Recall that an (N, K) two-source hitter is a function $f\colon [N]^2 \to \{0,1\}$ that is not constant on any product of sets $S \times T$ of size K each. We showed $(N, 2\log N + 1)$ two-source hitters exist, but even after optimizations the best construction of such a hitter took us around 2^{n^2} time, where $N = 2^n$.

Here is a simple proposal for a two-source hitter, due to Chor and Goldreich:

$$f(x,y) = \langle x, y\rangle \bmod 2 = x_1y_1 + x_2y_2 + \cdots + x_ny_n \mod 2$$

where x_i and y_i are the ith bist of x and y, respectively.

How well does this work? Actually it can never work too well: S and T must have size at least $\sqrt{N}$ (at least when n is even; can you see why)? However, if S and T are larger than that, we have

Theorem 1. *The function $\langle x, y\rangle \bmod 2$ is a $(N, \sqrt{N}+1)$ two-source hitter.*

Proving this theorem from scratch is not impossible, although I suspect it would be a bit challenging. Using Fourier analysis, the answer comes out staring at us.

1 Fourier analysis

Fourier analysis allows us to (literally) look at Boolean functions from a different angle. Instead of Boolean functions, we look at more general functions that take inputs from $\{0,1\}^n$, but produce a real-valued output. A Boolean function is the special case when there are only two possible output values among all reals, usually the values $\{0,1\}$ or $\{-1,1\}$.

We can think of a function f from $\{0,1\}^n$ to the real numbers as a vector in 2^n dimensional space. The standard basis for this space consists of those vectors that have entry 1 in exactly one of the 2^n positions, and entry 0 everywhere else. These are the 2^n functions δ_a ($a \in \{0,1\}^n$), which take value 0 everywhere except at a, where they take value 1.

Now any function f over $\{0,1\}^n$ can be written as a linear combination of the functions δ_a. The coefficient in front of δ_a – the value $f(a)$ – is then simply just the inner product between f and δ_a.

The Fourier transform allows us to express f in a different orthogonal basis. This is the basis consisting of the character functions $\chi_a\colon \{0,1\}^n \to \mathbb{R}$

$$\chi_a(x) = (-1)^{\langle x,a\rangle}, \qquad a \in \{0,1\}^n.$$

It is easy to verify that, viewed as vectors, the functions χ_a are orthogonal and each has norm $2^{n/2}$. Instead of normalizing these functions, it is more convenient to work with respect to the normalized inner product

$$\text{inner product of } f \text{ and } g = \mathrm{E}_{x\sim\{0,1\}^n}[f(x)g(x)] = \frac{1}{2^n}\sum_{x\in\{0,1\}^n} f(x)g(x).$$

In this basis, a function $f\colon \{0,1\}^n \to \mathbb{R}$ can be written as a linear combination of the χ_as:

$$f(x) = \sum_{a\in\{0,1\}^n} \hat{f}_a \cdot \chi_a(x)$$

where the coefficient $\hat{f}_a$ is just the inner product of f and χ_a:

$$\hat{f}_a = \mathrm{E}_{x \sim \{0,1\}^n}[f(x)\chi_a(x)]. \tag{1}$$

Parseval's identity Since the bases $\{\delta_a\}$ and $\{\chi_a\}$ are related by an orthonormal transformation (up to the normalization factor $2^{n/2}$), the sum of the squares of the coefficients of f in these two bases should be the same. This is Parseval's identity:

$$\sum_{a \in \{0,1\}^n} \hat{f}_a^2 = \mathrm{E}_{x \sim \{0,1\}^n}[f(x)^2].$$

We can also prove this algebraically by a calculation:

$$\begin{aligned}
\mathrm{E}_{x \sim \{0,1\}^n}[f(x)^2] &= \mathrm{E}_{x \sim \{0,1\}^n}\left[\left(\textstyle\sum_{a \in \{0,1\}^n} \hat{f}_a \chi_a(x)\right)^2\right] \\
&= \mathrm{E}_{x \sim \{0,1\}^n}\left[\textstyle\sum_{a,b \in \{0,1\}^n} \hat{f}_a \hat{f}_b \chi_a(x)\chi_b(x)\right] \\
&= \textstyle\sum_{a,b \in \{0,1\}^n} \hat{f}_a \hat{f}_b \, \mathrm{E}_{x \sim \{0,1\}^n}[\chi_a(x)\chi_b(x)] \\
&= \textstyle\sum_{a \in \{0,1\}^n} \hat{f}_a^2
\end{aligned}$$

as the only surviving terms in the summation over a and b are those where $a = b$.

If f takes values in the range $[-1, 1]$, we obtain that the sum of the squares of the Fourier coefficients is at most 1. If f takes only the values -1 and 1, then the sum of the squares is exactly 1, and so the squares of the Fourier coefficients can be thought of as a probability distribution over $\{0,1\}^n$.

2 Analysis of the Chor-Goldreich construction

Let us see immediately how this can be used to analyze the Chor-Goldreich construction. The first step in doing Fourier analysis is to come up with an algebraic representation of the problem we are looking at. In the Chor-Goldreich construction, we want to say that there is some variation among the values $\langle x, y \rangle$, where x and y range from S and T respectively. One way to say this is that when x and y are chosen at random from their respective sets, the expected value $\mathrm{E}[\langle x, y \rangle]$ is not identically zero or identically one:

$$0 < \mathrm{E}_{x \sim S, y \sim T}[\langle x, y \rangle \bmod 2] < 1.$$

Instead of working with a separate lower bound and upper bound, it is more convenient to just have one. We can accomplish this by the usual "change of constants" from $\{0,1\}$ to $\{1,-1\}$. In this notation we want to show that:

$$|bias| < 1 \quad \text{where} \quad bias = \mathrm{E}_{x \sim S, y \sim T}[(-1)^{\langle x,y \rangle}].$$

Now comes a useful trick in Fourier analysis: We convert sets into functions. In our case, we will replace S and T by their respective *indicator functions*:

$$f(x) = \begin{cases} 1, & \text{if } x \in S \\ 0, & \text{otherwise} \end{cases} \qquad g(y) = \begin{cases} 1, & \text{if } y \in T \\ 0, & \text{otherwise} \end{cases}$$

We can write:

$$\begin{aligned} bias &= \frac{1}{|S| \cdot |T|} \sum_{x,y \in \{0,1\}^n} f(x) g(y) (-1)^{\langle x,y \rangle} \\ &= \frac{2^n}{|S| \cdot |T|} \sum_{x \in \{0,1\}^n} f(x) \, \mathrm{E}_{y \sim \{0,1\}^n} [g(y) (-1)^{\langle x,y \rangle}] \\ &= \frac{2^n}{|S| \cdot |T|} \sum_{x \in \{0,1\}^n} f(x) \cdot \hat{g}_x. \end{aligned}$$

Another salient feature of these proofs is the Cauchy-Schwarz inequality.

$$\sum_{x \in \{0,1\}^n} f(x) \cdot \hat{g}_x \leq \sqrt{\sum_{x \in \{0,1\}^n} f(x)^2} \cdot \sqrt{\sum_{x \in \{0,1\}^n} \hat{g}_x^2}.$$

By definition, the first term equals $\sqrt{|S|}$. By Parseval's identity, the second term equals $\sqrt{|T|/2^n}$ and

$$bias \leq \sqrt{2^n / |S||T|}$$

so if $|S|, |T| > 2^{n/2}$, as we assumed, then $\langle x, y \rangle$ is indeed a two-source hitter.

3 The linearity test

There are two different ways of saying that a function $f \colon \{0,1\}^n \to \{0,1\}$ is linear:

1. f is linear if it has the form $f(x) = a_1 x_1 + \cdots + a_n x_n$.
2. f is linear if $f(x+y) = f(x) + f(y)$ for all $x, y \in \{0,1\}^n$.

It is clear that the first characterization implies the second. What about the other way? You can argue this direction with some algebra. But there is an elegant proof via Fourier analysis that has nice extensions relevant in computer science.

So let's assume that $f(x+y) = f(x) + f(y)$ for all $x, y \in \{0,1\}^n$ and see what we can deduce about f itself. To apply Fourier analysis, we replace the $0, 1$ outputs of f by $1, -1$ via the usual substitution $F(x) = (-1)^{f(x)}$. Then the assumption becomes $F(x+y) = F(x)F(y)$, or

$$F(x)F(y)F(x+y) = 1 \quad \text{for all } x, y \in \{0,1\}^n.$$

Averaging over all pairs x, y, we get

$$\mathrm{E}_{x,y \sim \{0,1\}^n} [F(x)F(y)F(x+y)] = 1. \tag{2}$$

Fourier analysis is often helpful with expressions that look like the one on the left. Let's see what we get:

$$\begin{aligned} \mathrm{E}[F(x)F(y)F(x+y)] &= \mathrm{E}_{x,y \sim \{0,1\}^n} \Big(\sum_{a \in \{0,1\}^n} \hat{F}_a \chi_a(x) \Big) \Big(\sum_{b \in \{0,1\}^n} \hat{F}_b \chi_b(y) \Big) \Big(\sum_{c \in \{0,1\}^n} \hat{F}_c \chi_c(x+y) \Big) \\ &= \mathrm{E}_{x,y \sim \{0,1\}^n} \sum_{a,b,c \in \{0,1\}^n} \hat{F}_a \hat{F}_b \hat{F}_c \chi_a(x) \chi_b(y) \chi_c(x+y) \\ &= \sum_{a,b,c \in \{0,1\}^n} \hat{F}_a \hat{F}_b \hat{F}_c \cdot \mathrm{E}_{x,y \sim \{0,1\}^n} [\chi_a(x) \chi_b(y) \chi_c(x+y)]. \end{aligned}$$

Recalling that $\chi_a(x) = (-1)^{\langle a,x\rangle}$, we can write

$$\mathrm{E}_{x,y\sim\{0,1\}^n}[\chi_a(x)\chi_b(y)\chi_c(x+y)] = \mathrm{E}_{x,y\sim\{0,1\}^n}[\chi_{a+c}(x)\chi_{b+c}(y)] = \mathrm{E}_{x\sim\{0,1\}^n}[\chi_{a+c}(x)]\,\mathrm{E}_{y\sim\{0,1\}^n}[\chi_{b+c}(y)].$$

Now notice that this expression equals zero unless $a = c$ and $b = c$, and so

$$\mathrm{E}[F(x)F(y)F(x+y)] = \sum_{\substack{a,b,c\in\{0,1\}^n:\\ a=c \text{ and } b=c}} \hat{F}_a\hat{F}_b\hat{F}_c = \sum_{a\in\{0,1\}^n} \hat{F}_a^3.$$

Therefore, from (2) we must have $\sum \hat{F}_a^3 = 1$. But by Parseval's identity $\sum \hat{F}_a^2 = 1$; this is only possible if $\hat{F}_a = 1$ for exactly one value of a, which means that $F(x) = \chi_a(x)$.

One important advantage of this Fourier-analytic proof is that it is *robust*: Even if the constraint $f(x)+f(y) = f(x+y)$ fails to hold sometimes, we cannot say anymore that f is linear, but we can still say that it is "close" to linear.

Theorem 2. *Suppose that $f(x) + f(y) = f(x + y)$ with probability $1 - \delta$ when x and y are chosen independently at random from $\{0,1\}^n$. Then $\Pr[f(x) = \langle a, x\rangle \bmod 2] \geq 1 - \delta$ for some $a \in \{0,1\}^n$.*

Proof. Writing $F(x) = (-1)^{f(x)}$, we have

$$\mathrm{E}_{x,y\sim\{0,1\}^n}[F(x)F(y)F(x+y)] = \Pr[F(x)F(y)F(x+y) = 1] - \Pr[F(x)F(y)F(x+y) = -1] = 1 - 2\delta$$

and so

$$\sum_{a\in\{0,1\}^n} \hat{F}_a^3 = 1 - 2\delta.$$

Since $\sum \hat{F}_a^3 \leq (\max \hat{F}_a)\sum \hat{F}_a^2 = \max \hat{F}_a$, there must exist $a \in \mathbb{F}^n$ such that $\hat{F}_a \geq 1 - 2\delta$. But $\hat{F}_a = \mathrm{E}[F(x)\chi_a(x)] = \Pr[F(x) = \chi_a(x)] - \Pr[F(x) \neq \chi_a(x)]$, so $\Pr[F(x) = \chi_a(x)] \geq 1 - \delta$. □

This theorem can be interpreted as a statement about the following scenario. We are given an unknown function f and we want to test if f is a linear function. A natural idea is to choose two random inputs x and y and check if $f(x) + f(y) = f(x + y)$. If f is linear, the test will certainly pass. The theorem tells us that if f passes the test with high probability, then we can be sure that f is close to a linear function.

4 Local list-decoding of the Hadamard code

Recall that the Hadamard encoding of a message a in $\{0,1\}^n$ (here we change the usual convention and we use n to denote message length instead of block length) consists of the evaluations $\langle a, x\rangle \bmod 2$ over all x in $\{0,1\}^n$. If we think of Hadamard encodings as (truth tables of) boolean functions from $\{0,1\}^n$ to $\{1,-1\}$, then the codewords of the Hadamard code correspond to the linear functions, namely the character functions χ_a.

In this language, the problem of list-decoding for the Hadamard code at relative distance $1/2-\varepsilon/2$ can be stated like this. We are given a "corrupted codeword" f, which is nothing but some function $f\colon \{0,1\}^n \to \{1,-1\}$, and we want a list of all codewords χ_a such that f agrees with χ_a a $1/2-\varepsilon/2$ fraction of the time, or equivalently

$$\hat{f}_a = \mathrm{E}[f(x)\chi_a(x)] \geq \varepsilon.$$

From this Fourier-analytic point of view, the list size of the Hadamard code can be bounded immediately via Parseval's identity: Every codeword χ_a in the list must contribute $\hat{f}_a^2 \geq \varepsilon^2$ to the square sum of the Fourier coefficients, so there cannot be more than $1/\varepsilon^2$ codewords in the list.

But how do we find all these codewords? There are two algorithms for doing so. The original algorithm of Goldreich and Levin has certain advantages, but today we show a different one, due to Kushilevitz and Mansour, which is very natural from a Fourier-analytic perspective.

We will generalize our objective a little bit and seek to find all a such that $\hat{f}_a^2 \geq \varepsilon^2$, and maybe even allow for a few as that don't quite satisfy this condition. The idea is to try to locate these relevant as by a divide-and-conquer strategy. One nice way to visualize this strategy is as a search process along the following full binary tree of depth n. The root of this binary tree is labeled by the value $\sum_{a \in \{0,1\}^n} \hat{f}_a^2$. Its left and right children are labeled by the partial sums

$$\sum_{a\colon a_1=0} \hat{f}_a^2 \quad \text{and} \quad \sum_{a\colon a_1=1} \hat{f}_a^2.$$

In general, a node at level i can be indexed by a string $v \in \{0,1\}^i$ and is labeled by the value

$$\sum_{a\colon a_1=v_1,\ldots,a_i=v_i} \hat{f}_a^2$$

so that the leaf indexed by a is labeled by $\hat{f}_a^2$.

Say a node v is relevant if its label is at least ε^2. Although there are exponentially many nodes in the tree, there can be at most n/ε^2 relevant ones because the labels in each level sum to 1. If we could calculate the labels, it would be easy to identify all the relevant nodes via depth-first search starting at the root and pruning the search path at irrelevant nodes.

How do we calculate the values of the labels? Using the Fourier coefficient formula (1), we can obtain these values in time exponential in n. But if we are willing to settle for a probabilistic approximation, we can do much better. Let's start at the leaves. From the formula (1) we get

$$\hat{f}_a^2 = \mathrm{E}[f(x)\chi_a(x)]\,\mathrm{E}[f(y)\chi_a(y)] = \mathrm{E}[f(x)f(y)\chi_a(x+y)].$$

This suggests that to estimate $\hat{f}_a^2$, we ought to sample some number of random pairs (x, y) and output the average of the values $f(x)f(y)\chi_a(x+y)$.

Now let v be an arbitrary node in the tree and $FIX(v)$ be the set of those $a \in \{0,1\}^n$ with $a_1 = v_1, \ldots, a_i = v_i$. We want to estimate the value

$$\sum_{a \in FIX(v)} \hat{f}_a^2 = \mathrm{E}\, f(x)f(y) \sum_{a \in FIX(v)} \chi_a(x+y).$$

The set $FIX(v)$ could be exponentially large so we have to be a bit careful here. Recall that $\chi_a(z) = (-1)^{\langle a,z\rangle}$ so:

$$\sum_{a \in FIX(v)} \chi_a(z) = \sum_{a \in FIX(v)} (-1)^{\langle a,z\rangle}$$

If z is nonzero along any of the coordinates $i+1$ up to n, this sum vanishes; otherwise, it equals $2^{n-i}\chi_v(z)$. So the only (x, y) pairs that contribute to the sum are those in which x and y agree on the last $n-i$ coordinates, and we can rewrite the identity as

$$\sum_{a \in FIX(v)} \hat{f}_a^2 = \mathrm{E}_{x',y' \sim \{0,1\}^i, u \sim \{0,1\}^{n-i}}[f(x'u)f(y'u)\chi_v(x'+y')].$$

Here, the first i bits x' and y' of x and y are chosen independently at random, while the last $n-i$ bits are random but identical in x and y. (When $i=0$ the right side equals $\mathrm{E}[f(u)^2]=1$, which is a good sign.)

We now have all the ingredients for the Kushilevitz-Mansour algorithm. First, we have a probabilistic procedure $\hat{\mathbf{S}}\mathbf{amp}(f,v)$ which estimates the label of node v as follows: Sample $O(n/\varepsilon^4)$ random triples (x',y',u) and output the average of the values $f(x'u)f(y'u)\chi_v(x'+y')$. The correctness of this sampler follows from Chebyshev's inequality:

Lemma 3. *With probability* $1-20\varepsilon^2/n$, $\hat{\mathbf{S}}\mathbf{amp}(f,v)$ *outputs a value between* $\ell(v)/2$ *and* $3\ell(v)/2$, *where*

$$\ell(v)=\sum_{a\colon a_1=v_1,\ldots,a_i=v_i}\hat{f}_a^2.$$

Now here is the Kushilevitz-Mansour algorithm:

> **Algorithm KM:** On input a function $f\colon\{0,1\}^n\to\{0,1\}$ and $\varepsilon>0$,
> Apply the following recursive procedure $\mathbf{P}(v)$ starting with v equal to the empty string:
> If $\hat{\mathbf{S}}\mathbf{amp}(f,v)\geq\varepsilon^2/2$:
> If v has length n, output v.
> Otherwise, call $\mathbf{P}(v0)$ and $\mathbf{P}(v1)$.

Theorem 4. *With probability at least* $1/2$, *the outputs of* $\mathbf{KM}(f,\varepsilon)$ *include all* a *such that* $\hat{f}_a^2\geq\varepsilon^2$, *but it produces no more than* $O(n/\varepsilon^2)$ *outputs in total.*

Proof. Recall that there are at most n/ε^2 nodes in the tree whose label is at least ε. By a union bound, $\hat{\mathbf{S}}\mathbf{amp}$ will output at least $\varepsilon/2$ on all these nodes with probability at least $3/4$, and so all a such that $\hat{f}_a^2\geq\varepsilon$ will be included in the output of $\mathbf{KM}$.

Let B be the set of nodes whose label exceeds $\varepsilon^2/3$. There must be fewer than $3n/\varepsilon^2$ such nodes in the tree, so they have fewer than $3n/\varepsilon^2+1$ children outside B. By a union bound, with probability at least $3/4$, $\hat{\mathbf{S}}\mathbf{amp}$ will output a value smaller than than $\varepsilon^2/2$ on all these children, so $\mathbf{KM}(f,\varepsilon)$ will not make more than $3n/\varepsilon^2$ recursive calls to $\mathbf{P}$. □