Electronic Colloquium on Computational Complexity, Report No. 102 (2001)

Using the FGLSS-reduction to Prove Inapproximability Results for Minimum Vertex Cover in Hypergraphs

Oded Goldreich*
Department of Computer Science
Weizmann Institute of Science
Rehovot, ISRAEL.
oded@wisdom.weizmann.ac.il

December 18, 2001

Abstract

Using known results regarding PCP, we present simple proofs of the inapproximability of vertex cover for hypergraphs. Specifically, we show that

1. Approximating the size of the minimum vertex cover in $O(1)$-regular hypergraphs to within a factor of 1.99999 is NP-hard.
2. Approximating the size of the minimum vertex cover in 4-regular hypergraphs to within a factor of 1.49999 is NP-hard.

Both results are inferior to known results (by Trevisan and Holmerin), but they are derived using much simpler proofs. Furthermore, these proofs demonstrate the applicability of the FGLSS-reduction in the context of reductions among combinatorial optimization problems.

Keywords: Complexity of approximation, combinatorial optimization problems, Vertex Cover, PCP, regular hypergraphs.

*Supported by a MINERVA Foundation, Germany.

ISSN 1433-8092

Introduction

This note is inspired by a recent work of Dinur and Safra [4]. Specifically, what we take from their work is the realization that the so-called FGLSS-reduction is actually a general paradigm that can be applied in various ways and achieve various purposes.

The FGLSS-reduction, introduced by Feige, Goldwasser, Lovász, Safra and Szegedy [5], is typically understood as a reduction from languages having certain PCP systems to approximation versions of Max-Clique (or Max Independent Set). The reduction maps inputs (either in or out of the language) to graphs that represent the pairwise consistencies among possible views of the corresponding PCP verifier. It is instructive to think of these possible verifier views as of possible partial solutions to the problem of finding an oracle that makes the verifier accept.

Dinur and Safra apply the same underlying reasoning to derive graphs that represent pairwise consistencies between partial solutions to a combinatorial problem [4]. In fact, they use two different instantiations of this reasoning. Specifically, in one case they start with the vertex-cover problem and consider the restrictions of possible vertex-covers to all possible $O(1)$-subsets of the vertex set. The partial solutions in this case are the vertex-covers of the subgraphs induced by all possible $O(1)$-subsets, and pairwise consistency is defined in the natural way. Thus, we claim that in a sense, the work of Dinur and Safra [4] suggests that the FGLSS-reduction is actually a general paradigm that can be instantiated in various ways. Furthermore, the goal of applying this paradigm may vary too. In particular, the original instantiatation of the FGLSS-reduction by Feige *et. al.* [5] was aimed at linking the class PCP to the complexity of approximating combinatorial optimization problems. In contrast, in the work of Dinur and Safra [4] one instantiation is aimed at deriving instances of very low "degree" (i.e., co-degree at most 2), and the other instantiation is aimed at moving the "gap location" (cf. [12] and further discussion below).

We fear that the complexity of the work of Dinur and Safra [4] may cause researchers to miss the above observation (regarding the wide applicability of the FGLSS-reduction). This would be unfortunate, because we believe in the potential of that observation. In fact, this note grew out of our fascination with the above observation and our attempt to find a simple illustration of it.

Our concrete results: Combining known results regarding PCP with the FGLSS-reduction, we present simple proofs of inapproximability results regarding the *minimum vertex cover problem for hypergraphs*. Specifically, we show that:

1. For every $\epsilon > 0$, approximating the size of the minimum vertex cover in $O(1)$-regular hypergraphs to within a $(2-\epsilon)$-factor is NP-hard (see Section 1). In fact, the hypergraphs we use are $O(\epsilon^{-o(1)})$-regular.

   This result is inferior to Holmerin's result [10], by which approximating vertex cover in 4-regular hypergraphs to within a $(2-\epsilon)$-factor is NP-hard. We also mention Trevisan's result [13], by which for every constant k, approximating vertex cover in k-regular hypergraphs to within a $\Omega(k^{1/19})$-factor is NP-hard. Clearly, in terms of achieving a bigger inapproximation factor, Trevisan's result is superior, but in terms of achieving an inapproximation result for k-regular graphs when k is small (e.g., $k < 2^{19}$) it seems that our result is better.

2. For every $\epsilon > 0$, approximating the size of the minimum vertex cover in 4-regular hypergraphs to within a $(1.5-\epsilon)$-factor is NP-hard (see Section 2).

   Again, this result is inferior to Holmerin's result [10].

We note that our work was done independently of Holmerin's work [10], but after the publication of Trevisan's work [13].

Preliminaries

A k-regular hypergraph is a pair (V, E) such that E is a collection of k-subsets (called hyper-edges) of V; that is, for every $e \in E$ it holds that $e \subseteq V$ and $|e| = k$. For a k-regular hypergraph $H = (V, E)$ and $C \subseteq V$, we say that C is a vertex cover of H if for every $e \in E$ it holds that $e \cap C \neq \emptyset$.

Free-bit complexity and the class $\mathcal{FPCP}$. We assume that the reader is familiar with the basic PCP-terminology (cf. [1, 2, 3] and [6, Sec. 2.4]). (For sake of simplicity we consider non-adaptive verifiers.) We say that the free-bit complexity of a PCP system is bounded by $f : \mathsf{N} \to \mathsf{R}$ if on every input x and any possible random-pad ω used by the verifier, there are at most $2^{f(|x|)}$ possible sequence of answers that the verifier may accept (on input x and random-pad ω). Clearly, the free-bit complexity of a PCP system is bounded by the number of queries it makes, but the former may be much lower. Free-bit complexity is a key parameter in the FGLSS-reduction. For functions $c, s : \mathsf{N} \to [0, 1]$, $r : \mathsf{N} \to \mathsf{N}$ and $f : \mathsf{N} \to \mathsf{R}$, we denote by $\mathcal{FPCP}_{c,s}[r, f]$ the class of languages having PCP systems of completeness bound c, soundness bound s, randomness complexity r and free-bit complexity f. That is, for every input x in the language, there exist an oracle that makes the verifier accept with probability at least $c(|x|)$, whereas for every input x not in the language and every oracle the verifier accepts with probability at most $s(|x|)$.

The FGLSS-graph. For $L \in \mathcal{FPCP}_{c,s}[r, f]$, the FGLSS-reduction maps x to a graph G_x having $2^{r(|x|)}$ layers, each having at most $2^{f(|x|)}$ vertices. The vertices represent possible views of the verifier, where the $N \stackrel{\text{def}}{=} 2^{r(|x|)}$ layers correspond to all possible choices of the random-tape and the vertices in each layer correspond to the up-to $2^{f(|x|)}$ possible sequences of answers that the verifier may accept. The edges represent inconsistencies among these views. In particular, each layer consists of a clique (because only one sequence of answers is possible for a fixed random-tape and a fixed oracle). If the random-tapes $\omega_1, \omega_2 \in \{0, 1\}^{r(|x|)}$ both lead the verifier to make the same query q (and both answers are acceptable) then the corresponding layers will have edges between vertices encoding views in which different answers are given to query q. In case $x \in L$ the graph G_x will have an independent set of size $c(|x|) \cdot N$, whereas if $x \notin L$ then the maximum independent set in G_x has size at most $s(|x|) \cdot N$. Thus, the inapproximability factor for the maximum independent set problem shown by such a reduction is $c(|x|)/s(|x|)$, and the fact the maximum independent set is always at most a $2^{-f(|x|)}$ fraction of the size of G_x does not effect the gap. However, inapproximability factor for the minimum vertex cover shown by such a reduction is

$$\frac{2^{f(|x|)} \cdot N - s(|x|) \cdot N}{2^{f(|x|)} \cdot N - c(|x|) \cdot N} = \frac{2^{f(|x|)} - s(|x|)}{2^{f(|x|)} - c(|x|)} < \frac{2^{f(|x|)}}{2^{f(|x|)} - 1} \tag{1}$$

This is the reason that while the FGLSS-reduction allows to establish quite optimal inapproximability factors for the maximum independent set problem, it failed so far to establish optimal inapproximability factors for the minimum vertex cover problem (although, it was used by Hastad [8] in deriving the 7/6 hardness factor by using Eq. (1) with $f = 2$, $c \approx 1$ and $s = 1/2$). In a sense, the gap between the size of the maximum independent set of G_x for $x \in L$ versus for $x \notin L$ is at the "right" location for establishing inapproximability factors for the maximum independent set problem, but is at the "wrong" location for establishing inapproximability factors for the minimum vertex cover problem. In a sense, what we do below is "move the gap location": Specifically, in Section 1, we take a maximum independent set gap of $c2^{-f}$ versus $s2^{-f}$ (which means a minimum vertex cover gap of $1 - c2^{-f}$ versus $1 - s2^{-f}$), and transform it into a minimum vertex cover gap of $(2 - c) \cdot 2^{-f}$ versus $(2 - s) \cdot 2^{-f}$).

1 A $2-\epsilon$ Hardness Factor for $O(1)$-Regular Hypergraphs

We start with the usual FGLSS-graph, denoted G, derived from the FGLSS-reduction as applied to input x of a $\mathcal{FPCP}_{1-\epsilon,s}[\log, f]$ scheme (for a language in $\mathcal{NP}$). For simplicity, think of f as being a constant such that 2^f is an integer. Without loss of generality, each layer of G has $\ell = 2^f$ vertices.

We now apply the "FGLSS paradigm" by considering vertex-covers of G, and their projection on each layer. Such projections (or "partial assignments") have either ℓ or $\ell-1$ vertices. We focus on the good vertex covers, having exactly $\ell-1$ vertices in each projection. Thus, for each such $(\ell-1)$-subset we introduce a vertex in the hypergraph, to be denoted H. That is, for layer $L = (v_1, ..., v_\ell)$ in G, we introduced a corresponding layer in H containing ℓ vertices such that each H-vertex corresponds to an $(\ell-1)$-subset of L; that is, we introduce ℓ vertices that correspond to $L \setminus \{v_1\}$,...,$L \setminus \{v_\ell\}$. For each pair of layers $L' = (v'_1, ..., v'_\ell)$ and $L'' = (v''_1, ..., v''_\ell)$ if (v'_i, v''_j) is an edge in G then we introduce the $2 \cdot (\ell-1)$-hyperedge containing the H-vertices that correspond to the subsets $\{L' \setminus \{v'_k\} : k \neq i\}$ and $\{L'' \setminus \{v''_k\} : k \neq j\}$; that is, the hyper-edge consists of all the H-vertices of these two layers except for the two that correspond to the subsets $L' \setminus \{v'_i\}$ and $L'' \setminus \{v''_j\}$. In addition, for each layer in H, we introduce an ℓ-size hyper-edge containing all ℓ vertices of that layer.

To get rid of the non-regularity of this construction, we augment each layer with a sets of $\ell-2$ auxiliary vertices, and augment the abovementioned ℓ-size hyper-edge by a hyper-edge containing all vertices of that layer (i.e., the original ℓ vertices as well as the $\ell-2$ auxiliary vertices). This completes the construction of H.

Fixing any input x, we consider the corresponding FGLSS-graph $G = G_x$, and the hypergraph $H = H_x$ derived from G by following the above construction. Let N denote the number of layers in G (and H).

Claim 1.1 *If x is in the language then the hypergraph H_x has a vertex-cover of size at most $(1+\epsilon) \cdot N$.*

Proof: Since x is in the language the graph $G = G_x$ has an independent set (IS) of size at least $(1-\epsilon) \cdot N$. Consider this IS or actually the corresponding vertex-cover (i.e., VC) of G. Call a layer in G good if it has $\ell-1$ vertices in this VC, and note that at least $(1-\epsilon) \cdot N$ layers are good. We create a vertex-cover for $H = H_x$ as follows. For each good layer, place in C the corresponding H-vertex; that is, the H-vertex corresponding to the $(\ell-1)$-subset (of this layer in G) that is in the VC of G. For the rest of the layers (i.e., the non-good layers), place in C any two H-vertices of each (non-good) layer.

In total we placed in C at most $(1-\epsilon)N + 2\epsilon N = (1+\epsilon)N$ vertices. We show that C is a vertex cover of H by considering all possible hyper-edges, bearing in mind the correspondence between layers of G and layers of H.

- The intra-layer hyper-edges of H are definitely covered because we placed in C at least one H-vertex from each layer.
- Each hyper-edge connecting H-vertices from two good layers is covered.

  This is shown by considering the edge, denoted (x, y), of G that is "responsible" for the introduction of each hyper-edge (in H). Since we started with a vertex cover of G, either x or y must be in that cover. Suppose, without loss of generality, that x is in the VC of G. Then, we must have placed in C one of the H-vertices that corresponds to a $(\ell-1)$-subset that contains x. But then this H-vertex covers the said hyper-edge (because the latter contains all $(\ell-1)$-subsets that contain x).

- Hyper-edges containing H-vertices from non-good layers are covered because we placed in C two H-vertices from such layers, whereas each hyper-edge containing H-vertices of some layer contains all but one vertex of that layer.

The claim follows. ■

Claim 1.2 *If x is not in the language then every vertex-cover of the hypergraph H_x has size at least* $(2-s(|x|))\cdot N$.

Proof: Consider any vertex cover C of H. Note that due to the intra-layer hyper-edges, C must contain at least one vertex in each layer. Furthermore, without loss of generality, C contains only original vertices (rather than the $\ell-2$ auxiliary vertices added to each layer). Denote by C' the set of layers that have a single vertex in C. Then, $|C| \geq |C'| + 2(N - |C'|) = 2N - |C'|$. The claim follows by proving that $|C'| \leq sN$, where $s \stackrel{\text{def}}{=} s(|x|)$.

Suppose, towards the contradiction, that $|C'| > sN$. We consider the set of G-vertices, denoted I, that correspond to the (single) H-vertices in these layers; that is, for layer L (in C') such that C contains the H-vertex (which corresponds to) $L \setminus \{v\}$, place $v \in G$ in I. We show that I is an independent set in G (and so derive a contradiction to $G = G_x$ not having an independent set of size greater than sN, because x is a no-instance). Specifically, for every $u, v \in I$, we show that (u,v) cannot be an edge in G. Suppose (u,v) is an edge in G, then the corresponding hyper-edge in H cannot be covered by C; that is, the hyper-edge $\{L\setminus\{w\} : w \neq u\} \cup \{L'\setminus\{w\} : w \neq v\}$ (which must be introduced due to the edge (u,v)) cannot be covered by the H-vertices that correspond to the $(\ell-1)$-subsets $L\setminus\{u\}$ and $L'\setminus\{v\}$. The claim follows. ■

Conclusion: Starting from a $\mathcal{FPCP}_{1-\epsilon,s}[\log, f]$ system for $\mathcal{NP}$, we have shown that the minimum vertex-cover in $(2^{f+1}-2)$-regular hypergraphs is NP-hard to approximate to a $(2-s)/(1+\epsilon)$-factor. Now, if we start with any $\mathcal{FPCP}_{1,s}[\log, f]$ for $\mathcal{NP}$, with $s \approx 0$, then we get a hardness result for a factor of $2 - s \approx 2$. Any $\mathcal{NP} \subseteq \mathcal{PCP}[\log, O(1)]$ result (starting from [1]) will do for this purpose, because a straightforward error-reduction will yield $\mathcal{NP} \subseteq \mathcal{FPCP}_{1,s}[\log, O(1)]$, for any $s > 0$. The (amortized) free-bit complexity only effects the growth of the hyper-edge size as a function of the deviation of the hardness-factor from 2. Specifically, if we start with an "amortized free-bit complexity zero" result (i.e., $\mathcal{NP} \subseteq \mathcal{FPCP}_{1,s}[\log, o(\log_2(1/s))]$ for every $s > 0$) then we get a factor of $2-s$ hardness for $(1/s)^{o(1)}$-regular hypergraphs. That is, starting with Hastad's first such result [7] or from the simplest one currently known [9], we state the above as our first little result:

Theorem 1.3 *For every $\epsilon > 0$, approximating the size of the minimum vertex cover in $\epsilon^{-o(1)}$-regular hypergraphs to within a $(2-\epsilon)$-factor is NP-hard.*

Alternatively, if we start with Hastad's "maxLIN3 result" [8] (i.e., $\mathcal{NP} \subseteq \mathcal{FPCP}_{1-\epsilon,0.5}[\log, 2]$ for every $\epsilon > 0$), then we get a hardness factor of $(2-0.5)/(1+\epsilon) \approx 1.5$ for 6-regular hypergraphs. Below we show that the same hardness factor holds also for 4-uniform hypergraphs (by starting with the same "maxLIN3 result" [8] but capitalizing on an additional property of it).

2 A $1.5 - \epsilon$ Hardness Factor for 4-Regular Hypergraphs

We start with the FGLSS-graph derived from applying the FGLSS-reduction to Hastad's "maxLIN3 system" [8]; that is, the $\mathcal{FPCP}_{1-\epsilon,0.5}[\log, 2]$ system for $\mathcal{NP}$ ($\forall\epsilon > 0$). The key observation is that in this system for any two queries all four answer pairs are possible (as accepting configurations). This observation is relied upon when establishing (below) simple structural properties of the derived FGLSS-graph.

As before, there will be a correspondence between the vertex set of G and the vertex set of H. Here it is actually simpler to just identify the two sets. So it just remains to specify the hyper-edges of H. Again, we place hyper-edges between all (i.e., four) vertices of each layer. As for the construction of inter-layer hyper-edges, we consider three cases regarding each pair of layers:

1. *The trivial case*: In case there are no edges between these two layers in G, there would be no hyper-edges between these layers in H. This case corresponds to the case that these two layers correspond to two random-tapes that induce two query sets with empty intersection.

2. *The interesting case* is when these two layers correspond to two random-tapes that induce two query sets having a *single* query, denoted q, in common. Relying on the property of the starting PCP system, it follows that both answers are possible to this query and that each possible answer is represented by two vertices in each corresponding layer. Accordingly, we denote the vertices of the first layer by $u_1^0, u_2^0, u_1^1, u_2^1$, where u_i^b is the ith configuration in this layer in which query q is answered by the bit b. Similarly, denote the vertices of the second layer by $v_1^0, v_2^0, v_1^1, v_2^1$. (We stress that this notation is used only for determining the hyper-edges between the current pair of layers, and when considering a different pair of layers a different notation may be applicable.) In this case we introduce the two hyper-edges $\{u_1^0, u_2^0, v_1^1, v_2^1\}$ and $\{u_1^1, u_2^1, v_1^0, v_2^0\}$.

   Intuition: Note that the edges in G between these two layers are two $K_{2,2}$'s (i.e., for each $b \in \{0, 1\}$, between the two u_i^b's on one side and the two v_i^{1-b}'s on the other side). These two $K_{2,2}$'s enforce that if some u_i^b is in some IS then v_j^{1-b} is not in the IS. For a H-VC having a single vertex in each layer, the (two) hyper-edges will have the same effect.

3. *The annoying case* is when these two layers (correspond to two random-tapes that induce two query sets that) have *two* or more queries in common. In this case, we label the vertices in these two layers according to these two answers; that is, we denote the four vertices of the first layer by $u_{0,0}, u_{0,1}, u_{1,0}, u_{1,1}$, where $u_{a,b}$ is the unique configuration in this layer in which these two queries are answered by a and b, respectively. Similarly, denote the vertices of the second layer by $v_{0,0}, v_{0,1}, v_{1,0}, v_{1,1}$. (Again, this notation is used only for determining the hyper-edges between the current pair of layers.) In this case, we introduce four hyper-edges between these two layers, each has one vertex of the first layer and the three "non-matching" vertices of the second layer; that is, the hyper-edges are $\{u_{a,b}, v_{a,1-b}, v_{1-a,b}, v_{1-a,1-b}\}$, for $a, b \in \{0, 1\}$.

   Intuition: The pair $(u_{a,b}, v_{a',b'})$ is an edge in G if and only if either $a \neq a'$ or $b \neq b'$. Similarly, the pair $(u_{a,b}, v_{a',b'})$ participates in an hyper-edge of H if and only if either $a \neq a'$ or $b \neq b'$.

This completes the construction. Note that $H = H_x$ is a 4-regular hypergraph.

Claim 2.1 *If x is in the language then the hypergraph H_x has a vertex-cover of size at most $(1 + 3\epsilon) \cdot N$, where N denotes the number of layers.*

Proof: Since x is in the language the graph $G = G_x$ has an independent set (IS) of size $(1 - \epsilon)N$. Consider such an IS, denoted I. Call a layer in G good if it has a vertex in I, and note that at least $(1 - \epsilon)N$ layers are good. Augment I by the set of all vertices residing in non-good layers. In total we took at most $(1 - \epsilon)N + 4\epsilon N = (1 + 3\epsilon)N$ vertices. We show that these vertices cover all hyper-edges of H.

- The intra-layer hyper-edges are definitely covered (since we took at least one vertex from each layer).
- Each hyper-edge connecting vertices from two good layers is covered.

  This is shown by considering each of the two non-trivial cases (in the construction). In the interesting case, I (having a single vertex in each good layer) must have a single vertex in *each* $K_{2,2}$. But then this vertex covers the corresponding hyper-edge. In the annoying case, I (having a single vertex in each good layer) must contain vertices with matching labels in these two layers. But then these two vertices cover all 4 hyper-edges, because each hyper-edge contains a (single) vertex of each label.
- Hyper-edges containing H-vertices from non-good layers are covered trivially (because we took all vertices of each non-good layer).

The claim follows. ■

Claim 2.2 *If x is not in the language then every vertex-cover of the hypergraph H_x has size at least $1.5 \cdot N$.*

Proof: Consider a cover C of H. Note that (due to the intra-layer hyper-edges) C must contain at least one vertex in each layer. Denote by C' the set of layers that have a single vertex in C. Then, $|C| \geq |C'| + 2(N - |C'|)$. The claim follows by proving that $|C'| \leq 0.5N$. Suppose, towards the contradiction, that $|C'| > 0.5N$. Consider the set of vertices, denoted I, that correspond to these layers (i.e., for a layer in C' consider the layer's vertex that is in C). We show that I is an independent set in G (and so we derive contradiction).

Suppose that $u, v \in I$ and (u, v) is an edge in G. In the interesting case, this (i.e., (u, v) being an edge in G) means that u and v are in the same hyper-edge in H, and being the only vertices in C that are in these layers, no vertex covers the other (vertex-disjoint) hyper-edge between these layers. In the annoying case, this (i.e., (u, v) being an edge in G) means that u and v do not have the same label and one of the four hyper-edges in H cannot be covered by them; specifically, without loss of generality, suppose that u is in the first layer, then neither $v = v_{a,b}$ nor $u \neq u_{a,b}$ covers the hyper-edge $\{u_{a,b}, v_{a,1-b}, v_{1-a,b}, v_{1-a,1-b}\}$. ■

Conclusion: Starting from the abovementioned $\mathcal{NP} \subseteq \mathcal{FPCP}_{1-\epsilon,0.5}[\log, 2]$ result of Hastad [8], we have shown that the minimum vertex-cover in 4-regular hypergraphs is NP-hard to approximate to a factor of $1.5/(1 + 3\epsilon)$. Let us state this as our second little result:

Theorem 2.3 *For every $\epsilon > 0$, approximating the size of the minimum vertex cover in 4-regular hypergraphs to within a $(1.5 - \epsilon)$-factor is NP-hard.*

Postscript

Following this work, Holmerin has applied related FGLSS-type reductions to different PCP systems and obtained improved inapproximability results for vertex cover in hypergraphs [11]. Specifically, for every $\epsilon > 0$, he showed that:

1. Approximating the size of the minimum vertex cover in k-regular hypergraphs to within a factor of $\Omega(k^{1-\epsilon})$ is NP-hard.

2. Approximating the size of the minimum vertex cover in 3-regular hypergraphs to within a factor of $1.5 - \epsilon$ is NP-hard.

Acknowledgments

We are grateful to Johan Hastad for referring us to the works of Trevisan [13] and Holmerin [10].

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof Verification and Intractability of Approximation Problems. *JACM*, Vol. 45, pages 501–555, 1998. Preliminary version in *33rd FOCS*, 1992.

[2] S. Arora and S. Safra. Probabilistic Checkable Proofs: A New Characterization of NP. *JACM*, Vol. 45, pages 70–122, 1998. Preliminary version in *33rd FOCS*, 1992.

[3] M. Bellare, O. Goldreich and M. Sudan. Free Bits, PCPs and Non-Approximability – Towards Tight Results. *SICOMP*, Vol. 27, No. 3, pages 804–915, 1998.

[4] I. Dinur and S. Safra. The Importance of Being Biased. Manuscript, 2001.

[5] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Approximating Clique is almost NP-complete. *JACM*, Vol. 43, pages 268–292, 1996. Preliminary version in *32nd FOCS*, 1991.

[6] O. Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness. Algorithms and Combinatorics series* (*Vol. 17*), *Springer*, 1999.

[7] J. Hastad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, Vol. 182, pages 105–142, 1999. Preliminary versions in *28th STOC* (1996) and *37th FOCS* (1996).

[8] J. Hastad. Some optimal in-approximability results. In *29th STOC*, pages 1–10, 1997.

[9] J. Hastad and S. Khot. Query efficient PCPs with Perfect Completeness. In *42nd FOCS*, pages 610–619, 2001.

[10] J. Holmerin. Vertex Cover on 4-regular Hypergraphs is Hard to Approximate within $2 - \epsilon$. TR01-094, *ECCC*, 2001.

[11] J. Holmerin. Improved Inapproximability Results for Vertex Cover on k-regular Hypergraphs. Manuscript, 2001.

[12] E. Petrank. The Hardness of Approximations: Gap Location. *Computational Complexity*, Vol. 4, pages 133–157, 1994.

[13] L. Trevisan. Non-approximability Results for Optimization Problems on Bounded-Degree Instances. In *33rd STOC*, pages 453–461, 2001.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'