Electronic Colloquium on Computational Complexity, Report No. 9 (2001)

Towards proving strong direct product theorems

Ronen Shaltiel
Hebrew University, Jerusalem, Israel.
and
Institute for advanced study, Princeton, NJ.
E-mail:`ronens@cs.huji.ac.il`.

Abstract

A fundamental question of complexity theory is the direct product question. Namely weather the assumption that a function f is hard on average for some computational class (meaning that every algorithm from the class has small advantage over random guessing when computing f) entails that computing f on k independently chosen inputs is exponentially harder on average. A famous example is Yao's XOR-lemma, [Yao82] which gives such a result for boolean circuits. This question has also been studied in other computational models, such as decision trees [NRS94], and communication complexity [PRW97].

In Yao's XOR-lemma one assumes f is hard on average for circuits of size s and concludes that $f^{\oplus k}(x_1, \cdots, x_k) = f(x_1) \oplus \cdots \oplus f(x_k)$ is essentially exponentially harder on average for circuits of size s'. All known proofs of this lemma, [Lev85, Imp95, IW97, GNW95] have the feature that $s' < s$. In words, the circuit which attempts to compute $f^{\oplus k}$ is *smaller* than the circuit which attempts to compute f on a single input!

This paper addresses the issue of proving *strong* direct product assertions, that is ones in which $s' \approx ks$ and is in particular *larger* than s. Our first result is a counterexample which shows that strong direct product assertions are simply not true. This holds for every "reasonable" model of computation. While this counterexample rules out the possibility of proving strong direct product assertions, it seems to exploit defects in the formulation of the problem rather than show that our general intuition for strong direct product assertions is false. Still, in order to prove theorems with a strong direct product flavor we must change the formulation of the question and either strengthen the assumption or weaken the conclusion.

An example of strengthening the assumption is given in our second result in which we prove that if a function f is proved to be hard on average for c-bit communication protocols via the "discrepancy method" then $f^{\oplus k}$ is exponentially harder on average for $\Omega(kc)$-bit communication protocols. This follows by showing that $disc(f^{\oplus k}) = disc(f)^{\Omega(k)}$ which we prove using algebraic techniques inspired by [NW94].

As for weakening the conclusion, we introduce the notion of *fair* algorithms which restricts the algorithm attempting to compute $f^{\oplus k}$ to spend its computational resource evenly between the k instances. Our third result is an optimal direct product theorem for fair decision trees. We show that if f is hard on average for depth d decision trees then $f^{\oplus k}$ is exponentially harder for fair decision trees of depth kd. This result seems incomparable to that of [NRS94] in which the algorithm is a "decision forest" of k parallel decision trees. It is our hope that these notions could be extended to stronger computational models.

ISSN 1433-8092

1 Introduction

1.1 The direct product question

Given a boolean function f over domain X and an integer k, we define a function $f^{\oplus k} : X^k \rightarrow \{0, 1\}$.

$$f^{\oplus k}(x_1, \cdots, x_k) = f(x_1) \oplus \cdots \oplus f(x_k)$$

Intuitively, if f is hard on average, (say f can be computed correctly by some computational class on at most a $(1/2 + p)$-fraction of of the inputs), then we expect $f^{\oplus k}$ to be computed correctly on at most a $(1/2 + p^k)$-fraction of the inputs. This intuition is based on an information theoretic analog. In the information theoretic setup f is a biased random coin with probability of heads $1/2 + p$, and $f^{\oplus k}$ is the exclusive-or of k such independent coins which indeed induces a coin with probability of heads roughly equal to $1/2 + p^k$. Transferring this intuition from the information theoretic setting to computational settings is much more involved than one can expect at first glance. Such assertions are referred to as direct product assertions.

Let us present the direct product question in a general computational setting. Consider some computational resource (such as circuit size, decision tree depth, number of bits exchanged in a communication protocol...). Let Res_r denote the class of all functions computable using r "units" of the resource. In this paper we consider the following classes:

- $Size_s$ the family of functions computed by circuits of size s.
- $Comm_c$ the family of functions (on two inputs) computed by a communication protocol which exchanges c bits.
- $Depth_d$ the family of functions computed by decision trees of depth d.

Saying that a function f is "hard on average" for Res_r means that every algorithm from Res_r computes f correctly on a bounded fraction of the inputs[1]. We use the following notation:

$$Suc_r^{Res}(f) = max_{P \in Res_r} \Pr_{x \in_R X}[P(x) = f(x)]$$

Since f is boolean it can always be computed on at least half the inputs. We will be interested in the advantage the algorithm can get over guessing.

$$Adv_r^{Res}(f) = 2(Suc_r^{Res}(f) - 1/2)$$

The direct product question can now be presented as follows:[2]

The direct product problem: Is it true that for all f and r, k:

$$Adv_r^{Res}(f) \leq p \Rightarrow Adv_{r'}^{Res}(f^{\oplus k}) \leq p'$$

Where r' and p' are parameters. In words, one supposes that f is hard on the average to algorithms with r units of the resource, and concludes that $f^{\oplus k}$ is hard on the average to algorithms having

[1]In this paper we restrict ourselves to average case hardness relative to the uniform distribution. Indeed, some of out results do not generalize to arbitrary probability distributions. See the discussion at the end of the paper.

[2] A different variant that is sometimes considered is the *concatenation variant*. It involves replacing the function $f^{\oplus k}$ with $f^{(k)}(x_1, \cdots, x_k) = (f(x_1), \cdots, f(x_k))$, and asking weather $Suc_r^{Res}(f) \leq p \Rightarrow Suc_{r'}^{Res}(f) \leq p^k$. These two variants are closely related and in particular a strong direct product assertion for $f^{\oplus k}$ implies a strong direct product assertion for $f^{(k)}$. Exact details are omitted from this version.

r' units of the resource. Naturally, the assertion is stronger when r' is large and p' is small. we say that the assertion is *optimal* when $r' = kr$ and $p' = p^k$. It seems reasonable to allow the algorithm attempting to compute $f^{\oplus k}$ to use kr units of the resource, as its input is k times larger than that of the algorithm attempting to compute f. In this paper we are interested in proving direct product assertions for large r'. We will call such assertions *strong* if $r' = \Omega(kr)$ and $p' = p^{\Omega(k)}$.

The strong-direct product problem: Is it true that for all f and r, k:

$$Adv_r^{Res}(f) \le p \Rightarrow Adv_{\Omega(kr)}^{Res}(f^{\oplus k}) \le p^{\Omega(k)}$$

In our results r' is a parameter, and the fixing of $r' = \Omega(kr)$ is made to simplify the presentation[3].

1.2 Previous work

The most studied model for direct product results is circuit complexity. The so called "Yao's XOR-lemma", [Yao82] can be stated this way in our terminology:

$$Adv_s^{Size}(f) \le p \Rightarrow Adv_{s'}^{Size}(f^{\oplus k}) \le p^k + \epsilon$$

where $s' = s\left(\frac{\epsilon}{n}\right)^{O(1)}$, and n is the number of inputs of f. Note that in this result s' is actually *smaller* than s. In other words the circuit which tries to compute f on many instances is smaller than the one which tries to compute f on one instance. This is unavoidable in the sense that all known proofs of this lemma, [Lev85, Imp95, GNW95, IW97] work by proving the contra-positive claim: $Adv_{s'}^{Size}(f^{\oplus k}) > p^k + \epsilon \Rightarrow Adv_s^{Size}(f) > p$, and use the circuit which computes $f^{\oplus k}$ as a sub-circuit in the circuit that computes f. (See [GNW95] for a survey on Yao's XOR-lemma).

Another unpleasant feature of this result is that p' is always larger than $1/s$ which means that one does not benefit from taking $k > \log s$. An unpublished result which is commonly attributed to Steven Rudich shows that all "black-box"[4] proofs of the XOR-lemma suffer from this flaw. Thus, proving a result in which $s' > s$ or $p' < 1/s$ seems to be beyond our current ability, as we do not know how to handle boolean circuits other than using them as black boxes.

The direct product question was also studied in other computational models. Nisan et al, [NRS94] consider a specific variant of decision trees which they call "decision forests". A k-decision forest of depth d consists of k decision trees of depth d. Each is allowed to query all k inputs, and the i'th tree is supposed to compute $f(x_i)$. The final output of the decision forest is the concatenation of outputs of individual trees. Let us denote the class of all functions computable by depth d k-decision forests by $Forest_{k,d}$. With this terminology their result could be stated this way:

$$Suc_d^{Depth}(f) \le p \Rightarrow Suc_{k,d}^{Forest}(f^{(k)}) \le p^k$$

(Here $f^{(k)}(x_1, \cdots, x_k) = (f(x_1), \cdots, f(x_k))$, see footnote 2).

Parnafes et al, [PRW97] used the technique of Raz's parallel repetition theorem [Raz98] to prove a product theorem for "forests of c-bit communication protocols"[5]. Their result is similar in flavor

[3]In particular, our negative results work whenever $p' < p$ and apply to "intermediate" values of r'. Our positive result for communication complexity gives a tradeoff between r' and p'

[4]"Black box" refers to proofs like the ones mentioned above, that use a circuit which computes $f^{\oplus k}$ too well as a black box in a circuit that computes f too well.

[5]A forest of c-bit communication protocol is a collection of k c-bit communication protocols each is over all k inputs, and the i'th protocol is supposed to compute the function on the i'th input.

to that of [NRS94] with the exception that $p' = p^{\Omega(k/c)}$. This dependence on c comes from the technique of Raz, but whereas a dependence on c is unavoidable in the parallel repetition theorem (as was shown by [FV96]), it is open weather the result of [PRW97] is best possible for forests of communication protocols.

1.3 Our results

Our first result is a general counterexample which shows that strong direct product assertions, (or even ones with r' sufficiently larger than r) are simply not true. We can apply this counterexample whenever there is a function such that $Adv_r^{Res}(f)$ is small, yet $f \in Res_{\bar{r}}$ for $\bar{r}$ not much larger than r. This means that whenever a computational model has functions which are hard when given r units of the resource and easy when given slightly more units we cannot expect a strong direct product assertion to hold. The counterexample applies to boolean circuits, communication protocols and decision trees.

While this counterexample rules out the possibility of having strong direct product assertions, it seems to exploit defects in the formulation of the problem rather than show that our general intuition for direct product assertions is false. Intuitively, the algorithm of the counterexample is able to compute $f^{\oplus k}$ correctly with high probability by using its resources in an imbalanced way allocating a lot of its resources to specific instances. This is beneficial to the algorithm as the function of the counterexample is with high probability easy on many instances. This does not contradict our intuition why strong direct product assertions are true as this is not a counterexample to our belief that it is not beneficial for the algorithm to correlate computations on different inputs. We elaborate on this point in section 3.3. In any case, as the assertion is not true as is, in order to capture our intuition and prove a strong direct product assertion we would have to either strengthen the assumption or weaken the conclusion.

1.3.1 Strengthening the assumption: demanding more information on the function

The function presented in the counterexample has the property that it has a large subset of its inputs on which it is easy, and checking weather an input belongs to this subset is feasible for the algorithm. It is natural to ask weather a strong direct product assertion holds for functions which do not have this property. More generally, what kind of restrictions can we place on the function in order to make a strong product assertion hold?

We provide an answer to these questions for communication protocols. We do this by analyzing the discrepancy of $f^{\oplus k}$. (The discrepancy of f, denoted by $disc(f)$ measures how imbalanced is f in large rectangles). Using ideas from [NW94], we are able to show that:

$$disc(f^{\oplus k}) = O\left(disc(f)\right)^{k/3}$$

It is standard that $Adv_c^{Comm}(f) \le disc(f)2^c$, this immediately entails:

$$Adv_{kc/3}^{Comm}(f^{\oplus k}) \le O(disc(f)2^c)^{k/3}$$

This inequality has the following interpretation: If the fact that f is hard on average for c-bit communication protocols follows from the fact that f has low discrepancy then a strong product theorem holds for f. We would like to point out that the "discrepancy method" is the most common way to prove that f is hard on average for communication protocols.

1.3.2 Weakening the conclusion: imposing restrictions on the algorithm

The algorithm presented in the counterexample has the property that it uses its resource in an "unfair" way spending more than r units on particular inputs. It is natural to ask weather a strong direct product assertion holds for "fair" algorithms. More generally, what kind of restrictions can we place on the algorithm in order to make a strong direct product assertion hold?

Intuitively, the forest model of [NRS94] is such a restriction. However, we would like our algorithm to be a restricted algorithm from $Res_{\Omega(kr)}$. We suggest to impose a "fairness" restriction on the algorithm. Some evidence to the potential of this direction is that we can prove an optimal direct product theorem for "fair" decision trees.

A decision tree of depth kd over variables $x_1, \cdots, x_k$ is *fair* if on every path from the root to the leaf at most d bits from each variable are queried. Let us denote the class of fair decision trees of depth kd by $FairDepth_{kd}$. It is not hard to prove the following theorem:

$$Adv_d^{Depth}(f) \leq p \Rightarrow Adv_{kd}^{FairDepth}(f^{\oplus k}) \leq p^k$$

It is our hope that the two directions we present here can be extended to prove strong direct product assertions for stronger computational models.

1.4 Organization of the paper

In section 3 we present our counterexample. In section 4 we prove a strong direct product theorem for communication protocols via the discrepancy method. In section 5 we prove a strong direct product theorem for fair decision trees.

2 Preliminaries

We use $\oplus$ to denote the exclusive or. For two matrices A and B of size $N \times N$, we use $A \otimes B$ to denote the *tensor product* of the two matrices. More precisely $A \otimes B$ is an $N^2 \times N^2$ matrix. We we think of this matrix as an $N \times N$ matrix with entries being matrices of size $N \times N$ and place a copy of the matrix $A_{ij} \cdot B$ in the i'th row and j'th column. The tensor product of A with itself k times is denoted by $A^{\otimes k}$.

We use $Size_s$ to denote the class of all functions (over arbitrary number of inputs) computable by boolean circuits of size s.

We use $Comm_c$ to denote the class of all functions of two arguments which can be computed by a c-bit communication protocol. The exact definition of a communication protocol can be found in any textbook on this subject, (i.e. [KN97]). The only property of such protocols used in this paper is that such a protocol induces a partition of the inputs into 2^c rectangles.

We use $Depth_d$ to denote the class of all functions computed by a decision tree of depth d. Whereas a decision tree is a binary tree in which every internal node is labeled with a specific bit of the input, and leafs are labeled with outputs. An input to the decision tree defines a path from root to leaf in the obvious way, and the output of the tree on this input is the leaf label.

3 A general counterexample

In this section we give a general counterexample to direct product results with $r' = kr$ which works for boolean circuits, decision trees and communication protocols. We show that given a function which is hard given r units of the resource and easy given slightly more units, we can construct a function which is hard given r units of the resource, and yet computing it on k independent inputs is easy. We present the example using our general notation in section 3.1 and then draw conclusions for specific models in section 3.2. In section 3.3 we discuss the implications of this counterexample.

3.1 The general setting

We will present a function which is hard on average given r units of the resource, yet $f^{\oplus k}$ can be computed correctly with probability $1-2^{-k}$ given r' units of the resource, for r' sufficiently larger than r.

The counterexample works assuming the existence of a function which is hard given r units of the resource, and easy given slightly more units. Formally, we assume the existence of a function $g:\{0,1\}^n \rightarrow \{0,1\}$ and $r < \bar{r}$ such that:

- $Suc_r^{Res}(g) \leq 3/4$
- $g \in Res_{\bar{r}}$.

Another ingredient is an easy function (over few inputs) which answers one on a prescribed fraction of its inputs. Formally, Given a number $q < 1$ we assume the existence of a function $h:\{0,1\}^l \rightarrow \{0,1\}$ and a small number r^* such that

- $h \in Res_{r^*}$
- $\Pr_{y\in_R\{0,1\}^l}[h(y)=1]=q$

Our counterexample function is a combination of the easy and hard functions.

Definition 1 *We define a function $f_q:\{0,1\}^n \times \{0,1\}^l \rightarrow \{0,1\}$ in the following way.*

$$f(x,y)=\begin{cases} g(x) & h(y)=1 \\ 0 & h(y)=0 \end{cases}$$

An algorithm for computing $f^{\oplus k}$ can utilize its resource smartly by spending a lot of the resource on the (expectedly few) inputs in which f involves the hard function, and spend a very small amount on other inputs. This is made formal in the following lemma.

Lemma 1 *The following inequalities hold:*

- $Suc_r^{Res}(f) \leq 1-q/4$
- *If $r' \geq 2qk\bar{r}+kr^*$ then $Suc_{r'}^{Res}(f^{\oplus k}) \geq 1-2^{-\Omega(k)}$*

Proof: (of lemma 1) For the first item note that an algorithm which is correct on f with probability greater than $1-q/4$ must be correct on g with probability greater than 3/4. For the second and third item, note that when $(y_1,\cdots,y_k)$ are randomly chosen, we expect qk of them to have $h(y_i)=1$. By Chernoff's inequality The probability that more than $2qk$ of them have $h(y_i)=1$ is bounded by $2^{-\Omega(k)}$. We can check which of the y_i's have $h(y_i)=1$ using kr^* units of the resource. Assuming the constant function zero can be computed using 0 units of the resource, we can compute the function f on (x_i,y_i)'s such that $h(y_i)$ Assuming that there are at most $2qk$ of these we can use $2qk\bar{r}$ units to compute the outputs of the "hard inputs". Thus, $Suc_{kr^*+2qk\bar{r}}^{Res}(f^{\oplus k}) \geq 1-2^{-\Omega(k)}$. •

Corollary 1 *If $\bar{r} \leq (r - r^*)/2q$ then $Suc^{Res}_{kr}(f^{\oplus k}) \geq 1 - 2^{-\Omega(k)}$*

Remark 1 *It should be noted that the function f constructed here isn't as pathological as it may seem at first glance. Impagliazzo's hard core theorem [Imp95], shows that (at least in the boolean circuit model) every function f with $Suc^{Size}_s(f) \leq 1 - q$ has a large subset of the inputs on which any (slightly smaller) circuit succeeds with probability roughly $1/2$. In our example the "hard core" of f is the function g. The unnatural state of affairs in our example is that the function is easy outside of the hard core, and deciding weather an input is in the hard core is an easy computational task.*

3.2 Conclusions for specific models

In order to use the counterexample from the previous section we will show the existence of the required "building block" functions g and h for various computational models.

We will use the same function as "h" in all constructions. Namely we choose $q = 2^{-l}$ for integer l, and define $h : \{0,1\}^l \to \{0,1\}$ to take the value one if all its inputs are zeroes, and zero otherwise. It is immediate to verify that h is in $Size_{O(l)}, Comm_l, Depth_l$, and accepts a q-fraction of its inputs.

3.2.1 Boolean Circuits

There is a hierarchy theorem for circuits which provides us with a function ψ which is computable in size s but not in size $s - n$. The following formulation is weaker than the best known result.

Theorem 1 *[PW86] If $\bar{s} < \frac{2^n}{n}$ then there is a function $\phi : \{0,1\}^n \to \{0,1\}$, such that $\phi \in Size_{\bar{s}}$ and $\phi \notin Size_{\bar{s}-n}$.*

We can now use hardness amplification techniques to convert ϕ into a function which is hard on average for slightly smaller size[6].

Theorem 2 *[Imp95, STV99] If $\phi \notin Size_{\hat{s}}$ then there exists a function $g : \{0,1\}^{O(n)} \to \{0,1\}$ such that $Suc^{Size}_s(g) \leq 3/4$, for $s = \hat{s}/n^{O(1)}$. Moreover, $g \in EXP^{\Phi}$.*

By combining theorems 1,2 we get the existence of the function g we wanted.

Corollary 2 *If $s < 2^{\gamma n}$, (for some constant γ) then there exists a function $g : \{0,1\}^n \to \{0,1\}$ such that $Suc^{Size}_s(g) \leq 3/4$ and $g \in Size_{\bar{s}}$ for $\bar{s} = sn^{O(1)}$.*

Using lemma 1 we conclude that $Adv^{Size}_{\Omega(ks)}(f^{\oplus k}) > Adv^{Size}_s(f)$ for $q = 1/n^c$ and $k = d \log n$. Where c and d are constants which can depend on the constant hidden in the Ω-notation. This means that strong direct product assertions are not true for boolean circuits[7].

[6]Some of these hardness amplification techniques involve proving XOR-lemmas for circuits. Thus, the counterexample is actually based on a true direct product assertion! The hardness amplification results used are explicit in the sense that the new function can be efficiently computed given access to the initial one. This is reflected in the "moreover clause" in the next lemma, and is not necessary for our purposes. The result stated is a strong variant of hardness amplification in the sense that the new function has roughly the same number of inputs as the initial one.

[7]We can also draw some conclusion for general s'. The only weakness of this example is that the starting advantage $p = 1 - 1/n^{O(1)}$ is not a constant.

3.2.2 Communication Protocols

For communication complexity we use the inner product function $g(x, y) = \sum_{1\leq i\leq n} x_i y_i$. As all functions over n bit inputs, $g \in Comm_n$. It is known that g is very hard on average given $n/4$ bits of communication[8].

Theorem 3 *[CG88]* $Adv^{Comm}_{n/4}(g) \leq 2^{-\Omega(n)}$.

Using lemma 1 we conclude that $Adv^{Comm}_{kc}(f^{\oplus k}) > Adv^{Comm}_{c}(f)$ for a sufficiently small constant q and a sufficiently large constant k. Thus, there are no strong direct product assertions for communication protocols evan when starting from a constant advantage.

Remark 2 *In this example c is very large and in particular $\bar{c} = n$ this can be avoided by "padding" the inputs of the two players to increase n.*

3.2.3 Decision Trees

For decision trees we choose g to be the parity function $g(x) = \oplus_{1\leq i\leq n} x_i$. It is immediate that $g \in Depth_n$ and $Adv^{Depth}_{n-1}(g) = 0$. Once again we can use lemma 1 to get a counterexample with the same behavior of parameters as the counterexample for communication protocols.

3.3 How bad is this counterexample?

It seems that the counterexample is not so bad in the sense that it does not contradict our intuition as to why direct product assertions are true. When proving a direct product assertion, the main task is to show that the algorithm does not benefit from correlating computations on different inputs. In the counterexample we presented no such correlations occur. Instead, the algorithm uses its resource in an unbalanced way, spending a lot of it on particular inputs. In particular, the algorithm is able to compute f on single instances with advantage greater than p. This is something which is ruled out when $r' \leq r$.

We would like to change the formulation of the direct product problem to rule out such cases. In the next two sections we suggest two such strengthenings. One involves adding assumptions on the function f, in hope that such assumptions can prevent the situation of the counterexample. Intuitively, if f is hard in a "robust way", any additional resources spent on f on one instance will result in a "loss" in another instance, and it will not be beneficial to treat the inputs unfairly. The other involves restricting the algorithm in a way that insures that it cannot have advantage greater than p when attempting to compute f on any single coordinate.

4 A discrepancy product theorem

In the previous section we've seen that a direct product assertion for communication protocols is not true if we allow the protocol trying to compute $f^{\oplus k}$ to communicate more bits than the protocol attempting to compute f. In this section we show that if f has "low discrepancy" then a strong direct product theorem holds for f.

It will be convenient to think of the outputs of a communication complexity problem as being $\{-1, 1\}$ rather than $\{0, 1\}$. Thus, such a problem can be viewed as a matrix with entries in $\{-1, 1\}$.

[8]Preparing for section 4, we remark that the proof of that statement works by showing that g has very low discrepancy.

This matrix is not necessarily a square matrix, as the input of the different players may be chosen from sets of different sizes. Still, in the remainder of this section we will assume that the matrix is a square matrix. Our results follow for the general case as well, and this assumption is made only to simplify the presentation. The choice of $\{-1, 1\}$ is made so that the tensor product of A with itself k times, (denoted by $A^{\otimes k}$) is exactly the matrix of the communication problem $A^{\oplus k}$.

For a set $C \subseteq [N]$, we use χ_C to denote the *characteristic vector* of C, that is $(\chi_C)_i = 1$ for $i \in C$ and $(\chi_C)_i = 0$ for $i \notin C$.

Definition 2 *Let A be an N by N matrix with entries in $\{-1, 1\}$. For a rectangle $R = C \times D$ where $C, D \subseteq [N]$ we define:*

$$disc_R(A) = \frac{|\chi_C^t A \chi_D|}{N^2}$$

The discrepancy of A is defined in the following way:

$$disc(A) = max_R disc_R(A)$$

where the maximum is taken over all rectangles $R = C \times D$ with $C, D \subseteq [N]$.

For a fixed rectangle $R = C \times D$, $\frac{|\chi_C^t A \chi_D|}{|C \times D|}$ measures how imbalanced is the matrix A in the rectangle R. If R is a rectangle reached in a leaf of a communication protocol then half this quantity is the advantage the protocol gets over random guessing in the rectangle R. The definition of $disc_R(A)$ multiplies this quantity by $\frac{|C \times D|}{N^2}$ to take into account the volume of the rectangle. More precisely, the advantage over random guessing is multiplied by the volume of the rectangle R to give the contribution of R to the advantage of the protocol over random guessing. This normalization is made so that low discrepancy will imply that the problem is hard on average. This is made formal in the following lemma.

Lemma 2 $Adv_c^{Comm}(A) \leq disc(A)2^c$

Proof: (of lemma 2) Let P be the c-bit communication protocol which achieves $Adv_c^{Comm}(A)$. A c-bit communication protocol partitions A into 2^c disjoint rectangles. On each rectangle $R_i = C_i \times D_i$ the advantage of the protocol is bounded by $\frac{|\chi_{C_i}^t A \chi_{D_i}|}{|C_i \times D_i|}$. We can now bound the advantage of P:

$$Adv_c^{Comm}(A) \leq \sum_{1 \leq i \leq 2^c} \frac{|C_i \times D_i|}{N^2} \cdot \frac{|\chi_{C_i}^t A \chi_{D_i}|}{|C_i \times D_i|}) \leq 2^c disc(A)$$

•

The requirement that $disc(A)$ is small is stronger than that A is hard on average. Still, the most common way of showing that communication problems are hard on average is by showing that they have low discrepancy. Intuitively, low discrepancy provides a "smooth" hardness condition which rules out the counterexample of the previous section.

Remark 3 *Note that for a sub-matrix B of A, the discrepancy of B is upper bounded by the discrepancy of A times the ratio of the volumes of B and A. Thus, by lemma 2, low discrepancy means that any large sub-matrix of A is average case hard. Another nice feature of this setup is that checking membership in B is easy for communication protocols and can be done by exchanging 2 bits. Thus, it captures our intuition that the subsets of inputs on which we bound the function are easily computable. This should be compared to remark 1.*

The main theorem of this paper shows that the discrepancy of a taking the tensor product of A with itself k times goes down exponentially with k.

Theorem 4 $disc(A^{\otimes k}) = O(disc(A))^{\frac{k}{3}}$

This has the following interpretation: Suppose, (as is often the case), that fact that $Adv_c^{Comm}(A) < p$ follows from the fact that $disc(A) < p2^{-c}$. In that case:

$$Adv_{kc/3}^{Comm}(A^{\oplus k}) \leq disc(A^{\otimes k})2^{kc/3} \leq O(2^c disc(A))^{k/3} \leq O(p)^{k/3}$$

In words, we get a strong direct product theorem for A. This is stated with more generality in the next corollary.

Corollary 3 *For every c', $Adv_{c'}^{Comm}(A^{\oplus k}) \leq O(disc(A))^{k/3}2^{c'}$.*

Note that in this formulation one can get rid of the constant 3, and get a result on $Adv_{kc}^{Comm}(A^{\oplus k})$ in terms of $disc(A)$.
The proof of the theorem will require the definition of the spectral norm of a matrix A.

Definition 3 *For a vector x we use $||x||_2$ to denote the L2-norm of x. For a matrix A, $||A||_2$ is defined to be $max_{x:||x||_2=1}||Ax||_2$.*

It will be useful to consider equivalent definitions of this norm.

Fact 1 *Equivalent definitions for $||A||_2$ are:*

1. $||A||_2 = max_{x:||x||_2=1,y:||y||_2=1}|x^tAy|$.
2. $||A||_2 = max\{\sqrt{\lambda}|\ \lambda$ *is an eigenvalue of* $A^tA\}$

A useful property of $||A||_2$ is that it is multiplicative under tensor product.

Fact 2 $||A^{\otimes k}||_2 = ||A||_2^k$

Proof: (of fact 2) The proof of fact 2 consists of two steps: The first is to show this is true for symmetric matrices. If A is symmetric then there is an orthonormal basis of eigenvectors of A and $||A||_2 = |\lambda|$, where λ is the maximal eigenvalue of A in absolute value. It is easy to see that the eigenvalues of $A^{\otimes k}$ are exactly all products of eigenvalues of A. Thus, $||A||_2 = |\lambda|^k$. The fact now follows for non symmetric matrices by using the second item of fact 1, and the observation that $(A^{\otimes k})^tA^{\otimes k} = (A^tA)^{\otimes k}$. •

In the remainder of this section we prove theorem 4. The first step is to express the discrepancy in terms of the spectral norm. We then use the multiplicativity of the spectral norm to get the conclusion. The second item of fact 1 enables us to upper bound the discrepancy using the spectral norm.

Lemma 3 $disc(A) \leq \frac{||A||_2}{N}$

Proof: (of lemma 3) Let $R = C \times D$ be the rectangle such that $disc(A) = disc_R(A)$. We have that $disc(A) = \frac{|\chi_C^t A \chi_D|}{N^2}$. We define $x = (\chi_C)/\sqrt{|C|}$ and $y = (\chi_D)/\sqrt{|D|}$. Note that $||\chi_C||_2 = ||\chi_D||_2 = 1$. it follows from the first item of fact 1 that

$$||A||_2 \geq |x^t A y| = \frac{|\chi_C^t A \chi_D|}{\sqrt{|C|}\sqrt{|D|}} \geq \frac{|\chi_C^t A \chi_D|}{N} = disc(A)N$$

•

In [NW94], Nisan and Wigderson address the so called "log-rank conjecture" and show that $disc(A) = \Omega(1/rank(A)^{3/2})$. We will use ideas from that paper to lower bound the discrepancy using the spectral norm.

Lemma 4 $disc(A) = \Omega\left(\frac{||A||_2}{N}\right)^3$

We start by showing that theorem 4 easily follows from lemma 4.

Proof: (of theorem 4)

$$disc(A^{\otimes k}) \leq \frac{||A^{\otimes k}||_2}{N^k} = \left(\frac{||A||_2}{N}\right)^k = O\left(disc(A)\right)^{k/3}$$

The first inequality follow from lemma 3. The second follows from fact 2, and the third follows from lemma 4. •

We want to prove lemma 4 in a similar way to the previous lemma. The first step is a way to transform a bilinear form with arbitrary vectors into one with characteristic vectors. Such a transformation was given by Nisan and Wigderson in [NW94].

Lemma 5 *[NW94] Let u, v be vectors such that $||u||_\infty, ||v||_\infty \leq 1$, then there exist a rectangle $R = C \times D$ such that $|\chi_C^t A \chi_D| \geq |u^t A v|/4$.*

For completeness we give the proof of this lemma.

Proof: (of lemma 5) We "split" u and v to their positive coordinates and negative coordinates, having $u = u^+ - u^-$ and $v = v^+ - v^-$. We now have that $||u^+||_\infty, ||u^-||_\infty, ||v^+||_\infty, ||v^-||_\infty \leq 1$ and the four vectors are non-negative. Since $|u^t A v| = |(u^+)^t A(v^+) - (u^+)^t A(v^-) - (u^-)^t A(v^+) + (u^-)^t A(v^-)|$ then one of the four terms is larger in absolute value than $|u^t A v|/4$. Setting C to be the non-negative coordinates of the u-part of this term and D to be the non-negative coordinates of the v-part of this term, we have that $|\chi_C^t A \chi_D| \geq |u^t A v|/4$. •

We are now tempted to use lemma 5 directly to prove lemma 4. That is start from u, v such that $||A||_2 = u^t A v$ and get a rectangle $R = C \times D$ with roughly the same value. This will not do since to get a bound on $disc(A)$ we have to divide by N^2. Thus, the above argument only gives the non-impressive estimate: $disc(A) \geq \frac{||A||_2}{4N^2}$. Had it been the case that u and v had $||u||_\infty, ||v||_\infty \leq \rho < 1$ we could deduce that $\rho^2 disc(A) \geq \frac{||A||_2}{4N^2}$ and do better. We will now show that $||A||_2$ is obtained by such u and v.

Lemma 6 *There are vectors u, v with $||u||_\infty, ||v||_\infty \leq \rho$ such that $|u^t A v| \geq ||A||_2 - \frac{2\sqrt{N}}{\rho}$*

Proof: (of lemma 6) Let x and y be vectors such that $||x||_2 = ||y||_2 = 1$ and $||A||_2 = |x^t Ay|$. Let $I = \{i|x_i > \rho\}$ and $J = \{j|y_j > \rho\}$. Let u be the vector obtained from x by setting the coordinates in I to zero, and v be the vector obtained from y by setting the coordinates in J to zero. Note that $|I|, |J| \leq 1/\rho^2$, (since otherwise the contribution of elements in I (J) to the norm of x (y) is greater than one). We now have that:

$$|u^t Av| \geq ||A||_2 - |\sum_{i \in I, j \in [N]} x_i a_{ij} y_j + \sum_{i \in [N], j \in J} x_i a_{ij} y_j|$$

We will now argue that the two terms on the right hand side are small because they involve small rectangles. We will bound the first term, and the second will follow the same way.

$$|\sum_{i \in I, j \in [N]} x_i a_{ij} y_j| \leq \sum_{i \in I, j \in [N]} x_i y_j = \sum_{i \in I} x_i \sum_{j \in [N]} y_j \leq \sqrt{|I|}\sqrt{N}||x||_2||y||_2 \leq \frac{\sqrt{N}}{\rho}$$

(where the second inequality follows from the Cauchy-Schwartz inequality). Plugging this in the previous calculation we have that:

$$|u^t Av| \geq ||A||_2 - \frac{2\sqrt{N}}{\rho}$$

•

Lemma 4 follows from the previous lemmas.

Proof: (of lemma 4) Given a matrix A, we set $\rho = 3\sqrt{N}||A||$. Let u, v be the vectors which existence is given by lemma 6. We now define new vectors $\bar{u} = u/\rho$ and $\bar{v} = v/\rho$. Note that $||\bar{u}||_\infty, ||\bar{v}||_\infty \leq 1$. Using lemma 5 there exists a rectangle $R = C \times D$ such that

$$|\chi_C^t A \chi_D| \geq \frac{|\bar{u}^t A \bar{v}|}{4} = \frac{|u^t Av|}{4\rho^2} = \frac{||A||_2 - \frac{2\sqrt{N}}{\rho}}{4\rho^2} = \frac{||A||_2^3}{108N}$$

(Note that this result is valid even if $\rho > 1$. In that case, it follows directly without using anything). We conclude that $disc(A) = \Omega\left(\frac{||A||_2}{N}\right)^3$. •

5 Fair decision trees

In this section we prove an optimal direct product theorem for fair decision trees. We start with the definition of "fairness" for decision trees.

Definition 4 *A decision tree over inputs $x_1, \cdots, x_k$ is $(d_1, \cdots, d_k)$-*fair *if for every $1 \leq i \leq k$ and on every path from root to leaf the decision tree queries at most d_i bits from x_i. A decision tree is d-fair if it is $(d, \cdots, d)$-fair.*

Let $FairDepth_{d_1,\cdots,d_k}$ denote the class of functions over inputs $x_1, \cdots, x_k$ computed by $d_1, \cdots, d_k$ decision trees. Let $FairDepth_{kd}$ denote $FairDepth_{d,\cdots,d}$.

Theorem 5 $Adv_{kd}^{FairDepth}(f^{\oplus k}) \leq Adv_d^{Depth}(f)^k$.

The proof of this theorem is by induction and is almost similar to that of [NRS94]. To simplify the notation we will prove it for $k = 2$. The proof for general k follows the same way. To have a stronger induction hypothesis we will prove the following stronger version.

Lemma 7 *For every two functions f_1, f_2 and numbers d_1, d_2*

$$Adv_{d_1,d_2}^{FairDepth}(f \oplus g) \leq Adv_{d_1}^{Depth}(f_1) \cdot Adv_{d_2}^{Depth}(f_2)$$

Proof: (of lemma 7) We prove the lemma by induction over $d_1 + d_2$. If $d_1 + d_2 = 0$ then the decision tree does not base its answer on the inputs. It is now standard to check that indeed $Adv_{0,0}^{FairDepth}(f_1 \oplus f_2) = Adv_0^{Depth}(f_1) \cdot Adv_0^{Depth}(f_2)$. To bound $Adv_{d_1,d_2}^{FairDepth}(f_1 \oplus f_2)$ for $d_1 + d_2 > 0$, let T be a decision tree which achieves this advantage. Without loss of generality the first query of T is from x_1. We will use the notation $x_1 = (y, b)$ where b is the bit queried by T and y is the remaining bits. We denote the two sub-trees of T by T^0 and T^1 respectively. For $b \in \{0, 1\}$, we define functions $g^b(y) = f_1((y, b))$. We now have:

$$Adv_{d_1,d_2}^{FairDepth}(f_1 \oplus f_2) = \sum_{b \in B} (1/2) \cdot Adv^{T_b}(g^b \oplus f_2)$$

Here, $Adv^T(f)$ is used to denote the advantage of T on f. However, T^0 and T^1 are of depth $d_1 - 1 + d_2$, and are in $FairDepth_{d_1-1,d_2}$. Applying our induction hypothesis we continue and get that:

$$\leq \sum_{b \in B} (1/2) \cdot Adv_{d_1-1,d_2}^{FairDepth}(g^b \oplus f_2) \leq Adv_{d_2}^{Depth}(f_2) \sum_{b \in B} (1/2) \cdot Adv_{d_1-1}^{Depth}(g^b)$$

Consider trees P^0, P^1 which achieve the advantage on g^0, g^1. We now construct a tree P of depth d_1 which starts by querying b and depending on the outcome activates P^b. We now have that:

$$\sum_{b \in B} (1/2) \cdot Adv_{d_1-1}^{Depth}(g^b) = Adv^p(f_1) \leq Adv_{d_1}^{Depth}(f_1)$$

The first equality follows from the definition of P and the second from the fact that P is of depth d_1. Plugging this in our previous calculation we prove the lemma. •

6 Open problems

The natural direction is to try and extend the ideas of this paper to stronger computational models. Particular such extensions are:

Extend the technique of section 4 to non-uniform probability distributions. The reason the current argument does not extend is that we use the fact that small rectangles have small probability.

It will also be nice to be able to handle one sided discrepancy, that is removing the absolute value from the definition 2. This may require totally different techniques as we are unaware of an algebraic interpretation of one sided discrepancy. Both extension are motivated by the lower bound on the disjointness function [BFS86, KS87, Raz92] which uses a one sided discrepancy in a non-uniform probability distribution.

The next model on which we may want to prove a strong product assertion by imposing a fairness restriction is communication complexity. It seems that the result of [PRW97] regarding the forest model does not extend to a fair kc-bit communication protocol.

For some models of computations it may not be obvious how to define "fairness". We suggest the following definition: An algorithm for $x_1, \cdots, x_k$ is fair, if for every i and for every $a_1, \cdots, a_{i-1}, a_{i+1}, \cdots, a_k$ the "computation" of the algorithm over $a_1, \cdots, a_{i-1}, x, a_{i+1}, \cdots, a_k$ for fixed a's and varying x can be simulated by an algorithm with r units of the resource when given x.

An alternative approach is to replace the "syntactic" fairness condition by a "semantic" one. Intuitively, the fairness restriction is supposed to guarantee that the algorithm cannot compute the function on individual coordinates with advantage greater than p. If it is hard to impose fairness restrictions, one may try to prove the direct product assertion only for algorithms which obey this semantic restriction.

Acknowledgments

I would like to thank my advisor Avi Wigderson for introducing me to this area, for countless conversations and for his guidance and support.

References

[BFS86] László Babai, Péter Frankl, and János Simon. Complexity classes in communication complexity theory (preliminary version). In *27th Annual Symposium on Foundations of Computer Science*, pages 337–347, Toronto, Ontario, Canada, 27–29 October 1986. IEEE.

[CG88] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, April 1988.

[FV96] Uriel Feige and Oleg Verbitsky. Error reduction by parallel repetition—a negative result (preliminary version). In *Proceedings, Eleventh Annual IEEE Conference on Computational Complexity*, pages 70–76, Philadelphia, Pennsylvania, 24–27 May 1996. IEEE Computer Society Press.

[GNW95] Oded Goldreich, Noam Nisan, and Avi Wigderson. On yao's xor-lemma. Technical report, Electronic Colloquium on Computational Complexity, 1995. `http://www.eccc.uni-trier.de/eccc`.

[Imp95] Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *36th Annual Symposium on Foundations of Computer Science*, pages 538–545, Milwaukee, Wisconsin, 23–25 October 1995. IEEE.

[IW97] Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 220–229, El Paso, Texas, 4–6 May 1997.

[KN97] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge unuversity press, 1997.

[KS87] Balasubramanian Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection (preliminary version). In *Proceedings, Structure in*

Complexity Theory, Second Annual Conference, pages 41–47, Cornell University, Ithca, NY, 16–19 June 1987. IEEE Computer Society Press.

[Lev85] Leonid A. Levin. One-way functions and pseudorandom generators. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, pages 363–365, Providence, Rhode Island, 6–8 May 1985.

[NRS94] Noam Nisan, Steven Rudich, and Michael Saks. Products and help bits in decision trees. In *35th Annual Symposium on Foundations of Computer Science*, pages 318–329, Santa Fe, New Mexico, 20–22 November 1994. IEEE.

[NW94] Noam Nisan and Avi Wigderson. On rank vs. communication complexity. In *35th Annual Symposium on Foundations of Computer Science*, pages 831–836, Santa Fe, New Mexico, 20–22 November 1994. IEEE.

[PRW97] Itzhak Parnafes, Ran Raz, and Avi Wigderson. Direct product results and the GCD problem, in old and new communication models. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 363–372, El Paso, Texas, 4–6 May 1997.

[PW86] Michael S. Paterson and Ingo Wegener. Nearly optimal hierarchies for network and formula size. *Acta Informatica*, 23(2):217–221, 1986.

[Raz92] A. A. Razborov. On the distributed complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 14 December 1992. Note.

[Raz98] Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, June 1998.

[STV99] Maduh Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the xor lemma. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, 1999.

[Yao82] Andrew C. Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, 3–5 November 1982. IEEE.