

Lecture Notes for CCT06, 13/12-2006

Peter Sebastian Nordholt

January 12, 2007

1 Hitting Set Generators

In this lecture we looked at more hitting set generators besides the π -generator of the previous lecture.

However first we refresh some definitions and results from previous lectures. Firstly we defined a hitting set as follows:

Definition 1.1 (Hitting Set). A *hitting set* is a set, $S \subseteq \{0, 1\}^n$, so that for all circuits $C : \{0, 1\}^n \rightarrow \{0, 1\}$ of size n , the following property holds:

$$\mu(C) \geq \frac{1}{2} \Rightarrow \exists z \in S : C(z) = 1$$

Where μ is the function measuring the fraction of inputs in $\{0, 1\}^n$ that makes the circuit output 1. That is $\mu(C) = \frac{|\{x \in \{0, 1\}^n \mid C(x) = 1\}|}{2^n}$.

A hitting set generator was then defined as:

Definition 1.2 (Hitting Set Generator). A *hitting set generator* is an efficient algorithm that given the input 1^n outputs a hitting set $S \subseteq \{0, 1\}^n$.

Hitting set generators as defined in definition 1.2 are interesting, because if they exist hypothesis H holds, and thus we have $\mathbf{prRP} = \mathbf{prBPP} = \mathbf{prP}$, which also means $\mathbf{RP} = \mathbf{BPP} = \mathbf{P}$. However we do not need quite as strong a definition of a hitting set and a hitting set generator to achieve this, and thus we define a weak hitting set or an (ϵ, q) -*hitting set*.

Definition 1.3 (ϵ, q) -Hitting Set). An (ϵ, q) -*hitting set* is a set, $S \subseteq \{0, 1\}^n$, so that for all circuits $C : \{0, 1\}^n \rightarrow \{0, 1\}$ of size n^q , the following property holds:

$$\mu(C) \geq 1 - 2^{-n+n^\epsilon} \Rightarrow \exists z \in S : C(z) = 1$$

The corresponding generators are then defined as follows.

Definition 1.4 (ϵ, q) -Hitting Set Generator). A (ϵ, q) -*hitting set generator* is an efficient algorithm that given the input 1^n outputs a hitting set $S \subseteq \{0, 1\}^n$.

Now for all $\epsilon > 0$ there is a q so that if there exists an (ϵ, q) -hitting set generator, we have that $\mathbf{prRP} = \mathbf{prBPP} = \mathbf{prP}$. This is because given such an (ϵ, q) -hitting set generator, we can make a normal hitting set generator using the Trevisan extractor as a disperser, and we can then use the statement from above. For $\epsilon > 0$ we will call the corresponding q , for which there exists a (ϵ, q) -hitting set generator, q_ϵ .

In the following sections we will show the existence of (ϵ, q) -hitting set generators, under certain assumptions.

2 Sipers Hitting Set Generator

In this section we will prove the following.

Theorem 2.1. *Assume there exists a language*

$$L \in \mathbf{DTIME}(2^{cn}) - \mathbf{DSpace}_{i.o.}(2^{(c-\delta)n})$$

Where $\delta = \frac{1}{10q_{\frac{1}{2}}}$, and c is some constant. Then there also exists a $(\frac{1}{2}, q_{\frac{1}{2}})$ -hitting set generator.

Notice that if L exists, this means that any machine M that decides L must use at least $2^{(c-\delta)n}$, space for any input longer than some threshold. To prove theorem 2.1 we will define a candidate for a hitting set generator, and show that under the assumption that L exists it is indeed a $(\frac{1}{2}, q_{\frac{1}{2}})$ -hitting set generator.

Definition 2.1 (Sipers Hitting Set Generator). Assuming L exists let it be decided by a $\mathbf{DTIME}(2^{cn})$ Turing machine M . On input 1^m Sipers hitting set generator simulates M on all inputs of size $2q_{\frac{1}{2}} \log m$. We can then divide the work tapes of M into blocks of size m . The hitting set is then the set of all blocks of m -bits appearing on the work tapes during the computation of M .

Now we are ready to prove theorem 2.1. We will do this by showing that if Sipers hitting set generator is not a $(\frac{1}{2}, q_{\frac{1}{2}})$ -hitting set generator, the computation of M can be compressed to use less than $2^{(c-\delta)n}$ space.

Proof of theorem 2.1. Assume L exists, and let M be a $\mathbf{DTIME}(2^{cn})$ Turing machine deciding L . We assume the work tapes of M are of length at most 2^{cn} (This is safe assumption since M does not have time to read longer work tapes).

There then exists a circuit $C : \{0, 1\}^m \rightarrow \{0, 1\}$ of size $m^{q_{\frac{1}{2}}}$ so that $\mu(C) \geq 1 - 2^{-m+m^{\frac{1}{2}}}$, that is at most $2^{m^{\frac{1}{2}}}$ of the possible inputs maps to 0. Let Z_C be the set of all inputs to C that maps to 0.

Now assume Sipers hitting set generator is not a $(\frac{1}{2}, q_{\frac{1}{2}})$ -hitting set generator. This means all m -bit blocks appearing on M 's work tape on inputs of size $2q_{\frac{1}{2}} \log(m)$ are in Z_C . Since, by the definition of C , the size of Z_C is at most $2^{m^{\frac{1}{2}}}$, we see that we can represent each m -bit block of M 's work tape with at most

$m^{\frac{1}{2}}$ bits. We can do this by representing each m -bit block as its lexicographical order in Z_C . The mapping from the m -bit blocks to the compressed forms, can be done using C in space something like $m^{\frac{q_1}{2}} + m$ (Space for C and one uncompressed block). Since we assume M 's work tapes has length at most 2^{cn} , the compressed blocks takes up space at most $\frac{2^{cn}}{m^{\frac{1}{2}}}$. This means we can create a machine M' that does the computation of M but keeps the works tapes compressed. This machine uses $\frac{2^{cn}}{m^{\frac{1}{2}}} + m^{\frac{q_1}{2}} + m$ space which is less than $2^{(c-\delta)n}$, this however is a contradiction to our assumption on L . \square

Notice that if the language L exists and thereby also the Sipser hitting set generator exists, we have that $\mathbf{prRP} = \mathbf{prBPP} = \mathbf{prP}$, as stated in section 1.

3 Miltersen-Vinodchandran Hitting Set Generator

In this section we show the existence of an other hitting set generator, given certain assumptions. First however we prove the following theorem:

Theorem 3.1. *If there is an $\epsilon > 0$ and a language L so that:*

$$L \in \mathbf{DTIME}(2^{cn}) - \mathbf{DSPACE}_{i.o.}(2^{\epsilon n})$$

Then there is also an $\epsilon' > 0$ and a language $L' \in \mathbf{DTIME}(2^{cn})$, such that for all sufficiently large n : $\mathbf{SIZE}^{\mathbf{SAT}}(L'_{\{0,1\}^n}) \geq 2^{\epsilon' n}$. Where $\mathbf{SIZE}^{\mathbf{SAT}}$ is \mathbf{SIZE} defined as usual only allowing SAT solving gates.

Proof. We assume that the language L exists. Assume now that theorem 3.1 does not hold. That is assume that for all ϵ' and all languages $L' \in \mathbf{DTIME}(2^{cn})$, $\mathbf{SIZE}^{\mathbf{SAT}}(L'_{\{0,1\}^n}) < 2^{\epsilon' n}$, for infinitely many n . We will show that this is a contradiction to the assumption that L exists.

Let M_L be a $\mathbf{DTIME}(2^{cn})$ machine deciding L . We define L' to be the language of tuples $(x, (i, t), k)$, so that $(x, (i, t), k) \in L'$ if and only if the k 'th bit in the cell description of cell i , of M_L 's work tape, is 1 at time t of M_L 's computation on x . The language L' is then *locally checkable*. That means that for some input length, n' , and circuit C that supposedly decides $L'_{\{0,1\}^{n'}}$, we can check if C correctly decides $L'_{\{0,1\}^{n'}}$ in time linear in C 's size.

Now the assumption above means, that for all ϵ' there are circuits deciding $L'_{\{0,1\}^n}$ of size at most $2^{\epsilon' n}$, for infinitely many n (possibly using SAT-gates). Given some $x \in \{0,1\}^n$ we can then for questions $x \in L$? reduce the question to a question $x' \in L'$, for x' of length $n' \leq 10cn$. We can then search through all circuits of size $2^{\frac{\epsilon'}{10c} n'}$ checking if they decide $L'_{\{0,1\}^{n'}}$. If we find such a circuit we use this to decide if $x \in L$ otherwise we can just use M_L . However since we can find such a circuit for infinitely many n 's we can now decide L in space $2^{\epsilon n}$ for infinitely many n 's, a contradiction to the assumption that L exists. \square

We then have the following theorem, regarding a hitting set generator:

Theorem 3.2. *Assume there is an ϵ' so that there exists a language $L' \in \text{DTIME}(2^{cn})$, such that for all sufficiently large n : $\text{SIZE}^{\text{SAT}}(L'_{\{0,1\}^n}) \geq 2^{\epsilon' n}$. Then there exists an $(\epsilon'', q_{\epsilon''})$ -hitting set generator.*

The hitting set generator we will use for theorem 3.2 is defined as follows:

Definition 3.1 (Miltersen-Vinodchandran). On input 1^n the Miltersen-Vinodchandran hitting set generator generates the truth table of a function $f : \{0, 1\}^m \rightarrow \{0, 1\}$ of circuit complexity at least $2^{\epsilon' m}$, where $m = \frac{\log n}{10} k$. Interpreting each $\frac{\log n}{10}$ -bit block of input to f as decimal numbers we write f as:

$$f : \{1, \dots, \sqrt[10]{n}\}^k \rightarrow \{0, 1\}$$

This way the truth table of f can be arranged as a k -dimensional matrix. The hitting set generator then produces the k -dimensional truth table matrix of a function:

$$\tilde{f} : \{1, \dots, n\}^k \rightarrow \{0, 1\}$$

With the properties:

- $\tilde{f}(i_1, \dots, i_k) = f(i_1, \dots, i_k)$, for $i_1, \dots, i_k \leq n$
- Each vector:

$$v = \begin{pmatrix} \tilde{f}(i_1, \dots, i_{j-1}, 1, i_{j+1}, \dots, i_k) \\ \tilde{f}(i_1, \dots, i_{j-1}, 2, i_{j+1}, \dots, i_k) \\ \vdots \\ \tilde{f}(i_1, \dots, i_{j-1}, n^{10}, i_{j+1}, \dots, i_k) \end{pmatrix}$$

For any j and $i_1, \dots, i_{j-1}, i_{j+1}, \dots, i_k$, is a codeword for an error correcting code e .

Notice the first property tells us that the truth table matrix of \tilde{f} has the matrix of f as one small corner.

The hitting set of the generator will then be the set of all vectors v as described in the second property.

Making the assumption of theorem 3.2 it can be shown that the hitting set generated by the hitting set generator of definition 3.1 is a $(\epsilon'', q_{\epsilon''})$ -hitting set for circuits $C : \{0, 1\}^n \rightarrow \{0, 1\}$ of size $n^{q_{\epsilon''}}$ where $\epsilon'' = \frac{\epsilon'}{100}$. The proof of theorem 3.2 then goes along the lines of the proof of 2.1. We assume the hitting set generator does not generate a correct hitting set, then show that this means we can compress circuits for some language, L' , using a circuit that was not hit, and thus contradicting the assumption of the theorem.