Electronic Colloquium on Computational Complexity - Reports Series 1996 - available via:
ECCC
TR96-055
FTP: ftp.eccc.uni-trier.de:/pub/eccc/
WWW: http://www.eccc.uni-trier.de/eccc/
Email: ftpmail@ftp.eccc.uni-trier.de with subject 'help eccc'

Hitting Properties of Hard Boolean Operators and their Consequences on BPP

Alexander E. Andreev
University of Moscow

Andrea E. F. Clementi*
University of Rome

José D. P. Rolim
University of Geneva

October 22, 1996

Abstract

We present the first worst-case hardness conditions on the circuit complexity of EXP functions which are sufficient to obtain $P = BPP$. In particular, we show that from such hardness conditions it is possible to construct quick Hitting Sets Generators with logarithmic prize. As proved in [8], such generators can efficiently derandomize any BPP-algorithm.

*Contact author: Dip. di Scienze dell'Informazione, Universitá di Roma "La Sapienza", Via Salaria 113, 00198 Roma. E-mail: clementi@dsi.uniroma1.it

1 Introduction

- *Motivations and previous results.* A major goal in complexity theory is the study of the real power of randomized algorithms. To this aim, several recent studies have been devoted to the area of derandomization, i.e., the design of general methods that permit an efficient deterministic simulation of algorithms which make use of random bits. A central question in this area is the relationship between the existence of computationally-hard functions and the existence of efficient derandomization methods. Yao [26], and Blum and Micali [10] introduced the concept of *Pseudo-Random Generators* (PSRG's), boolean operators that stretch a short truly random sequence of bits into a long string of bits that "looks" random to any machine which has limited computational power. More formally, A PSRG is a boolean operator

$$G \;=\; \{\; G_n \;:\; \{0,1\}^{k(n)} \rightarrow \{0,1\}^n \;,\; n > 0 \;\},$$

denoted by $G : k(n) \rightarrow n$, that, for a.e. n and for any boolean circuit $C : \{0,1\}^n \rightarrow \{0,1\}$ whose size is at most n, we have:

$$|\mathbf{Pr}\,(C(\vec{y}) = 1) - \mathbf{Pr}\,(C(G_n(\vec{x})) = 1)| \leq \frac{1}{n}$$

(where $\vec{y}$ is chosen uniformly at random in $\{0,1\}^n$, and $\vec{x}$ in $\{0,1\}^{k(n)}$). According to the definition used in [23], a boolean operator $Op : k(n) \rightarrow n$ is *quick* if it can be computed in time polynomial in n (note in passing that if $k(n) = O(\log n)$ then the "quick" condition implies that Op belongs to EXP).

Nisan and Wigderson [23] showed a method to construct *quick* PSRG's based on the existence of boolean functions in EXP that have exponential *hardness* [22, 23]. The hardness condition used by Nisan and Wigderson requires the existence of a function in EXP that not only has a hard *worst-case* circuit complexity[1] but also a hard *average-case* circuit complexity. More formally, a function $f : \{0,1\}^n \rightarrow \{0,1\}$ is (ϵ, L)-*hard* if for any circuit C of size at most L

$$|\mathbf{Pr}\,(C(\vec{x}) = f(\vec{x})) \;-\; \frac{1}{2}| \leq \frac{\epsilon}{2}.$$

Given a boolean function $F = \{F_n : \{0,1\}^n \rightarrow \{0,1\},\; n > 0\}$, the *hardness* at n $H_F(n)$ of F is defined as the maximum integer h_n such that F_n is $(1/h_n, h_n)$-hard. Then, F has exponential hardness if $H_F(n) \geq 2^{\Omega(n)}$. Nisan and Wigderson [23] showed the following fundamental result.

Theorem 1.1 *If a boolean function F exists such that* i) $F \in EXP$, *and* ii) F *has exponential* hardness, *then there exists a quick PSRG $G : k(n) \rightarrow n$ where $k(n) = O(\log n)$, and consequently $BPP = P$.*

The above theorem provides an explicit result on the "hardness vs randomness" trade-offs. Furthermore, more recent results on the hardness of boolean functions [2, 6] led to a rather general opinion that the class EXP actually contains functions having exponential hardness, thus providing positive indications on the Nisan and Wigderson's conjecture claiming that the gap between deterministic and randomized computational power is not large at least in a pure theoretical framework. As observed above, the hardness required by Nisan and Wigderson's construction of quick PSRG's refers to average-case complexity. The research over the last ten years suggests that determining strong

[1] As circuit complexity of a finite boolean function f, we will always mean the size of the smallest circuit that computes f.

hardness results on the average-case complexity for uniform, recursive languages seems to be a much harder task than that in the case of worst-case complexity (it is sufficient to observe, for instance, the enormous difference between the number of known NP-hard problems and the number of known RNP-hard problems [15, 14, 11]). This paper thus investigates the following central question: which "worst-case" hardness assumption should a function satisfy in order to yield an efficient derandomization method (i.e. to obtain $P = BPP$)? We give two answers to this question. Both answers make use of a general method to derandomize algorithms (different from PSRG's) that has been recently introduced by Andreev *et al* in [7]. This method is based on quick *Hitting Set Generators* ([24, 16, 5]).

Definition 1.1 *Let $\epsilon(n)$ and $\beta(n)$ be polynomial-time computable functions such that, for any $n \geq 1$, $0 < \epsilon(n) < 1$ and $n \leq \beta(n) \leq 2^n$. Then, a boolean operator $H : k(n) \rightarrow n$ is an $(\epsilon(n), \beta(n))$-*Hitting Set Generator *(in short, $(\epsilon(n), \beta(n))$-HSG) if, for any boolean circuit C such that $L(C) \leq \beta(n)$ and $\mathbf{Pr}\,(C = 1) \geq \epsilon(n)$, H is required to provide one "example" $\vec{y}$ for which $C(\vec{y}) = 1$, i.e., there exists $\vec{a} \in \{0,1\}^{k(n)}$ such that $C(H_n(\vec{a})) = 1$.*

Observe first that any quick PSRG is also a quick $(1/n, n)$-HSG but the *converse* is not true. Informally speaking, a PSRG provides a precise approximation of the value $\mathbf{Pr}\,(C(\vec{y}) = 1)$, i.e., the fraction of 1's in the output of C, for any "small" circuit C. Thus, if C has a large fraction of 1's in its output then the PSRG must generate an input space for which this fraction has about the same large size (see also the definition of *Discrepancy Sets* [9]). On the other hand, HSG's are not required to have this property: a HSG provides, for any "small" circuit C having a "large" number of $1's$ in its output, only a witness of the fact that C is not a *null* function (for a further analysis of the differences between HSG's and PSRG's see [7]). In particular, Andreev *et al* proved the following result.

Theorem 1.2 *Let $k(n) = O(\log n)$ and ϵ be any constant such that $0 < \epsilon < 1$. If there exists a quick (ϵ, n)-HSG $H : k(n) \rightarrow n$ then $BPP = P$.*

- *Our results.* Given any boolean sequence $\vec{x} \in \{0,1\}^n$, we will use the term *complexity of $\vec{x}$* to refer to the circuit complexity of the corresponding boolean function $x : \{0,1\}^{\lceil \log(n+1) \rceil} \rightarrow \{0,1\}$ where $x(i)$ is the i-th bit of $\vec{x}$. The circuit complexity of a finite function f (a finite sequence $\vec{x}$) will be denoted as $L(f)$ ($L(\vec{x})$). The circuit complexity of a boolean operator H will be denoted as $L^{op}(H)$.

In this paper, we give two worst-case complexity conditions which are sufficient to construct quick HSG's that satisfy Theorem 1.2, thus obtaining $P = BPP$. The first condition deals with the worst-case circuit-complexity of characteristic functions of sets generated by boolean operators. The construction of this quick HSG envolves the use of *expanders* graphs [3, 17, 20]. An undirected graph $G(V, E)$ is a (d, c)-*expander* if the maximum degree of a vertex is d, and for every set $W \subseteq V$ of cardinality $|W| \leq |V|/2$, the inequality $|N(W) - W| \geq c|W|$ holds, where $N(W)$ denotes the set of all vertices adjacent to some vertex in W. Expanders share many of the properties of sparse *random* graphs, and have been strongly used in several applications. The expanding properties of a graph can be established by determining the value of its second largest eigenvalue (see for example [4]). Indeed, if λ is an upper bound on the second largest eigenvalue of any d-regular graph $G(V, E)$, then G is a (d,c)-expander for $c = (d - \lambda)/2d$.

In [17], a polynomial-time algorithm is shown which, given $n > 0$, and $d \leq n$, constructs a d'-regular expanders G such that $d' = O(d)$, $|V| = O(n)$ and its second largest eigenvalues $\lambda > 0$ is such that $\lambda \leq 2\sqrt{d-1}$ (such families of graphs are called *Ramanujan* graphs).

Using the expanding properties of Ramanujan graphs, we can demonstrate the following result.

Theorem 1.3 *Let d and λ be as in the above definition of Ramanujan graphs. Let δ be any positive constant and let $k(n) = (1 + \Theta(1)) \log n$. If there exists a quick operator*

$$H = \{H_n : \{0,1\}^{k(n)} \to \{0,1\}^n , n = 1, 2, ...\} ,$$

such that the characteristic function of its output sets

$$F^H = \{F_n^H : \{0,1\}^n \to \{0,1\} , \text{ where } F_n^H(\vec{x}) = 1 \text{ iff } \exists \vec{y} \in \{0,1\}^{k(n)} s.t. \ H_n(\vec{y}) = \vec{x}, \ n > 0\}$$

satisfies

$$L(F_n^H) \geq \left(\frac{\log(4\lambda)}{\log d} + \delta \right) \frac{2^{k(n)} n}{k(n) + \log n} ,$$

then it is possible to construct a quick operator $H' : \{0,1\}^{k'(n)} \to \{0,1\}^n$ where $k'(n) = \Theta(\log n)$ and H' is an $(1 - \epsilon, n)$-HSG for some positive constant $0 < \epsilon < 1$, thus obtaining $P = BPP$.

Observe that if $L^{op}(k, n)$ denotes the worst-case circuit complexity of boolean operators $H : k(n) \to n$, then it is known that [19, 25], for any $\log n \leq k \leq n$,

$$L^{op}(k, n) = (1 + o(1)) \frac{2^k n}{k + \log n} .$$

Furthermore, for a.e. boolean operator $H : k \to n$, we have $L^{op}(H) = \Theta((2^k n)/(k + \log n))$.

The second sufficient condition to obtain a quick HSG is stronger but refers directly to the worst-case circuit complexity of a boolean operator instead of the characteristic functions of its ouput sets.

Theorem 1.4 *Let $k(n) = \Theta(\log n)$. Let $H : k(n) \to n$ be a quick operator such that for a.e. n,*

$$L^{op}(H_n) \geq L^{op}(k, n) - \frac{2^{k(n)}}{k(n)^2} .$$

Then, for any constant $0 < \epsilon < 1$, and for any positive integer q, it is possible to construct a quick $(1 - \epsilon, n^q)$-HSG $H' : k'(n) \to n$, where $k'(n) = \Theta(\log n)$, thus obtaining $P = BPP$.

Organization of the paper. In Section 2, we provide an asymptotically-optimal complexity bound for partial boolean functions which depends on the number of inputs on which they ouput 1. This bound is the generalization of Lupanov's result [19] which holds only for total boolean functions. In Section 3, we describe the construction of the HSG for proving Theorem 1.3. Finally, in Section 4 we describe the construction of our second HSG that allows us to prove Theorem 1.4.

2 A preliminary result on the circuit complexity of partial boolean functions

One of the key ingredients in deriving the HSG's in both of our theorems consists of a new precise bound on the Shannon function describing the trade-offs between the worst-case circuit complexity of partial boolean functions and the number of inputs on which they ouput 1.

Let $\mathcal{F}(n,N,m)$ be the set of all partial boolean functions $f(x_1,\ldots,x_n)$ defined on $N \leq 2^n$ inputs and assuming 1 on $m \leq N$ inputs. Furthermore, $L(n,N,m)$ denotes the worst-case circuit complexity of functions in $\mathcal{F}(n,N,m)$. Lupanov [19] proved an optimal bound for $L(n,N,m)$ when $N = 2^n$ (i.e. for total functions). In order to use this bound in the construction of our HSG's, we instead require the precise bound for partial boolean functions.

The method used to derive the above upper bound is based on a probabilistic construction of *linear operators* having some new variants of the "well-distibution" property previously shown in [6] to obtain optimal bounds on the circuit complexity of approximating boolean functions.

Definition 2.1 *A boolean function $l : \{0,1\}^n \to \{0,1\}$ is linear if it can be represented in the following way:*

$$l(x_1,\ldots,x_n) = \alpha_1 x_1 \oplus \ldots \oplus \alpha_n x_n \oplus \beta \ ,$$

where $\alpha_1,\ldots,\alpha_n,\beta$ are boolean constants. The set of all linear functions with n variables is denoted as $\mathcal{L}_n$.

Moreover, a vector function $\vec{l} = (l_1, l_2, ..., l_s) \in (\mathcal{L}_n)^s$ $(s \geq 1)$ is called *linear operator*. The circuit complexity of linear operators has been studied in [21]. In particular, we will use the following result.

Theorem 2.1 [21] *For any linear operator $\vec{l} = (l_1,\ldots,l_s) \in (\mathcal{L}_n)^s$ $(s \geq 1)$, we have*

$$L(\vec{l}) \ = \ O\left(\frac{ns}{\log n}\right) + O(n) \ .$$

Let $\mathcal{F}(n,N,m)$ be the set of all partial boolean functions $f(x_1,\ldots,x_n)$ defined on $N \leq 2^n$ inputs and assuming 1 on $m \leq N$ inputs. Furthermore, $L(n,N,m)$ denotes the worst-case circuit complexty of functions in $\mathcal{F}(n,N,m)$. Lupanov demonstrated the following result for the case of total boolean functions.

Theorem 2.2 [19] *Let $L^{tot}(n,m) = L(n,2^n,m)$. Then*

$$L^{tot}(n,m) \ = \ (1+o(1))\frac{\log\left(\begin{array}{c}2^n\\ m\end{array}\right)}{\log\log\left(\begin{array}{c}2^n\\ m\end{array}\right)} \ .$$

Theorem 2.3

$$L(n,N,m) \ = \ (1+o(1))\frac{\log\left(\begin{array}{c}N\\ m\end{array}\right)}{\log\log\left(\begin{array}{c}N\\ m\end{array}\right)} + O(n) \ .$$

Proof. For the sake of brevity, we will prove the theorem only in the restricted case

$$n^{1+\epsilon} \leq m \leq n^{O(1)} \ \text{ for some } \ \epsilon > 0, \ \text{ and } \ N = 2^{\Omega(n)} \ , \tag{1}$$

that is what we need to contruct our HSG's in the next sections. The proof of the general case will be given in the full version of this paper.

The proof consists of a reduction from our case to that of total functions for which we can apply Theorem 2.2. Consider a partial boolean function $f(x_1, \ldots, x_n)$, let M_α be the set of vector $\vec{a} \in \{0,1\}^n$ for which $f(\vec{a}) = \alpha$. We have $|M_1| = m$ and $|M_0| = N - m$. Consider a randomly chosen linear operator (with uniform distribution) $\vec{l} = (l_1, \ldots, l_k) \in (\mathcal{L}_n)^k$ where $k = \lceil \log N + \log m \rceil + 2$. Then, observe that for any choice of two fixed elements $\vec{a}, \vec{b} \in \{0,1\}^n$ such that $\vec{a} \neq \vec{b}$, we have

$$\mathbf{Pr}\left(\vec{l}(\vec{a}) = \vec{l}(\vec{b})\right) = 2^{-k} .$$

Consequently,

$$\mathbf{Pr}\left(\exists \vec{a} \in M_1 \ \exists \vec{b} \in M_0 \ : \ \vec{l}(\vec{a}) = \vec{l}(\vec{b}) \ \right) \leq |M_0| * |M_1| * 2^{-k} \leq 1/4 .$$

Hence, for the negation of the above event we have

$$\mathbf{Pr}\left(\forall \vec{a} \in M_1 \ \forall \vec{b} \in M_0 \ : \ \vec{l}(\vec{a}) \neq \vec{l}(\vec{b}) \ \right) \geq 3/4 .$$

From the above probabilistic argument, we can state that there exists $\vec{l} \in (\mathcal{L}_n)^k$ such that

$$\forall \vec{a} \in M_1 \ \forall \vec{b} \in M_0 \ : \ \vec{l}(\vec{a}) \neq \vec{l}(\vec{b}) \tag{2}$$

We define the total boolean function $g(y_1, \ldots, y_k)$ as follows

$$g(\vec{y}) = 1 \text{ iff } \exists \ \vec{a} \in M_1 \text{ such that } \vec{l}(\vec{a}) = \vec{u}.$$

Note that if f is defined on $\vec{a}$ then $f(\vec{a}) = g(\vec{l}(\vec{a}))$; Then, using Theorem 2.1, we obtain

$$L(f) \leq L(\vec{l}) + L(g) \leq O(n(\log N + \log m + 2)) + L(g) \leq O(n^2) + L(g) .$$

Furthermore, Condition 1 implies that

$$\log \left(\begin{array}{c} N \\ m \end{array} \right) = (1 + o(1)) m \log N \ , \ \text{ and } \ n^2 = o\left(\frac{\log \left(\begin{array}{c} N \\ m \end{array} \right)}{\log \log \left(\begin{array}{c} N \\ m \end{array} \right)}\right) .$$

Since g is a total function we can apply Theorem 2.2, i.e.

$$L(g) \leq (1 + o(1)) \frac{\log \left(\begin{array}{c} N \\ m \end{array} \right)}{\log \log \left(\begin{array}{c} N \\ m \end{array} \right)} ,$$

and consequently

$$L(f) \leq O(n^2) + L(g) \leq (1 + o(1)) \frac{\log \left(\begin{array}{c} N \\ m \end{array} \right)}{\log \log \left(\begin{array}{c} N \\ m \end{array} \right)} .$$

Note for the general case. If we remove Condition 1, a harder proof is required since we need to derive a stronger bound on the circuit complexity of $\vec{l}$. For general linear operators this is not possible and we have to show a new special probabilistic construction. □

3 Hard characteristic functions and HSG's

The following theorem provides a first trade-offs between the hardness of characteristic functions of boolean subsets and their hitting properties.

Theorem 3.1 *Let $0 < c_2 < 1$ be a constant, and let $S_n \subseteq \{0,1\}^n$ be any subset such that $|S_n| \leq b_n$, where $b_n = n^{\Theta(1)}$. Suppose that for the characteristic function F_n of S_n we have*

$$i) \quad L(F_n) \geq c_2 \frac{b_n n}{\log b_n + \log n} .$$

Then, for any constant c_1, such that $0 < c_1 < c_2$, for any boolean function $f(x_1, \ldots, x_n)$ such that

$$ii) \ \mathbf{Pr}\,(f = 1) \geq 1 - 2^{(c_1 - 1)n} , \quad and \quad iii) \ L(f) \leq b_n ,$$

there exists $\vec{a} \in S_n$ for which $f(\vec{a}) = 1$.

Proof. Suppose, by contradiction, that f satisfies conditions $ii)$ and $iii)$ but for any $\vec{a} \in S_n$ we have $f(\vec{a}) = 0$. Let $Z \subseteq \{0,1\}^n$ be the subset of all inputs on which $f = 0$. Clearly, we have $S_n \subseteq Z \subseteq \{0,1\}^n$. Then consider the partial boolean function $g(x_1, \ldots, x_n)$ defined as follows:

$$g(\vec{a}) = \begin{cases} 1 & \textbf{if } \vec{a} \in S_n \\ 0 & \textbf{if } \vec{a} \in Z \setminus S_n \\ not\ defined & \textbf{otherwise} \end{cases}$$

Since $|Z| \leq 2^{c_1 n}$ and $|S_n| \leq b_n$, from Theorem 2.3, we have

$$L(g) \leq (1 + o(1)) \frac{\log \binom{2^{c_1 n}}{b_n}}{\log \log \binom{2^{c_1 n}}{b_n}} + O(n) \leq (1 + o(1)) c_1 \frac{b_n n}{\log b_n + \log n} .$$

From $S_n \subseteq Z$, it is easy to prove that $F_n^H = g \cdot \neg f$. It follows that

$$L(F_n) \leq L(g) + L(f) + O(1) \leq (1 + o(1)) c_1 \frac{b_n n}{\log b_n + \log n} + b_n + O(1) \leq$$

$$\leq (1 + o(1)) c_1 \frac{b_n n}{\log b_n + \log n}$$

For sufficiently large n, this last upper bound is in contradiction with hypothesis (i) of the theorem. □

In which follows, we will consider HSG's which always have a monotone function prize $k(n)$ such that, for any $n > 0$,

$$k(n+1) - k(n) \leq 1 \ \text{ and } \ n^\alpha \geq k(n) \geq \log n \ \text{ where } 0 < \alpha < 1.$$

Let $H : \{0,1\}^{k(n)} \to \{0,1\}^n$ be a boolean operator with $k(n) = \Theta(\log n)$, and let $F^H = \{F_n^H : \{0,1\}^n \to \{0,1\}\}$ be the family of the characteristic functions of the output sets of H. Given any $\vec{a} \in \{0,1\}^n$, $[\vec{a}]_{i_1,i_2}$ denotes the substring $\vec{a}_{i_1}, \ldots, \vec{a}_{i_2}$.

Corollary 3.1 *Suppose that a quick operator*

$$H \;=\; \{H_n \;:\; \{0,1\}^{k(n)} \to \{0,1\}^n \;,\; n = 1,2,...\} \;,$$

exists such that $k(n) = (1+\Theta(1))\log n$ *and, for a.e.* n,

$$L(F_n^H) \;\geq\; c_2 \frac{2^{k(n)}n}{k(n)+\log n} \text{ for some constant } 0 < c_2 < 1 \;.$$

Then, for any positive constant q *and for any constant* c_1 *such that* $0 < c_1 < c_2$, *it is possible to construct a quick operator* $H' : k'(n) \to n$ *such that* $k'(n) = \Theta(\log n)$ *and* H' *is a* $(1-2^{(c_1-1)n}, n^q)$*-HSG.*

Proof. Since $k(n) = (1+\Theta(1))\log n$, we can assume that $k(n) = (1+\delta)\log n$. Define $s(n) = \lceil \frac{2q}{\delta}\log n \rceil$. We consider the quick operator $H' : k'(n) \to n$ where $k'(n) = k(2^{s(n)}n) + s(n)$, defined as follows:

$$H'_n(\vec{a},\vec{b}) \;=\; [H_{n2^{s(n)}}(\vec{a})]_{t_1,t_2} \;,\;\; \text{where } t_1 = n(\phi(\vec{b})-1) \;,\;\; \text{and } t_2 = t_1 + n - 1$$

(observe that $\vec{a} \in \{0,1\}^{k(2^{(s(n)}n)}$, $\vec{b} \in \{0,1\}^{s(n)}$, and $\phi(\vec{b})$ is the decimal representation of $\vec{b}$). Consider a boolean function $f(x_1,\ldots,x_n)$ such that $\mathbf{Pr}\,(f=1) \geq 1-2^{(c_1-1)n}$ and $L(f) \leq n^q$. Then for function

$$f^*(x_1,\ldots,x_{2^{s(n)}n}) \;=\; \bigvee_{t=1}^{2^{s(n)}} f(x_{(t-1)n+1}, x_{(t-1)n+2}, ..., x_{tn}) \;,$$

it is easy to prove that

$$\mathbf{Pr}\,(f^*=1) \geq 1 - 2^{(c_1-1)2^{s(n)}n} \;,$$

and

$$L(f^*) \;\leq\; 2^{s(n)} + 2^{s(n)}L(f) \leq (1+o(1))2^{s(n)}n^q \leq (1+o(1))2^{s(n)}2^{\frac{\delta}{2}s(n)} \;\leq$$

$$\leq\; (1+o(1))\left(2^{s(n)}n\right)^{1+\frac{\delta}{2}} \;.$$

Thus for sufficiently large n, we have

$$L(f^*) \leq \left(2^{s(n)}n\right)^{1+\delta} \leq 2^{k(2^{s(n)}n)} \;.$$

By applying Theorem 3.1 with

$$b_n = 2^{k(2^{s(n)}n)},$$

we have that there exist $\vec{a} \in \{0,1\}^{2^{s(n)}n}$ and $t \in [1,\ldots,2^{s(n)}]$ such that

$$f^*\left([H_{2^{s(n)}n}(\vec{a})]_{(t-1)n,tn}\right) \;=\; 1.$$

It follows that there exists

$$\vec{a} \in \{0,1\}^{k(2^{s(n)}n)} \;\text{ and }\; \vec{b} \in \{0,1\}^{s(n)} \;\text{ such that }\; f(H'_n(\vec{a},\vec{b})) = 1 \;.$$

□

3.1 Improved HSG's using expanders

We will use the following important "hitting" property of expander graphs proved by *Ajtai et al* [1] (for a proof of this claim see for example Theorem 2.7 - p.124 - of [4]).

Theorem 3.2 *Let $G(V,E)$ be a d-regular graph, and assume that its second largest eigenvalue is at most $\lambda > 0$. Given any subset $W \subseteq V$ such that $|W| = \alpha n$ ($\alpha < 1$). Then, for every $t > 0$, the number of walks of length t in G that avoid C is at most*

$$n(1-\alpha)^{1/2}((1-\alpha)d^2 + \lambda^2)^{t/2} \ .$$

As mentioned in the Introduction, there exists a polynomial-time algorithm that, given $n > 0$, and $d \leq n$, constructs an d'-regular expanders G such that $d' = O(d)$, $|V| = O(n)$, and its second largest eigenvalues $\lambda > 0$ is such that $\lambda \leq 2\sqrt{d-1}$ [17] (such families of graphs are called *Ramanujan* graphs).

For any $n > 0$, consider a d-regular *Ramanujan expander* $EP_n = (V_n, X_n)$ where $2^n < |V_n| \leq 2^{n+1}$ [17]. Observe that the boolean strings with last component equal 0 correspond to the input set of the function we want to hit. This assumption is required when EP_n cannot be contructed on vertex sets whose size is exactly a power of 2. Let $l = \lceil \log d \rceil$. We suppose that d is a large but constant value. Then, we consider the operator $EPR_{n,t} : \{0,1\}^{n+l\cdot(2^t-1)+t} \to \{0,1\}^n$, such that

$$EPR_{n,t}(\vec{a}, \vec{u}_1, \ldots, \vec{u}_{2^t-1}, \vec{s}) \ , \qquad \vec{a} \in \{0,1\}^n \ , \ \vec{u}_i \in \{0,1\}^l \ , \ \vec{s} \in \{0,1\}^t \ ,$$

are the first n components of the $\phi(\vec{s})$-th vertex of the EP_n-walk of length 2^t which starts from vertex $(\vec{a}, 0)$ and is uniquely determined by the sequence of edge choices in the neighborhood of each vertex: $\phi(\vec{u}_1), \ldots, \phi(\vec{u}_{2^t-1})$. Observe that if $t = \Theta(\log n)$, the operator $EPR_{n,t}$ can be computed in time polynomial in n. Consider now a boolean function $g(x_1, \ldots, x_n)$, and the operator $EPR^g_{n,t} : \{0,1\}^{n+l\cdot 2^t} \to \{0,1\}$ that performs the OR among the values of g computed on the input points visited by a fixed EP_n-walk of length 2^t, i.e.,

$$EPR^g_{n,t}(\vec{a}, \vec{u}_1, \ldots, \vec{u}_{2^t}) \ = \ \bigvee_{\vec{s} \in \{0,1\}^t} g\left(EPR_{n,t}(\vec{a}, \vec{u}_1, \ldots, \vec{u}_{2^t-1}, \vec{s})\right) \ . \tag{3}$$

Lemma 3.1 *If $\mathbf{Pr}\,(g = 0) \leq c < \frac{1}{2}$, then*

$$\mathbf{Pr}\left(EPR^g_{n,t} = 0\right) \leq \left(c + \frac{\lambda}{d}\right)^{2^t-2} \ .$$

Proof. Let $0 < \alpha < 1$, and let $C \subseteq V_n$ be any subset such that $|C| \leq \alpha n$. From Theorem 3.2, we know that the number of walks of length m in EP_n that avoid C is at most

$$|V_n|(1-\alpha)^{1/2}((1-\alpha)d^2 + \lambda^2)^{m/2} \ .$$

Furthermore, observe that the number of all walks of length in EP_n is $|V_n|d^m$, and the value $\mathbf{Pr}\,(g = 0)$ computed on the set of vertices representing strings with last components equal to 0 is at most $2c$. It follows that

$$\mathbf{Pr}\left(EPR^g_{n,t} = 0\right) \leq \sqrt{2c}\left(2c + \frac{\lambda^2}{d^2}\right)^{2^t-1} \ \leq \ \left(c + \frac{\lambda}{d}\right)^{2^t-2} \ .$$

□

Theorem 3.3 *Assume that there exists a quick operator* $H = \{H_n : \{0,1\}^{k(n)} \to \{0,1\}^n,\ n = 1,2,...\}$ *, such that* $k(n) = (1+\Theta(1))\log n$ *and the characteristic functions of its output sets satisfies*

$$L(F_n^H) \ \geq \ \left(\frac{\log(4\lambda)}{\log d} + \delta\right)\frac{2^{k(n)}n}{k(n)+\log n}$$

for some constant $\delta > 0$*. Then it is possible to construct a quick operator*

$$H'' \ = \ \{H''_n \ : \ \{0,1\}^{k''(n)} \to \{0,1\}^n \ , \ n = 1,2,...\} \ ,$$

such that $k''(n) = \Theta(\log n)$ *and* H'' *is an* $(1-\epsilon, n)$*-HSG for some constant* $0 < \epsilon < 1$*, thus* $P = BPP$*.*

Proof. Corollary 3.1 implies that, for any positive integer q, a quick operator $H' : \{0,1\}^{k'(n)} \to \{0,1\}^n$ such that $k'(n) = \Theta(\log n)$ and H' is a quick $(1-2^{(c_1-1)n}, n^q)$-HSG, where

$$c_1 \ = \ \frac{\log(4\lambda)}{\log d} + \frac{\delta}{2} \ .$$

Let $l = \lceil \log d \rceil$ and $t(n) = \lceil 2\log n \rceil$. Then we define a new quick operator $H'' : \{0,1\}^{k''(n)} \to \{0,1\}^n$, such that

$$H''_n(\vec{a},\vec{b}) = EPR_{n,t(n)}(H'_{n+l\cdot(2^{t(n)}-1)}(\vec{a}),\vec{b}) \ , \ \text{ where } \ k''(n) = n + l\cdot(2^{t(n)}-1) + t(n) \ .$$

From the construction of Ramanujan expanders shown in [17], we can assume that $\frac{1}{2} \leq \lambda \leq 2\sqrt{d-1}$. Define $\epsilon = \frac{\lambda}{2d}$ and consider any boolean function $g(x_1,\ldots,x_n)$ such that $\mathbf{Pr}\,(g=1) \geq 1-\epsilon$, and $L(g) \leq n$. Then, we define $f(x_1,\ldots,x_N) = EPR^g_{n,t(n)}$ where $N = n + l\cdot(2^{t(n)}-1)$ (see Eq. 3). Clearly, f has polynomial-size circuit, thus we can choose q in the definition of H' such that $L(f) \leq N^q$. Lemma 3.1 implies that

$$\mathbf{Pr}\,(f=0) \ = \ \mathbf{Pr}\left(EPR^g_{n,t(n)} = 0\right) \ \leq \ \left(\epsilon + \frac{\lambda}{d}\right)^{2^{t(n)}-2} \ \leq \ \frac{2\lambda}{d}^{\frac{N-n-2}{l}} \ \leq$$

$$\leq 2^{\frac{\log(2\lambda)-\log d}{\log d+1}(N-n-2)} 2^{\left(\frac{\log(4\lambda)}{\log d+1}-1\right)(N-n-2)} \leq 2^{\left(\frac{\log(4\lambda)}{\log d}-1\right)N}$$

(observe that the last inequality holds for a.e. n). By definition, we know that H' hits function f, i.e., there exists $\vec{a} \in \{0,1\}^{k'(n+l\cdot(2^{t(n)}-1)}$ such that $f(H'_{n+l(2^{t(n)}-1)}(\vec{a})) = 1$. From the definition of f, there exists $\vec{b} \in \{0,1\}^{t(n)}$ such that

$$g(H''_n(\vec{a},\vec{b})) = g(EPR_{n,t(n)}(\vec{a},\vec{b})) = 1 \ .$$

□

4 Hitting sets from hard boolean operators

The construction of an efficient HSG from a boolean operator which has hard circuit-complexity is based on the following "contradiction" argument. Suppose that a boolean operator $T : \{0,1\}^m \longrightarrow \{0,1\}^n$ is not a HSG for a certain class of circuits defined by the parameters $\epsilon(n)$ and $\beta(n)$ (see Def. 1.1). Roughly speaking, this negative fact implies that the output sequence of T can be represented by a new binary sequence which contains a "large" number of 0's (this number depends on $\epsilon(n)$ and $\beta(n)$). Then, using the Andreev *et al*'s technique shown in [7], it is possible to compress this new binary sequence in order to prove a new upper bound on the circuit-size complexity of the output sequence. This bound is obtained by a better analysis of the compression rate achieved by this technique and by applying our new result on the Shannon function $L(n,N,m)$ (Theorem 2.3). Thus, we get an upper bound on the circuit complexity of T. If T is supposed to have a hard circuit complexity, we have a contradiction.

4.1 Compressing boolean operators

Let $T : \{0,1\}^m \to \{0,1\}^n$ and $C(x_1,\ldots,x_n)$ be a circuit. Given $\vec{\alpha} \in \{0,1\}^n$, consider the function

$$Med(C,T,\vec{\alpha}) \;=\; 2^{-m} \sum_{\vec{u}\in\{0,1\}^m} C(T(\vec{u}) \oplus \vec{\alpha}) \; .$$

Note that it is immediate to prove that, for any $\vec{\alpha} \in \{0,1\}^n$,

$$\mathbf{E}\left(Med(C,T,\vec{\alpha})\right) = \mathbf{Pr}\left(C(x_1,\ldots,x_n)=1\right) \; .$$

We describe here the *Andreev et al*'s technique introduced in [7]. Let $\vec{\alpha}_1$ and $\vec{\alpha}_2$ be two different elements in $\{0,1\}^n$. Define

$$d_1 = Med(C,T,\vec{\alpha}_1) \;\text{ and }\; d_2 = Med(C,T,\vec{\alpha}_2)$$

and assume that $D = d_2 - d_1 > 0$. The j-th component of $\vec{a}$ will be denoted as $[\vec{a}]^j$. Since we are considering the case in which $D > 0$, without loss of generality, we can assume that there exists an index s for which $[\vec{\alpha}_1]^s \neq [\vec{\alpha}_2]^s$. Consider the operator $T^{\#} : \{0,1\}^m \to \{0,1\}^n$ defined as follows

$$T^{\#}(\vec{u}) = T(\vec{u}) \oplus ([T(\vec{u})]^s \cdot (\vec{\alpha}_1 \oplus \vec{\alpha}_2))$$

where the operation "$\oplus$" between two boolean vectors is performed component by component and the operation "$\cdot$" is the standard scalar product. The s-th component of $T^{\#}(\vec{u})$ satisfies the following equations:

$$[T^{\#}(\vec{u})]^s = [T(\vec{u})]^s \oplus ([T(\vec{u})]^s \cdot ([\vec{\alpha}_1]^s \oplus [\vec{\alpha}_2]^s)) = [T(\vec{u})]^s \oplus [T(\vec{u})]^s \cdot 1 \;= 0 \; . \tag{4}$$

Observe also that the set $\{T^{\#}(\vec{u}) \oplus \vec{\alpha}_1, T^{\#}(\vec{u}) \oplus \vec{\alpha}_2\}$ is equal to the set $\{T(\vec{u}) \oplus \vec{\alpha}_1 \;,\; T(\vec{u}) \oplus \vec{\alpha}_2\}$. Let

$$N(\sigma,\phi_1,\phi_2) \;=\; |\{i \;:\; [T(\vec{u})]^s = \sigma \;,\; C(T(\vec{u}) \oplus \vec{\alpha}_1) = \phi_1 \text{ and } C(T(\vec{u}) \oplus \vec{\alpha}_2) = \phi_2\}| \; . \tag{5}$$

We can now introduce the function which approximates the s-th component of $T(\vec{u})$. Consider the function Q defined as follows:

$$Q_{N(\sigma,\phi_1,\phi_2)}(x,y) \;=\; \begin{cases} x & \textbf{if} \quad x \neq y \\ 1 & \textbf{if} \quad x=y=0 \;\text{ and }\; N(1,0,0) \geq N(0,0,0) \\ 0 & \textbf{if} \quad x=y=0 \;\text{ and }\; N(1,0,0) < N(0,0,0) \\ 1 & \textbf{if} \quad x=y=1 \;\text{ and }\; N(1,1,1) \geq N(0,1,1) \\ 0 & \textbf{if} \quad x=y=1 \;\text{ and }\; N(1,1,1) \geq N(0,1,1) \end{cases}$$

In which follows we will consider the function N as a fixed parameter, and thus we will omit the index $N(\sigma,\phi_1,\phi_2)$ in the definition of Q. Then the approximation function for the s-th bit of $T(\vec{u})$ is

$$Z(\vec{u}) \;=\; Q(C(T^{\#}(\vec{u}) \oplus \vec{\alpha}_1), C(T^{\#}(\vec{u}) \oplus \vec{\alpha}_2)) \;,\; i = 1, \ldots, m \;.$$

Our next goal is to estimate the number of errors generated by $Z(\vec{u})$. Let $ND(\sigma,\phi_1,\phi_2)$ be the number of indexes i such that the following conditions are satisfied:

i) $[T(\vec{u})]^s \oplus Z(\vec{u}) = 1$ (i.e. there is an error);

ii) $[T(\vec{u})]^s = \sigma$;

iii) $C(T(\vec{u}) \oplus \vec{\alpha}_1) = \phi_1$;

iv) $C(T(\vec{u}) \oplus \vec{\alpha}_2) = \phi_2$.

The following Lemma gives an upper bound on the number of errors in approximating the s-th bit of $T(\vec{u})$.

Lemma 4.1 [7]

$$\sum_{(\sigma,\phi_1,\phi_2)\in\{0,1\}^3} ND(\sigma,\phi_1,\phi_2) \leq m\left(\frac{1}{2} - \frac{d_2-d_1}{2}\right) \;.$$

4.1.1 Some new hardness-compression trade-offs

Using Lemma 4.1, we are now able to perform a better analysis of the circuit complexity of T. Observe that the function

$$U(\vec{u}) \;=\; [T(\vec{u})]^s \oplus Z(\vec{u}) \;,\; \vec{u} \in \{0,1\}^m$$

singles out the positions in the operator T in which an error occurs. We thus have that

$$[T(\vec{u})]^s = U(\vec{u}) \oplus Z(\vec{u}) = U(\vec{u}) \oplus Q(C(T^{\#}(\vec{u}) \oplus \alpha_1), C(T^{\#}(\vec{u}) \oplus \alpha_2)).$$

Lemma 4.2

$$L(T) \;\leq\; L^{op}(m, n-1) + L(U) + O(L(C)) + O(n) \;.$$

Proof. Observe first that T can be represented as

$$T(\vec{u}) \;=\; T^{\#}(\vec{u}) \oplus \vec{e}_s^{\,n} \cdot (Q(C(T^{\#}(\vec{u}) \oplus \vec{\alpha}_1), C(T^{\#}(\vec{u}) \oplus \vec{\alpha}_2)) \oplus U(\vec{u})) \qquad (6)$$

where $\vec{e}_s^{\,n} \in \{0,1\}^n$ is the boolean vector having only the s-th component equal to 1. Furthermore, the operator $T^{\#}(\vec{u})$ satisfies Eq. 4. It follows that the thesis is an immediate consequence of Eq. 6. □

Furthermore, the circuit complexity of U satisfies the following upper bound.

Lemma 4.3 *If, for some constant c_1, we have that $D \geq c_1$, then there exists a constant $c_2 < 1$ for which*

$$L(U) \ \leq \ c_2 \frac{2^m}{m} \ .$$

Proof. Let $|U|$ denote the the number of 1's in U; then, from Lemma 4.1, we have that

$$|U| \ \leq \ 2^m \left(\frac{1}{2} - \frac{d_2 - d_1}{2} \right) \ . \tag{7}$$

Finally, the thesis follows from the upper bound shown in Theorem 2.3. □

4.2 The Hitting Set Generator

In order to derive our HSG, we will make use of the following result given by Lupanov (see also [25]). Let $L^{op}(k,n)$ denote the worst-case circuit complexity of boolean operators having k variables and n outputs.

Theorem 4.1 *[19]*

$$L^{op}(k,n) = (1 + o(1)) \frac{2^k n}{k + \log n} \ ,$$

Theorem 4.2 *Assume that a quick operator*

$$H \ = \ \{H_n \ : \ \{0,1\}^{k(n)} \rightarrow \{0,1\}^n \ , \ n = 1, 2, ...\} \ ,$$

exists such that $k(n) = (1 + \Theta(1)) \log n$, and for a.e. n

$$L^{op}(H_n) \ \geq \ L(k(n), n) - \frac{2^{k(n)}}{k(n)^2} \ .$$

Then, it is possible to construct a $(\frac{1}{2}, n)$-HSG $H' : k'(n) \rightarrow n$ such that $k'(n) = \Theta(\log n)$. Hence, $P = BPP$.

Proof. Let $k'(n) = k(n + 2^{t(n)}) + t(n)$, where $t(n) = \lceil 2 \log n \rceil$, and define, for any $\vec{a} \in \{0,1\}^{k(n+2^{t(n)})}$ and $\vec{b} \in \{0,1\}^{t(n)}$,

$$H'_n(\vec{a}, \vec{b}) = H_{n+\phi(\vec{b})}(\vec{a}) \ ,$$

where $\phi(\vec{b})$ denotes the standard decimal representation of $\vec{b}$ (note that if the length of $\vec{a}$ is greater than $k(n + \phi(\vec{b}))$, we simply erase the last bits of $\vec{a}$). Observe that there exists a constant $\delta > 0$ such that, for any sufficiently large n, two distinct integers n_1, n_2 exist such that

$$n < n_1 < n_2 \leq n + 2^{t(n)} \ , \quad n_2 = \lceil (1 + \delta n_1) \rceil \ \text{ and } \ k(n_1) = k(n_2) \ .$$

Indeed, if this would be false then

$$k(n + 2^{t(n)}) \geq k(n) + \frac{\frac{n + 2^{t(n)}}{n}}{\log(1 + \delta)} \geq k(n) + \Theta(1) \frac{1}{\delta} \log n \ ,$$

but for small δ this bound will be more than $k(n+2^{t(n)})$. Let $k = k(n_1) = k(n_2)$. From Theorem 4.1, we have

$$L^{op}(k,n_2) - L^{op}(k,n_1) \sim \frac{2^k n_2}{k+\log n_2} - \frac{2^k n_1}{k+\log n_1} \sim (n_2-n_1)\frac{2^k}{k+\log n_1} ,$$

and there exists n_3, such that $n_1 \leq n_3 < n_2$, for which

$$L^{op}(k,n_3+1) - L^{op}(k,n_3) \geq (1-o(1))\frac{2^k}{k+\log n_1} . \tag{8}$$

Suppose now that our generator does not hit a circuit $C(x_1,\ldots,x_n)$ such that

$$\mathbf{Pr}\,(C=1) \geq \frac{1}{2} \text{ , and} \qquad L(C) \leq n .$$

It follows that for $d_1 = Med(f, H_{n_3+1}, \vec{0})$, we have $d_1 = 0$. But we also have that an $\alpha_2 \in \{0,1\}^n$ exists for which $d_2 = Med(f, H_{n_3+1}, \alpha_2) \geq \frac{1}{2}$. By applying Lemma 4.2 and Lemma 4.3, there exists a constant $0 < c_2 < 1$ such that

$$L^{op}(H_{n_3+1}) \leq L^{op}(k,n_3) + c_2\frac{2^k}{k} + O(L(C)) + O(n) ,$$

and

$$L^{op}(k,n_3) \geq L^{op}(H_{n_3+1}) - c_2\frac{2^k}{k} - O(n) .$$

Since

$$L^{op}(H_{n_3+1}) \geq L^{op}(k,n_3+1) - \frac{2^k}{k^2} ,$$

it follows that

$$L^{op}(k,n_3+1) \leq L^{op}(H_{n_3+1}) + \frac{2^k}{k^2} ,$$

and

$$L^{op}(k,n_3+1) - L^{op}(k,n_3) \leq \frac{2^k}{k^2} + c_2\frac{2^k}{k} + O(n) . \tag{9}$$

The value $c_2 < 1$ depends only on $D = d_2 - d_1 = \frac{1}{2}$, consequently is a constant. Whithout loss of generality, we can assume that $k(n) \geq q \log n$, for some convenient big constant q (it is sufficient to consider the first n output bits of H_{n^r} , where r depends on q). Then, comparing Eq. 8 and Eq. 9, we have a contradiction. It follows that H' is an $(\frac{1}{2},n)$-HSG. □

References

[1] Ajtai M, Komlos J, and Szemeredi E. (1987), Deterministic simulation in LOGSPACE, Proc. of *19th ACM STOC*, 132-140.

[2] E. Allender and M. Strauss (1994), "Measure on small complexity classes, with applications for BPP", *Proc. of 35th IEEE- FOCS*, 807-818.

[3] Alon N. (1986), "Eigenvalues and Expanders", *Combinatorica*, 6, pp. 83-96.

[4] Alon N. and Spencer J.H. (1992), *The Probabilistic Method*, Wiley-Interscience Publication.

[5] Andreev A. (1995), "The complexity of nondeterministic functions", *Information and Computation*, to appear.

[6] Andreev A., Clementi A., and Rolim J. (1996), "Optimal Bounds on the Approximation of Boolean Functions, with Consequences on the Concept of Hardness", in *XIII Annual Symposium on Theoretical Aspects of Computer Science (STACS'96)*, LNCS, 1046, 319-329. Also available via ftp/WWW in the electronic journal *ECCC* (TR95-041).

[7] Andreev A., Clementi A., and Rolim J. (1996), "Hitting Sets Derandomize BPP", in *XXIII International Colloquium on Algorithms, Logic and Programming (ICALP'96)*, LNCS. Also available via ftp/WWW in the electronic journal *ECCC* (TR95-061)

[8] Andreev A., Clementi A., and Rolim J. (1996), "Towards efficient constructions of Hitting Sets that derandomize BPP", Technical Report available via ftp/WWW in the electronic journal *ECCC* (TR96-029).

[9] Armoni R., Saks M., Wigderson A., Zhou S. (1996) "Discrepancy Sets and Pseudorandom Generators for Combinatorial Rectangles", Proc. of *IEEE - FOCS'96*, to appear.

[10] Blum M., and Micali S. (1984), "How to generate cryptographically strong sequences of pseudo-random bits", *SIAM J. of Computing*, 13(4), 850-864.

[11] Clementi A., Impagliazzo R., and Pierini P. (1994), "On the average-case complexity of the reversibility problem for finite cellular automata", Proc. of *IEEE PhysComp'94*, 151-155.

[12] Chor B., and Goldreich O. (1989), "On the Power of Two-Point Based Sampling", *J. of Complexity*, 5, 96-106.

[13] Karp R., Pippenger N., and Sipser M. (1982) "Time-Randomness, Tradeoff", presented at *AMS Conference on Probabilistic Computational Complexity.*

[14] Gurevich Y. (1991), "Average-Case Completeness", *JCSS*, 42, 346-360.

[15] Levin L. (1986), "Average-Case Complete Problems", *SIAM JCOMP*, 15, 285-285.

[16] Linial N., Luby M., Saks M., and Zuckerman D. (1993), "Efficient construction of a small hitting set for combinatorial rectangles in high dimension", in *25th ACM STOC*, 258-267.

[17] A. Lubotzky, R. Phillips, and P. Sarnak. (1988), "Ramanujan graphs", *Combinatorica*, 8(3):261–277, 1988.

[18] Lupanov, O.B. (1956) "About gating and contact-gating circuits", *Dokl. Akad. Nauk SSSR* 111, 1171-11744.

[19] Lupanov, O.B. (1965), "About a method circuits design – local coding principle", *Problemy Kibernet.* 10, 31-110 (in Russian).

[20] G.A. Margulis (1973), "Explicit Construction of Concentrators", *Problems of Inform. Transmission*, 325-332.

[21] Nechiporuk E.I. (1965) , "About the complexity of gating circuits for the partial boolean matrix", *Dokl. Akad. Nauk SSSR*, 163, 40-42.

[22] Nisan N. (1990), *Using Hard Problems to Create Pseudorandom Generators*, *ACM Distinguished Dissertation*, MIT Press.

[23] Nisan N., and Wigderson A. (1994), "Hardness vs Randomness", *J. Comput. System Sci.* 49, 149-167 (also presented at the *29th IEEE FOCS*, 1988).

[24] Sipser M. (1986), "Expanders, Randomness or Time vs Space", in *Proc. of the 1st Conference on Structures in Complexity Theory*, LNCS 223, 325-329.

[25] Wegener, I. (1987), *The complexity of finite boolean functions*, *Wiley-Teubner Series in Computer Science.*

[26] Yao A. (1982), "Theory and applications of trapdoor functions", in *23th IEEE FOCS*, 80-91.