore, the condition always holds.

III. Related Work

Vertex Cover is one of the fundamental problems in computer science, and a testbed for new techniques in parameterized complexity. The problem admits a polynomial time 2-approximation, which cannot be improved under the *Unique Games Conjecture (UGC)* [10]. Vertex Cover has been widely studied from the viewpoint of parameterized complexity. We say that a problem (with a particular parameter k) is *fixed-parameter tractable (FPT)* if it can be solved in time $f(k) \cdot \mathrm{poly}(n)$, where f is some computable function depending only on k. Vertex Cover parameterized by the solution size is well known to be FPT (see, e.g., [8]). The fastest running time of an FPT algorithm for the problem is $O^*(1.273^k)$ due to Chen et at. [11]. It is also known that there is no $2^{o(k)} \cdot \mathrm{poly}(n)$ algorithm for the problem, under the *exponential time hypothesis (ETH)*.

In [12] it was shown that there is no $(7/6 - \varepsilon)$ approximation for Vertex Cover with running time $O(2^{n^{1-\delta}})$ for any $\delta > 0$ under ETH. In [13] Manurangsi and Trevisan showed a $(2 - 1/O(r))$-approximation for the problem with running time $O^*(\exp(n \cdot 2^{-r^2}))$, improving upon earlier results of [14]. To the best of our knowledge, the existence of a $(2-\varepsilon)$-approximation for Vertex Cover with running time $2^{o(n)}$ is still open.

The above results suggest that for $\alpha < 7/6$ subexponential α-approximation algorithms are unlikely to exist, and even as the approximation ratio approaches 2 the barrier of exponential running time remains unbreached. This motivates our study of parameterized α-approximation algorithms for Vertex Cover, for $1 < \alpha < 2$, whose running times are exponential in the solution size, k.

Brankovic and Fernau presented in [2] a branching algorithm that yields a parameterized 1.5-approximation for Vertex Cover with running time $O^*(1.0883^k)$. In [5] Fellows et at. presented an α-approximation algorithm whose running time is $O^*(1.273^{(2-\alpha)k})$, for any $1 \leq \alpha \leq 2$. A similar result was obtained in [15] using a different technique.

Similar to Vertex Cover, 3-Hitting Set cannot be approximated within a constant factor better than 3 under UGC [10], and there is no subexponential algorithm for the problem under ETH. The best known parameterized algorithm for the problem has running time $O^*(2.076^k)$ [16]. Previous works on parameterized approximation for 3-Hitting Set resulted in an α-approximation in time $O^*(2.076^{k(3-\alpha)/2})$ due to [5], for any $1 \leq \alpha \leq 3$, and a 2-approximation in time $O^*(1.29^k)$ using a branching algorithm by Brankovic and Fernau [6].

Randomized branching is a well known approach for algorithm design (see, e.g, [17], [18], [19]). Often, the analysis of such algorithms evaluates the probability that in *every* branching step the algorithm makes a correct branching choice (in contrast, our analysis aims at bounding the number of incorrect steps). This leads to a one-variable recurrence which can be simply solved. Randomized branching has been used for approximation in [14], along with a tailored analysis for the approximation ratio.

The idea of sampling leaves from a branching tree was studied in the past from a different perspective. Specifically, it was used in [20] to justify one-sided probabilistic polynomial algorithms as a computational model for branching algorithms. Within this model, the authors derived lower bounds for branching algorithms.

Previous works on parameterized approximations for both Vertex Cover and 3-Hitting Set either considered approximative preprocessing [5] or used approximative (worsening) steps within branching algorithms [6], [2]. While these techniques use the approximative step explicitly at given stages of the algorithm execution, in randomized branching the approximative step takes the form of an incorrect branching decision, which may add unnecessary vertices to the solution. As incorrect branching is not restricted to a specific stage, a degree of

[7] Consider, for example, the redundant recurrence $p(b,k) = p(b-6, k-4)$ with $p(b,k) = 0$ for $b < 0$ and $p(b,k) = 1$ for $k \leq 0 \leq b$. In this case, $\lim_{k\to\infty} \frac{1}{k} \log p(\lfloor 1.5k \rfloor, k)$ does not exist.

Fig. 2. Algorithm VC3$_\gamma$

Input: An undirected graph G

1: **if** G has a vertex v with degree 3 or more **then**
2: Let u_1, u_2, u_3 be 3 of v's neighbors.
3: With probability γ set $S = \{v\}$ and $S = \{u_1, u_2, u_3\}$ with probability $1-\gamma$.
4: Use a recursive call to evaluate $R = \text{VC3}_\gamma(G \backslash S)$.
5: Return $R \cup S$.
6: **else** the maximal degree in G is no greater than 2
7: Find an optimal cover S of G in polynomial time.
8: Return S

freedom is added to the number of *good paths* within the branching tree. This degree of freedom in turn increases the probability of finding an approximate solution. This gives some intuition to the improved performance of our algorithms.

A. Recurrence Relations and the Method of Types

The analysis of single variable recurrence relations (e.g., $f(n) = \sum_{i=1}^{N} f(n - a_i)$) is a cornerstone in the analysis of parameterized branching algorithms that is often included in introductory textbooks on parameterized algorithms (see, e.g., [8], [1]).

In [21] Eppstein introduced a technique for computing the asymptotic behavior of multivariate recurrences of the form $f(x) = \max_i \sum_j f(x - \delta_{i,j})$, where $f : \mathbb{Z}^d \to \mathbb{Z}$ and $\delta_{i,j} \in \mathbb{N}^d$. For any $t \in \mathbb{N}^d$, the technique shows how to compute a constant c such that $f(nt) \approx c^n$ up to a polynomial factor. The technique is based on a tight reduction of the multivariate recurrence to a solvable single variable recurrence. A matching lower bound to the result of the quasiconvex program is derived using a random walk, which bears some similarity to the reduction used in this paper from a recurrence to a stochastic process. Nevertheless, the analysis in this paper is substantially different.

The result in [21] is often used in the analysis of parameterized algorithms, and specifically within the context of Measure and Conquer [22]. To the best of our knowledge, these works typically utilize solutions for recurrences, in a wise and sophisticated manner, to derive running times for algorithms, but do not deal with solving the recurrence relations themselves.

We emphasize that the recurrences considered in [21] are different from the recurrences studied is this paper. The difference seems to be more than merely technical. The recurrences in [21] commonly measure the size of a branching tree, while our recurrence relations aim at bounding the number of leaves adhering to certain property within the tree. In fact, the size of the branching trees considered in this paper can be easily evaluated using standard single variable recurrence relations. We are not aware of other works relating to the analysis of similar multivariate recurrences.

The *method of types* is a powerful technique developed mostly within the context of information theory in a line of works, starting from the early works of Sanov [3] and Hoeffding [23]. The current form of the method is attributed to the works of Csiszar et al. [24]. Along with Sanov's theorem (see Section VI-A), the prominent results derived using the method of types are universal block coding and hypothesis testing (we refer the reader to the survey in [24] and to Chapter 11 in [4]). While the method of types is considered a basic tool in information theory, it seems much less known in theoretical computer science.

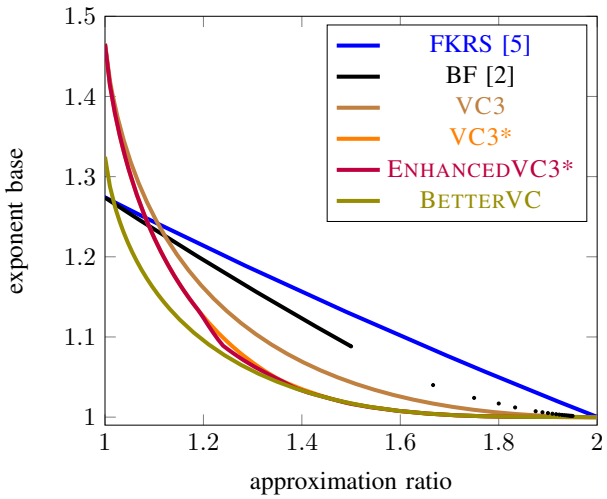


Fig. 3. Results of Section IV. A dot at (α, c) means that the respective algorithm yields an α-approximation for Vertex Cover in time $O^*(c^k)$ or $O^*\left((c+\varepsilon)^k\right)$ for any $\varepsilon > 0$.

IV. Parameterized Approximation for Vertex Cover

We start by completing the analysis of the algorithm presented in Section I. A formal description of the algorithm, VC3$_\gamma$, is given in Figure 2. While the performance of VC3$_\gamma$ can be significantly improved, as we show below, it demonstrates the main tools and concepts developed in this paper, and its analysis involves only few technical details. Interestingly, already this simple algorithm improves the previous state of the art results for a wide range of approximation ratios. Sections IV-A and IV-B present variants of VC3$_\gamma$, which perform even better. Each section introduces some new ideas. Section IV-C presents a more complicated algorithm which has a better running time for approximation ratios close to 1. The results of the algorithms presented in this section are depicted in Figure 3.

Clearly, VC3$_\gamma$ has a polynomial running time. Also, it always returns a vertex cover of the input graph G. Let $\mathcal{G}_k$ be the set of graphs with a vertex cover of size k or less. Also, let $P_\gamma(b,k)$ be the minimal probability that VC3$_\gamma$ returns a solution of size at most b, given a graph $G \in \mathcal{G}_k$. That is, $P_\gamma(b,k) = \min_{G \in \mathcal{G}_k} \Pr[\ |\text{VC3}_\gamma(G)| \leq b\ ]$. By the arguments given in Section I, it is easy to prove that $P_\gamma(b,k) \geq p_\gamma(b,k)$, where $p_\gamma(b,k)$ is defined by the following recurrence relation.

Fig. 4. Algorithm α-APPROX

Input: An undirected graph G, a parameter k, an algorithm $\mathcal{A}$ and a recurrence relation p.

1: Evaluate $r = p(\alpha k, k)$ using dynamic programming.
2: Execute $\mathcal{A}(G)$ r^{-1} times. Return the minimal cover found.

$$p_\gamma(b,k) = \min \begin{cases} \gamma p_\gamma(b-1,k-1) \\ \quad + \ (1-\gamma)p_\gamma(b-3,k) & k \geq 1 \\ \gamma p_\gamma(b-1,k) \\ \quad + \ (1-\gamma)p_\gamma(b-3,k-3) & k \geq 3 \end{cases} \tag{4}$$

$$p_\gamma(b,k) = 0 \quad \forall b < 0$$

$$p_\gamma(b,k) = 1 \quad \forall b \geq 0, k = 0$$

That is, p_γ is the composite recurrence of $\left\{(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j) |\ j = 1,2\right\}$ with $\bar{b}^1 = \bar{b}^2 = (1,3)$, $\bar{\gamma}^1 = \bar{\gamma}^2 = (\gamma, 1-\gamma)$, $\bar{k}^1 = (1,0)$ and $\bar{k}^2 = (0,3)$. Note that in this case $N = 2$, and $r_1 = r_2 = 2$ (recall that a composite recurrence is defined in Section II-B).

Hence, by repeating the execution of VC3$_\gamma$ $p_\gamma(b,k)^{-1}$ times, we have a constant probability to find a cover of size b or less, for any $G \in \mathcal{G}_k$. This is achieved by using the algorithm α-APPROX (Figure 4) taking VC3$_\gamma$ as $\mathcal{A}$ and $p = p_\gamma$. We call the resulting algorithm α-VC3$_\gamma$.

We note that if $G \in \mathcal{G}_k$ then α-VC3$_\gamma$ returns a cover of size at most αk with constant probability. Clearly, the running time of the algorithm is $O^*((p_\gamma(\alpha k, k))^{-1})$. We resort to Theorem 2 to obtain a better understanding of the running time.

For any $\alpha > 1$ and $\gamma \in (0,1)$, we can calculate the α-branching numbers $M_1^{\alpha,\gamma}, M_2^{\alpha,\gamma}$ of $(\bar{b}^1, \bar{k}^1, \bar{\gamma}^1), (\bar{b}^2, \bar{k}^2, \bar{\gamma}^2)$, respectively, by numerically solving the optimization problem (3). Let $M^{\alpha,\gamma} = \max\{M_1^{\alpha,\gamma}, M_2^{\alpha,\gamma}\}$. Then, by Theorem 2 we have $\lim_{k\to\infty} \frac{\log p_\gamma(\alpha k,k)}{k} = -M^{\alpha,\gamma}$. Thus, for any $\varepsilon > 0$ and large enough k, it holds that $\frac{\log p_\gamma(\alpha k,k)}{k} > -M^{\alpha,\gamma} - \varepsilon$, and equivalently $(p_\gamma(\alpha k,k))^{-1} < \exp(M^{\alpha,\gamma} + \varepsilon)^k$. We conclude that the running time of α-VC3$_\gamma$ is $O^*((p_\gamma(\alpha k,k))^{-1}) = O^*(\exp(M^{\alpha,\gamma} + \varepsilon)^k)$ for any $\varepsilon > 0$.

For any $\alpha > 1$, we can numerically find the value of γ for which $M^{\alpha,\gamma}$ is minimal. Let γ_α be this value. Then, for any $\alpha > 1$ algorithm α-VC3$_\gamma$ is a parameterized random α-approximation for Vertex Cover with running time $O^*(\exp(M^{\alpha,\gamma_\alpha} + \varepsilon)^k)$, for any $\varepsilon > 0$. For example, taking $\alpha = 1.5$ we get that α-VC3$_\gamma$ has a running time $O^*(1.043642^k)$. In Figure 3 the value of $\exp(M^{\alpha,\gamma_\alpha})$ is presented as a function of α. An overview of the standard methods used for the numerical optimizations is given in the full version of the paper [9].

A. A Refined Analysis of Incorrect Branching

Standard branching algorithms derive several simpler sub-instances from a given instance with a guarantee that an optimal solution to one (specific yet unknown) of the sub-instances leads to an optimal solution. Therefore, the analysis is focused on this specific sub-instance and ignores the effect of other sub-instances on the optimum. This is not the case when using randomized branching for *approximation*, where the reduction in the minimal cover size by an incorrect branching can lead to an improved running time, as we demonstrate below.

Fig. 5. Algorithm VC3*$_{\gamma_3,\gamma_4,\ldots,\gamma_\Delta}$

Input: An undirected graph G

1: **if** G has a vertex v with degree $d \geq 3$ **then**
2: If $d < \Delta$ let $U = N(v)$, otherwise let U be a subset of $N(v)$ of size exactly Δ.
3: With probability γ_d set $S = \{v\}$ and $S = U$ with probability $1 - \gamma_d$.
4: Use a recursive call to evaluate $R =$ VC3*$_{\gamma_3,\gamma_4,\ldots,\gamma_\Delta}(G \setminus S)$, and return $R \cup S$
5: **else** the maximal degree in G is 2
6: Find an optimal cover S of G in polynomial time and return S.

Consider the following observation. If v is a vertex of degree exactly 3 and the algorithm (e.g., VC3$_\gamma$) selects its three neighbors $\{u_1, u_2, u_3\}$ to the cover, then even if none of $\{u_1, u_2, u_3\}$ belongs to an optimal cover, the size of the optimal cover decreases by one (as v is a part of an optimal cover, but is no more required). This observation can be extended to any fixed degree of v.

Algorithm VC3*$_{\gamma_3,\gamma_4,\ldots,\gamma_\Delta}$ (Figure 5) takes advantage of this property by using a different probability for selecting v or its neighbors depending on its degree, as well as by selecting all the neighbors of v in case the degree of v is smaller than Δ, for some fixed $\Delta \in \mathbb{N}$.

Clearly, VC3*$_{\gamma_3,\gamma_4,\ldots,\gamma_\Delta}$ is polynomial and always returns a cover of G. Similar to VC3$_\gamma$, it can be shown that the probability VC3*$_{\gamma_3,\gamma_4,\ldots,\gamma_\Delta}$ returns a solution of size b, given a graph $G \in \mathcal{G}_k$, is at least $p(b,k)$, where p is given by

$$p(b,k) = \min \begin{cases} \gamma_d \cdot p(b-1,k-1) \\ \quad + \ (1-\gamma_d) \cdot p(b-d,k-1) & 3 \leq d < \Delta \\ \gamma_d \cdot p(b-1,k) \\ \quad + \ (1-\gamma_d) \cdot p(b-d,k-d) & 3 \leq d < \Delta \\ & k \geq d \\ \gamma_\Delta \cdot p(b-1,k-1) \\ \quad + \ (1-\gamma_\Delta) \cdot p(b-\Delta,k) \\ \gamma_\Delta \cdot p(b-1,k) \\ \quad + \ (1-\gamma_\Delta) \cdot p(b-\Delta,k-\Delta) & k \geq \Delta \end{cases} \tag{5}$$

with $p(b,k) = 0$ for $b < 0$ and $p(b,k) = 1$ for $b \geq 0$ and $k = 0$. Clearly, p is a composite recurrence relation of the

$N = 2(\Delta - 2)$ triplets

$$\{\ ((1,d),(1,1),(\gamma_d, 1-\gamma_d))\mid 3 \le d < \Delta\ \}\ \cup$$
$$\{\ ((1,d),(0,d),(\gamma_d, 1-\gamma_d))\mid 3 \le d < \Delta\ \}\ \cup$$
$$\{\ ((1,\Delta),(1,0),(\gamma_\Delta, 1-\gamma_\Delta))\ \}$$
$$\{\ ((1,\Delta),(0,\Delta),(\gamma_\Delta, 1-\gamma_\Delta))\ \}.$$

And as before, we can derive an approximation algorithm by using α-APPROX with Algorithm VC3*$_{\gamma_3,\gamma_4,\ldots,\gamma_\Delta}$ as $\mathcal{A}$ and p as defined in (5). Let α-VC3* be this algorithm. Clearly, α-VC3* is a random parameterized α-approximations algorithm for Vertex Cover.

Arbitrarily, we select $\Delta = 100$. As before, for every $1 < \alpha < 2$ and $1 \le d < \Delta$ we can find the value $\gamma_{\alpha,d}$ such that the maximal α-branching number of $((1,d),(1,1),(\gamma_{\alpha,d}, 1-\gamma_{\alpha,d}))$ and $((1,d),(0,d),(\gamma_{\alpha,d}, 1-\gamma_{\alpha,d}))$ is minimal. Let $M_{\alpha,d}$ be this value. Also, we can find the value $\gamma_{\alpha,\Delta}$ such that the maximal α-branching number of $((1,\Delta),(1,0),(\gamma_{\alpha,\Delta}, 1-\gamma_{\alpha,\Delta}))$ and $((1,\Delta),(0,\Delta),(\gamma_{\alpha,\Delta}, 1-\gamma_{\alpha,\Delta}))$ is minimal; let $M_{\alpha,\Delta}$ be this value. Let M_α be the maximal branching number of these triplets for a given value of α ($M_\alpha = \max_{1\le d\le\Delta} M_{\alpha,d}$). Then, by Theorem 2, for any $\varepsilon > 0$ and large enough k, it holds that $p(\alpha k, k) \ge \exp(-M_\alpha - \varepsilon)$, and therefore the running time of α-VC3* is $O^*\left(\exp(M_\alpha+\varepsilon)^k\right)$. For $\alpha = 1.5$ the running time is $O^*(1.0172^k)$. Figure 3 shows $\exp(M_\alpha)$ as a function of α.

B. Further Insights from using α-Branching Numbers

In the context of classic branching algorithms, the running time of an algorithm is dominated by the highest branching number of the branching rules used by the algorithm (see, e.g., [8], [1]). This observation is commonly used in the design of (exact) branching algorithms. Theorem 2 asserts that essentially the same holds for parameterized approximation using randomized branching. In the following we show how it can be used to improve the running time of α-VC3*. With some surprise, we were unable to find a similar result referring to the recurrence relations in [21].

Consider algorithm α-VC3* of Section IV-A, whose time complexity is the inverse of the function in (5). As an example, for $\alpha = 1.5$ we can sort the values $M_{\alpha,d}$ to understand which value of d dominates the running time. The four highest values are given in the table below.

d	5	6	4	7
$\exp(M_{1.5,d})$	1.0171	1.0165	1.0164	1.0156

The values in the table suggest that avoiding branching over degree 5 vertices leads to an $O^*(1.0165^k)$ algorithm. In fact, tools to do so have already been used in previous works, such as [25]. The basic idea is that as long as there is a vertex v in the graph of degree different than 5 the algorithm branches on it. If all vertices in the graph are of degree 5, the algorithm has to perform a branching on a degree 5 vertex; however, such event cannot happen more than once along a branching path. Therefore, the algorithm can use non-randomized branching in this case while maintaining a polynomial running time.

Fig. 6. Algorithm ENHANCEDVC3*

Input: An undirected graph $G = (V, E)$
Configuration Parameters: The algorithm depends on several parameters that should be configured. These include $\Delta \in \mathbb{N}$, $\delta \in \mathbb{N}$, $2 \le \delta < \Delta$, and $\gamma_2, \ldots, \gamma_{\delta-1}, \gamma_{\delta+1}, \ldots, \gamma_\Delta \in (0,1)$.

1: If the empty set is a cover of G return $\emptyset$.
2: **if** G is not connected **then**
3: Let C be a component of G. Return ENHANCEDVC3*$(C) \cup$ ENHANCEDVC3*$(G - C)$.
4: If G has a vertex v of degree 1, let u be its neighbor. Return ENHANCEDVC3*$(G \setminus \{u\}) \cup \{u\}$.
5: **if** G has a vertex v of degree $d \ne \delta$ **then**
6: Let $U = N(v)$ if $d < \Delta$ and $U \subseteq N(v)$ with $|U| = \Delta$ otherwise.
7: Let $S = \{v\}$ with probability γ_d and $S = U$ otherwise.
8: Return ENHANCEDVC3*$(G \setminus S) \cup S$
9: If G is a regular graph (of degree δ), select an arbitrary edge $(v_1, v_2) \in E$. Evaluate $S_1 =$ ENHANCEDVC3*$(G \setminus v_1) \cup v_1$ and $S_2 =$ ENHANCEDVC3*$(G \setminus v_2) \cup v_2$. Return the smaller set between S_1 and S_2.

Consider the algorithm ENHANCEDVC3* given in Figure 6. It can be shown that its running time is polynomial, and the probability that the algorithm returns a solution of size b, given $G \in \mathcal{G}_k$, is at least

$$p(b,k) = \min \begin{cases} \gamma_d \cdot p(b-1,k-1) \\ \quad + \ (1-\gamma_d)\cdot p(b-d,k-1) & 2 \le d < \Delta,\ d \ne \delta \\ \gamma_d \cdot p(b-1,k) \\ \quad + \ (1-\gamma_d)\cdot p(b-d,k-d) & 2 \le d < \Delta,\ d \ne \delta, k \ge d \\ \gamma_\Delta \cdot p(b-1,k-1) \\ \quad + \ (1-\gamma_\Delta)\cdot p(b-\Delta,k) \\ \gamma_\Delta \cdot p(b-1,k) \\ \quad + \ (1-\gamma_\Delta)\cdot p(b-\Delta,k-\Delta) & k \ge d \end{cases} \quad (6)$$

As before, we use the lower bound derived from the recurrence relation to obtain a random parameterized α-approximation algorithm with running time $O^*\left(\frac{1}{p(\alpha k,k)}\right)$, by using Algorithm α-APPROX with ENHANCEDVC3* as $\mathcal{A}$ and the recurrence relation p as given in (6). Let α-ENHANCEDVC3* be this algorithm.

For any $1 < \alpha < 2$ and $2 \le d \le \Delta$ we can find the value $M_{\alpha,d}$ as in Section IV-A. If $\delta' = \arg\max_{2\le d\le N} M_{\alpha,d} \ne \Delta$

TABLE II
RUNNING TIMES OF α-ENHANCEDVC3* AND α-VC3*.

α	1.2	1.3	1.4	1.5	1.6	1.7
α-VC3*	1.12548^k	1.06804^k	1.03501^k	1.01713^k	1.00754^k	1.00280^k
α-ENHANCEDVC3*	1.12386^k	1.06420^k	1.03320^k	1.01657^k	1.00751^k	1.00277^k

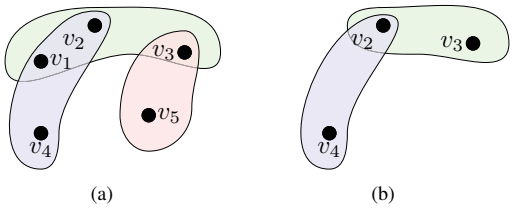


Fig. 7. An example of a neighbors graph. A hypergraph H is illustrated in 7(a). The neighbors graph of v_1, $\mathsf{NG}(v_1)$, is given in 7(b).

we can set $\delta = \delta'$; therefore, the running time of α-ENHANCEDVC3* is $O^*(\exp(M_\alpha + \varepsilon)^k)$ when M is the *second* largest number of $M_{\alpha,2}, \ldots, M_{\alpha,\Delta-1}$ (or $M_{\alpha,\Delta}$ if $\delta' = \Delta$). The value of $\exp(M_\alpha)$ as a function of α is shown in Figure 3. For $\alpha = 1.5$ the running time of the algorithm is $O^*(1.01657^k)$. This is the best running time for the given approximation ratio presented in this paper. Table II gives a comparison of the running times of α-ENHANCEDVC3* and α-VC3* for various values of α.

C. Advanced Randomized Branching for Vertex Cover

The techniques presented for parameterized approximations can be incorporated in conjunction with more sophisticated parameterized (exact) algorithm for Vertex Cover. This commonly results in faster algorithms for approximation ratios close to 1.

In the full version of this paper [9] we give a parameterized approximation algorithm, BETTERVC for Vertex Cover building on the exact $O^*(1.33^k)$ algorithm presented in [8]. To design BETTERVC, the branching rules of the exact algorithm are replaced with randomized branching. The analysis of the running time is similar to the analysis presented in the previous sections and requires a careful accounting of the decrease in the minimal Vertex Cover size in case of incorrect branching. We refer the reader to [9] for additional details.

V. PARAMETERIZED APPROXIMATION FOR 3-HITTING SET

In this section we present a parameterized approximation algorithm for 3-Hitting Set. The algorithm draws some ideas from VC3* (see Section IV-A), which relies on two basic observations. The first is that for any vertex v of a graph G and a vertex cover S, either $v \in S$ or $N(v) \subseteq S$. The second observation is that, even if v is in a minimum vertex cover, removing $N(v)$ from the graph decreases the size of a minimum cover at least by one.

Consider the following analog of the above statement for 3-Hitting Set. Given a 3-hypergraph $H = (V, E)$, for any $v \in V$ define the *neighbors graph* of v as the hypergraph $\mathsf{NG}(v) = (V_v, E_v)$ with $V_v = \{u | \exists e \in E : u, v \in e\}$ and $E_v = \{e \setminus \{v\} | \ e \in E, v \in e\}$ (see an example in Figure 7). Clearly, for every $e \in \mathsf{NG}(v)$ it holds that $|e| \leq 2$ (the neighbors graph is essentially a standard undirected graph with the addition of single node edges). Similar to the case of Vertex Cover, for any $v \in V$ and a hitting set S of H, either $v \in S$ or there is a minimal hitting set T of $\mathsf{NG}(v)$ such that $T \subseteq S$.[8] We note that if v belongs to a minimum hitting set of H then removing a minimal hitting set of $\mathsf{NG}(v)$ from H decreases the minimum hitting set size at least by 1.

Let $v \in V$ such that $\{v\} \notin E$, then the neighbors graph of v admits a specific structure. It has up to $2 \cdot \deg(v)$ vertices, exactly $\deg(v)$ edges (there may be edges with a single vertex) and no isolated vertices. Therefore, the number of possible graphs $\mathsf{NG}(v)$ for vertices v of bounded degree is finite up to isomorphism.

For some fixed $\Delta \in \mathbb{N}$, we construct a finite set $\mathcal{G}_\Delta$ of hypergraphs, such that $\mathsf{NG}(v)$ is isomorphic to a hypergraph in $\mathcal{G}_\Delta$ for any v with $\deg(v) \leq \Delta$. Also, for every $G \in \mathcal{G}_\Delta$, let $C_1^G, \ldots, C_{m^G}^G$ be all the minimal hitting sets of G. Clearly, the set $\{C_i^G | \ G \in \mathcal{G}_\Delta, 1 \leq i \leq m^G\}$ has a finite cardinality.

We need one more technical definition before introducing our algorithm. Given a 3-hypergraph $H = (V, E)$, a vertex $v \in V$ and $F \subseteq E$ such that $v \in e$ for any $e \in F$, define the *induced graph* of v and F as the hypergraph $\mathsf{Ind}(v, F) = (V_{v,F}, E_{v,F})$ with $V_{v,F} = \{u | \ \exists e \in F : u \in e \setminus \{v\}\}$ and $E_{v,F} = \{e \setminus \{v\} | \ e \in F\}$. By definition, it also holds that the cardinality of edges in $\mathsf{Ind}(v, F)$ is at most 2 and $\mathsf{Ind}(v, F)$ has no isolated vertices. It follows that $\mathsf{NG}(v) = \mathsf{Ind}(v, \{e \in E | \ v \in e\})$. Our algorithm uses induced graphs to handle vertices of degree larger than Δ. Similar to the neighbors graph, the induced graph $\mathsf{Ind}(v, F)$ satisfies the following. Let S be a hiting set of the hypergraph H, then either $v \in S$ or there is a hitting set T of $\mathsf{Ind}(v, F)$ such that $T \subseteq S$.

The above observations are used to derive algorithm 3HS, given in Figure 8. It is easy to see that the algorithm always returns a hitting set of the input hypergraph H. Also, the size of H strictly decreases between recursive calls, and the processing time of each recursive call is polynomial. Therefore, the algorithm has polynomial running time (note that since Δ is a fixed constant, finding a graph G isomorphic to N takes constant time). It is also easy to verify the algorithm always finds a hypergraph $G \in \mathcal{G}_\Delta$ isomorphic to N in Line 4.

Let $\|G\|$ be the number of edges in G. We set $1_{\|G\|<\Delta} = 1$ if $\|G\| < \Delta$ and $1_{\|G\|<\Delta} = 0$ otherwise. Consider the recurrence

[8] T is a *minimal* hitting set of a hypergraph H if T is a hitting set, and no strict subset $T' \subsetneq T$ is also a hitting set of H.

Fig. 8. Algorithm 3HS

Input: A 3-hypergraph $H = (V, E)$
Configuration Parameters: $\bar{\gamma}^G \in \mathbb{R}_{\geq 0}^{m^G+1}$ with $\sum_{i=1}^{m^G+1} \bar{\gamma}_i^G = 1$ for any $G \in \mathcal{G}_\Delta$.
Notation: Define $H \setminus U = (V', E')$ with $V' = V \setminus U$ and $E' = \{e \in E |\ e \cap U = \emptyset\}$

1: If the empty set is a hitting set of H return $\emptyset$.
2: If there is $\{v\} \in E$ then return $3\text{HS}(H \setminus \{v\}) \cup \{v\}$.
3: Pick an arbitrary vertex v. If $\deg(v) \leq \Delta$ set $N = \mathsf{NG}(v)$. Otherwise, set $N = \mathsf{Ind}(v, F)$ with an arbitrary set $F \subseteq E$ of Δ edges such that $\forall e \in F : v \in e$.
4: Find a hypergraph $G \in \mathcal{G}_\Delta$ such that N and G are isomorphic. Let φ be the vertex isomorphism function from G to N.
5: Select $S = \{v\}$ with probability $\bar{\gamma}_{m^G+1}^G$ and $S = \varphi(C_i^G)$ with probability $\bar{\gamma}_i^G$ for $1 \leq i \leq m^G$.
6: Return $3\text{HS}(H \setminus S) \cup S$.

relation p defined by

$$p(b,k) = \min \begin{cases} \bar{\gamma}_{m^G+1}^G \cdot p\left(b-1, k\right) + \\ \quad + \sum_{i=1}^{m^G} \bar{\gamma}_i^G \cdot p\left(b - |C_i^G|, k - |C_i^G \cap C_j^G|\right) \\ \quad\quad \forall G \in \mathcal{G}_\Delta, 1 \leq j \leq m^G : |C_j^G| \leq k \\ \\ \bar{\gamma}_{m^G+1}^G \cdot p\left(b-1, k-1\right) + \\ \quad + \sum_{i=1}^{m^G} \bar{\gamma}_i^G \cdot p\left(b - |C_i^G|, k - 1_{\|G\|<\Delta}\right) \\ \quad\quad \forall G \in \mathcal{G}_\Delta, 1 \leq j \leq m^G \\ \\ p(b-1, k-1) \end{cases} \tag{7}$$

along with $p(b,k) = 0$ for $b < 0$, and $p(b,0) = 1$ for $b \geq 0$. Let $P(b,k)$ be the minimal probability that algorithm 3HS returns a hitting set of size b or less, given the 3-hypergraph H which has a hitting set of size k or less. Then it can be easily shown that for every $b \in \mathbb{Z}$ and $k \in \mathbb{N}$, $P(b,k) \geq p(b,k)$.

Accordingly, an α-approximation algorithm for 3-Hitting Set can be derived by the same approach used for Vertex Cover. This leads to algorithm α-3HS given in Figure 9.

It follows that algorithm α-3HS yields an α-approximation for 3-Hitting Set with running time $\frac{1}{p(\alpha k, k)}$. For any value of α, it is possible to optimize the value of $\bar{\gamma}^G$ for each $G \in \mathcal{G}_\Delta$ and evaluate the asymptotic behavior of $p(\alpha k, k)$ as k goes to infinity using Theorem 2.

However, the size of $\mathcal{G}_\Delta$ grows rapidly as Δ increases, rendering the above computation less and less practical. With a little technical sophistication we were able to evaluate the running time of the algorithm with $\Delta = 7$ for various approximation ratios. Figure 1(b) shows the running times of the algorithm with $\Delta = 7$ as function of α. A list of running times for several approximation ratios is given in Table III. For

Fig. 9. Algorithm α-3HS

Input: A 3-hypergraph H, a parameter k

1: Evaluate $r = p(\alpha k, k)$ using dynamic programming (p is defined in (7)).
2: **loop** r^{-1} times
3: Execute $3\text{HS}(H)$.
4: Return the minimal hitting set found.

$\alpha = 2$ the running time is $O^*(1.0659^k)$, yielding a significant improvement over the previous best result of $O^*(1.29^k)$ due to [6].

VI. Recurrence Relations

In the following we introduce the main tools and ideas used in the proof of Theorem 2. We first outline (in Section VI-A) the tools and techniques. We then present (in Section VI-B) a proof sketch for a variant of Theorem 2.

A. Tools and Techniques

We start by studying "simpler" recurrence relations, which help us motivate the formula for an α-branching number, as given in (3). Given $(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j)$ for $1 \leq j \leq N$ as in Theorem 2, define N "simpler" functions $p_j : \mathbb{Z} \times \mathbb{Z} \to [0,1]$ for $1 \leq j \leq N$, satisfying

$$p_j(b,k) = \sum_{i=1}^{r_j} \bar{\gamma}_i^j \cdot p_j(b - \bar{b}_i^j, k - \bar{k}_i^j), \tag{8}$$

along with $p_j(b,k) = 1$ for any $b \geq 0$ and $k \leq 0$, and $p_j(b,k) = 0$ for $b < 0$. One can easily give a probabilistic interpretation for p_j. For $1 \leq j \leq N$, let $(Y_n^j)_{n=1}^\infty$ be a series of *i.i.d.* random variables such that $\Pr(Y_n^j = i) = \bar{\gamma}_i^j$ for any $1 \leq i \leq r_j$. It is easy to show that

$$p_j(b,k) = \Pr\left(\exists n : \sum_{\ell=1}^{n} \bar{b}_{Y_\ell^j}^j \leq b \text{ and } \sum_{\ell=1}^{n} \bar{k}_{Y_\ell^j}^j \geq k\right). \tag{9}$$

In the context of Vertex Cover, we can interpret $Y_n^j = i$ as the event: "In the nth step of the algorithm, $\bar{b}_i^j$ vertices were added to the cover and reduced the minimal vertex cover by $\bar{k}_i^j$". With this interpretation, $p_j(b,k)$ is the probability that at some point in the algorithm execution the size of the minimal vertex cover has decreased by k while at most b vertices were added to the cover.

We utilize the *method of types* for analyzing the asymptotic behavior of p_j. The *type* $\mathcal{T}(a_1, \ldots, a_n)$ of $(a_1, \ldots, a_n) \in \{1, \ldots, r_j\}^n$ is the relative frequency of each $i \in \{1, \ldots, r_j\}$ in $(a_1, \ldots, a_n)$; that is, $\mathcal{T}(a_1, \ldots, a_n) = T \in \mathbb{R}_{\geq 0}^{r_j}$, where $T_i = \frac{|\{\ell |\ a_\ell = i\}|}{n}$. It follows that the type of a sequence is a distribution.

One of the important results attributed to the method of types is Sanov's theorem [3]. Given a set Q of distributions,

i.e., $Q \subseteq \{\bar{q} \in \mathbb{R}_{\geq 0}^{r_j} |\ \bar{q} \text{ is a distribution}\}$, the theorem states (under a few technical conditions omitted here) that

$$\lim_{n\to\infty} \frac{1}{n} \log \Pr\left(\mathcal{T}(Y_1^j, \ldots, Y_n^j) \in Q\right) = -\min_{\bar{q}\in Q} D\left(\bar{q}\|\bar{\gamma}^j\right). \tag{10}$$

Informally, equation (10) implies that the probability the type of a random sequence is in Q is dominated by the distribution in Q closest to the distribution $\bar{\gamma}^j$ from which the sequence is drawn, with the distance measured by $D(\cdot\|\cdot)$, the Kullback-Leibler divergence.

For $\beta > 0$, let Q_β^j be the set of all distributions $\bar{q}$ such that, if a random variable X is sampled according to $\bar{q}$ (i.e., $\Pr(X = i) = \bar{q}_i$), then it holds that $E[\bar{k}_X^j] \geq \beta$ and $E[\bar{b}_X^j] \leq \alpha\beta$. Formally,

$$Q_\beta^j = \left\{ \bar{q} \;\middle|\; \begin{array}{c} \sum_{i=1}^{r_j} \bar{q}_i \bar{b}_i^j \leq \alpha\beta, \\ \sum_{i=1}^{r_j} \bar{q}_i \bar{k}_i^j \geq \beta, \\ \bar{q} \text{ is a distribution} \end{array} \right\}.$$

Then, $\mathcal{T}(a_1, \ldots, a_n) \in Q_{\frac{k}{n}}^j$ *iff* $\sum_{\ell=1}^n \bar{k}_{a_\ell}^j \geq \frac{k}{n} \cdot n = k$ and $\sum_{\ell=1}^n \bar{b}_{a_\ell}^j \leq \alpha\frac{k}{n} \cdot n = \alpha k$. The sets Q_β^j can be used to write the event in (9) in terms of types.

$$\begin{aligned} p_j(\alpha k, k) &= \Pr\left(\exists n : \sum_{\ell=1}^n \bar{b}_{Y_\ell^j}^j \leq b, \sum_{\ell=1}^n \bar{k}_{Y_\ell^j}^j \geq k\right) \\ &= \Pr\left(\exists n : \mathcal{T}(Y_1^j, \ldots, Y_n^j) \in Q_{\frac{k}{n}}^j\right) \end{aligned}$$

It follows from Sanov's theorem that

$$\begin{aligned} &\Pr\left(\mathcal{T}(Y_1^j, \ldots, Y_{\frac{k}{\beta}}^j) \in Q_\beta^j\right) \\ &\quad\approx \exp\left(-\frac{k}{\beta} \cdot \min_{\bar{q}\in Q_\beta^j} D\left(\bar{q}\|\bar{\gamma}^j\right)\right). \end{aligned}$$

Let $\beta^{j,*}, \bar{q}^{j,*} = \arg\min_{\beta,\bar{q}\in Q_\beta^j} \frac{1}{\beta} D\left(\bar{q}\|\bar{\gamma}^j\right)$. We can then lower bound $p_j(\alpha k, k)$ by focusing on sequences of length $n = \frac{k}{\beta^{j,*}}$ (we ignore integrality issues in this informal overview).

$$\begin{aligned} p_j(\alpha k, k) &= \Pr\left(\exists n : \mathcal{T}(Y_1^j, \ldots, Y_n^j) \in Q_{\frac{k}{n}}^j\right) \\ &\geq \Pr\left(\mathcal{T}(Y_1^j, \ldots, Y^j_{\frac{k}{\beta^{j,*}}}) \in Q_{\beta^{j,*}}^j\right) \\ &\approx \exp\left(-k\frac{1}{\beta^{j,*}} D\left(\bar{q}^{j,*}\|\bar{\gamma}^j\right)\right). \end{aligned}$$

It is easy to show that we can limit in (9) the values of n such that $c^j \cdot b \leq n \leq b$, where c^j is a constant (in fact, $c^j = \left(\max_{1\leq i\leq r_j} \bar{b}_i^j\right)^{-1}$). This can be used to establish a matching upper bound.

$$\begin{aligned} p_j(\alpha k, k) &= \Pr\left(\exists n : \mathcal{T}(Y_1^j, \ldots, Y_n^j) \in Q_{\frac{k}{n}}^j\right) \\ &= \Pr\left(\exists c^j \alpha k \leq n \leq \alpha k : \mathcal{T}(Y_1^j, \ldots, Y_n^j) \in Q_{\frac{k}{n}}^j\right) \\ &\leq \sum_{n=c^j\alpha k}^{\alpha k} \Pr\left(\mathcal{T}(Y_1^j, \ldots, Y_n^j) \in Q_{\frac{k}{n}}^j\right) \\ &\approx \sum_{n=c^j\alpha k}^{\alpha k} \exp\left(-k \cdot \frac{1}{\left(\frac{k}{n}\right)} \cdot \min_{\bar{q}\in Q_{\frac{k}{n}}^j} D\left(\bar{q}\|\bar{\gamma}^j\right)\right) \\ &\leq \sum_{n=c^j\alpha k}^{\alpha k} \exp\left(-k\frac{1}{\beta^{j,*}} D\left(\bar{q}^{j,*}\|\bar{\gamma}^j\right)\right) \\ &= k \cdot \alpha(1-c^j) \exp\left(-k\frac{1}{\beta^{j,*}} D\left(\bar{q}^{j,*}\|\bar{\gamma}^j\right)\right). \end{aligned}$$

Hence, we can expect that $\lim_{k\to\infty} \frac{1}{k} \log p_j(\alpha k, k) = \frac{1}{\beta^{j,*}} D\left(\bar{q}^{j,*}\|\bar{\gamma}^j\right)$. It can be easily shown that $\beta^{j,*} = \sum_{i=1}^{r_j} \bar{q}_i^{j,*} \bar{k}_i^j$ (otherwise the values are not optimal, contradicting their definition). Thus, we conclude that $\frac{1}{\beta^{j,*}} D\left(\bar{q}^{j,*}\|\bar{\gamma}^j\right) = M_j$, where M_j is the α-branching number of $(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j)$, as given in Definition 1.

Using the ideas in the above discussion, it can be shown that for any $\varepsilon > 0$,

$$\begin{aligned} &\liminf_{k\to\infty} \frac{1}{k} \log p_j((\alpha+\varepsilon)k, k) \geq -M_j \\ &\text{and} \\ &\limsup_{k\to\infty} \frac{1}{k} \log p_j(\alpha k, k) \leq -M_j. \end{aligned} \tag{11}$$

While (11) provides a clear indication for the asymptotic behavior of $p_j(\alpha k, k)$, it only makes a little progress in understanding the asymptotics of $p(\alpha k, k)$. By definition, it is expected that $p \leq p_j$ for $1 \leq j \leq N$. Thus, following (11) we expect that $\limsup_{k\to\infty} \frac{1}{k} \log p(\alpha k, k) \leq -M$, where $M = \max_{1\leq j\leq N} M_j$. Our main result, stated in Theorem 2, asserts that the asymptotic behavior of $\frac{1}{k} \log p(\alpha k, k)$ can also be *lower bounded* by $-M$.

B. Proof Sketch for Theorem 2

Similar to the above sketch of analysis for p_j, the proof of Theorem 2 uses a probabilistic analysis based on the method of types. The stochastic process $(X_n)_{n=1}^\infty$ used in the proof is one in which X_n is drawn according to one of the distributions $\bar{\gamma}^j$, where j is selected by an (arbitrary) function of $X_1, \ldots, X_{n-1}$. While the method of types is mostly used in the context of *i.i.d.* random variables, we show that some

TABLE III
RESULTS FOR 3-HITTING SET

α	1.2	1.4	1.6	1.8	2.0	2.2	2.4	2.6	2.8
$\alpha - HS$	1.59^k	1.29^k	1.18^k	1.11^k	1.0659^k	1.039^k	1.021^k	1.0085^k	1.0026^k

of the basic properties of types can be adapted to the process $(X_n)_{n=1}^{\infty}$. These properties are used for proving Theorem 2. While the result, formally stated as Lemma 7, is weaker than Theorem 2, it suffices for deriving all the algorithmic results shown in this paper.

Let $\alpha > 0$, $N \in \mathbb{N}_+$ and $r_j \in \mathbb{N}_+$ for $1 \leq j \leq N$. Also, let $\bar{b}^j \in \mathbb{N}_+^{r_j}$, $\bar{k}^j \in \mathbb{N}^{r_j}$ and $\bar{\gamma}^j \in \mathbb{R}_+^{r_j}$ for $1 \leq j \leq N$. We assume that $\bar{\gamma}^j$ is a distribution for any j, and there is $1 \leq j \leq N$ such that $\bar{k}^j \leq 1$. Let p be the composite recurrence of $\{(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j) |\ 1 \leq j \leq N\}$ as defined in (2). Finally, let M_j be the α-branching number of $(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j)$, and $M = \max_{1 \leq j \leq N} M_j$. We assume that $M < \infty$.

We start with a few technical definitions. Let B (K, Γ) be the result of concatenating the vectors $\bar{b}^1, \bar{b}^2, \ldots, \bar{b}^N$ ($\bar{k}^1, \bar{k}^2, \ldots, \bar{k}^N$ and $\bar{\gamma}^1, \bar{\gamma}^2, \ldots, \bar{\gamma}^N$, respectively). Formally, set $s_j = \sum_{k=1}^{j} r_k$ (therefore, $s_0 = 0$) and $r = s_N$. For any $1 \leq j \leq N$ and $1 \leq i \leq r_j$ set $B_{s_{j-1}+i} = \bar{b}_i^j$, $K_{s_{j-1}+i} = \bar{k}_i^j$ and $\Gamma_{s_{j-1}+i} = \bar{\gamma}_i^j$. Also, define $\mathcal{X}_j = \{i \in \mathbb{N} |\ s_{j-1} < i \leq s_j\}$ and $\mathcal{X} = \bigcup_{j=1}^{N} \mathcal{X}_j = \{1, 2, \ldots, r\}$.

We say that $\Upsilon \in \mathbb{R}_{\geq 0}^r$ is an **extended distribution** if for any $1 \leq j \leq N$ it holds that $\sum_{i \in \mathcal{X}_j} \Upsilon_i = 1$; for example, Γ is an extended distribution. A **rules mapping** is a function $R : \mathcal{X}^* \to \{1, 2, \ldots, N\}$.[9]

Given an extended distribution Υ and a rules mapping R, we define a stochastic process $(X_n)_{n=1}^{\infty}$, where $X_n \in \mathcal{X}$ for any $n \in \mathbb{N}$. For $1 \leq j \leq N$, let $(Y_n^j)_{n=1}^{\infty}$ be *i.i.d.* random variables where $\Pr(Y_n^j = i) = \Upsilon_i$ for $i \in \mathcal{X}_j$.[10] We set $X_n = Y_n^j$ where $j = R(X_1, \ldots, X_{n-1})$. We use $\Pr_{R,\Upsilon}$ to denote the probability distribution function (formally, this is a probability measure) of the process defined by R and Υ. In particular, we use this distribution function with respect to two specific extended distributions: Γ and Q. We use Υ for showing generic results which apply to both Γ and Q.

Define $\kappa(a_1, \ldots, a_n) = \sum_{\ell=1}^{n} K_{a_\ell}$ and $\mu(a_1, \ldots, a_n) = \sum_{\ell=1}^{n} B_{a_\ell}$ to be the coverage and cost of $(a_1, \ldots, a_n) \in \mathcal{X}^n$.

For $b, k \in \mathbb{R}_{\geq 0}$ define the event

$$S^{b,k} = \left\{ \exists n \in \mathbb{N} : \begin{array}{l} \mu(X_1, \ldots, X_n) \leq b \text{ and} \\ \kappa(X_1, \ldots, X_n) \geq k \end{array} \right\} \tag{12}$$

Finally, we say that a rules mapping R is k**-consistent** for $k \in \mathbb{N}$ if for any $\bar{a} \in \mathcal{X}^*$ such that $\kappa(\bar{a}) < k$ it holds that $K_i \leq k - \kappa(\bar{a})$ for any $i \in \mathcal{X}_{R(\bar{a})}$ (equivalently, $\bar{k}^{R(\bar{a})} \leq k - \kappa(\bar{a})$). The notion of k-consistent mapping corresponds to the requirement that the minimum in (2) is only taken over j such that $\bar{k}^j \leq k$. Let $\mathcal{R}$ be the set of all mappings and $\mathcal{R}(k)$ be the set of all k-consistent mappings.

The next lemma shows a strong connection between the process $(X_n)_{n=1}^{\infty}$ and p.

Lemma 3: For any $b \in \mathbb{Z}$ and $k \in \mathbb{N}$ it holds that

$$p(b, k) = \inf_{R \in \mathcal{R}(k)} \Pr_{R,\Gamma}\left(S^{b,k}\right).$$

[9]For a set A we use A^* to denote all vectors of elements in A. That is, $A = \bigcup_{n=0}^{\infty} A^n$. We use ϵ to denote the vector of dimension 0 ($\epsilon \in A^0$).

[10]Note that the definition of Y_n^j in this section is slightly different from the definition used in Section VI-A.

While the proof of the lemma is fairly straightforward, it requires multiple technical steps. By Lemma 3, in order to evaluate the asymptotic behavior of $p(\alpha k, k)$, we can focus on the asymptotics of $\inf_{R \in \mathcal{R}(k)} \Pr_{R,\Gamma}(S^{\alpha k, k})$.

As before, denote the **type** of $(a_1, \ldots, a_n) \in \mathcal{X}^n$ by $\mathcal{T}(a_1, \ldots, a_n) = T \in \mathbb{R}_{\geq 0}^r$, where $T_i = \frac{|\{\ell | a_\ell = i\}|}{n}$ is the frequency of each $i \in \mathcal{X}$ in $(a_1, \ldots, a_n)$. It can be easily verified that $\kappa(a_1, \ldots, a_n) = n \cdot \mathcal{T}(a_1, \ldots, a_n) \cdot K$ and $\mu(a_1, \ldots, a_n) = n \cdot \mathcal{T}(a_1, \ldots, a_n) \cdot B$ for any $(a_1, \ldots, a_n) \in \mathcal{X}^n$, where $\cdot$ is the standard dot product of two vectors. While $(X_n)_{n=1}^{\infty}$ is not a sequence of *i.i.d.* random variables, several properties of types (see [4]) are preserved.

Recall that for two distributions $\bar{c}, \bar{d} \in \mathbb{R}^t$, the classic Kullback-Leibler divergence is given by[11] $D\left(\bar{c} \middle\| \bar{d}\right) = \sum_{i=1}^{t} \bar{c}_i \log \frac{\bar{c}_i}{\bar{d}_i}$. We define the **extended Kullback-Leibler divergence** for any $\Upsilon^1, \Upsilon^2 \in \mathbb{R}_{\geq 0}^r$ by

$$D_e\left(\Upsilon^1 \middle\| \Upsilon^2\right) = \sum_{i \in \mathcal{X}} \Upsilon_i^1 \log \frac{\Upsilon_i^1}{\Upsilon_i^2} - \sum_{j=1}^{N} \lambda_j \log \lambda_j$$

where $\lambda_j = \sum_{i \in \mathcal{X}_j} \Upsilon_i^1$. There is a simple interpretation for $D_e\left(\Upsilon^1 \middle\| \Upsilon^2\right)$ when Υ^1 is a type and Υ^2 is an extended distribution. In this case, $D_e\left(\Upsilon^1 \middle\| \Upsilon^2\right) = \sum_{j=1}^{N} \lambda_j \cdot D_j + H(\lambda_1, \ldots, \lambda_N)$, where D_j is the Kullback-Liebler divergence between the distributions $\left(\frac{\Upsilon_i^1}{\lambda_j}\right)_{i \in \mathcal{X}_j}$ and $\left(\Upsilon_i^2\right)_{i \in \mathcal{X}_j}$ for $1 \leq j \leq N$, and H is the entropy function.

Lemma 4: For any $R \in \mathcal{R}$ and extended distribution Υ the following hold.

1) Let $(a_1, \ldots a_n) \in \mathcal{X}^n$ and $T = \mathcal{T}(a_1, \ldots, a_n)$. If $\Pr_{R,\Upsilon}(\forall 1 \leq \ell \leq n : X_\ell = a_\ell) > 0$ then
$$\Pr_{R,\Upsilon}(\forall 1 \leq \ell \leq n : X_\ell = a_\ell) = \exp\left(-n \sum_{i \in \mathcal{X}} T_i \log \frac{1}{\Upsilon_i}\right).$$
2) Let
$$K_n = \left\{ \left(\frac{n_1}{n}, \ldots, \frac{n_r}{n}\right) \middle| \begin{array}{c} \forall 1 \leq i \leq r : n_i \in \mathbb{N} \\ \text{and } \sum_{i=1}^{r} n_i = n \end{array} \right\}.$$
Then, $|K_n| \leq (n+1)^r$ and for any $\bar{a} \in \mathcal{X}^n$, $\mathcal{T}(\bar{a}) \in K_n$.
3) Let $T \in K_n$, then
$$\Pr_{R,\Upsilon}(\mathcal{T}(X_1, \ldots, X_n) = T) \leq \exp(-n D_e(T \| \Upsilon)).$$

The proof of the lemma relies on standard arguments relating to the method of types (see [9]).

Let $\bar{q}^j$ be the vector which solves (3) with respect to $(\bar{b}^j, \bar{k}^j, \bar{\gamma}^j)$. That is,

$$\bar{q}^j = \operatorname*{arg\,min}_{\bar{q} \cdot \bar{b}^j \leq \alpha \bar{q} \cdot \bar{k}^j,\ \bar{q} \text{ is a distribution}} \frac{1}{\bar{q} \cdot \bar{k}^j} D\left(\bar{q} \middle\| \bar{\gamma}^j\right). \tag{13}$$

Then, $\frac{1}{\bar{q}^j \cdot \bar{b}^j} D\left(\bar{q}^j \middle\| \bar{\gamma}^j\right) = M_j$. As before, let Q be the concatenation of $\bar{q}^1, \bar{q}^2, \ldots, \bar{q}^N$. Formally, define $Q \in \mathbb{R}_{\geq 0}^r$ with $Q_{s_{j-1}+i} = \bar{q}_i^j$ for any $1 \leq j \leq N$ and $1 \leq i \leq r_j$.

[11]We follow the standard convention $0 = 0 \log 0 = 0 \log \frac{0}{0}$.

The next lemma shows that a lower bound for the probability of events in $\Pr_{R,\Gamma}$ can be derived using a lower bound for the probability of the same events in $\Pr_{R,Q}$. The lemma is a simple implication of Lemma 4.

Lemma 5: Let $A \subseteq \mathcal{X}^n$ such that $\mathcal{T}(\bar{a}) = T$ for any $\bar{a} \in A$. Then, for any $R \in \mathcal{R}$,

$$\begin{aligned}&\log \Pr_{R,\Gamma}((X_1, \dots X_n) \in A)\\ &\geq \log \Pr_{R,Q}(X_1, \dots X_n \in A) - n \sum_{i \in \mathcal{X}} T_i \log \frac{Q_i}{\Gamma_i}. \end{aligned} \quad (14)$$

Note that the distribution function in (14) is $\Pr_{R,\Gamma}$ in the LHS and $\Pr_{R,Q}$ in the RHS.

To utilize Lemma 5 we need to specify a subset $A \subseteq S^{\alpha k,k}$ such that all the elements in A have the same type and A has high probability. The next lemma shows a slightly relaxed claim.

Lemma 6: For any $\varepsilon > 0$ there is $L > 0$ such that, for any $k > L$ and rules mapping $R \in \mathcal{R}$, there is a length $n^{k,R} \in \mathbb{N}$ and a type $T^{k,R}$ such that:

1)
$$\lim_{k\to\infty} \inf_{R \in \mathcal{R}} \frac{1}{k} \log \Pr_{R,Q} \left(\mathcal{T}(X_1, \dots, X_{n^{k,R}}) = T^{k,R} \right) = 0$$

2) For any $k > L$, $R \in \mathcal{R}$ and $1 \leq j \leq N$ set $\lambda_j^{k,R} = \sum_{i \in \mathcal{X}_j} T_i^{k,R}$. Then,
$$\lim_{k\to\infty} \sup_{R \in \mathcal{R}} \sum_{j=1}^{N} \sum_{i \in \mathcal{X}_j} \left| T_i^{k,R} - \lambda_j^{k,R} Q_i \right| = 0.$$

3) For any $k \geq L$ and $R \in \mathcal{R}$ it holds that $n^{k,R} \cdot (K \cdot T^{k,R}) \geq k$, $n^{k,R} \cdot (B \cdot T^{k,R}) \leq \alpha(1+\varepsilon)k$, and $n^{k,R} \leq \alpha(1+\varepsilon)k$.

The proof of the lemma uses a generative approach which bears some similarity to the *probabilistic method* [26], along with a repetitive application of Lemma 4. By combining the results of Lemmas 5 and 6 one can show that for any $\varepsilon > 0$,

$$\liminf_{k\to\infty} \inf_{R \in \mathcal{R}} \frac{1}{k} \log \Pr_{R,\Gamma} \left(S^{\alpha(1+\varepsilon)k,k} \right) \geq -M(1+\varepsilon).$$

Thus, by Lemma 3, we obtain the following.

Lemma 7: For any $\varepsilon > 0$,

$$\liminf_{k\to\infty} \frac{1}{k} \log p(\lfloor \alpha(1+\varepsilon)k \rfloor, k) \geq -M(1+\varepsilon).$$

REFERENCES

[1] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, 1st ed. Springer Publishing Company, Incorporated, 2015.

[2] L. Brankovic and H. Fernau, "A novel parameterised approximation algorithm for minimum vertex cover," *Theoretical Computer Science*, vol. 511, pp. 85 – 108, 2013, exact and Parameterized Computation.

[3] I. N. Sanov, "On the probability of large deviations of random variables," *Matematicheskii Sbornik*, vol. 42, pp. 11–44, in Russian. English translation in: Selected Translations in Mathematical Statistics and Probability I, pages 213–244, 1961.

[4] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*, 2nd ed. New York, NY, USA: Wiley-Interscience, 2006.

[5] M. R. Fellows, A. Kulik, F. A. Rosamond, and H. Shachnai, "Parameterized approximation via fidelity preserving transformations," *J. Comput. Syst. Sci.*, vol. 93, pp. 30–40, 2018.

[6] L. Brankovic and H. Fernau, "Parameterized approximation algorithms for hitting set," in *WAOA 2011*, Saarbrcken, Germany, 2012, pp. 63–76.

[7] D. Lokshtanov, F. Panolan, M. S. Ramanujan, and S. Saurabh, "Lossy kernelization," in *STOC 2017*, New York, NY, USA, 2017, pp. 224–237.

[8] R. Niedermeier, *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[9] A. Kulik and H. Shachnai, "Analysis of two-variable recurrence relations with application to parameterized approximations," *arXiv preprint arXiv:1911.02653*, 2019.

[10] S. Khot and O. Regev, "Vertex cover might be hard to approximate to within 2-ϵ," *J. Comput. Syst. Sci.*, vol. 74, no. 3, pp. 335–349, 2008.

[11] J. Chen, I. A. Kanj, and G. Xia, "Improved upper bounds for vertex cover," *Theoretical Computer Science*, vol. 411, no. 40, pp. 3736 – 3756, 2010.

[12] E. Bonnet, B. Escoffier, E. J. Kim, and V. T. Paschos, "On subexponential and fpt-time inapproximability," *Algorithmica*, vol. 71, no. 3, pp. 541–565, Mar 2015.

[13] P. Manurangsi and L. Trevisan, "Mildly exponential time approximation algorithms for vertex cover, balanced separator and uniform sparsest cut," in *APPROX/RANDOM'18*, 2018, pp. 20:1–20:17.

[14] N. Bansal, P. Chalermsook, B. Laekhanukit, D. Nanongkai, and J. Nederlof, "New tools and connections for exponential-time approximation," *Algorithmica*, vol. 81, no. 10, pp. 3993–4009, 2019.

[15] N. Bourgeois, B. Escoffier, and V. T. Paschos, "Approximation of max independent set, min vertex cover and related problems by moderately exponential algorithms," *Discrete Appl. Math.*, vol. 159, no. 17, pp. 1954–1970, 2011.

[16] M. Wahlström, "Algorithms, measures and upper bounds for satisfiability and related problems," Ph.D. dissertation, Department of Computer and Information Science, Linköpings University, Sweden, 2007.

[17] A. Becker, R. Bar-Yehuada, and D. Geiger, "Random algorithms for the loop cutset problem," in *UAI-99*, Stockholm, Sweden,, 1999, pp. 49–56.

[18] R. Beigel and D. Eppstein, "3-coloring in time o(1.3289n)," *Journal of Algorithms*, vol. 54, no. 2, pp. 168 – 204, 2005.

[19] D. Loksanov, M. S. Ramanuan, and S. Saurabh, "Linear time parameterized algorithms for subset feedback vertex set," *ACM Trans. Algorithms*, vol. 14, no. 1, pp. 7:1–7:37, 2018. [Online]. Available: https://doi.org/10.1145/3155299

[20] A. Drucker, J. Nederlof, and R. Santhanam, "Exponential Time Paradigms Through the Polynomial Time Lens," in *ESA 2016*, vol. 57, Aarhus, Denmask, 2016, pp. 36:1–36:14.

[21] D. Eppstein, "Quasiconvex analysis of multivariate recurrence equations for backtracking algorithms," *ACM Trans. Algorithms*, vol. 2, no. 4, pp. 492–509, Oct. 2006.

[22] F. V. Fomin, F. Grandoni, and D. Kratsch, "A measure & conquer approach for the analysis of exact algorithms," *J. ACM*, vol. 56, no. 5, pp. 25:1–25:32, 2009.

[23] W. Hoeffding, "Asymptotically optimal tests for multinomial distributions," *Ann. Math. Statist.*, vol. 36, no. 2, pp. 369–401, 04 1965.

[24] I. Csiszar, "The method of types [information theory]," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2505–2523, Oct 1998.

[25] R. Niedermeier and P. Rossmanith, "Upper bounds for vertex cover further improved," in *STACS 99*, C. Meinel and S. Tison, Eds., 1999, pp. 561–570.

[26] N. Alon and J. H. Spencer, *The Probabilistic Method*, 4th ed. Wiley Publishing, 2016.