# A threshold of $\ln n$ for approximating set cover

Uriel Feige*
Department of Applied Math and Computer Science
The Weizmann Institute
Rehovot 76100, Israel
feige@wisdom.weizmann.ac.il.

April 2, 1998

**Abstract**

Given a collection $\mathcal{F}$ of subsets of $S = \{1, \ldots, n\}$, *set cover* is the problem of selecting as few as possible subsets from $\mathcal{F}$ such that their union covers $S$, and *max k-cover* is the problem of selecting $k$ subsets from $\mathcal{F}$ such that their union has maximum cardinality. Both these problems are NP-hard. We prove that $(1 - o(1)) \ln n$ is a threshold below which set cover cannot be approximated efficiently, unless NP has slightly superpolynomial time algorithms. This closes the gap (up to low order terms) between the ratio of approximation achievable by the greedy algorithm (which is $(1 - o(1)) \ln n$), and previous results of Lund and Yannakakis, that showed hardness of approximation within a ratio of $(\log_2 n)/2 \simeq 0.72 \ln n$. For max $k$-cover we show an approximation threshold of $(1 - 1/e)$ (up to low order terms), under the assumption that $P \neq NP$.

## 1 Introduction

Let $S$ be a set of $n$ points and $\mathcal{F} = \{S_1, S_2, \ldots S_s\}$ a collection of subsets of $S$. *Set cover* is the problem of selecting as few as possible subsets from $\mathcal{F}$ such that every point in $S$ is contained in at least one of the selected subsets. *Max k-cover* is the problem of selecting $k$ subsets from $\mathcal{F}$ such that their union contains as many points as possible. Both these problems are NP-hard. A common approach of coping with NP-hard problems is by approximation algorithms that run in polynomial time and deliver solutions that are close to optimal. For set cover, we evaluate an approximation algorithm by considering the ratio between the number of subsets used in the cover output by the algorithm and the number of subsets used by the optimal solution. This ratio is always at least one, and the largest value

*Incumbent of the Joseph and Celia Reskin Career Development Chair.

that it can attain on an input instance is the approximation ratio of the algorithm. For max $k$-cover, we consider the ratio between the number of points covered by the $k$ subsets selected by the algorithm and the number of points covered by the optimal solution. This ratio is always at most one, and the smallest value that it can attain on an input instance is the approximation ratio of the algorithm. (Slighly extending the class of algorithms of interest, we also allow for randomized polynomial time algorithms, in which case we want the solution output by the algorithm to be close to optimal with high probability, where probability is computed over the coin tosses of the randomized algorithm.) It is well known that set cover can be approximated within a ratio of $\ln n$, where ln denotes the natural logarithm, and that max $k$-cover can be approximated within a ratio of $1 - 1/e \simeq 0.632$. The results and techniques in [1, 28] imply that there is a constant $\delta < 1$ such that it is NP-hard to approximate max $k$-cover within a ration better than $\delta$. Lund and Yannakakis [26] showed (under a complexity assumption that will be presented in Section 1.1) that it is hard to approximate set cover within a ratio of $(\log n)/2$, where log denotes logarithms in base 2. We extend these hardness results, and show that for any $\epsilon > 0$, set cover cannot be approximated within a ratio of $(1 - \epsilon)\ln n$ unless NP has $n^{O(\log\log n)}$-time deterministic algorithms, and that max $k$-cover cannot be approximated within a ratio of $1 - 1/e + \epsilon$ unless P=NP. This implies that known approximation algorithms for these problems are essentially best possible in terms of the approximation ratios that they guarantee. In all instances of set cover and max $k$-cover that we construct, $s$ (the number of subsets) is smaller than $n$ (the number of points). Our results are based on a reduction from a new multi-prover proof system for NP (see Section 2), designed specifically for this purpose. Our proof technique extends that of [26].

## 1.1 Related work

Set cover was among the first problems for which approximation algorithms were analysed. Johnson [23] showed that the greedy algorithm gives an approximation ratio of $\ln n$. (This was extended by Chvatal [7] to the weighted version of set cover.) Lovasz [24] showed that a linear programming relaxation approximates set cover within a ratio of $\ln n$. In both cases, the authors were interested mainly in the leading term of the approximation ratio. Analysis of the low order terms of the approximation ratio was provided by Srinivasan [34] (for the linear programming approach) and by Slavik [33] (for the greedy algorithm). For max $k$-cover, the greedy algorithm gives an approximation ratio of $1 - 1/e$ (up to low order terms). See [20] and references therein, and also Proposition 11. (A similar approximation ratio can be obtained via a linear programming relaxation, though the author is not aware of an explicit reference for this.)

The first hardness of approximation results for set cover followed from work on probabilistically checkable proof systems (PCPs). The notion of PCPs grew out of the theory of interactive proofs [14, 4, 6, 13] (parts of which we will review shortly) and from major breakthroughs in understanding their power [25, 32, 3]. The relevance of interactive proofs for proving hardness of approximation results was demonstrated in [9], and further developments in [2, 1] led to the PCP notion as stated below. Informally, a PCP for an NP language is a method of encoding NP witnesses, coupled with a *verifier* – a very efficient randomized method for verifying the validity of the witness. For any instance of the input language, the verifier reads only a constant number of bits from the corresponding PCP witness. The indices of these bits depend on random coin tosses of the verifier and on the input instance. The verifier accepts or rejects based on a simple predicate evaluated on these bits. If the input instance is in the NP language, then there is a way of encoding the PCP witness such that regardless of the bits read by the verifier, the verifier accepts. If the input instance is not in the NP language, then any string given as a PCP witness will be accepted by the verifier with probability at most 1/2 (probability taken over the coin tosses of the verifier). The gap between the probabilities that the verifier accepts inputs in the NP-language and inputs not in the NP-language is a key property that makes PCPs useful in proving hardness of approximation results.

As shown by Arora *et.al.* [1], the above PCP characterization of NP-languages (the "PCP theorem") is equivalent to the statement that it is NP-hard to approximate *MAX 3SAT*, meaning that for some $\delta < 1$, it is NP hard to distinguish between satisfiable 3CNF formulas and 3CNF formulas in which at most a $\delta$-fraction of the clauses can be satisfied. This immediately implies constant factor hardness of approximation results for a variety of other problems – all those that are MAX SNP-hard [28]. One of these problems is *minimum vertex cover* in bounded degree graphs, that is, selecting as few as possible vertices in a graph of bounded degree such that for each edge, at least one of its endpoints is selected. As vertex cover is a special case of set cover, this implies that for some $\epsilon > 0$, it is NP-hard to approximate set cover within a ratio of $1+\epsilon$. Using the fact that the graph is of bounded degree, one can also show that it is NP-hard to approximate max $k$-cover within a ratio of $1-\epsilon$.

To present subsequent hardness of approximation results, we let TIME($t$) denote the class of languages that have a deterministic algorithm that runs in time $t$, and let ZTIME($t$) denote the class of languages that have a probabilistic algorithm that runs in expected time $t$ (with zero error). We shall ignore low order terms in the approximation ratios presented below.

Lund and Yannakakis [26] showed that set cover cannot be approximated within a ratio of $\log n/4$ unless $NP \subset TIME(n^{O(\text{polylog } n)})$, and that set cover cannot be approximated within a ratio of $\log n/2$ unless $NP \subset ZTIME(n^{O(\text{polylog } n)})$. Their proof was based on a reduction from efficient two prover proof systems for NP [6]. For our purposes, a two prover proof system can be described as a PCP with some special properties. The alphabet in which the PCP witness is encoded is no longer binary, and its cardinality may depend on the input size. The PCP witness is partitioned into two segments. The verifier reads one character from each segment (the choice of which character to read is based on the input instance and on random coin tosses of the verifier), and accepts or rejects based on a predicate evaluated on the two characters. More generally, one may view $k$-prover proof systems as PCPs in which the PCP witness is partitioned into $k$ segments, and the verifier reads one character from each segment. In the terminology of multiprover proof systems [6], each segment of the PCP witness is thought of as being controlled by one prover. The contents of the segment are called the *strategy* of the prover. Reading a character from a segment corresponds to the verifier querying the respective prover as to the value of the indexed character, and the prover responding with the value of the requested character. As each prover is queried only once, our description corresponds to one round multiprover proof systems. More general multiround proof systems are also described in [6], but are beyond the scope of the current paper.

Lund and Yannakakis obtained their hardness results for approximating set cover under complexity assumptions that are stronger than $P \neq NP$. In order to get hardness results under weaker complexity assumptions, subsequent work focused on reducing the probability of falsely accepting in a multiprover proof system (this probability is known as the *error* of the proof system), while maintaining small values for other parameters such as the number of provers, the cardinality of the alphabet and the number of random bits used by the verifier. Bellare *et.al.* [5] constructed four prover proof systems that implied that unless P=NP set cover cannot be approximated within any constant ratio, and unless $NP \subset TIME(n^{O(\log\log n)})$ then set cover cannot be approximated within a ratio of $\log n/8$. Improved analysis of two prover proof systems by Raz [30] implies that unless $NP \subset TIME(n^{O(\log\log n)})$ then set cover cannot be approximated within a ratio of $\log n/4$, and that unless $NP \subset ZTIME(n^{O(\log\log n)})$ then set cover cannot be approximated within a ratio of $\log n/2$.

Improved deterministic constructions by Naor *et.al.* [27] closed the gap (up to low order terms) between the consequences achievable under the assumption that NP is not contained in a deterministic time class and the assumption that NP is not contained in a probabilistic time class. It follows that unless $NP \subset TIME(n^{O(\log\log n)})$ then set cover cannot be approximated within a ratio of $\log n/2$.

In our work we close the gap between the known $\ln n$ approximation ratio and the hardness result of $\log n/2$. We show that the upper bound is tight (up to low order terms) under the assumption that $NP \not\subset TIME(n^{O(\log\log n)})$.

There are only few NP optimization problems that are known to have a threshold of nontrivial nature (e.g., not located at approximation ratio 1). A very sharp example is the *minimum p-center* problem, for which Hsu and Nemhauser [22] showed that it is NP-hard to obtain approximation ratios below 2, whereas Hochbaum and Shmoys [21], and Dyer and Frieze [8], showed how to approximate minimum $p$-center within a factor of 2. For the *minimum maximal independent set* problem, Halldorsson [16] shows that it cannot be approximated within a ratio of $n^{1-\epsilon}$, for any $\epsilon > 0$, which is tight up to multiplicative low order terms. Another example of a well characterized approximation problem is presented in [19]. Our work shows that also set cover and max $k$-cover have a threshold of a nontrivial nature. This can be extended to other problems as well, as the same threshold of $\ln n$ holds for all problems that are equivalent to set cover in terms of approximation ratio, such as *dominating set* (see [29] and [26] for more details).

A survey of hardness of approximation results and the techniques involved is provided by Arora and Lund in [20].

**Remark:** A preliminary version of this paper, without the results on max $k$-cover, appeared in the Proceedings of the 28th Annual ACM Symposium on the Theory of Computing, 1996. Since then, several other thresholds for approximation were discovered. Essentially tight $O(n^{1-\epsilon})$ hardness of approximation results where obtained for clique and independent set [17], and for chromatic number [11]. Tight constant factor hardness of approximation results were obtain for several problems in [18], including a threshold of 7/8 for MAX 3SAT. As for set cover, Raz and Safra [31] constructed new low error constant prover proof systems and used them to show that for some constant $c > 0$, it is NP-hard to approximate set cover within a ratio of $c \log n$. It is not known whether hardness of approximating set cover within a ratio of $\ln n$ (up to low order terms) can be shown under the assumption that $P \neq NP$, rather than $NP \not\subset TIME(n^{O(\log\log n)})$.

### 1.2 Overview

We give a high level overview of the main ideas in our proof that set cover is hard to approximate within a ratio of $\ln n$. The proof that max $k$-cover is hard to approximate within a ratio of $1 - 1/e$ is based on similar ideas.

The proof of Lund and Yannakakis involves a combinatorial construction, and a reduction from two prover proof systems to set cover which uses the combinatorial construction. The ratio of $\log n/2$ comes up from the following construction. There is a set $S$ of $m$ points, and a collection $\mathcal{F}$ of subsets of $S$ and their complements, each of size $m/2$. A *good* way of covering $S$ is by taking a subset and its complement, thus using only two subsets. The combinatorial construction is such that any *bad* cover of $S$ that does not include a subset and its complement must use at least roughly $\log m$ subsets from $\mathcal{F}$. Hence the ratio between the good case and the bad case is $\log m/2$. We remark that a combinatorial construction with properties as described above is easy to come by: standard probabilistic arguments show that if the subsets in $\mathcal{F}$ are chosen at random, then with high probability every bad cover requires at least roughly $\log m$ subsets, as desired. Lund and Yannakakis showed how to reduce two prover proof systems for satisfiability of a formula $\phi$ to a collection of sets as described above, such that if $\phi$ is satisfiable, all sets are covered by the good way, and if $\phi$ is not satisfiable, most sets need to be covered by the bad way.

To prove a $\ln n$ ratio, we consider a modified construction which we call a *partition system*. There is a set $S$ of $m$ points, and a collection $\mathcal{F}$ of subsets of $S$, each of size $m/k$, where $k$ is a large constant. Each subset is associated with $k-1$ other pairwise disjoint subsets of size $m/k$ that together partition $S$ into $k$ equal parts. A good cover of $S$ by disjoint subsets requires only $k$ subsets. A bad cover needs roughly $d$ subsets (not belonging to the same partition) in order to cover $S$, where $(1-1/k)^d \simeq 1/m$. As $k$ grows, $d$ tends to $k \ln m$. The ratio between the two cases approaches $\ln m$, as desired. Again, a construction based on random subsets of size $m/k$ will with high probability have properties as described above.

To make use of the above setting, we design a new $k$-prover proof system for satisfiability. We remark that already in [5] hardness results for set cover were proved using $k$-prover proof system, where $k = 4$. However, these hardness results gave poorer bounds on the ratio of approximation than those obtainable from two prover proof systems. The reason why we obtain stronger bounds is that we introduce a new ingredient into $k$-prover proof systems – that of having two different acceptance predicates, a *strong* acceptance predicate, and a *weak* acceptance predicate. In our proof system, the difference between the case when $\phi$ is satisfiable and the case when $\phi$ is not satisfiable is not only in the acceptance probability, but also in the acceptance predicate. If $\phi$ is satisfiable, the provers have a strategy that always satisfies the strong acceptance predicate (and hence also the weak one). If $\phi$ is not satisfiable, then any strategy of the provers satisfies the weak acceptance predicate on only a small fraction of the possible queries of the verifier. The gap that we obtain for approximating set cover is due in part to the difference in acceptance probability between the cases that $\phi$ is satisfiable and $\phi$ is not satisfiable, and in part to the difference in acceptance predicate.

In Section 2 we describe our $k$-prover proof system. It is fortunate that we can invoke a recent theorem of Raz [30] regarding reduction of error by parallel repetition. In contrast, Lund and Yannakakis used the more complicated two prover proof system of Feige and Lovasz [12] (the result of [30] was not available at the time), without actually describing it.

In Section 3 we explain how to construct the partition systems mentioned above. In Section 4 we describe the reduction from our $k$-prover proof system to set cover. In Section 5 we show that *max k-cover* has an approximation threshold at $1 - 1/e$ under the assumption that $P \neq NP$. In Section 6 we analyze the low order terms for hardness of approximation for set cover. Under the assumption that $NP \not\subset ZTIME(2^{n^\eta})$ for some $\eta > 0$, we show that set cover cannot be approximated within $\ln n - c(\ln \ln n)^2$, for some constant $c$ that depends on $\eta$.

## 2 A multi-prover proof system

Our result is based on a reduction from a multi-prover proof system. In Section 2.1 we describe the NP-hard problem MAX 3SAT-5 for which we construct the multiprover proof system. This specific problem has a regular structure that will later be used in proving the hardness of approximating set cover. In Section 2.2 we use standard techniques to construct a low error two prover proof system for MAX 3SAT-5. In Section 2.3 we construct our $k$-prover proof system for MAX 3SAT-5. This proof system has the new feature of having two different acceptance predicates, which is used in proving the hardness of approximating set cover. We remark that the two prover proof system of Section 2.2 is presented only so as to provide intuition and help in the analysis of the final $k$-prover proof system, and is not used elsewhere in our paper.

### 2.1 The underlying NP-complete language

Our starting point is the problem of **MAX 3SAT-B**.

*Input:* A CNF formula with $n$ variables in which every clause contains at most three literals (a literal is a Boolean variable in either positive or negated form), and every variable appears in a bounded number of clauses.

*Output:* The maximum number of clauses that can be satisfied simultaneously by some assignment to the variables.

The following well known theorem appears in [1, 28].

**Theorem 1** *It is MAX-SNP hard to approximate MAX 3SAT-B: for some $\epsilon > 0$, it is NP-hard to distinguish between satisfiable 3CNF-B formulas, and 3CNF-B formulas in which at most an $(1 - \epsilon)$-fraction of the clauses can be satisfied simultaneously.*

We would like to work with 3CNF-B formulas that have a very regular structure, and hence define the problem of **MAX 3SAT-5**.

*Input:* A CNF formula with $n$ variables and $5n/3$ clauses, in which every clause contains exactly three literals, every variable appears in exactly five clauses, and a variable does not appear in a clause more than once.

*Output:* The maximum number of clauses that can be satisfied simultaneously by some assignment to the variables.

**Proposition 2** *For some $\epsilon > 0$, it is NP-hard to distinguish between satisfiable 3CNF-5 formulas, and 3CNF-5 formulas in which at most a $(1 - \epsilon)$-fraction of the clauses can be satisfied simultaneously.*

**Proof:** The proof uses known techniques, and is only sketched below. It is based on the hardness of approximating MAX 3SAT-B (Theorem 1). We change an arbitrary 3CNF-B formula $\psi$ to a new 3CNF-5 formula $\phi$ (on a different set of variables).

Consider any variable $x$ and let $b$ be the number of occurrences of $x$ in $\psi$. $b$ is bounded from above by some universal constant, and w.l.o.g, we also assume that $b \geq 2$. Replace each occurrence of $x$ by a fresh variable $x_i$, for $0 \leq i \leq b-1$, and add the $2b$ clauses $(x_i \vee \bar{x}_{i+1})$, $(\bar{x}_i \vee x_{i+1})$, where $i+1$ is computed mod $b$. These clauses are satisfied only if $x_i = x_{i+1}$ for every $i$. Now each variable appears exactly 5 times, and no variable appears more than once in the same clause. For clauses that are shorter than three, add a fresh dummy literal $\bar{y}$, and add the following clauses with additional dummy variables $z_1$ and $z_2$: $(y \vee z_1 \vee z_2)$, $(y \vee \bar{z}_1 \vee z_2)$, $(y \vee z_1 \vee \bar{z}_2)$, and $(y \vee \bar{z}_1 \vee \bar{z}_2)$. These clauses are satisfied only if $y = 1$, in which case $\bar{y}$ has no influence on the original clause to which it was added. Add a constant number of additional dummy variables $w_i$ so that the total number of variables (of the types $x_i$, $y_i$, $z_i$, and $w_i$) is divisible by 3, and add dummy 3-CNF clauses that contain distinct dummy variables $z_i$ and $w_i$ in positive form, until each dummy variable occurs exactly five times.

The above reduction has the following properties (proof left to the reader):

1. The reduction takes polynomial time.
2. If $\psi$ is satisfiable then so is $\phi$.
3. The number of clauses increases by at most a constant multiplicative factor. (This is a consequence of the fact that each clause in $\psi$ is of bounded length).
4. The number of unsatisfiable clauses decreases by at most a constant multiplicative factor. (This is a consequence of the fact that each variable appears a bounded number of times in $\psi$).

Properties 3 and 4 above imply that if a $\delta$-fraction of the causes of $\psi$ are not satisfiable, then an $\epsilon$-fraction of the causes of $\psi$ are not satisfiable, for some $\epsilon$ that depends on $\delta$. Hence if one could distinguish in polynomial time between satisfiable 3CNF-5 formulas and 3CNF-5 formulas in which at most a $(1-\epsilon)$-fraction of the clauses can be satisfied simultaneously, then one could distinguish in polynomial time between satisfiable 3CNF-B formulas and 3CNF-B formulas in which at most a $(1-\delta)$-fraction of the clauses can be satisfied simultaneously. $\Box$

Following the NP-hardness result of Proposition 2, we shall assume that the input to the multiprover proof systems that we construct is either a satisfiable 3CNF-5 formula (a *true* input), or a 3CNF-5 formula in which every assignment to the variables fails to satisfy an $\epsilon$-fraction of the clauses, for some universal constant $\epsilon > 0$ (a *false* input).

## 2.2 A two prover proof system for MAX 3SAT-5

Using known techniques, we construct a one-round two-prover proof system for 3SAT-5. In our two-prover proof system, the first prover receives as a query the index of a clause, and returns as an answer a sequence of three bits (i.e., a value between 0 and 7). These three bits can be viewed as Boolean assignments to the three variables of the clause. The second prover receives as a query the index of a variable, and returns one bit as an answer. This bit can be viewed as a Boolean assignment to the variable. The verification procedure is as follows. The verifier selects an index of a clause at random, sends it to the first prover, and selects a random variable in the clause, and sends its index to the second prover. The verifier interprets the reply of the first prover as an assignment to the three variables in the clause, and the reply of the second prover as an assignment to the variable selected from the clause. The verifier accepts if the following two conditions hold:

1. **Clause check:** the assignment sent by the first prover satisfies the clause.
2. **Consistency check:** the assignment sent by the second prover is identical to the assignment for the same variable sent by the first prover.

**Proposition 3** *Let $\phi$ be a 3CNF-5 formula and let $\epsilon$ be the fraction of unsatisfied clauses in the assignment to the variables that satisfies the largest number of clauses. Then under the optimal strategy of the provers, the verifier in the above two prover proof system accepts with probability* $(1-\epsilon/3)$.

**Proof:** To see that regardless of the strategy of the provers the acceptance probability is at most $(1-\epsilon/3)$, observe that the strategy of the second prover defines an assignment $\chi$ to the variables of $\phi$. When the verifier selects a clause that is not satisfied by $\chi$ (this happens with probability at least $\epsilon$), then in order to pass the clause check, the first prover must set at least one of the three variables differently from $\chi$, and then the consistency check fails with probability at least $1/3$.

A strategy that guarantees acceptance probability of at least $(1-\epsilon/3)$ is to let $\chi$ be an assignment that satisfies a $(1-\epsilon)$-fraction of the clauses, and to have the first prover set exactly one variable differently from $\chi$ for clauses not satisfied by $\chi$. $\square$

The probability of accepting a false input is known as the *error* of the two prover proof system. For the above two-prover proof system, the error may be as high as $1-\epsilon/3$. We now modify our construction so as to substantially lower the error. This is done via a method known as *parallel repetition*. Rather than choose at random one clause, the verifier chooses at random $\ell$ clauses (the value of $\ell$ will be determined later). The indices of these $\ell$ clauses are sent to the first prover, who now replies with a sequence of $3\ell$ bits. From each clause the verifier chooses at random one variable, and sends the indices of the $\ell$ variables to the second prover. The second prover replies with a sequence of $\ell$ bits. The verifier interprets the sequence of bits sent by the first prover as an assignment to the $3\ell$ variables that appear in the $\ell$ random clauses, and interprets the sequence of bits sent by the second prover as an assignment to the $\ell$ variables that were queried of the second prover. The verifier accepts if the following two conditions hold for every one of the $\ell$ clauses: the assignment sent by the first prover satisfies the clause, and the assignment sent by the second prover is identical to the assignment for the same variable sent by the first prover. Hence from the point of view of the verifier, the new proof system is composed of $\ell$ parallel repetitions of the original proof system, where each repetition uses fresh random bits. As the verifier accepts in the modified proof system only if all repetitions are accepting, it is natural to expect that the error of the modified proof system will be at most $(1-\epsilon/3)^{\ell}$. Unfortunately, this is in general not true, due to subtle reasons that are best explained by explicit counter examples [13]. However, it is true that parallel repetition reduces the error at an exponential rate. The following theorem was proven by Raz [30].

**Theorem 4** *If a one round two prover proof system is repeated $\ell$ times independently in parallel, then the error is $2^{-c\ell}$, where $c > 0$ is a constant that depends only on the error of the original proof system (assuming this error was less than one) and on the length of the answers of the provers in the original proof system.*

As the error in our original two prover proof system was a constant $(1 - \epsilon/3)$ that is independent of $n$, and the answer length was also a constant (three for the first prover, one for the second prover), it follows from Theorem 4 that the error of our modified two prover proof system is at most $2^{-c\ell}$, for some universal constant $c$.

### 2.3 The $k$-prover proof system

We are now ready to describe our $k$ prover proof system for MAX 3SAT-5 which has the nonstandard feature of two different acceptance predicates. For reasons of efficiency in the construction, we consider a binary code that contains $k$ code words, each of length $\ell$ and weight $\ell/2$, and Hamming distance at least $\ell/3$ between any two code words. For our main result we shall choose $\ell = \Theta(\log\log n)$ and $k$ an arbitrarily large constant. In this case, assuming w.l.o.g. that $\ell$ is an exact power of 2 and that $k < \ell$, the rows of a Hadamard matrix give a code with the desired properties (in fact, with Hamming distance $\ell/2$). For refined results (see Section 6), it is useful to choose $k > \ell$, and use some other standard code instead of the Hadamard code.

In our $k$-prover proof system, the verifier selects $\ell$ clauses uniformly and independently at random. Call these clauses $C_1, \ldots, C_\ell$. From each clause, the verifier selects a single variable uniformly and independently at random. These are called the *distinguished* variables $x_1, \ldots, x_\ell$. (So far, this is identical to the modified two prover proof system.) With each prover the verifier associates a code word. Prover $P_i$ receives $C_j$ for those coordinates $j$ in its code word that have the bit 1, and $x_j$ for those coordinates in its code word that have the bit 0. Each prover replies with a string of $2\ell$ bits. This string is interpreted by the verifier as an assignment to all the variables that the prover received ($\ell/2$ distinguished variables plus three variables in each of the $\ell/2$ clauses). For simplicity in describing the acceptance predicate, we assume that for each of the $\ell/2$ clauses received by the prover, the corresponding bits in the prover's answer encode a satisfying assignment for that clause. (This assumption is without loss of generality, as whenever it does not hold, the verifier may simply complement the first of the three bits that correspond to the variables of the unsatisfied clause, thereby obtaining a canonical reply that satisfies the clause.) Hence in this $k$-prover proof system, the acceptance predicates need not involve clause checks, and will only involve consistency checks.

Observe that the answer of a prover induces an assignment to the distinguished variables. (Namely, if on the respective coordinate the answer gives an assignment to all three variables in the clause rather than an assignment just to the distinguished variable, remove the assignment for the other two variables. If the same variable appears several times in the sequence of distinguished variables, different occurrences of the same variable may receive different assignments.) We say that the answers of two provers are *consistent* if the induced assignments to the distinguished variables is identical.

We can now describe our acceptance predicates:

- **Weak acceptance predicate:** at least one pair of provers is consistent.
- **Strong acceptance predicate:** every pair of provers is consistent.

**Lemma 5** *Consider the $k$-prover proof system defined above and a 3CNF-5 formula $\phi$. If $\phi$ is satisfiable, then the provers have a strategy that causes the verifier to always strongly accept. If at most a $(1-\epsilon)$-fraction of the clauses in $\phi$ are simultaneously satisfiable, then the verifier weakly accepts with probability at most $k^2 \cdot 2^{-c\ell}$, where $c > 0$ is a constant that depends only on $\epsilon$.*

**Proof:** If $\phi$ is satisfiable, then the provers can base their answers on a canonical satisfying assignment (e.g., on the lexicographically first such assignment). Then all clauses are satisfied and the answers of all provers are mutually consistent.

We now consider the case in which only a $(1-\epsilon)$-fraction of the clauses of $\phi$ are satisfiable. Assume that the verifier weakly accepts with probability at least $\delta$. Then with respect to two of the provers, the verifier accepts with probability at least $\delta/k^2$. By the property of the code, there are at least $\ell/6$ coordinates on which one of these provers receives a clause, and the other prover receives a variable in this clause. Fix the question pairs in the other $5\ell/6$ coordinates in a way that maximizes the acceptance probability, which by averaging remains at least $\delta/k^2$. Now omit the questions on these $5\ell/6$ coordinates (the provers can reconstruct them anyway). It follows that the two provers have a strategy that succeeds with probability at least $\delta/k^2$ on $\ell/6$ parallel repetitions of the original two prover proof system. From Theorem 4, $\delta/k^2 < 2^{-c\ell}$. $\square$

# 3 Construction of partition systems

**Definition 1** *A* partition system $B(m, L, k, d)$ *has the following properties.*

1. *There is a ground set $B$ of $m$ points.*
2. *There is a collection of $L$ distinct* partitions $p_1, \ldots, p_L$.
3. *For $1 \le i \le L$, partition $p_i$ is a collection of $k$ disjoint subsets of $B$ whose union is $B$.*
4. *Any cover of the $m$ points by subsets that appear in pairwise different partitions requires at least $d$ subsets.*

**Lemma 6** *For every $c \ge 0$ and $m$ sufficiently large there is a* partition system $B(m, L, k, d)$ *whose parameters satisfy the following inequalities:*

1. $L \simeq (\log m)^c$.
2. *$k$ can be chosen arbitrarily as long as $k < \frac{\ln m}{3 \ln \ln m}$.*
3. *$d = (1 - f(k))k \ln m$, where $f(k) \to 0$ as $k \to \infty$.*

*A partition system with parameters as described above and $f(k) = 2/k$ can be constructed $ZTIME(m^{O(\log m)})$.*

**Proof:** Consider the following randomized construction for $B(m, L, k, d)$.

*For each point in the set $B$, for each partition $p_i$, decide independently at random in which subset of the partition to place the point.*

We show that with high probability $d$ subsets, each belonging to a different partition, cannot cover the set $B$ (for parameters as in the lemma). Consider a particular choice of $d$ subsets, no two of which belong to the same partition. Then the probability for a point to be covered by at least one of the $d$ subsets is $1 - (\frac{k-1}{k})^d$. Using $(1 - 1/k)^k > e^{-1-1/k}$ (for $k \ge 2$), and $(1 + 1/k)(1 - 2/k) < (1 - 1/k)$, this probability is at most $1 - m^{-1+1/k}$. As there are $m$ points, the probability that all $m$ points are covered by the same $d$ subsets is $(1-m^{-1+1/k})^m < e^{-m^{1/k}}$. There are $k^d\binom{L}{d} < L^d$ ways of choosing the subsets (the inequality holds since $d >> k$). Substituting $L \simeq (\log m)^c$ and $d < k \ln m$, the probability that some collection of $d$ subsets covers all points is at most $(\log m)^{ck \ln m} e^{-m^{1/k}}$. For $k < \frac{\ln m}{3 \ln \ln m}$ and $m$ sufficiently large, this probability tends to 0, proving that the probabilistic construction works.

The randomized construction described above requires time polynomial in $m$, and succeeds with probability at least 1/2. Exhaustively checking that the construction indeed gives a partition system can be done in time roughly $\binom{kL}{d} < m^{O(\log m)}$. The expected number of times the randomized construction needs to be tried until it succeeds is less than 2. $\square$

The randomized construction can be replaced by a deterministic construction using techniques developed in [27]. There, partition systems are called *anti-universal* sets. Theorem 9 in [27] says that for any $k$ one can in time linear in $m$ construct a partition system for which $m = (\frac{k}{k-1})^d d^{O(\log d)} \log L$. (Here $k$ is assumed to be an arbitrary constant, and $m$ grows as a function of $L$ and $d$.) Expressing the ratio $d/k$ as a function of $m$ one gets $(1 - f(k)) \ln m$, where $f(k) \to 0$ as $k \to \infty$, provided that $d$ is sufficiently large as a function of $k$, and $L$ is bounded by a polynomial in $d$. This will hold when we use partition systems in Section 4. (The reader may use the following table to translate from our notation to that of [27]: a point in set $B \to$ a function $h$ in collection $H$, $m \to |H|$, $L \to n$, $k \to b$, $d \to k$.)

## 4 The reduction to set cover

Our reduction extends that of Lund and Yannakakis [26].

The verifier of the $k$-prover proof system of Section 2.3 uses its randomness, which we assume that is given in form of a random string $r$, to select $\ell$ clauses and a distinguished variable in each clause. We call these $\ell$ distinguished variables the *sequence of distinguished variables*. The length of the random string is $(\log 5n/3 + \log 3)\ell = \ell \log 5n$. Let $R = (5n)^\ell$ denote the number of possible random strings for the verifier. With each random string $r$, we associate a distinct partition system $B_r(m, L, k, d)$ as in Lemma 6, where $L = 2^\ell$, $m = n^{\Theta(\ell)}$, and $d = (1 - f(k))k \ln m$. (Altogether there are $N = mR$ points in our set cover problem.) Each of the $L$ partitions is labeled by an $\ell$-bit string $p$, that corresponds to an assignment to the respective sequence of distinguished variables. Each subset in a partition is labeled by a unique prover $i$. We let $B(r, j, i)$ denote the $i$th subset of partition $j$ in partition system $r$. With each question-answer pair $(q, a)$ of prover $P_i$, where $1 \leq i \leq k$, we associate a subset $S_{(q,a,i)}$ as follows. (Remark: the notation $S_{(q,a,i)}$ is somewhat redundant, but is used for clarity. The index $i$ of the prover can be deduced from the syntax of $(q, a)$ by observing which coordinates have clauses and which have variables.) We use the notation $(q, i) \in r$ to say that on random string $r$, prover $P_i$ receives question $q$. For $r$ such that $(q, i) \in r$, consider the induced sequence of distinguished variables, and extract from $a$ on a coordinate by coordinate basis an assignment $a_r$ to this sequence of variables. One of the partitions of partition system $B_r(m, L, k, d)$ has label $a_r$. The subset $S_{(q,a,i)}$ contains the points of subset $B(r, a_r, i)$, for all $r$ with $(q, i) \in r$.

Let $Q$ denote the number of possible different questions that a prover may receive. A question to a single prover includes $\ell/2$ variables, for which there are $n^{\ell/2}$ possibilities (with repetition), and $\ell/2$ clauses, for which there are $(5n/3)^{\ell/2}$ possibilities. Hence $Q = n^{\ell/2} \cdot (5n/3)^{\ell/2}$. Observe that this number is the same for all provers.

**Lemma 7** *If $\phi$ is satisfiable, then the above set of $N = mR$ points can be covered by $kQ$ subsets. If only a $(1-\epsilon)$ fraction of the clauses in $\phi$ are simultaneously satisfiable, the above set requires $(1 - 2f(k))kQ \ln m$ subsets in order to be covered, where $f(k) \to 0$ as $k \to \infty$.*

**Proof:** If $\phi$ is satisfiable, consider a satisfying assignment $A$ for $\phi$, and fix for the provers the strategy of answering each question consistently with this satisfying assignment. Now consider the subsets $S_{(q,a,i)}$, for which $a$ is indeed the answer given by prover $P_i$ on question $q$ under the above strategy. For any $r$, consider only the subsets $S_{(q_1,a_1,1)}$, $S_{(q_2,a_2,2)}$, $\ldots$, $S_{(q_k,a_k,k)}$, where for $1 \leq i \leq k$, $(q_i, i) \in r$, and $a_i$ is the answer given by prover $P_i$ on this question under the strategy described above. Then the partition system $B_r(m, L, k, d)$ is completely covered by these $k$ sets $S_{(q_i,a_i,i)}$, since for the partition whose label $p$ agrees with assignment $A$, the $i$th such set contains subset $B(r, p, i)$, for every $i$. A similar argument applies for every $r$. Hence the collection of subsets described above covers all points. The number of subsets used is $k$ times the number of possible questions to a single prover. Interestingly, the $kQ$ subsets used in the cover happen to be disjoint.

If only a $(1 - \epsilon)$-fraction of the clauses in $\phi$ are simultaneously satisfiable, then by Lemma 5 any strategy of the provers (weakly) succeeds with probability at most $k^2 \cdot 2^{-c\ell}$. Assume a cover of size $(1 - \delta)kQ \ln m$, where $\delta = 2f(k)$, and derive a contradiction.

Let $\mathcal{C}$ be a collection of subsets that covers $S$, where $|\mathcal{C}| = (1 - \delta)kQ \ln m$. With each question $q$ to a prover $P_i$ associate a weight $w_{q,i}$ equal to the number of answers $a$ such that $S_{(q,a,i)} \in \mathcal{C}$. Hence $\sum_{q,i} w_{q,i} = |\mathcal{C}|$. With each random string $r$ associate a weight $w_r = \sum_{(q,i)\in r} w_{q,i}$. This weight is equal to the number of subsets that participate in covering the $m$ points of $B_r(m, L, k, d)$. Call $r$ *good* if $w_r < (1 - \delta/2)k \ln m$.

**Proposition 8** *The fraction of good $r$ is at least $\delta/2$.*

**Proof:** Assume otherwise. Then $\sum_r w_r \geq (1 - \delta/2)^2 kR \ln m > (1 - \delta)kR \ln m$, where $R$ denotes the number of possible random strings of the verifier. On the other hand,

$$\sum_r w_r = \sum_r \sum_{(q,i)\in r} w_{q,i} = \sum_{q,i} \frac{R}{Q} w_{q,i} = \frac{R}{Q}|\mathcal{C}|$$

where the middle equality follows from the fact that there are exactly $R/Q$ random strings that cause the verifier to send out question $q$. Hence $|\mathcal{C}| > (1 - \delta)kQ \ln m$. Contradiction. $\square$

**Proposition 9** *Let $\mathcal{C}$ be a collection of subsets that covers $S$, where $|\mathcal{C}| = (1-\delta)kQ \ln m$. Then for some strategy for the $k$ provers, the verifier accepts $\phi$ with probability at least $2\delta/(k \ln m)^2$.*

**Proof:** Based on $\mathcal{C}$, we describe a randomized strategy for the $k$ provers. On question $q$ addressed to prover $P_i$, prover $P_i$ selects an answer $a$ uniformly at random from the set of answers that satisfy $S_{(q,a,i)} \in \mathcal{C}$. We show that under this strategy for the provers, the verifier weakly accepts with probability at least $2\delta/(k \ln m)^2$, where this probability is taken over the joint distribution of the coin tosses of the provers and of the verifier. Clearly, by fixing the optimal coin tosses for the provers, one obtains a deterministic strategy for the provers that satisfies the weak acceptance predicate with a probability that is at least as high.

Observe that for a fixed $r$, there is a one to one correspondence between sets $B(r,p,i)$ that participate in the cover of $B_r(m,L,k,d)$ and sets $S_{(q,a,i)}$ that belong to $\mathcal{C}$. For this correspondence we need $(q,i) \in r$ and the projection of $a$ on the sequence of distinguished variables to be $p$.

Concentrate now only on good $r$, and compute a lower bound on the probability that the verifier accepts when he chooses a good $r$. Observe that by property 4 of partition systems, and by the fact that for good $r$ the respective $B_r(m,L,k,d)$ is covered by at most $(1-\delta/2)k \ln m$ subsets, the cover $\mathcal{C}$ must have used two subsets from the same partition $p$ in the cover of $B_r(m,L,k,d)$ (by our choice of $\delta = 2f(k)$). Denote these two subsets by $B(r,p,i)$ and $B(r,p,j)$, where $i \neq j$, and their corresponding subsets in $\mathcal{C}$ by $S_{(q_i,a_i,i)}$ and $S_{(q_j,a_j,j)}$, respectively. Consider what happens when the verifier chooses random string $r$. Prover $P_i$ then receives question $q_i$ and prover $P_j$ receives question $q_j$. Let $A_{r,i}$ denote the set of answers satisfying $a \in A_{r,i}$ if and only if $S_{(q_i,a,i)} \in \mathcal{C}$. Define $A_{r,j}$ in an analogous manner. By the strategy described above, prover $P_i$ selects an answer $a \in A_{r,i}$ at random (and $P_j$ selects $a \in A_{r,j}$). Observe that for $a_i$ and $a_j$ above, $a_i \in A_{r,i}$ and $a_j \in A_{r,j}$, and furthermore, for good $r$, $|A_{r,i}| + |A_{r,j}| < k \ln m$. Hence the joint probability that the provers choose to answer with $a_i$ and $a_j$ is at least $4/(k \ln m)^2$. Since both these answers are consistent with the label $p$ of the same partition, the verifier weakly accepts.

To complete the proof, use Proposition 8, which shows that the probability that a verifier chooses a good $r$ is at least $\delta/2$. $\square$

To complete the proof of Lemma 7, observe that $2\delta/(k \ln m)^2 > k^2 \cdot 2^{-c\ell}$, for sufficiently large $\ell$ (made possible by $\ell = \Theta(\log\log n)$ and $m = n^{\Theta(\ell)}$). $\square$

We now prove our main theorem.

**Theorem 10** *If there is some $\epsilon > 0$ such that a polynomial time algorithm can approximate set cover within $(1-\epsilon)\ln n$, then $NP \subset TIME(n^{O(\log\log n)})$.*

**Proof:** Assume that there is a polynomial time algorithm $A$ that approximates set cover within $(1-\epsilon)\ln n$. Consider now an arbitrary NP-problem. Reduce it to the NP-complete problem of approximating an instance of max 3SAT-5. Now follow the reduction to set cover described above, with $k$ sufficiently large so that $f(k)$ in Lemma 7 is smaller than $\epsilon/4$, and with $m = (5n)^{2\ell/\epsilon}$. Using the deterministic construction of partition systems described in [27], and observing that $m$, $R$ and $Q$ are bounded by $n^{O(\log\log n)}$, the time to perform this reduction is $n^{O(\log\log n)}$. Recall that the number of points in the set cover problem is $N = mR$ where $R = (5n)^{\ell}$, and observe that for $m$ as above, $\ln m > (1-\epsilon/2)\ln N$. By Lemma 7, if the original NP instance was satisfiable, all points can be covered by $kQ$ subsets, and if the original NP instance was not satisfiable, all points cannot be covered by $(1-2f(k))kQ\ln m$. For our choice of $k$ and $m$, the ratio between the two cases is $(1-2f(k))\ln m > (1-\epsilon)\ln N$. Hence by applying algorithm $A$ to the set cover problem, one can tell whether the original NP instance was satisfiable or not. $\square$

## 5 Max $k$-cover

We say that a polynomial time algorithm *constructively approximates* max $k$-cover within a ratio of $\delta < 1$ if on any input, the number of points covered by the $k$ sets selected by the algorithm is at least a $\delta$-fraction of the number of points covered by the optimal solution. The following proposition is well known and is presented for completeness.

**Proposition 11** *The greedy algorithm (iteratively selecting the sets that cover the largest number of yet uncovered points) constructively approximates max $k$-cover within a ratio of at least* $1 - 1/e \simeq 0.632$.

**Proof:** Let $S' \subset S$ be the set of points covered by the optimal solution, and let $n' = |S'|$. Let $n_i$ be the number of new points covered by the $i$th set selected by the greedy algorithm. Then since $S'$ can be covered by $k$ sets, it follows that

$$n_i \geq \frac{n' - \sum_{j=1}^{i-1} n_j}{k}$$

Hence $\sum_{j=1}^{i} n_j \geq n' - n'(1-\frac{1}{k})^i$ and

$$\sum_{i=1}^{k} n_i \geq n' - n'(1-\frac{1}{k})^k \geq n'(1-1/e)$$

$\square$

Using a Turing reduction, Theorem 10 can be used to show that the performance guarantee of the greedy algorithm is optimal up to low order terms. This has also been observed by others (see [15], for example).

**Proposition 12** *If max $k$-cover can be constructively approximated in polynomial time within a ratio of $(1 - 1/e + \epsilon)$ for some $\epsilon > 0$, then $NP \subset TIME(n^{O(\log\log n)})$.*

**Proof:** Assume that a polynomial time algorithm $A$ approximates max $k$-cover within a ratio of $1 - 1/e + \epsilon$ for some $\epsilon > 0$. We use algorithm $A$ as a subroutine in a polynomial time algorithm $B$ that approximates set cover within $(1 - \delta) \ln n$. By Theorem 10, this implies that $NP \subset TIME(n^{O(\log\log n)})$.

Given an instance of set cover, try out all possible values of $1 \leq k \leq n$ as the number of sets that suffice to cover all points. One of those choices of $k$ is the true optimal $k$, and we concentrate on the one case in which this $k$ is tried out. Algorithm $B$ repeatedly applies algorithm $A$ on max $k$-cover problems, where after each application the points already covered by previous applications are removed (but $k$ remains unchanged).

Since all of $S$ can be covered by $k$ of the sets, then each time algorithm $A$ is applied a fraction of at least $(1 - 1/e + \epsilon)$ of the remaining points are covered. Hence the number of times that $A$ is applied is at most $\ell$ where $\ell$ satisfies $(1/e - \epsilon)^\ell = 1/n$, and the number of sets used in the cover is at most $\ell k$ (recall that $k$ is the number of sets used by the optimum cover). Simple manipulations show that $\ell = \ln n/(1 - \ln(1 - e\epsilon)) < (1 - \delta) \ln n$ for some $\delta > 0$ that depends only on $\epsilon$. $\Box$

We say that a polynomial time algorithm *approximates* max $k$-cover within a ratio of $0 < \delta < 1$ if on any input, the algorithm outputs a number that is between opt and $\delta \cdot \text{opt}$, where opt denotes number of points covered by the optimal solution. The following theorem improves on Proposition 12 in two respects: approximation need not be constructive, and the assumption $NP \not\subset TIME(n^{O(\log\log n)})$ is weakened to $P \neq NP$.

**Theorem 13** *For any $\epsilon > 0$, max $k$-cover cannot be approximated in polynomial time within a ratio of $(1 - 1/e + \epsilon)$, unless $P = NP$.*

**Proof:** We show a reduction from approximating max 3SAT-5 to approximating max $k$-cover. The value of $k$ for the $k$-cover problem will be denoted by $k'$, so as to distinguish it from the number of provers in the underlying $k$-prover proof system, and from the parameter $k$ that this number induces for partition systems.

The proof closely mimics that for set cover, and the reader is assumed to be familiar with the reduction of Section 4 and the proof of Lemma 7. Unlike the case for set cover, we set the parameter $\ell$ (number of repetitions) to be some large constant (rather than $\Theta(\log\log n)$). For this reason we shall get NP-hardness results rather than results under the assumption that $NP \not\subset TIME(n^{O(\log\log n)})$. The explicit construction of partition systems becomes simpler. Recall that $L = 2^\ell$ and let $m = k^L$ (the number of points in the partition system is now a constant that depends on the number of provers and the number of repetitions). Treat the points in a partition system as vectors in $\{0, \ldots, k-1\}^L$, and let the $i$th *partition*

partition the points into $k$ disjoint subsets according to their value on the $i$th coordinate. Clearly, any collection of $j$ subsets that appear in pairwise disjoint partitions covers exactly $(1-(1-1/k)^j)m$ points.

By performing the reduction of Section 4, we create an instance of max $k'$-cover with $k' = kQ$. If the original 3CNF-5 formula is satisfiable, then all $N$ points can be covered by $kQ$ sets. We sketch the proof that if only a $(1-\epsilon')$-fraction of the clauses of the original formula are simultaneously satisfiable, then $kQ$ sets can cover at most $(1-1/e+g(k))N$ points, where $g(k) \to 0$ as $k \to \infty$.

Assume that a $(1-1/e+\epsilon)$-fraction of the points are covered, and derive a contradiction. In analogy to the proof of Lemma 7, call $r$ *good* if two conditions hold: $w_r \leq 3k/\epsilon$, and the $w_r$ sets that participate in covering points in the partition system $r$ contain at least two sets from the same partition.

**Proposition 14** *The fraction of good $r$ is at least $\epsilon/3$.*

**Proof:** The average value (over the choice of $r$) of $w_r$ is exactly $k$ (similar to the proof of Proposition 8). Hence the fraction of $r$ with $w_r > 3k/\epsilon$ is at most $\epsilon/3$. Even if all points of the respective partition systems of these $r$ are covered, the average number of points that need to be covered from each other partition system is at least $(1-1/e+2\epsilon/3)m$. The average value of $w_r$ for these other $r$ is not larger than $k$.

Now assume that the fraction of good $r$ is less than $\epsilon/3$ and derive a contradiction. Even if all points of the good partition systems are covered, the average number of points that need to be covered from each remaining partition system is at least $(1-1/e+\epsilon/3)m$. The average value of $w_r$ for the remaining partition systems is at most $(1+\epsilon/3)k$. For the function $h(j) = (1-(1-1/k)^j)m$ which describes how many points are covered by $j$ subsets, the second derivative is never positive. Hence the average number of points that are covered per partition is maximized when all $w_r$ are equal, and then it is $(1-(1-1/k)^{(1+\epsilon/3)k})m$, which is smaller than $(1-1/e+\epsilon/3)m$ (for large enough $k$). $\square$

Now proceed as in Proposition 9, using the fact that $w_r \leq 3k/\epsilon$ for good $r$. This will give a strategy for the provers that causes the verifier to weakly accept with probability $\frac{\epsilon}{3}(\frac{\epsilon}{3k})^2$. This is larger than $k^2 2^{-c\ell}$, for large enough $\ell$. Contradiction. $\square$

# 6 Refinements

In our hardness of approximation result for set cover, $\epsilon$ need not be constant. It may be a decreasing function of $n$. To make $\epsilon$ as small as possible, we strengthen the $NP \not\subseteq TIME(n^{\log\log n})$ assumption. Observe that the low order terms in Proposition 15 are not far from optimal, as the greedy algorithm approximates set cover within $\ln n - \ln\ln n + O(1)$ [33].

**Proposition 15** *If for some $\eta > 0$, $NP \not\subseteq ZTIME(2^{n^\eta})$, then for some constant $c' > 0$, there is no polynomial time algorithm that approximates set cover within* $\ln n - c'(\ln\ln n)^2$.

**Proof:** By the NP-completeness of approximating MAX 3SAT-5, if there is a problem in NP that does not have $ZTIME(2^{n^\eta})$ algorithms for some $\eta > 0$, then max 3SAT-5 is not approximable in $ZTIME(2^{O(n^{2\eta})})$, for some (other) $0 < \eta < 1$. Performing the reduction of Section 4 under this stronger assumption, we can choose parameters of the reduction (such as $k$ and $m$) to be larger, obtaining smaller low order terms in the hardness of approximation result for set cover. Specifically, we choose:

Number of points in a partition system: $m = e^{n^\eta}$.
Number of provers: $k = \frac{\ln m}{3\ln\ln m} \simeq n^\eta$.
Number of random bits of verifier: $\ell = c''\log n$, for some sufficiently large constant $c'' > 0$.

We first verify that the above combination of parameters is possible. Recall from Section 2 that we need $k$ codewords of length $\ell$ such that the Hamming distance between any two codewords is $\Omega(\ell)$. This is possible whenever $k$ is at most mildly exponential in $\ell$ (e.g., by taking a random code), which holds for the choice of $\ell = c''\log n$. Another thing that needs to be checked is that the proof of Lemma 7 still goes through, and we shall verify this shortly.

We analyze the hardness of approximation ratio that these parameters give. Observe that for $\ell$ as above, the number of random strings available to the verifier is $R = (5n)^\ell = (5n)^{c\log n}$. The total number of points in the set cover problem is $N = mR$. Recall that if the original 3CNF-5 formula is satisfiable then $kQ$ sets may be used to cover all points, where $k$ is the number of provers, and $Q$ is the number of different questions that a single prover can receive. If only a $(1-\epsilon')$-fraction of the clauses of the original 3CNF-5 formula are simultaneously satisfiable then $(1-2f(k))kQ\ln m$ sets are required in order to cover all points. The ratio between these two cases is $(1-2f(k))\ln m$, which we need to express as a function of $N$, the total number of points in the instance of set cover.

From Lemma 6 it follows that in $ZTIME(m^{O(\log m)}) = ZTIME(2^{O(n^{2\eta})})$ we can construct partition systems with $f(k) = 2/k = 6\ln\ln m/\ln m$. From our choice of parameters, $\ln m = n^\eta$ and $\ln N = \ln mR = n^\eta + O((\log n)^2)$, implying that $\ln m \geq \ln N - O((\ln\ln N)^2)$. Altogether we have that $(1-2f(k))\ln m \geq \ln N - O((\ln\ln N)^2)$, as needed.

Finally, recall that for the proof of Lemma 7 we required that $4f(k)/(k\ln m)^2 > k^2 \cdot 2^{-c\ell}$, which indeed holds when $\ell$ is a sufficiently large multiple of $\log n$. $\square$

An open question that is "traditionally" (ever since [26]) associated with the hardness of approximating set cover is that of constructing two prover one round proof systems for NP, in which the amount of randomness used by the verifier is logarithmic, the answer length of the provers is logarithmic, and the error is polynomially small. Recall that our new

$k$-prover proof system is a variation on the low error two prover proof system of Section 2.2. Conceivably, if we had as a starting point a low error two prover proof system in which the verifier uses $O(\log n)$ random bits, our techniques would lead to a proof of hardness of approximating set cover within $(1-\epsilon)\ln n$ under the assumption that $P \neq NP$, rather than $NP \not\subset TIME(n^{O(\log\log n)})$. The number of random bits used by the verifier is relevant here because we construct instances of set cover with $N = mR$ points. For the reduction to be polynomial, $R$ must be polynomial in $n$, implying that the number of random bits used by the verifier must be logarithmic in $n$. The error in the proof system has to be at most $O(1/(\log n)^2)$ for the proof of Lemma 7 (or a similar lemma) to go through. The answer length must remain logarithmic so that the number of subsets and the number of partitions in a partition system will remain polynomial.

The open question of trying to decrease $R$ is also related to the analysis of the low order terms as in Proposition 15. For $m = 2^{n^\eta}$, if $R$ is decreased to $O(n^c)$, then $\ln m = \ln N - O(\ln\ln N)$. This may allow to reduce the low order term to $\Theta(\log\log n)$ (under the complexity assumption of Proposition 15). We remark that for this choice of parameters we need the error in the $k$-prover proof system to be polynomially small in order for the proof of Lemma 7 to go through.

We do not know that reducing the number of random bits used by the verifier in two prover proof systems is a necessary requirement for obtaining tight (up to low order terms) NP-hardness results for set cover. Moreover, it may not even be a sufficient requirement, since current techniques require that the proof systems have very regular structure.

Some of the difficulties involved in reducing the number of random bits in two prover proof systems are discussed in [10].

### Acknowledgements

I thank Moni Naor, Leonard Shulman, and Aravind Srinivasan for a preview of [27], and Mihir Bellare for his comments on an earlier version of this manuscript.

## References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy. "Proof verification and hardness of approximation problems". In *Proc. of 33rd Annual Symposium on Foundations of Computer Science, 14–23, 1992.*

[2] S. Arora, S. Safra. "Probabilistic checking of proofs: a new characterization of NP". In *Proc. of 33rd Annual Symposium on Foundations of Computer Science, 2–13, 1992.*

[3] L. Babai, L. Fortnow, C. Lund, "Non-deterministic exponential time has two-prover interactive protocols", *Computational Complexity, 1:3–40, 1991.*

[4] L. Babai, S. Moran. "Arthur-Merlin games: a randomized proof system, and a hierarchy of complexity classes." *J. Computer and Sys. Sci.* 36 (1988), 254–276.

[5] M. Bellare, S. Goldwasser, C. Lund, A. Russell. "Efficient probabilistically checkable proofs and applications to approximation". In *Proc. of 25th Annual ACM Symposium on the Theory of Computing*, 294–304, 1993.

[6] M. Ben-Or, S. Goldwasser, J. Kilian, A. Wigderson. "Multi-prover interactive proofs: how to remove intractability assumptions". In *Proc. of 20th Annual ACM Symposium on the Theory of Computing*, 113–131, 1988.

[7] V. Chvatal. "A greedy heuristic for the set covering problem". *Math. Oper. Res., 4, 1979, 233–235.*

[8] M. Dyer, A. Frieze. "A simple heuristic for the $p$-center problem". *Oper. Res. Lett., 3, 1985, 285–288.*

[9] U. Feige, S. Goldwasser, L. Lovasz, S. Safra, S. Szegedy. "Interactive proofs and the hardness of approximating cliques". *Journal of the ACM, Vol. 43, No. 2, 1996, 268–292.*

[10] U. Feige, J. Kilian. "Impossibility results for recycling random bits in two prover proof systems". *Proc. of 27th Annual ACM Symposium on the Theory of Computing*, 457–468, 1995.

[11] U. Feige, J. Kilian. "Zero knowledge and the chromatic number". *Proc. of 11th Annual IEEE Conference on Computational Complexity*, 1996.

[12] U. Feige, L. Lovasz. "Two prover one round proof systems: their power and their problems". In *Proc. of 24th Annual ACM Symposium on the Theory of Computing*, 733–744, 1992.

[13] L. Fortnow, J. Rompel, M. Sipser, "On the Power of Multi-Prover Interactive Protocols", *Theoretical Computer Science* 134, 545–557, 1994.

[14] S. Goldwasser, S. Micali, C. Rackoff. "The knowledge complexity of interactive proof-systems." *SIAM J. Comp.* **18** (1989), 186–208.

[15] S. Guha, S. Khuller. "Greedy strikes back: improved facility location algorithms". In *Proc. Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, 1998.*

[16] M. Halldorsson. "Approximating the minimum maximal independence number". *Inform. Process. Lett. 46 (1993) 169–172.*

[17] J. Håstad. "Clique is hard to approximate within $n^{1-\epsilon}$." In *Proc. 37th IEEE Symp. on Foundations of Computer Science*, 627–636, 1996.

[18] J. Håstad. "Some optimal inapproximability results". In *proc. 29th Annual ACM Symposium on Theory of Computing*, 1–10, 1997.

[19] J. Hastad, S. Phillips, S. Safra. "A well characterized approximation problem". *Proc. 2nd Israel Symp. on Theory of Computing and Systems, 261–265, 1993.*

[20] D. Hochbaum (editor). Approximation Algorithms for NP-hard Problems. PWS Publishing Company, Boston, 1997.

[21] D. Hochbaum, D. Shmoys. "A best possible approximation algorithm for the $k$-center problem". *Math. Oper. Res., 10, 1985, 180–184.*

[22] W. Hsu, G. Nemhauser. "Easy and hard bottleneck location problems". *Discrete Applied Math., 1, 1979, 209–216.*

[23] D. Johnson. "Approximation algorithms for combinatorial problems". *J. Comput. System Sci. 9, 1974, 256–278.*

[24] L. Lovasz. "On the ratio of the optimal integral and fractional covers". *Discrete Mathematics 13 (1975) 383–390.*

[25] C. Lund, L. Fortnow, H. Karloff, N. Nisan. "Algebraic Methods for Interactive Proof Systems." *J. ACM*, **39** (1992), 859–868.

[26] C. Lund, M. Yannakakis. "On the hardness of approximating minimization problems". *JACM 41(5), 1994, 960–981.*

[27] M. Naor, L. Schulman, A. Srinivasan. "Splitters and near-optimal derandomization". *Proc. of 36th Annual Symposium of Foundations of Computer Science*, 182–191, 1995.

[28] C. Papadimitriou, M. Yannakakis. "Optimization, approximation, and complexity classes". *JCSS, 43, 1991, 425–440.*

[29] A. Paz, S. Moran. "Nondeterministic polynomial optimization problems and their approximations", *Theoretical Computer Science, 15 (1981) 251–277.*

[30] R. Raz. "A parallel repetition theorem". *Proc. of 27th Annual ACM Symposium on the Theory of Computing*, 447–456, 1995.

[31] R. Raz, S. Safra. "A sub-constant error-probability low-degree test, and sub-constant error-probability PCP characterization of NP". *Proc. of 29th Annual ACM Symposium on Theory of Computing*, 475–484, 1997.

[32] A. Shamir. "IP=PSPACE". *Journal of the ACM, 39:869–877, 1992.*

[33] P. Slavik. "A tight analysis of the greedy algorithm for set cover". *Proc. of 28th Annual ACM Symposium on Theory of Computing*, 435–439, 1996.

[34] A. Srinivasan. "Improved approximations of packing and covering problems". *Proc. of 27th Annual ACM Symposium on the Theory of Computing*, 268–276, 1995.