Algorithmic Construction of Sets for k-Restrictions

Noga Alon * Dana Moshkovitz † Shmuel Safra ‡

Abstract

This work addresses *k-restriction problems*, which unify combinatorial problems of the following type: The goal is to construct a short list of strings in Σ^m that satisfies a given set of k-wise demands. For every k positions and every demand, there must be at least one string in the list that satisfies the demand at these positions. Problems of this form frequently arise in different fields in Computer Science.

The standard approach for deterministically solving such problems is via almost k-wise independence or k-wise approximations for other distributions. We offer a generic algorithmic method that yields considerably smaller constructions. To this end, we generalize a previous work of Naor, Schulman and Srinivasan [18]. Among other results, we greatly enhance the combinatorial objects in the heart of their method, called splitters, and construct *multi-way splitters*, using a new discrete version of the topological *Necklace Splitting Theorem* [1].

We utilize our methods to show improved constructions for *group testing* [19] and *generalized hashing* [3], and an improved inapproximability result for SET-COVER under the assumption $\mathcal{P} \neq \mathcal{NP}$.

1 Introduction

In the past decades, randomness was established as a fundamental notion in Computer Science. Consider, for example, the following simple algorithmic result (cf. [5]). Given a directed graph $G = (V, E)$, one wishes to find whether it contains a *simple* path of length k. A very simple randomized solution is to assign each vertex a uniformly chosen number in $[k]$, and consider only edges connecting a vertex assigned i, $1 \leq i < k$, to a vertex assigned $i + 1$. By scanning the resulting k-layered graph from the first layer to the last, one can easily discover in linear time whether it contains a simple path of length k. If G indeed contains such path, there is a probability of at least k^{-k} that the path remains in the random layered subgraph. If $k \leq O(\frac{\log|V|}{\log\log|V|})$, repeating this process, say, $2k^k$ is (a) efficient (b) discovers a sized-k simple path, if such exists, with good probability, and, (c) never discovers such path, unless it existed in the first place.

Many efforts were made to fully understand the role of randomness in computation. Those efforts revealed that many applications that incorporate randomness may be, in fact, *derandomized*, that is, simulated deterministically. The reason is that frequently one can settle for

*`nogaa@tau.ac.il`. Schools of Mathematics and Computer Science, Tel-Aviv University, Tel-Aviv, 69978, Israel. Research supported in part by a USA Israeli BSF grant, by a grant from the Israel Science Foundation, and by the Hermann Minkowski Minerva Center for Geometry at Tel Aviv University

†`danamo@tau.ac.il`. School of Computer Science, Tel-Aviv University, Tel-Aviv, 69978, Israel.

‡`safra@tau.ac.il`. Schools of Mathematics and Computer Science, Tel-Aviv University, Tel-Aviv, 69978, Israel.

objects that only *look* random to a limited observer, that is, *randomness is (many times) in the eyes of the beholder*. This motivated a series of works constructing small sample spaces for different limitations on a beholder: observing only k places (*k-wise independence*), considering only linear tests (*ϵ-bias* [17]), being unable to distinguish small deviations from such (*almost-*ϵ-biased, k-wise independent [4]) *etc.*

In the above example, one can consider only assignments $V \to [k]$ that are induced by k-wise independent sample-spaces, because one is only interested in a condition on any k vertices. As a matter of fact, any almost k-wise independent sample space suffices, as long as the distribution is, say, $\frac{1}{2}k^{-k}$-close to the uniform distribution. Furthermore, one can settle for even smaller sample-spaces, as the demands are merely *existential*; one only wishes to have one assignment for every k vertices, assigning the vertices $1, \ldots, k$ respectively. There is no need that a *noticeable* fraction of assignments would do that.

This work addresses problems of this type; given *any* set of existential k-wise limitations on the beholder as well as a distribution with respect to which the limitations have a good probability to be satisfied, it provides *algorithmically* a small set that satisfies the limitations.

The following definition formalizes the problems we address, namely, *k-restriction problems*:

Definition 1 (k-restriction problems)**.** *k-restriction problems are as follows:*

1. *The input is an alphabet Σ of size $|\Sigma| = q$, a length m and a set of s possible demands $f_i : \Sigma^k \to \{0,1\}$, $1 \le i \le s$. For every $1 \le i \le s$ there exists $a \in \Sigma^k$ so that $f_i(a) = 1$.*
2. *The task is to prepare a set $A \subseteq \Sigma^m$ so that: For any choice of k indices $1 \le i_1 < \cdots < i_k \le m$, and a demand j, $1 \le j \le s$, there is some $a \in A$, such that $f_j(a(i_1) \cdots a(i_k)) = 1$.*
3. *The quality of the solution is measured by how small $|A|$ is.*

For completeness, let us formulate the above problem as a k-restriction problem. $\Sigma = [k]$, $q = k$, $m = |V|$ and there are $s = k!$ demands; for every permutation $\sigma : [k] \to [k]$, corresponding to a possible ordering of the k observed vertices, there is a demand $f_\sigma : [k] \to \{0,1\}$, requiring the vertices would be assigned the appropriate numbers, that is, $f_\sigma(x_1, \ldots, x_k) = 1$ *if and only if* $x_1 = \sigma(1) \wedge \cdots \wedge x_k = \sigma(k)$.

1.1 Our Work

For every constant k, we devise an efficient algorithm that finds a solution whose size is arbitrarily close to the size of a probabilistic construction. For $k = O(\frac{\log m}{\log\log m})$, we show an efficient algorithm constructing solutions of a somewhat larger size. We also present a method for handling problems with larger k. The exact results are detailed in section 4 (theorems 1 and 2), after the required terminology and background. Let us present some of the applications of our work.

Multi-Way Splitters

A *k-wise t-way splitter* is a set of partitions of $[m]$ into b sets, so that for every k coordinates within $[m]$, each having a color in $[t]$, there exists a partition so that every set $1 \le j \le b$ contains the same number of coordinates of each color up to rounding. This is a generalization of the existing notion of a *splitter* introduced by [18]. Splitters are multi-way splitters for $t = 1$.

Splitters and multi-way splitters are used to split a problem of the form "for every k coordinates,..." into b problems of the form "for every $\lceil k/b \rceil$ coordinates,...". The advantage of multi-way splitters is that they give more control on the split. They allow us to split a problem of the form "for every k coordinates, for every partition of the coordinates into t types,..." into b problems of the form "for every $\lceil k/b \rceil$ coordinates, for every partition of the coordinates into t types,...".

We show how to construct succinct multi-way splitters

Theorem 3 *A k-wise t-way splitter for splitting m coordinates into b blocks of size*

$$O(k^4 \log m) \cdot b^{p+1} \cdot \binom{k^2}{p}$$

where $p = (b-1)t$, may be constructed in time polynomial in m, t^k and the size of the construction.

Our main tool is the Necklace Splitting Theorem [1], which is a theorem proved by topological techniques, that considers the number of cuts needed to split a necklace with beads of t types between b thieves. In order to use the theorem, we prove a new discrete version of it.

Group Testing

The problem of group testing [11, 19] was first presented during world war II. Among a large population of m soldiers, at most d individuals carry a fatal virus. We would like to blood test the soldiers to detect all carriers. The simple solution would be to test the blood of each soldier separately. But if we are to minimize the number of tests, we can mix the blood of several soldiers and test the mixture. If the test comes out negative, then none of the tested soldiers is a carrier. If the test comes out positive, we know at least one of them is a carrier.

The problem is to come up, given m and d, with a small set of group tests. Each group test is defined by the subset of soldiers whose blood samples are mixed. After performing all the tests there should be enough information to pin down the identities of the carriers. We focus on the non-adaptive case, namely, all tests should be decided upon before starting to test.

Using the results presented in this paper, we prove that there exists an efficient deterministic construction of a set of group-tests, whose size is – roughly – $ed^2 \ln m$, for a large, however *constant*, d. The reason that d need be constant, is that the running time of the algorithm depends on m^d.

Theorem 4 *For any fixed d, for every $\delta > 0$, a set of at most $(1+\delta)\cdot e(d+1)[(d+1)\ln m + \ln(d+1)]$ group tests may be found in time polynomial in m^d.*

In other words, for every $\delta > 0$, there exists $d(\delta)$, so that for every $d \geq d(\delta)$, there is an algorithm that given m can produce a set of group-tests for m people with at most d carriers, whose size is at most $(1+\delta)ed^2 \ln m$ in time polynomial in m.

Generalized Hashing

The classic problem of $(m, q, k)-$*perfect hashing* is to find a small set of functions $h_i : [m] \rightarrow [q]$ so that for every $X \subseteq [m]$, $|X| \leq k$, there exists some function which is 1-1 on X, i.e there exists i so that for every $x_1, x_2 \in X$, $x_1 \neq x_2$, it holds that $h_i(x_1) \neq h_i(x_2)$.

This problem has many variants. One example is (t, u)-*hash families* used for the design of parent identifying codes for digital fingerprinting [3]. Again, one wishes to find a small set of functions $h_i : [m] \to [q]$, but now, for every two subsets $T \subset U \subseteq [m]$, $|T| = t$, $|U| = u$, there should exist i so that for any $x_1 \in T$, $x_2 \in U$, $x_1 \neq x_2$, it holds that $h_i(x_1) \neq h_i(x_2)$.

[3] addressed the case of the minimal alphabet-size $q = t+1$, and concentrated on the *rate* $R = \frac{\log_q m}{n}$, where n is the number of functions produced. Our results derive a better explicit construction than theirs:

Theorem 5 *For any fixed $2 \leq t < u$, for any $\delta > 0$, one can construct efficiently a (t,u)-hash family over an alphabet of size $t+1$ whose rate is at least $(1-\delta)\frac{t!(u-t)^{u-t}}{u^{u+1}\ln(t+1)}$.*

Our methods are also applicable when the parameters u and t are growing, and not constant. For instance, we prove:

Theorem 6 *There exists an algorithm that given a and m, outputs an $(a, 2a)$-hash family of at most $(4e)^a \log^{O(\log a)} m$ functions $h_i : [m] \to [a+1]$ in time polynomial in m and this size.*

The same methods could be easily applied for other choices of parameters and other generalizations of perfect hashing as well.

Hardness of Approximating Set-Cover

Given a universe $\mathbb{U} = \{u_1, ..., u_n\}$ and a family of its subsets, $\mathbb{S} = \{S_1, ..., S_m\} \subseteq P(\mathbb{U})$, $\bigcup_{S_j \in \mathbb{S}} S_j = \mathbb{U}$, Set-Cover is the problem of finding a minimal sub-family $\mathbb{C} \subseteq \mathbb{S}$ that covers the whole universe, $\bigcup_{S_j \in \mathbb{C}} S_j = \mathbb{U}$. Set-Cover is a classic $\mathcal{NP}$-hard combinatorial optimization problem, and it is known that it can be approximated in polynomial time to within $\ln n - \ln\ln n + \Theta(1)$ [21, 15, 22].

Feige proved that for every $\varepsilon > 0$, there is no efficient approximation to within $(1-\varepsilon)\ln n$, under the assumption $\mathcal{NP} \nsubseteq DTIME(n^{\log\log n})$ [13].

The work of [8] facilitated hardness results for Set-Cover under the weaker, more traditional, assumption $\mathcal{P} \neq \mathcal{NP}$. However, it seems their reduction has an inherent inverse linear dependency in a parameter $d \geq 2$ of the underlying PCP, called its *dependency*, or *the number of provers*, which prevents the reduction from obtaining the optimal factor. Specifically, using the sub-constant error PCP of Raz and Safra [20], unless $\mathcal{P} = \mathcal{NP}$, Set-Cover cannot be efficiently approximated to within any number smaller than $\frac{\log_2 n}{2(d+1)}$. Arora and Sudan claim in [7] that d can be made 3 for the error probability we need, but do not provide the details.

We would like to comment that the work of [18] eliminated – up to low order terms – the advantage the randomized reductions had over the deterministic ones, and thus, under assumptions concerning efficient randomized computation, better results are *not* known.

We hence address the challenge of proving better inapproximability results for Set-Cover, under $\mathcal{P} \neq \mathcal{NP}$. Roughly speaking, we manage to reduce the inverse dependency of the inapproximability result of [8] in d to the considerably smaller inverse dependency in $\ln d$, which improves the result no matter how small $d \geq 2$ is made.

Theorem 7 *For any $\eta \geq 1$,* Set-Cover *cannot be efficiently approximated to within any number smaller than $c\ln n$, for $c = \frac{1}{(1+1/\eta)(\ln \eta d + 1 + 1/\eta d)}$, unless $\mathcal{P} = \mathcal{NP}$.*

For $d = 3$, we obtain a hardness factor of more than $0.2267 \ln n$ (for $\eta = 3$), instead of arbitrarily close to $0.125 \ln n$.

1.2 Comparison With Previous Work

The literature regarding derandomization is vast. We will reference only two papers that are of particular relevance.

The paper by Naor, Schulman and Srinivasan [18] is the starting point of our work. They consider a few important problems that fall into the category of k-restriction problems, such as perfect hash families and universal sets. They solve these problems using similar methods, focusing on decomposition of the problems into smaller problems. We take a *unifying* approach and address all k-restriction problems. Moreover, two ingredients are added. Naor *et al* consider problems in which a small set of strings chosen *uniformly* is (with high probability) a solution. We consider problems in which a small set of strings chosen from some (efficiently approximatable) distribution, not necessarily the uniform one, is (with high probability) a solution. Many intriguing k-restriction problems are so (see our applications). In addition, Naor *et al* use *splitters* to split a problem into sub-problems. We put *splitters* in a wider context, viewing the observation underlying their construction as a special case of the topological Necklace Splitting Theorem [1]. This realization allows us to construct *multi-way splitters* that form much finer splits, and are, hence, suitable for a larger class of applications.

Koller and Megiddo [14] take a similar approach to ours, and consider the problem of constructing small sample-spaces algorithmically given the constraints. They consider constraints of the form "the probability these k coordinates admit this string in Σ^k is this number". In one respect, this is a generalization of our definition, as we only guarantee a positive probability, and do not promise a restriction would be satisfied with some specified probability. However, their results are only applicable when the number of observed k-tuples is small, while we concentrate on demands on all $\binom{m}{k}$ possible k-tuples, and are still able to provide efficient solutions.

1.3 Organization

We start by laying out the foundations, when discussing the probabilistic construction in section 2 and k-wise approximations in section 3. In section 4 we describe our results simulating the probabilistic construction deterministically. The construction is done in three stages: greediness, approximation and concatenation. In section 5, we present a *Divide-and-Conquer* approach, that builds upon the first results and may derive efficient algorithms for a wider range of parameters. This approach is based on splitters [18] and extends them using the Necklace Splitting Theorem [1]. The applications are detailed in sections 6 (group testing), 7 (generalized hashing) and 8 (Set-Cover hardness of approximation).

2 Probabilistic Solution

Σ^m is a solution for every k-restriction problem on alphabet Σ and length m. However, a substantially smaller subset of Σ^m chosen at random forms a solution with good probability. In this section we will study this argument, and establish how small we can expect a solution to a general k-restriction problem to be.

To this end, we will define the *density* ε of a *k-restriction* problem with respect to some probability distribution D over Σ^m to be so that for any given demand at any given k-positions, when picking a random string from D, the string satisfies the demand at those positions with probability at least ε.

Definition 2 (density)**.** *Given a probability distribution, $D\colon \Sigma^m \to [0,1]$, the* density *of a k-restriction problem with respect to D is*

$$\varepsilon \doteq \min_{\substack{1\le i_1<\cdots<i_k\le m\\ 1\le j\le s}} \left\{ \Pr_{a\sim D}\left[f_j(a(i_1)\cdots a(i_k))=1\right]\right\}$$

Note that every k-restriction problem has density at least q^{-k} with respect to the uniform distribution. Some problems (e.g., group testing, or (t,u)-hashing) have a much larger density with respect to other distributions.

A simple union bound calculation shows that if a problem has a large density (with respect to some distribution), it has a small solution:

Proposition 1 (union bound)**.** *Every k-restriction problem with density ε with respect to some probability distribution D has a solution of size at most $\lceil \frac{k\ln m+\ln s}{\varepsilon}\rceil$.*

Proof. Fix a k-restriction problem with density ε with respect to some probability distribution D. Let $t > \frac{k\ln m+\ln s}{\varepsilon}$ be some natural number. consider $A=\{a_1,\ldots,a_t\}\subseteq \Sigma^m$ chosen probabilistically as follows: for each $1\le i\le t$, draw a_i independently from D. Let us bound the probability that there exist k indices $1\le i_1<\cdots<i_k\le m$ and a constraint $1\le j\le s$, such that $a_1,\ldots,a_t$ all do not match,

$$\Pr_{a_1,\ldots,a_t\sim D}\left[\exists i_1,\ldots,i_k\exists j\forall a\in A,\;\; f_j(a(i_1),\ldots,a(i_k))=0\right] \le \binom{m}{k} s(1-\varepsilon)^t \le e^{k\ln m+\ln s-\varepsilon t}$$

where we use the union bound and the inequality $1-x\le e^{-x}$.

For our choice of t, the probability above is smaller than 1, hence there exists $A\subseteq\Sigma^m$ of size t which is a solution for the k-restriction problem. ∎

Moreover, picking slightly more vectors at random from D should yield a solution with high probability. But note that we do not know of an efficient way to verify a given set of vectors really does form a solution, unless $k=\Theta(1)$.

3 k-wise Approximating Distributions

A relatively small solution for any k-restriction problem may be obtained using a succinct distribution that approximates the distribution, with respect to which the problem has a large density. The resulting solution is, however, considerably larger than the random solution presented in the previous section. In this section we will make the proper definitions and consider this argument.

We start by defining ways to measure distance between distributions.

Definition 3 (l_p-norm distance)**.** *The* distance in the l_p-norm *between two probability distributions D,P over some sample space Ω is $\|D-P\|_p \doteq \left(\sum_{a\in\Omega}|D(a)-P(a)|^p\right)^{\frac{1}{p}}$. The distance in max-norm is $\|D-P\|_\infty \doteq \max_{a\in\Omega}|D(a)-P(a)|$.*

We consider distributions over Σ^m, and are interested in k-wise approximations, i.e., distributions that approximate a distribution well when *restricted* to any k coordinates. The *restriction* of a probability distribution D over Σ^m to indices $1\le i_1<\cdots<i_k\le m$ is a probability distribution $D_{i_1,\ldots,i_k}$ over Σ^k, that assigns each $a\in\Sigma^k$ the probability a string drawn randomly from D has a in indices $i_1,\ldots,i_k$, i.e., $D_{i_1,\ldots,i_k}(a)\doteq \Pr_{X\sim D}\left[X(i_1)=a(1)\wedge\cdots\wedge X(i_k)=a(k)\right]$.

Definition 4 (k-wise ε-closeness)**.** *Two distributions D, P over Σ^m are said to be* k-wise ε-close in the l_p-norm*, if for any* $1 \leq i_1 < \cdots < i_k \leq m$, $\|D_{i_1,\ldots,i_k} - P_{i_1,\ldots,i_k}\|_p < \varepsilon$*.*

A distribution which is k-wise ε-close to the uniform distribution *for every* $\varepsilon > 0$ (no matter in what norm) is called *k-wise independent*, and hence the use of the term *almost k-wise independence*, when this holds for any $\varepsilon \geq \varepsilon_0$ *for some* small $\varepsilon_0 > 0$.

Approximations in the l_1-norm are very strong:

Lemma 1. *Let D, P be two probability distributions over Σ^m, such that P is k-wise ε-close to D in the l_1-norm. Then for every k-restriction $f : \Sigma^k \to \{0, 1\}$ and indices $1 \leq i_1 < \cdots < i_k \leq m$,*

$$\left| \Pr_{X \sim D}[f(X(i_1) \cdots X(i_k)) = 1] - \Pr_{X \sim P}[f(X(i_1) \cdots X(i_k)) = 1] \right| < \varepsilon$$

Proof.

$$\begin{aligned} \|D_{i_1,\ldots,i_k} - P_{i_1,\ldots i_k}\|_1 &= \sum_{a \in \Sigma^k} |D_{i_1,\ldots,i_k}(a) - P_{i_1,\ldots,i_k}(a)| \\ &\geq \sum_{a : f(a)=1} |D_{i_1,\ldots,i_k}(a) - P_{i_1,\ldots,i_k}(a)| \\ &\geq \left| \sum_{a : f(a)=1} D_{i_1,\ldots,i_k}(a) - \sum_{a : f(a)=1} P_{i_1,\ldots,i_k}(a) \right| \\ &= \left| \Pr_{X \sim D}[f(X(i_1) \cdots X(i_k)) = 1] - \Pr_{X \sim P}[f(X(i_1) \cdots X(i_k)) = 1] \right| \end{aligned}$$

The lemma follows from $|D_{i_1,\ldots,i_k} - P_{i_1,\ldots i_k}\|_1 < \varepsilon$. ■

Many works present *explicit* constructions of *succinct* distributions that are k-wise ε-close to different distributions in different norms. *Succinctness* refers to the size of their *support*, i.e the number of strings having a positive probability. We will use the following definition:

Definition 5 (k-wise approximation)**.** *A distribution D over Σ^m is said to be* k-wise efficiently approximatable*, if the support of a distribution which is k-wise ε-close to it in the l_1-norm may be enumerated in time polynomial in m, ϵ^{-1} and $|\Sigma|^k$.*

Perhaps the largest family of distributions that admit an efficient k-wise approximation is that of *product distributions*. A *product distribution* over Σ^m , $|\Sigma| = q$, is specified by a matrix $\mathcal{P}$ of dimensions $m \times q$, indicating the probability each of the m independent coordinates assumes each of the values in Σ. Note that the uniform distribution is a product distribution. We quote:

Lemma 2. *[12] Any product distribution on Σ^m is k-wise efficiently approximatable.*

Using the former definition we can now describe the folklore method for solving *k-restriction* problems deterministically. The underlying observation is the following:

Proposition 2. *[folklore] The support of a distribution over Σ^m which is k-wise ε-close in l_1 to D is a solution for any* k-restriction *problem with parameters m ,Σ, k, s and density ε with respect to D.*

Proof. Fix some k-restriction problem with density ε with respect to D. Consider some distribution P which is k-wise ε-close in l_1 to D. By way of contradiction, let us assume that there exist $1 \leq i_1 < \dots i_k \leq m$ and $1 \leq j \leq s$ so that $\Pr_{X \sim P}[f_j(X(i_1) \cdots X(i_k)) = 1] = 0$. However, $\Pr_{X \sim D}[f_j(X(i_1) \cdots X(i_k)) = 1] \geq \varepsilon$. This contradicts lemma 1. ■

Unfortunately, this yields solutions with a dependence on parameters that is inferior to that of proposition 1 (which was roughly $\frac{k \log m}{\varepsilon}$). To demonstrate this, consider the extensively studied uniform distribution over $\{0,1\}^m$. Known explicit constructions give distributions that are almost k-wise independent in l_1 of support size $O(2^k \frac{k^2 \log^2 m}{\varepsilon^2})$ [4]. The dependence is usually much worse for other distributions (e.g, see [12]).

Let us remark that for k-restriction problems in which each restriction f_j requires exactly one string in Σ^k (i.e., $\left|f_j^{-1}(1)\right| = 1$), one can settle for an approximation in l_∞. Unfortunately, even then the known constructions are large (e.g., $O(\frac{k^2 \log^2 m}{\varepsilon^2})$ [4], or $O(\frac{k \log m}{\varepsilon^3})$ [17, 2] for the uniform distribution on $\{0,1\}^m$).

4 Our Results

Our results provide solutions that are smaller than those implied by proposition 2.

We first make one technical definition. We will sometimes consider sets of demands that are *invariant under permutations*, i.e when permutating the substrings, the possible demands remain the same. Formally, if for a demand f, we use C_f to indicate $f_i^{-1}(1)$, the set of substrings it accepts, then a set of demands $\{f_1, \dots, f_s\}$ is *invariant under permutations*, if for any permutation $\sigma\colon [k] \to [k]$,

$$\{C_{f_1}, \dots, C_{f_s}\} = \{\sigma(C_{f_1}), \dots, \sigma(C_{f_s})\}$$

Natural problems, like the ones presented in the introduction, usually have demands with this property. Moreover, if the set of demands does not have this property, we can add it dummy demands. Since the dependence of our solutions in the number of demands s is only logarithmic, this usually does not enlarge the solution significantly.

Now to the results – for $k = O(1)$, an efficient algorithm that produces solutions of size arbitrarily close to that of proposition 1 is implied by the following theorem:

Theorem 1. *Fix some efficiently approximatable probability distribution D. For any k-restriction problem with density ε with respect to D, there is an algorithm, that given an instance of the problem and an accuracy parameter $0 < \delta < 1$, obtains a solution of size at most $\left\lceil \frac{k \ln m + \ln s}{(1-\delta)\varepsilon} \right\rceil$ in time polynomial in s, m^k, q^k, ε^{-1} and δ^{-1}.*

Let $|Hash_{m,q,k}|$ be the size of the smallest possible efficient construction of a (m,q,k)-perfect hash family (see the introduction). For $k = O(\frac{\log m}{\log \log m})$, an efficient algorithm is implied by the following theorem:

Theorem 2. *Fix some efficiently approximatable probability distribution D. For any k-restriction problem with density ε with respect to D and demands that are invariant under permutations, there exists an algorithm that given an instance of the problem and an accuracy parameter $0 < \delta < 1$, obtains a solution of size at most $\left\lceil \frac{2k \ln k + \ln s}{(1-\delta)\varepsilon} \times \left|Hash_{m,k^2,k}\right| \right\rceil$ in time $poly(m, s, k^k, q^k, \varepsilon^{-1}, \delta^{-1})$.*

Note that by [2] and the connection between perfect hashing and asymptotically optimal error correcting codes, $\left|Hash_{m,k^2,k}\right| \leq O(k^4 \log m)$:

Lemma 3. *For every m, k, there exists an explicit construction of a code with m codewords of length $n = O(k^4 \log m)$ over an alphabet of size k^2, so that every k codewords differ on some coordinate.*

Proof. Consider an explicit construction of an error correcting code with m codewords of length n over alphabet $[k^2]$ whose (normalized) Hamming distance is at least $1-\frac{2}{k^2}$ (such a construction exists by [2]). Denote the codewords by $c_1, \ldots, c_m$. Let $I = \{i_1, \ldots, i_k\} \subseteq [m]$. By the distance of the code,

$$\begin{aligned}\sum_{j=1}^{n} |\{i_1, i_2 \in I \mid i_1 < i_2,\ c_{i_1}(j) \neq c_{i_2}(j)\}| &= \sum_{i_1 < i_2 \in I} |\{j \in [n] \mid c_{i_1}(j) \neq c_{i_2}(j)\}| \\ &\geq \binom{k}{2}\left(1 - \frac{2}{k^2}\right) n\end{aligned}$$

By averaging, there exists $1 \leq j \leq n$ for which,

$$|\{i_1, i_2 \in I \mid i_1 < i_2,\ c_{i_1}(j) \neq c_{i_2}(j)\}| \geq \frac{1}{n} \cdot \binom{k}{2}\left(1 - \frac{2}{k^2}\right) n > \binom{k}{2} - 1$$

But since the left-hand is an integer, also bounded from above by the integer $\binom{k}{2}$,

$$|\{i_1, i_2 \in I \mid i_1 < i_2,\ c_{i_1}(j) \neq c_{i_1}(j)\}| = \binom{k}{2}$$

Thus, if $i_1 \neq i_2 \in I$, then $c_{i_1}(j) \neq c_{i_2}(j)$. ■

For larger values of k we present in section 5, an approach that builds upon the above theorems. To prove them we use three techniques. First, we formulate *k-restriction* problems as (huge) Set-Cover problems. This derives an algorithm with two flaws: The running-time is polynomial in m^k, which is inefficient for non-constant k, and it requires an enumeration of the distribution's support (sub-section 4.1). Then we k-wise approximate the distribution to get one that preserves most of the density, and has support that may be enumerated (sub-section 4.2), thus obtaining theorem 1. Lastly, we show a technique transforming inefficient solutions, such as the one obtained when combining the first two techniques, into efficient ones, by way of concatenation (sub-section 4.3). This results in theorem 2.

4.1 Greediness

Somewhat surprisingly – bearing in mind our Set-Cover application – the first step of the construction involves finding a small solution for some Set-Cover instance (see section 8 for background). We show that any k-restriction problem may be formulated as a Set-Cover problem, and invoking the greedy algorithm (i.e the one that chooses in any step the subset that covers the largest number of elements not covered yet) ensures the size anticipated by probabilistic considerations. This approach is really an implementation of the *method of conditional expectations.*

Unfortunately, the reduction to Set-Cover is not polynomial in m for $k = \omega(1)$. Hence, in this case, the algorithm we present is inefficient by itself. Nonetheless, it serves as a procedure inside the final algorithm, when invoked with much smaller parameters.

First, let us adhere to the Set-Cover setting, and analyze the behavior of the greedy algorithm when all elements are covered by a significant portion of the subsets.

Proposition 3. *Let $(\mathbb{U}, \mathbb{S})$ be a* Set-Cover *instance. Let $D \colon \mathbb{S} \to [0,1]$ be some probability distribution over $\mathbb{S}$. If for every $u \in \mathbb{U}$, $\Pr_{S \sim D}[u \in S] \geq \varepsilon$, then the greedy algorithm produces a set cover whose size is at most $\left\lceil \frac{\ln |\mathbb{U}|}{\varepsilon} \right\rceil$.*

Proof. For a subset $S \in \mathbb{S}$ and an element $u \in \mathbb{U}$, let $X_{u,S}$ be an indicator variable for the event $'u \in S'$. By linearity of expectations, for every $U' \subseteq \mathbb{U}$,

$$\mathop{\mathbf{E}}_{S \sim D}\left[|S \cap U'|\right] = \mathop{\mathbf{E}}_{S \sim D}\left[\sum_{u \in U'} X_{u,S}\right] = \sum_{u \in U'} \mathop{\mathbf{E}}_{S \sim D}\left[X_{u,S}\right] = \sum_{u \in U'} \Pr_{S \sim D}\left[u \in S\right] \geq \varepsilon \left|U'\right|$$

Hence, in each iteration the greedy algorithm covers at least ε of the sub-universe. Thus, after t iterations, at most $(1-\varepsilon)^t \leq e^{-\varepsilon t}$ of the elements remain. Therefore, any number of iterations larger than $\frac{\ln |\mathbb{U}|}{\varepsilon}$ suffices. ∎

The following lemma explains how k-restriction problems reduce to Set-Cover problems and uses the above analysis to derive the algorithm.

Lemma 4 (greediness)**.** *Given the support of some probability distribution $D : \Sigma^m \to [0,1]$ and a* k-restriction *problem with density ε with respect to D, one can obtain a solution whose size is at most $\left\lceil \frac{k \ln m + \ln s}{\varepsilon} \right\rceil$ in time polynomial in $\binom{m}{k}$, s and $|supp(D)|$.*

Proof. Define an element for every possible pair (k positions, demand on those positions), and a set for every support vector:

$$\mathbb{U} \doteq \{< i_1, \ldots, i_k, j > \mid 1 \leq i_1 < \cdots < i_k \leq m, 1 \leq j \leq s\}$$

$$\mathbb{S} \doteq \{\, S_a \mid a \in supp(D)\} \text{ where } S_a \doteq \{< i_1, \ldots, i_k, j > \in \mathbb{U} \mid f_j(a(i_1) \cdots a(i_k)) = 1\}$$

Note that a cover $\mathcal{C}$ for $(\mathbb{U}, \mathbb{S})$ corresponds to a solution of size $|\mathcal{C}|$. As for any $u \in \mathbb{U}$, $\Pr_{a \sim D}[u \in S_a] \geq \varepsilon$, the lemma follows from proposition 3. ∎

4.2 Approximating Distributions

The greedy algorithm immediately provides us an algorithm that solves a given k-restriction problem in time that depends on the size of the support of the distribution with respect to which we evaluate the density. This size may be very large for natural distributions, such as the uniform one, whose support is Σ^m.

The idea is to approximate the distribution first, in order to obtain a new distribution with considerably smaller support, that preserves most of the density. Then invoke the algorithm on the new distribution.

Proof. (of theorem 1) First construct a distribution D' which is k-wise $\delta\varepsilon$-close in the l_1-norm to the distribution at hand. By lemma 1, for any k indices $1 \leq i_1 < \cdots < i_k \leq m$ and any k-restriction $f : \Sigma^k \to \{0,1\}$,

$$\Pr_{X \sim D'}\left[f(X(i_1) \cdots X(i_k)) = 1\right] \geq \Pr_{X \sim D}\left[f(X(i_1) \cdots X(i_k)) = 1\right] - \delta\varepsilon \geq \varepsilon - \delta\varepsilon = (1-\delta)\,\varepsilon$$

The theorem follows from lemma 4 with D' instead of the original distribution. ∎

4.3 Concatenation

In this sub-section we show a general scheme for transforming inefficient algorithms for k-restriction problems, such as the greedy algorithm, into more efficient algorithms via concatenation. This follows an idea by [18].

Invoke the inefficient algorithm only on $m' \ll m$ tuples. In the case of the greedy algorithm, this reduces the $\binom{m}{k}$ factor in the running-time to the considerably smaller $\binom{m'}{k}$ factor. It is useful to think of the output of the inefficient algorithm as a table with m' columns.

Our plan is to interleave these columns in a clever way, so they compose a solution for the original problem on m tuples. This clever way is defined by an outer construction, that should be thought of as a table with m columns having entries from $[m']$. In this way, each entry references a column in the inner construction. The concatenated construction replaces each reference with the actual column.

The property of the outer construction is *perfect hashing* (see the introduction): for every choice of k columns out of m, there should be a row in which all k entries differ. That is, in the final construction we can find there k different columns of the inner construction. By the property of the inner construction and the invariance under permutations, they satisfy all demands.

Now we can describe the concatenation algorithm. It works for any *k-restriction* problem with demands that are invariant under permutations,

- **Input:** Solution A for the problem on m'-tuples, and (m, m', k)-perfect hash family B.
- **Output:** Solution C for the problem on m-tuples (instead of m'-tuples).
- **Process:** For every $a \in A$, and every $b \in B$, output the vector $a(b(1)) \cdots a(b(m))$.

The correctness of this algorithm may be easily checked:

Lemma 5 (concatenation)**.** *Given A and B as specified, the concatenation algorithm produces a solution C for the problem with $|C| = |A| \cdot |B|$.*

Proof. Note that clearly $C \subseteq \Sigma^m$, and $|C| = |A| \cdot |B|$. Let us prove C is indeed a solution. Fix a restriction $1 \le j \le s$ and k indices $1 \le i_1 < \cdots < i_k \le m$. As B is a (m, m', k)-perfect hash family, there exists $b \in B$ with $b(i_1), \ldots, b(i_k) \in [m']$ all distinct. Let us consider the permutation $\sigma : [k] \to [k]$ so that $b(i_{\sigma(1)}) < \cdots < b(i_{\sigma(k)})$. By the invariance under permutations, there exists a restriction $1 \le j' \le s$ that is equivalent to f_j when applying σ. As A is a solution for the problem, there exists $a \in A$ such that $f_{j'}(a(b(i_{\sigma(1)})) \cdots a(b(i_{\sigma(k)})) = 1$. Hence $f_j(a(b(i_1)) \cdots a(b(i_k))) = 1$. That is, the vector $a(b(1)) \cdots a(b(m))$ satisfies f_j at indices $1 \le i_1 < \cdots < i_k \le m$. As this holds for every k-restriction and k indices, the concatenated construction is solution for the problem on m tuples. ■

Recalling that there are small explicit constructions of (m, k^2, k)-perfect hash families (e.g of size $O(k^4 \log m)$ [2]), theorem 2 follows.

5 Divide and Conquer

We outline a Divide and Conquer approach, that may be used together with theorem 2 to obtain an efficient solution for many *k-restriction* problems, even for $k \ge \omega(\log m / \log\log m)$:

1. Give a short list of possible partitions of $\{1, \ldots, m\}$ into b blocks, typically $b = \Theta(\log k)$.

2. Find a solution for the sub-problem with parameters m, $\lceil k/b \rceil$ (e.g using theorem 2).

3. Combine the sub-solutions.

For the purpose of dividing into blocks, [18] defined a combinatorial entity called a *splitter*. We generalize their definition, based on the topological Necklace Splitting Theorem of [1].

5.1 Necklace Splitting

The Necklace Splitting Theorem states, that if b thieves steal a necklace with beads of t colors, they can cut it in only $(b-1)t$ places, in order to fairly divide it between them.

In [1], a continuous version of this theorem is proven using tools from algebraic topology, and a discrete version is deduced. We will need a slightly more general discrete version, which may be of independent interest, and show how to derive it.

Continuous Necklace Splitting

let $I = [0, 1]$ be the unit interval. An *interval t-coloring* is a coloring of the points of I by t colors, such that for each $1 \le i \le t$, the set of points colored by i is Lebesgue measurable. Given such coloring, a *b-splitting of size r* is a sequence of numbers $0 = y_0 \le y_1 \le \cdots \le y_{r+1} = 1$ and a partition of the family of $r+1$ intervals $F = \{ [y_i, y_{i+1}] \mid 0 \le i \le r\}$ into b pairwise disjoint subfamilies $F_1, \ldots, F_b$ whose union is F, such that for each $1 \le j \le b$, the union of the intervals in F_j captures precisely $1/b$ of the total measure of each of the t colors.

Clearly, if each color forms an interval, the size of a b-splitting is at least $(b-1)t$. The Necklace Splitting Theorem shows this is tight.

Lemma 6 (A Continuous Necklace Splitting Theorem,[1])**.** *Every interval t-coloring has a b-splitting of size $(b-1)t$.*

Let us describe the intuition behind the case of $t = b = 2$, which provides some insight to the role of topology in the proof. Call one of the types *red*. Instead of observing some coloring of the unit interval, observe the equivalent coloring of the one-dimensional sphere (*a necklace closed at its clasp*). Consider some half necklace. If the measure of red within this half is exactly $\frac{1}{2}$, this induces a fair partition, and we are done. Otherwise, assume without loss of generality that this measure is larger than $\frac{1}{2}$. When rotating the necklace 180°, the measure of red in the observed half is hence smaller than $\frac{1}{2}$. As the change in the measure is continuous, there must be a half in which this measure is exactly $\frac{1}{2}$. In general, the proof uses a generalization of the Borsuk-Ulam theorem.

Discrete Necklace Splitting

Suppose a necklace has n beads, each having a color i, where $1 \le i \le t$. Suppose there are a_i beads of color i, $1 \le i \le t$; $\sum_{i=1}^{t} a_i = n$. A b-splitting of a necklace is a partition of it into b parts, each consisting of a finite number of non-overlapping sub-necklaces of beads whose union captures either $\lfloor a_i/b \rfloor$ or $\lceil a_i/b \rceil$ beads of color i, for every $1 \le i \le t$. The size of the partition is the number of cuts that form the sub-necklaces.

The discrete version appearing in [1] considers only necklaces that can be accurately divided among b thieves, i.e $b|a_i$, for every $1 \le i \le t$. The following variant is clearly a generalization of this version.

Lemma 7 (A Discrete Necklace Splitting Theorem)**.** *Every necklace with a_i beads of type i, $1 \le i \le t$, has a b-splitting of size at most $(b-1)t$.*

Proof. Convert the discrete necklace into an interval coloring, by partitioning $I = [0,1]$ into n equal segments, and coloring the j'th segment by the color of the j'th bead of the necklace. By lemma 6, there exists a b-splitting of the interval into families of intervals $F_1, \dots, F_b$ with $(b-1)t$ cuts.

However, these cuts not necessarily occur at the endpoints of the segments, and thus, they do not necessarily correspond to a splitting of the discrete necklace; several thieves may have to share a single bead. Nevertheless, by giving each bead to one of the thieves $1 \le j \le b$ that have some share of it, (a) we do not increase the number of cuts, and, (b) we convert the splitting into a discrete splitting. Let us show how this can be done so (c) (almost) fairness holds as well: each thief gets either $\lfloor a_i/b \rfloor$ or $\lceil a_i/b \rceil$ beads of color i, for every $1 \le i \le t$.

For every $1 \le i \le t$, construct a flow network in which each edge has a capacity and a lower bound on the flow through this edge.

Define a directed graph G_i on the set of all a_i beads of color i, all b thieves, and two distinguished vertices: source v_s and sink v_t,

$$V_i = \{\, b_l \mid 1 \le l \le a_i \} \cup \{\, h_i \mid 1 \le j \le b \} \cup \{ v_s, v_t \}$$

$$E_i = \{\, (b_l, h_j) \mid \text{thief } h_j \text{ gets a share of bead } b_l \} \cup \{\, (v_s, b_l) \mid 1 \le l \le a_i \} \cup \{\, (h_j, v_t) \mid 1 \le j \le b \}$$

For every edge (v_s, b_l), let its capacity and lower-bound be $c(v_s, b_l) = l(v_s, b_l) = 1$. For every edge (b_l, h_j), let its capacity be $c(b_l, h_j) = 1$, and its lower-bound $l(b_l, h_j) = 0$. For every edge (h_j, v_t), let its capacity be $c(h_j, v_t) = \lceil a_i/b \rceil$, and let its lower-bound be $l(h_j, v_t) = \lfloor a_i/b \rfloor$. Note that the continuous theorem yields a non-integral legal flow in the network. As all capacities and lower-bounds are integral, there exists an integral flow in the network (folklore; this follows, for example, from the validity of Lawler's algorithm, cf., e.g, [23], page 602). Such a flow in each of the G_i's corresponds to a discrete, almost-fair, split. ∎

Note that the proof of this theorem, as a result of the underlying proof of the continuous version, is not constructive. Nevertheless, this is not a drawback in our context, as we invoke it on very small parameters, and consider all possible splits.

5.2 Multi-Way Splitters

A *partition* of m coordinates into b blocks is a function $\pi \colon [m] \to [b]$ assigning each coordinate the block that contains it. We say π splits a subset $I \subseteq [m]$ of the coordinates, if every block $1 \le j \le b$ sees the same number of coordinates up to rounding, i.e

$$\left\lfloor \frac{|I|}{b} \right\rfloor \le \left| \pi^{-1}(j) \cap I \right| \le \left\lceil \frac{|I|}{b} \right\rceil$$

To facilitate the handling of partitions, for a partition $\pi : [m] \to [b]$, let us define $\bar{\pi} \colon [m] \to [b] \times [m]$ assigning a coordinate its (block,ordinal within the block) pair, i.e

$$\bar{\pi}(i) = (\pi(i), |\{\, j \le i \mid \pi(j) = \pi(i) \}|)$$

[18] introduced the notion of (m,k,b)-splitters, referring to list of partitions such that every set of k coordinates is split. To use the analogy of the Necklace Splitting Theorem, say the necklace is composed of m seemingly identical jewels, but only k of them are real gemstones.

[18] wanted a small set of possible partitions of the necklace to b thieves, so that no matter which k jewels are real, there is a partition that splits these k jewels fairly between the thieves. We introduce a generalization of this notion, namely *multi-way* splitters. The only difference is that now jewels may be of various types, and a fair partition is one that gives each thief the same number of jewels of each type.

Definition 6 (multi-way splitter)**.** *Let m, b be natural numbers.*

- *A* t-way splitter *is a list A of partitions $\pi\colon [m] \to [b]$, s.t. for any series of t pairwise disjoint subsets of coordinates $S_1, \ldots, S_t \subseteq \{1, \ldots, m\}$, there exists a partition in A that splits each one of $S_1, \ldots, S_t$.*
- *A* k-wise t-way splitter *is a list A of partitions $\pi\colon [m] \to [b]$, s.t. for any series of t pairwise disjoint subsets $S_1, \ldots, S_t \subseteq \{1, \ldots, m\}$ containing at most k indices $\sum |S_i| \leq k$, there exists a partition in A that splits each one of $S_1, \ldots, S_t$.*

When there is only one type, i.e $t = 1$, our definition of a t-way splitter is equivalent to the [18] definition of a splitter. When $t > 1$, we need to use the Necklace Splitting Theorem.

Lemma 8. *There exists an explicit construction of a t-way splitter that splits m indices into b blocks, and has size $b^{p+1} \cdot \binom{m}{p}$, where $p \doteq (b-1)t$.*

Proof. Take all possible splits $\pi : [m] \to [b]$ of size p of a necklace with m beads of t types to b thieves. That is, for every $0 = i_0 < i_1 < \cdots < i_p \leq i_{p+1} = m$ positions, and every $s \in [b]^{p+1}$ determining which thief gets each piece, define a split π as follows: $\pi(j) = s(k)$ for every $i_{k-1} < j \leq i_k$.

To prove correctness, take t pairwise disjoint subsets $S_1, \ldots, S_t \subseteq \{1, \ldots, m\}$. They correspond to a necklace with $m' \doteq \left|\bigcup_{i=1}^{t} S_i\right| \leq m$ beads of t different types, determined by $S_1, \ldots, S_t$. As every split of a necklace of size m' is induced by at least one split of a necklace of size m, by lemma 7, there exists a split out of those we constructed, that splits evenly the necklace we consider, and thus satisifes our demand for $S_1, \ldots, S_t$. ∎

We are interested in constructions of (smaller) k-wise t-way splitters. This is a *k-restriction* problem over an alphabet of size b with demands that are invariant under permutations; There is a demand for every classification of k coordinates into t types, containing all partitions that split all types. Now, applying concatenation,

Theorem 3. *A k-wise t-way splitter for splitting m coordinates into b blocks of size $\left|Hash_{m,k^2,k}\right| \cdot b^{p+1} \cdot \binom{k^2}{p}$, $p = (b-1)t$, may be constructed in time polynomial in m, t^k and the size of the construction.*

Proof. Apply lemma 5 (concatenation) on lemma 8 (the exhaustive construction). ∎

6 Application to Group-Testing

In this section we will prove theorem 4 presented in the introduction.

Theorem 4. *For any fixed d, for every $\delta > 0$, a set of at most $(1+\delta) \cdot (d+1)e \cdot [(d+1)\ln m + \ln(d+1)]$ group tests may be found in time polynomial in m^d.*

Proof. Our plan is as follows:

1. Formulate the group-testing problem as a k-restriction problem.
2. Present a product distribution that induces a large density for the k-restriction problem.
3. Derive the conclusion from theorem 1.

Let us formulate the group-testing problem as a k-restriction problem. A test may be represented by a string in Σ^m, $\Sigma = \{0,1\}$, indicating which of the m blood samples to take and which not. The combinatorial requirement is that no matter who are the d carriers, for every person, there should be a test that examines his blood, and none of the d infected blood samples. In other words, for every string $w \in \Sigma^k$, $k = d+1$, with d zeros and a single one, there should be a demand $f_w : \Sigma^k \to \{0,1\}$, $f_w(x) = 1$ *iff* $x = w$ (hence $s = d+1$).

Consider the product distribution $D : \{0,1\}^m \to [0,1]$, so that each group-test $t \in \{0,1\}^m$ is chosen as follows: for every soldier $1 \le i \le m$, independently at random, let

$$t(i) = \begin{cases} 1 & \text{with probability } \frac{1}{d+1} \\ \\ 0 & \text{with probability } \frac{d}{d+1} \end{cases}$$

For any k indices $1 \le i_1 < \cdots < i_k \le m$ and a string $w \in \{0,1\}^k$ with d zeroes and a single one, the probability a group-test drawn randomly from D has w in positions $i_1, \ldots, i_k$ is

$$\frac{1}{d+1} \cdot \left(1 - \frac{1}{d+1}\right)^d \ge \frac{1}{e(d+1)}$$

Fix some $0 < \delta < 1$, and let $\delta' < \frac{1}{1-\delta}$. By theorem 1 and lemma 2, one can obtain a solution of size at most $(1+\delta) \cdot e(d+1)[(d+1)\ln m + \ln(d+1)]$ in time polynomial in m^d. ■

7 Application to Generalized Hashing

In this section we will prove theorems 5 and 6 concerning (t,u)-hashing, as presented in the introduction. First, let us formulate the problem as a k-restriction problem.

Claim 1. *The problem of finding a (t,u)-hash family is a k-restriction problem, for $k = u$.*

Proof. The underlying observation is that functions $h : [m] \to [q]$ may be thought of as strings $s \in \Sigma^m$, for $\Sigma = [q]$; for every $x \in [m]$, $h(x) = s(x)$. As to the restrictions, $k = u$, and for every $T \subseteq [u]$, $|T| = t$, there is a demand $f_T : \Sigma^u \to \{0,1\}$, $f_T(x) = 1$ *iff* for every $i \in T$ and $j \in [u] \setminus \{i\}$, $x(i) \ne x(j)$. Here $s = \binom{u}{t}$. ■

As aforementioned, [3] consider the minimal alphabet size, $q = t+1$, and observe the product distribution $D : [t+1]^m \to [0,1]$, for which a hash function $h : [m] \to [t+1]$ is chosen as follows: for every $x \in [m]$, independently at random,

$$h(x) = \begin{cases} t+1 & \text{with probability } 1 - \frac{t}{u} \\ 1 & \text{with probability } \frac{1}{u} \\ \vdots & \vdots \\ t & \text{with probability } \frac{1}{u} \end{cases}$$

The density of the problem with respect to this distribution is at least $t!(\frac{1}{u})^u(1-\frac{t}{u})^{u-t} = \frac{t!(u-t)^{u-t}}{u^u}$, as for every $T \subset U \subseteq [m]$, $|T| = t$, $|U| = u$, a function $h : [m] \to [t+1]$ satisfies the restriction T at place U, if (but not necessarily *only if*) $h(U-T) = \{t+1\}$, and $h(T) = \{1, \ldots, t\}$.

Note that this argument works even for a *restricted* (t, u)-hash problem, in which the demand associated with T is $f_T(x) = 1$ *iff* for every $i \in T$ and $j \in [u] \setminus \{i\}$, *either* $j \in U - T$ and then $x(j) = t+1$, as well as, $x(i) \le t$, *or* $j \in T$, and then $x(i) \neq x(j)$, as well as, $x(i), x(j) \le t$.

Now, we simply need to substitute the parameters in theorem 1:

Theorem 5. *For any fixed* $2 \le t < u$*, for any* $\delta > 0$*, one can construct efficiently a* (t, u)*-hash family over an alphabet of size* $t+1$ *whose rate is at least* $(1-\delta)\frac{t!(u-t)^{u-t}}{u^{u+1}\ln(t+1)}$.

Proof. Fix t, u. Let $\delta > 0$. By theorem 1, there is an algorithm, that given m and δ', outputs, in time polynomial in m, a (t, u)-hash family of size

$$\frac{(u \ln m + \ln \binom{u}{t})u^u}{(1-\delta')t!(u-t)^{u-t}}$$

The rate of this construction is, hence:

$$R \doteq \frac{\ln m \cdot (1-\delta')t!(u-t)^{u-t}}{\ln(t+1) \cdot (u \ln m + \ln \binom{u}{t})u^u}$$

If we choose δ', so that $\frac{1-\delta'}{u \ln m + \ln \binom{u}{t}} \ge \frac{1-\delta}{u \ln m}$,

$$R \ge \frac{(1-\delta)t!(u-t)^{u-t}}{\ln(t+1)u^{u+1}}$$

■

This rate resembles that of the probabilistic construction of [3], $\frac{t!(u-t)^{u-t}}{u^u(u-1)ln(t+1)} - \delta$, and improves their explicit construction whose rate is u^{-ct} for a large constant c. Importantly, our methods and the use of the *multi-way splitters* we define, provide small constructions even when u and t grow with the parameters of the problem. To ease the presentation, let us concentrate on the case $u = 2a$ and $t = a$, for some input parameter a.

Consider the following algorithm:

Input: a, m

Output: an $(a, 2a)$-hash family of functions $[m] \to [a+1]$.

1. Construct a $2a$-wise 2-way splitter S that splits m coordinates into $b = \lceil \log 2a \rceil$ blocks applying theorem 2.

2. Construct a *restricted* $(\lceil a/b \rceil, \lceil 2a/b \rceil)$-hash family H (as defined above) of functions $h_i : [m] \to [\lceil a/b \rceil + 1]$.

3. For every splitter partition $\pi \in S$, for every b (not necessarily different) strings, $h_1, \ldots, h_b \in H$, output a function $h_{\pi, h_1, \ldots, h_b}$ composed of $h_1, \ldots, h_b$ interleaved according to π: for every $x \in [m]$, if x is assigned to the k'th block by π, i.e., $\pi(x) = k$, then

$$h_{\pi, h_1, \ldots, h_b}(x) = \begin{cases} a+1 & h_k(x) = \lceil a/b \rceil + 1 \\ \\ h_k(x) + (k-1)\lceil a/b \rceil & \textit{otherwise} \end{cases}$$

Theorem 6. *There exists an algorithm that given a and m, outputs an $(a, 2a)$-hash family of at most $(4e)^a \log^{O(\log a)} m$ functions $[m] \to [a+1]$ in time polynomial in m and this size.*

Proof. Clearly, the algorithm produces a family of size $|S|\,|H|^b$. By corollary 3, $|S| \le a^{O(\log a)} \log m$. By theorem 2, $|H| \le 2^{O(a/b)} \log m$, since, the density of the problem solved is at least $\frac{c!c^c}{(2c)^{2c}}$, for $c = \lceil a/b \rceil$, and using Stirling's approximation, $\varepsilon^{-1} \le 2^{2c} e^c$. Therefore, the algorithm outputs a family of size $2^{2a} e^a \log^{O(\log a)} m$. Also note that the running-time of the algorithm is indeed as specified.

Let us prove correctness. Let $T \subset U \subseteq [m]$, $|T| = a$, $|U| = 2a$. By definition of $2a$-wise 2-way splitter, there exists a partition $\pi : [m] \to [b]$ that splits T and $U - T$. For every $1 \le k \le b$, let $T_k = T \cap \pi^{-1}(k)$, $U_k = U \cap \pi^{-1}(k)$, $T_k \subset U_k$, $|T_k| \le \lceil a/b \rceil$, $|U_k| \le \lceil 2a/b \rceil$. By definition of a $(\lceil a/b \rceil, \lceil 2a/b \rceil)$-hash family (together with the observation that a (t, u)-hash family is also a (t', u')-hash family for every $t' \le t$, $u' \le u$), for every $1 \le k \le b$, there exists $h_k \in H$ for which $h_k(U_k - T_k) = \{\lceil a/b \rceil + 1\}$ and $h_k(T_k) = \{1, \ldots, \lceil a/b \rceil\}$. Consider $h \doteq h_{\pi, h_1, \ldots, h_b}$. Let $x_1 \in T$, $x_2 \in U$, $x_1 \ne x_2$.

- If $\pi(x_1) \ne \pi(x_2)$, then necessarily $h(x_1) \ne h(x_2)$.
- Otherwise, let $k = \pi(x_1) = \pi(x_2)$. $x_1 \in T_k$, $x_2 \in U_k$. Thus, $h_k(x_1) \ne h_k(x_2)$.

In any case, the restriction is satisfied. ■

8 Application to Set-Cover

In this section we will prove theorem 7 presented in the introduction, and show an improved inapproximability result for Set-Cover under the assumption $\mathcal{P} \ne \mathcal{NP}$.

8.1 How to Prove That Set-Cover is Hard to Approximate?

To prove that Set-Cover cannot be efficiently approximated to within factor α unless $\mathcal{P} = \mathcal{NP}$, we prove that there exists t such that the following decision problem (a.k.a "the Set-Cover gap problem") is $\mathcal{NP}$-hard:

- Input: A Set-Cover instance $(\mathbb{U}, \mathbb{S})$; $\mathbb{U} = \{u_1, ..., u_n\}$ is a universe and $\mathbb{S} = \{S_1, ..., S_m\} \subseteq P(\mathbb{U})$ is a family of subsets, $\bigcup_{S_j \in \mathbb{S}} S_j = \mathbb{U}$.
- Problem: distinguish between the following two cases:
    1. There exists a small set-cover: there exists $\mathcal{C} \subseteq \mathbb{S}$, $\bigcup_{S \in \mathcal{C}} S = \mathbb{U}$ with $|\mathcal{C}| \le t$.
    2. Every set-cover is large: every $\mathcal{C} \subseteq \mathbb{S}$, $\bigcup_{S \in \mathcal{C}} S = \mathbb{U}$, satisfies $|\mathcal{C}| > \alpha t$.

If there were an efficient algorithm approximating Set-Cover to within α, there would have also been an efficient algorithm solving the gap problem for any t. Thus, proving the Set-Cover gap problem is $\mathcal{NP}$-hard for some t indeed suffices.

To prove that the problem is $\mathcal{NP}$-hard, we find another gap problem that is known to be $\mathcal{NP}$-hard, and reduce it to the Set-Cover gap problem. Specifically, PCP theorems provide gap problems that are $\mathcal{NP}$-hard. Let us describe the general structure of the problem we use. The input is a set of tests Φ over variables from a set X (e.g., quadratic equations). Each test depends on d variables (where d is some constant). $\mathbb{F}$ denotes the range of the variables. The problem is to distinguish between the case that there exists an assignment to the variables such that all tests are satisfied and the case that no assignment satisfies more than $\frac{2}{|\mathbb{F}|}$ of the tests (i.e., the tests are *far* from being satisfiable).

To show a reduction we need to find a way to "encode" the tests and variables by elements and subsets, such that a small cover would correspond to a satisfying assignment.

The general idea is as follows:

- For every variable $x \in X$ and possible assignment to it $a \in \mathbb{F}$, construct a set $S(x, a)$. Including $S(x, a)$ in the set-cover would correspond to assigning a to x.
- For every test $\varphi \in \Phi$, construct a sub-universe U_φ. Covering the sub-universe would correspond to satisfying the test.
- If a variable $x \in X$ appears in a test $\varphi \in \Phi$, each of its variable-sets $S(x, a)$ (for $a \in \mathbb{F}$) would cover some portion of the sub-universe U_φ.

Each sub-universe U_φ together with the portions covered within it is itself a Set-Cover instance. The challenge is to design each instance U_φ such that covers that do not correspond to satisfying assignments for φ would be large.

8.2 Using Universal Sets

Let us demonstrate how to design a sub-universe U_φ for a test $\varphi \in \Phi$ via *universal sets* [16, 8]. Our construction extends this idea.

An (m, k)*-universal set* is a set of binary strings in $\{0,1\}^m$, such that the restriction to any k indices contains all possible binary configurations. More formally, this is a solution to a k-restriction problem with $\Sigma = \{0, 1\}$, and a demand f_w for every $w \in \{0,1\}^k$ accepting only it. I.e., $f_w(x) = 1$ *iff* $x = w$ (hence, $s = 2^k$).

One can view an (m, k)-universal set B as a Set-Cover instance, where the set of elements is B and there are $2m$ subsets, indexed C_i^b, for $1 \le i \le m$, $b \in \{0, 1\}$. These is because every binary string $a \in \{0,1\}^m$ can be thought of as indicating to which of the $2m$ subsets it belongs: for every $1 \le i \le m$, it belongs to $C_i^{a(i)}$. This Set-Cover instance has two important properties:

1. *(legal covers are small)* For every $1 \le i \le m$, $\left\{C_i^0, C_i^1\right\}$ is a cover with only 2 sets (in other words, C_i^1 is the complement of C_i^0).
2. *(illegal covers are large)* Every cover of size at most k necessarily contains both C_i^0 and C_i^1 for some $1 \le i \le m$.

To see why the second property holds, note that every collection of the form $\left\{C_{i_l}^{b_l}\right\}_{l=1}^{k}$ has an element it misses: the element that has $1 - b_l$ in position i_l for every $1 \le l \le k$.

Clearly, the size of any (m, k)-universal set is at least 2^k. A probabilistic argument shows that there are (m, k)-universal sets of size $O(2^k k \ln m)$. This follows since the density of the k-restriction problem with respect to the uniform distribution is 2^{-k}. Moreover, there is a way to efficiently compute, given m, an (m, k)-universal set B such that $k \ge \log |B| (1 - o(1))$ [18].

Now we can describe how to construct a sub-universe. Fix a test $\varphi \in \Phi$ over variables $x_{i_1}, \ldots, x_{i_d} \in X$. Set $m = d |\mathbb{F}|$. Let U_φ be an (m, k) universal set where $k \ge \log |U_\varphi| (1 - o(1))$. Index the subsets $C_1^0, \ldots, C_m^0$ by $\{C_{j,a}\}$ for $j = 1, \ldots, d$, $a \in \mathbb{F}$.

- For every $1 \leq j \leq d$, $a \in \mathbb{F}$, the *variable-set* $S(x_{i_j}, a)$ covers $C_{j,a}$.
- For every *satisfying* assignment $a_1, \ldots, a_d \in \mathbb{F}$ for φ's variables, an extra *test-set* covers what the subsets corresponding to $a_1, \ldots, a_d$ do not cover, i.e., covers $(C_{1,a_1} \cup \cdots \cup C_{d,a_d})^c = C^c_{1,a_1} \cap \cdots \cap C^c_{d,a_d}$.

Clearly, for every satisfying assignment $a_1, \ldots, a_d \in \mathbb{F}$ for φ's variables, U_φ can be covered by the d variable-sets $S(x_{i_1}, a_1), \ldots, S(x_{i_d}, a_d)$ together with the extra test-set corresponding to $a_1, \ldots, a_d$. Moreover, by the property of the universal set, every cover with at most k subsets must contain $S(x_{i_1}, a_1), \ldots, S(x_{i_d}, a_d)$ for some satisfying assignment $a_1, \ldots, a_d$ for φ. The gap we get, i.e., the ratio between the size of the cover in the two cases, is $k/(d+1) \geq \frac{\log |U_\varphi|}{d+1}(1-o(1))$. Recall that ideally we would have liked the gap to be $\ln |U_\varphi| \, (1 - o(1))$.

8.3 How Do We Improve?

We note that in the above construction there is an asymmetry between variable-sets and test-sets. Variable-sets cover portions of many sub-universes: one for each test that depends on the variable, while test-sets only participate in the covering of a sub-universe of their test. Hence, if an adversary is to cover the construction composed of all sub-universes, taking variable-sets is worthier than taking test-sets.

This asymmetry can be used to design Set-Cover gadgets that disallow *larger* illegal covers, and, hence, give a better gap. Let us explain the idea by considering a probabilistic construction.

Consider some probability $p \in (0,1)$ (e.g., $p = \frac{1}{d}$). Pick m random subsets of a set B by letting each subset contain each possible element independently at random with probability p. Consider any collection of s subsets and c complements of subsets. The probability that the collection does not cover a specific element $b \in B$ is at least $(1-p)^s p^c \geq e^{-(sp(1+p) - c \ln p)}$. This probability is larger than $\frac{1}{|B|}$ even when $s \approx \frac{1}{p} \ln |B|$ as long as c is relatively small. Hence, we expect some element in B not to be covered by such a collection.

Define $f_p(s,c) \doteq sp(1+p) - c \ln p$.

Definition 7 (Set-Cover consistency gadget)**.** *Given m, l, and d, a* Set-Cover *consistency gadget is given by $(B, \{C_1, \ldots, C_m\})$ with the following property: for every cover $\mathcal{C}$ for B that contains*

- *s subsets of the form C_i for some $1 \leq i \leq m$.*
- *c subsets of the form $(C_{i_1} \cup \cdots \cup C_{i_d})^c$ for some $1 \leq i_1 < \cdots < i_d \leq m$.*

where $f_p(s,c) \leq l$, there necessarily exist $1 \leq i_1 < \cdots < i_d \leq m$ such that

$$C_{i_1}, \ldots, C_{i_d}, (C_{i_1} \cup \cdots \cup C_{i_d})^c \in \mathcal{C}$$

8.4 Constructing The New Gadget

Note that the problem of finding a Set-Cover consistency gadget can be viewed as a k-restriction problem for $k \leq \lceil l/p \rceil$. For every collection of s subsets and c complements, $f_p(s,c) \leq l$, not containing d subsets and their complement, we have a demand. The demand requires some element not to be covered by the collection (this is very similar to what was described in subsection 8.2 for universal sets).

Theorem 2 provides an efficient algorithmic construction as long as $k = O(\log m / \log\log m)$. We use the Divide and Conquer approach proposed in section 5 to find efficiently a construction for k which is logarithmic in m.
Input: parameters p, m, l.
Output: a SET-COVER consistency gadget.

1. Construct a k-wise 2-way splitter that splits m coordinates into $b \doteq \lceil \log l \rceil$ blocks.
2. Find a solution R for the re-scaled problem, with parameters p, m and $\lceil \frac{l}{b} - lnp \rceil$.
3. For every splitter partition π, for every b strings, $r_1, \ldots, r_b \in R$, output a string x composed of $r_1, \ldots, r_b$ interleaved according to π: for $1 \le i \le m$, $\bar{\pi}(i) = (k, j) \rightarrow x(i) = r_k(j)$.

The algorithm gives us the following

Lemma 9 (SET-COVER gadget)**.** *Fix some constant $p \in (0, 1)$. Given parameters m and l, a* SET-COVER *consistency gadget, whose universe is of size $e^l l^{O(\log l)} \log^{O(\log l)} m$, may be constructed in time polynomial in m and this size.*

Proof. The size of the construction is bounded by $|S|\,|R|^b$, where $|S| \le k^{O(\log l)} \log m$ bounds the size of the splitter and $|R| \le e^{l/b} k^{O(1)} \log m$ bounds the size of the solution for the re-scaled problem. The running-time of the algorithm is linear in the running-time of the splitter construction, the running-time of the algorithm of lemma 2 on the smaller instance and the size of the construction. Hence, it is polynomial in m and e^l. It remains to prove correctness. Fix some position and demand, i.e fix some function $v\colon [m] \rightarrow \{0, 1, *\}$ assigning each coordinate either the binary symbol expected there, or $*$, which means no constraints attached. By definition, $f_p(|v^{-1}(0)|, |v^{-1}(1)|) \le l$. By the property of the splitter, there exists a partition π that splits both $v^{-1}(0)$ and $v^{-1}(1)$. Hence, for every $1 \le j \le b$,

$$\begin{aligned} f_p\left(\left|\pi^{-1}(j) \cap v^{-1}(0)\right|, \left|\pi^{-1}(j) \cap v^{-1}(1)\right|\right) &\le -lnp\left\lceil \frac{|v^{-1}(0)|}{b} \right\rceil + p(1+p)\left\lceil \frac{|v^{-1}(1)|}{b} \right\rceil \\ &\le \frac{l}{b} - lnp \qquad (1) \end{aligned}$$

For every $1 \le j \le b$, define $v_j : [m] \rightarrow \{0, 1, *\}$ to represent the appropriate restriction according to the j'th block,

$$v_j(i) \doteq \begin{cases} v(z) & \exists z, \bar{\pi}(z) = (j, i) \\ * & otherwise \end{cases}$$

Note that calculation 1 shows that all v_j's indeed represent restrictions in the problem on parameters m, $\lceil \frac{l}{b} - \ln p \rceil$. Hence in any solution for the smaller problem each v_j appears in some string x_{k_j}. Thus, when combining $x_{k_1}, \ldots, x_{k_b}$ according to π, one gets a string that satisfies restriction v. ■

8.5 Improved Hardness Result for Set-Cover

In this subsection we prove theorem 7. We present a reduction from the *PCP* of [10] to a SET-COVER gap problem. The reduction generalizes that of [16, 8].

A few remarks:

- The reduction can be used with any PCP with poly-logarithmic small error-probability which has some constant dependency d. It does not rely on any additional property of the [10] PCP.
- We can get the same result of theorem 7 under the slightly stronger assumption $\mathcal{NP} \not\subseteq \mathcal{ZPP}$ without using the combinatorial tools presented in this paper. They allow us to relax the assumption to $\mathcal{P} \neq \mathcal{NP}$.

Preliminaries - Probabilistically Checkable Proofs

According to the traditional definition [9], $\mathcal{NP}$ is the class of all languages having a polynomially checkable membership proof. Rather surprisingly, $\mathcal{NP}$ may be equivalently characterized as the class of all languages having a *Probabilistically Checkable Proof* (PCP) [6]. Such proofs may be checked, with good probability of being correct, by inspecting only a constant number of their components, chosen at random. A PCP characterization of $\mathcal{NP}$ is derived by showing some sort of PCP *problem* – described shortly – is $\mathcal{NP}$-hard.

A PCP instance is a triplet $(X, \mathbb{F}, \Phi)$. $X = \{x_1, ..., x_n\}$ is a set of formal variables that range over a domain $\mathbb{F}$. $\Phi = \{\varphi_1, ..., \varphi_m\}$ is a collection of *local-tests*, $m = poly(n)$. Each local test is a Boolean function on $d = O(1)$ variables. Φ is *uniform*, i.e., each variable appears in the same number of tests.

The following PCP theorem was shown by [10]:

Lemma 10 (The PCP theorem of [10])**.** *For every $\varepsilon > 0$, given a PCP instance $(X, \mathbb{F}, \Phi)$, with $|\mathbb{F}| \leq 2^{log^{1-\varepsilon} n}$, it is $\mathcal{NP}$-hard to distinguish between the case in which there exists an assignment $A : X \to \mathbb{F}$ that satisfies all tests in Φ, and the case in which every such assignment $A : X \to \mathbb{F}$ satisfies at most $2\,|\mathbb{F}|^{-1}$ of the tests in Φ.*

In the context of Set-Cover, a different variant of the PCP theorem is more suitable. Instead of considering the maximal number of tests that can be satisfied simultaneously, we consider the minimal number of assignments per variable that is needed in order to satisfy many tests. More accurately, we consider multi-assignments $\hat{A}\colon X \to P(\mathbb{F})$, assigning each variable a set of values from $\mathbb{F}$. We say $\hat{A}$ satisfies a test φ over variables $x_{i_1}, \ldots, x_{i_d}$, if there exist $a_1 \in \hat{A}(x_{i_1}), \ldots, a_d \in \hat{A}(x_{i_d})$ such that $\varphi(a_1, \ldots, a_d) = true$. A multi-assignment $\hat{A}$ is said to be (t, γ)-*good*, if there exists a considerable fraction of tests $\Phi' \subseteq \Phi$, $|\Phi'| \geq \gamma\,|\Phi|$, so that for every $\varphi \in \Phi'$, $\hat{A}$ satisfies φ and the variables φ depends on, $x_{i_1}, \ldots, x_{i_d}$, have few assignments, $0 < |\hat{A}(x_{i_j})| \leq t$. The following follows from [10] using a simple probabilistic argument:

Lemma 11 (Multi-Assignment PCP)**.** *For any $0 < \beta, \gamma < 1$, given a PCP instance $(X, \mathbb{F}, \Phi)$ with $|\mathbb{F}| \leq 2^{log^{1-\beta} n}$, it is $\mathcal{NP}$-hard to distinguish between the case there exists a multi-assignment which is $(1,1)$-good, and the case every multi-assignment is* not *(t, γ)-good, as long as $\gamma > 2t^d / |\mathbb{F}|$.*

Proof. By reduction from the traditional unique-assignment PCP of lemma 10.

(i) If a PCP instance $(X, \mathbb{F}, \Phi)$ is satisfied by a unique-assignment $A\colon X \to \mathbb{F}$, than $\hat{A}\colon X \to P(\mathbb{F})$ defined by $\hat{A}(x) = \{A(x)\}$ is $(1,1)$-good.

(ii) Assume that for a PCP instance $(X, \mathbb{F}, \Phi)$ there exists a multi-assignment $\hat{A}\colon X \to P(\mathbb{F})$ which is (t, γ)-good for the parameters t, γ stated above. Let us show there exists a unique-assignment $A\colon X \to \mathbb{F}$ which satisfies more than $\hat{\varepsilon} = \gamma t^{-d}$ of the tests in Φ. Note that $\hat{\varepsilon}$ is larger than the error-probability of our PCP. The existence of such an A will be proven by the probabilistic method. Assign each variable $x \in X$ a value chosen uniformly at random from $\hat{A}(x)$. By definition, γ of the tests are satisfied and all their variables x have $\left|\hat{A}(x)\right| \leq t$. Such tests are satisfied by A with probability at least t^{-d}. Altogether, by the linearity of expectations, the expected fraction of tests satisfied is γ/t^d. Thus there exists a unique-assignment which satisfies that many tests. By [10], distinguishing between the two aforementioned cases is $\mathcal{NP}$-hard. ■

The Construction

Fix a PCP instance $(X, \mathbb{F}, \Phi)$ as in lemma 11, and let us construct a corresponding Set-Cover instance $(\mathbb{U}, \mathbb{S})$.

The construction is along the lines of the description in subsections 8.1 and 8.2. There is one addition though: the described construction is duplicated D times for $D = \Theta(|\Phi| / |X|)$. Each duplicate has its own variable-sets and sub-universes, but the test-sets are common to all duplicates. This way all subsets (variable-sets and test-sets) participate in covering $\Theta(|\Phi| / |X|)$ sub-universes, and the differences are only in the constants (Recall that by the uniformity of the PCP instance, each variable appears in $d |\Phi| / |X|$ tests).

Fix an arbitrary $\eta \geq 1$ and let $D = \left\lfloor \frac{|\Phi|}{\eta |X|} \right\rfloor$. Set $m = d \cdot |\mathbb{F}|$ and $p = 1/\eta d$. Denote the Set-Cover instance of lemma 9 above for p, m and l, which will be defined at will, by $(B, \{C_1, ..., C_m\})$. We associate every index $1 \leq j \leq d$ and element of the field $a \in \mathbb{F}$ with one of the $m = d \cdot |\mathbb{F}|$ sets, denoted $C_{j,a}$. Construct a Set-Cover instance $(\mathbb{U}, \mathbb{S})$ as follows. Let the universe be $\mathbb{U} \doteq [D] \times \Phi \times B$. For each index $1 \leq i \leq D$ let $\mathbb{U}_i \doteq \{i\} \times \Phi \times B$ denote the i-th duplicate. For each duplicate $1 \leq i \leq D$, a variable $x \in X$ and an assignment to that variable $a \in \mathbb{F}$, add a *variable-set* covering the appropriate portion of every relevant sub-universe,

$$S(i, x, a) = \bigcup_{x = \varphi[j]} \{i\} \times \{\varphi\} \times C_{j,a}$$

For each test $\varphi \in \Phi$ and a *satisfying* assignment to its variables $a_1, ..., a_d \in \mathbb{F}$, add a *test-set* covering the appropriate complements in all duplicates,

$$S(\varphi, a_1, ..., a_d) = \bigcup_{1 \leq i \leq D} \{i\} \times \{\varphi\} \times (C_{1,a_1} \cup ... \cup C_{d,a_d})^c$$

Correctness

Claim 2 (Completeness)**.** *If the tests system Φ has a $(1,1)$-good multi-assignment ρ, then* $(\mathbb{U}, \mathbb{S})$ *has a set cover of cardinality* $\left(1 + \frac{1}{\eta}\right) |\Phi|$.

Proof. Let $\mathcal{C}$ contain sets of two types: variable-sets and test-sets,

$$\mathcal{C} \doteq \bigcup_{1 \leq i \leq D, x \in X} S(i, x, \rho(x)) \cup \bigcup_{\varphi(x_{i_1}, ..., x_{i_d}) \in \Phi} S(\varphi, \rho(x_{i_1}), ..., \rho(x_{i_d}))$$

Note that $\mathcal{C}$ is a legal cover of the specified size. ■

Now let us proceed to prove soundness. We will need the following lemma:

Lemma 12. *If for some $1 \leq i \leq D$, $\mathbb{U}_i$ can be covered by at most $\delta l |\Phi|$ test-sets and $\eta \delta l |X|$ variable-sets, then Φ has a $(\eta d l, 1 - \delta c)$-good multi-assignment for Φ, where $c \doteq \ln \eta d + 1 + 1/\eta d$.*

Proof. Observe only $\mathbb{U}_i$, and let $\mathcal{C}$ indicate the given cover. For every $\varphi \in \Phi$, denote by V_φ the collection of variable sets that participate in covering it, and denote by T_φ the collection of such tests sets. Let $\mathcal{L}$ denote the tests $\varphi \in \Phi$ with low cost, $\mathcal{L} \doteq \{\varphi \in \Phi \mid f_p(|T_\varphi|, |V_\varphi|) \leq l\}$. Observe the multi-assignment $\hat{A}\colon X \to P(\mathbb{F})$ assigning each $x \in X$, $\hat{A}(x) = \{a \in \mathbb{F} \mid S(i,x,a) \in \mathcal{C}\}$. According to lemma 9, for every $\varphi(x_{i_1}, ..., x_{i_d}) \in \mathcal{L}$, there must exist $a_1, ..., a_d \in \mathbb{F}$, such that $\varphi(a_1, ..., a_d) = \mathit{true}$ and for every $1 \leq j \leq d$, $S(i, x_{i_j}, a_j) \in \mathcal{C}$. Therefore, for every $1 \leq j \leq d$, $0 < |\hat{A}(x_{i_j})| \leq \eta dl$. $\mathcal{L}$ contains a significant portion of the tests, $|\mathcal{L}| \geq |\Phi| \cdot (1 - \delta c)$, as $\sum_{\varphi \in \Phi - \mathcal{L}} f_p(|T_\varphi|, |V_\varphi|) \geq (|\Phi| - |\mathcal{L}|) \cdot l$, and

$$\begin{aligned}
\sum_{\varphi\in\Phi} f_p(|T_\varphi|, |V_\varphi|) &= \sum_{\varphi\in\Phi}\left(\frac{1}{\eta d}(1+1/\eta d)|V_\varphi| + \ln \eta d\, |T_\varphi|\right) \\
&= \frac{1}{\eta d}(1+1/\eta d)\sum_{\varphi\in\Phi}|V_\varphi| + \ln\eta d \sum_{\varphi\in\Phi}|T_\Phi| \\
&< \frac{1}{\eta d}(1+1/\eta d)\cdot\frac{d\,|\Phi|}{|X|}\eta\delta l\,|X| + \ln\eta d\cdot\delta l\,|\Phi| \\
&= \delta l\,|\Phi|\left(1+\frac{1}{\eta d}+\ln\eta d\right)
\end{aligned}$$

Hence, the multi-assignment $\hat{A}$ is $(\eta dl, 1-\delta c)$-good. ∎

Claim 3 (Soundness). *If no assignment for Φ is $(\eta dl, \gamma)$-good, then the cardinality of the minimal set cover of $(\mathbb{U}, \mathbb{S})$ is at least $\frac{l}{c}|\Phi| \cdot (1-\gamma)$, where $c \doteq \ln \eta d + 1 + 1/\eta d$.*

Proof. Assume, by way of contradiction, that our Set-Cover instance may be covered by less than $\frac{l}{c}|\Phi|\cdot(1-\gamma)$ sets. By averaging, there exists a duplicate $1 \leq i \leq D$ whose universe $\mathbb{U}_i$ is covered by less than $\frac{\eta l}{c}|X|\cdot(1-\gamma)$ variable-sets and $\frac{l}{c}|\Phi|\cdot(1-\gamma)$ test-sets. Lemma 12 implies there exists a multi-assignment for Φ which is $(\eta dl, \gamma)$-good. This contradicts the premise of this claim. ∎

Now we set the parameter l of the construction as to optimize the result and prove our improved hardness of approximation result for Set-Cover:

Theorem 7. *For any $\eta \geq 1$, Set-Cover cannot be efficiently approximated to within any number smaller than $c \ln n$, for $c = \frac{1}{(1+1/\eta)(\ln\eta d + 1 + 1/\eta d)}$, unless $\mathcal{P} = \mathcal{NP}$.*

Proof. Fix some $\varepsilon > 0$. Let us confine ourselves to values $l = O(\log|\Phi|)$, so by corollary 9, the construction is efficient. Also, for $\gamma \doteq 2\,|\mathbb{F}|^{-1}(\eta dl)^d$, it holds $0 < \gamma < 1$. By lemma 11, deciding if a PCP instance has either $(1,1)$-good assignments, or no $(\eta dl, \gamma)$-good assignment at all, is $\mathcal{NP}$-hard. For a PCP instance $(X, \mathbb{F}, \Phi)$, choose l, so that $l \geq \frac{1-\varepsilon}{1-\gamma}\ln|\mathbb{U}| \geq \frac{1-\varepsilon}{1-\gamma}(l + 4\ln|\Phi|)$, e.g, $l = \frac{4(1-\varepsilon)\ln|\Phi|}{1-\gamma+\varepsilon}$. By claims 2 and 3, the PCP problem reduces to deciding whether the constructed Set-Cover instance is coverable by $(1+1/\eta)|\Phi|$ sets or by at least $(1-\varepsilon)|\Phi|\frac{\ln|\mathbb{U}|}{c}$ for the constant c of claim 3. The theorem follows. ∎

Acknowledgements

We would like to thank Ran Raz for help with the adjustment of the [8] reduction and to Tal Moran for discussions and proofreading an early version of this paper.

References

[1] N. Alon. Splitting necklaces. *Advances in Mathematics*, 63(3):247–253, 1987.

[2] N. Alon, J. Bruck, J. Naor, M. Naor, and R. Roth. Construction of asymptotically good, low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38:509–516, 1992.

[3] N. Alon, G. Cohen, M. Krivelevich, and S. Litsyn. Generalized hashing and parent-identifying codes. *J. Combinatorial Theory, Ser. A*, 104:207–215, 2003.

[4] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple constructions of almost k-wise independent random variables. In *Proc. 31st IEEE Symp. on Foundations of Computer Science*, volume 2, pages 544–553, 1990.

[5] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Electronic Colloquium on Computational Complexity (ECCC)*, 1(009), 1994. Full paper appears in J.ACM 42:4, July 1995, 844-856.

[6] S. Arora and S. Safra. Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998.

[7] S. Arora and M. Sudan. Improved low-degree testing and its applications. In *Proc. 29th ACM Symp. on Theory of Computing*, pages 485–495, 1997.

[8] M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient multi-prover interactive proofs with applications to approximation problems. In *Proc. 25th ACM Symp. on Theory of Computing*, pages 113–131, 1993.

[9] S. A. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd ACM Symp. on Theory of Computing*, pages 151–158, 1971.

[10] I. Dinur, E. Fischer, G. Kindler, R. Raz, and S. Safra. PCP characterizations of NP: Towards a polynomially-small error-probability. In *Proc. 31st ACM Symp. on Theory of Computing*, pages 29–40, 1999.

[11] D. Z. Du and F. K. Hwang. *Combinatorial Group Testing and its Applications*, volume 12 of *Series on Applied Mathematics*. World Scientific, New York, second ed. edition, 2000.

[12] G. Even, O. Goldreich, M. Luby, N. Nisan, and B. Velickovic. Efficient approximation of product distributions. *Random Structures and Algorithms*, 13(1):1–16, 1998.

[13] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[14] D. Koller and N. Megiddo. Constructing small sample spaces satisfying given constraints. *Proc. 25th ACM Symp. on Theory of Computing*, pages 268–277, 1993.

[15] L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13:383–390, 1975.

[16] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM*, 41(5):960–981, 1994.

[17] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput*, 22(4):838–856, 1993.

[18] M. Naor, L. J. Schulman, and A. Srinivasan. Splitters and near-optimal derandomization. In *Proc. 36th IEEE Symp. on Foundations of Computer Science*, pages 182–191, 1995.

[19] H. Q. Ngo and D. Z. Du. *A Survey on Combinatorial Group Testing Algorithms with Applications to DNA Library Screening*, volume 55 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 171–182. Amer. Math. Soc., 2000.

[20] R. Raz and S. Safra. A sub-constant error-probability PCP characterization of NP. In *Proc. 29th ACM Symp. on Theory of Computing*, pages 475–484, 1997.

[21] P. Slavik. A tight analysis of the greedy algorithm for set cover. In *Proc. 28th ACM Symp. on Theory of Computing*, pages 435–441, 1996.

[22] A. Srinivasan. Improved approximations guarantees for packing and covering integer programs. *SIAM J. Comput.*, 29(2):648–670, 1999. Preliminary version in Proc. STOC 95.

[23] J. van Leeuwen. Graph Algorithms*, Handbook of Theoretical Computer Science (J. Van Leeuwen Ed.)*, volume A (Algorithms and Complexity), chapter 10, pages 525–631. MIT Press, 1990.