Electronic Colloquium on Computational Complexity, Report No. 137 (2011)

Isomorphism Testing of Boolean Functions Computable by Constant Depth Circuits

V.Arvind and Yadu Vasudev

The Institute of Mathematical Sciences, Chennai
{arvind,yadu}@imsc.res.in

Abstract. Given two n-variable boolean functions f and g, we study the problem of computing an *ε-approximate isomorphism* between them. I.e. a permutation π of the n variables such that $f(x_1, x_2, \ldots, x_n)$ and $g(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)})$ differ on at most an ε fraction of all boolean inputs $\{0,1\}^n$. We give a randomized $2^{O(\sqrt{n}\,\mathrm{polylog}(n))}$ algorithm that computes a $\frac{1}{2^{\mathrm{polylog}(n)}}$-approximate isomorphism between two isomorphic boolean functions f and g that are given by depth d circuits of poly(n) size, where d is a constant independent of n. In contrast, the best known algorithm for computing an *exact isomorphism* between n-ary boolean functions has running time $2^{O(n)}$ [Luk99] even for functions computed by poly(n) size DNF formulas. Our algorithm is based on a recent result for hypergraph isomorphism with bounded edge size [BC08] and the classical Linial-Mansour-Nisan result on approximating small depth and size boolean circuits by small degree polynomials using Fourier analysis.

1 Introduction

Given two Boolean functions $f, g : \{0,1\}^n \to \{0,1\}$ the *Boolean function isomorphism* is the problem of checking if there is a permutation π of the variables such that the Boolean functions $f(x_1, x_2, \ldots, x_n)$ and $g(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)})$ are equivalent. The functions f and g could be given as input either by Boolean circuits that compute them or simply by black-box access to them. This problem is known to be coNP-hard even when f and g are given by DNF formulas (there is an easy reduction from $\overline{\mathsf{CNFSAT}}$). The problem is in Σ_2^p but not known to be in coNP. Furthermore, Agrawal and Thierauf [AT96] have shown that the problem is not complete for Σ_2^p unless the polynomial hierarchy collapses to Σ_3^p.

On the other hand, the best known algorithm for Boolean function isomorphism, which reduces the problem to Hypergraph Isomorphism, runs in time $2^{O(n)}$ where n is the number of variables in f and g. This algorithm works even when f and g are given by only black-box access: First, the truth-tables of the functions f and g can be computed in time $2^{O(n)}$. The truth tables for f and g can be seen as hypergraphs representing f and g. Hypergraph Isomorphism for n-vertex and m-edge hypergraphs has a $2^{O(n)}m^{O(1)}$ algorithm due to Luks[Luk99] which yields the claimed $2^{O(n)}$ time algorithm for testing if f and g are isomorphic. This is the current best known algorithm for general hypergraphs and hence the current best algorithm for Boolean function isomorphism as well. Indeed, a hypergraph on n vertices and m edges can be represented as a DNF formula on n variables with m terms. Thus, even when f and g are DNF formulas the best known isomorphism test takes $2^{O(n)}$ time. In contrast, Graph Isomorphism has a $2^{O(\sqrt{n \log n})}$ time algorithm due to Luks and Zemlyachenko (see [BL83]). More recently, Babai and Codenotti [BC08] have shown for hypergraphs of edge size bounded by k that isomorphism testing can be done in $2^{\widetilde{O}(k^2\sqrt{n})}$ time.

ISSN 1433-8092

Our results

Since the exact isomorphism problem for Boolean functions is as hard as Hypergraph Isomorphism, and it appears difficult to improve the $2^{O(n)}$ bound, we investigate the problem of computing *approximate* isomorphisms (which we define below). An interesting question is whether the *circuit complexity* of f and g can be exploited to give a faster *approximate* isomorphism test. Specifically, in this paper we study the approximation version of boolean function isomorphism for functions computed by *small size and small depth* circuits and give a faster algorithm for computing approximate isomorphisms.

We note that, in a different context, approximate Boolean function isomorphism has been studied in the framework of *property testing*, and nearly matching upper and lower bounds are known ([AB10],[CSM10],[BO10]). In property testing the objective is to test whether two given Boolean functions are close to being isomorphic or far apart. The goal is to design a property test with low *query* complexity. In contrast, our result is algorithmic and the goal is to efficiently compute a good approximate isomorphism.

We also note that approximate versions of Graph Isomorphism have been studied in the literature as graph edit distance, graph similarity and graph matching with respect to various distance measures (e.g. [Bu00]). There are various heuristic algorithms for the problem. These results do not appear related to the topic of our paper.

The rest of the paper is organized into four sections. Section 2 gives basic definitions. In Section 3 we explain our approximate isomorphism algorithm for constant-depth small size boolean circuits. Finally, in Section 4 we study a general problem: given two n-variable boolean functions f and g consider the optimization problem where the objective is to find a permutation π that maximizes $|\{x \in \{0,1\}^n \mid f(x) = g^\pi(x)\}|$. This problem is coNP-hard under Turing reductions. We give a simple $2^{O(n)}$ time deterministic approximation algorithm that, when given as input Boolean functions f and g such that f and g^π agree on a constant fraction of the inputs for some permutation π, outputs a permutation σ such that f and g^σ agree on an $O(\frac{1}{\sqrt{n}})$ fraction of the inputs.

2 Some Notation and Definitions

Let $\mathcal{AC}_{s,d,n}$ denote the class of n-ary boolean functions computed by circuits of depth d and size s, where the gates allowed are unbounded fan-in AND and OR gates, and negation gates. Suppose $f, g \in \mathcal{AC}_{s,d,n}$ are isomorphic boolean functions. As a consequence of the main result, in Section 3, we show that there is a randomized algorithm that computes a $\frac{1}{2^{\log^{O(1)} n}}$-approximate isomorphism between f and g in time $2^{\log(ns)^{O(d)}\sqrt{n}}$. This is substantially faster than the $2^{O(n)}$ time algorithm for computing an exact isomorphism. We show how to achieve this running time by combining some Fourier analysis of boolean functions with the Babai-Codenotti algorithm mentioned above.

We will make precise the definition of closeness between two boolean functions and the notion of approximate isomorphism. Let $\mathcal{B}_n$ denote the set of all n-ary boolean functions $f : \{0,1\}^n \to \{0,1\}$. We remark that in Section 3, where we use Fourier analytic methods, it is convenient for us to consider boolean functions with domain $\{-1,1\}^n$ and range $\{-1,1\}$. It is an easy transformation to switch between these two domains and ranges for Boolean functions.

Let $g : \{0,1\}^n \to \{0,1\}$ be a boolean function and let $\pi : [n] \to [n]$ be any permutation. Then $g^\pi : \{0,1\}^n \to \{0,1\}$ is defined as follows $g^\pi(x_1, x_2, \ldots, x_n) = g(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)})$.

This clearly defines a faithful group action of the permutation group S_n on the set $\mathcal{B}_n$. I.e. $g^{(\pi\psi)} = (g^\pi)^\psi$ for all $g \in \mathcal{B}_n$ and $\pi, \psi \in S_n$, and $g^\pi = g^\psi$ for all $g \in \mathcal{B}_n$ if and only if $\pi = \psi$.

Definition 1. *Two boolean functions $f, g \in \mathcal{B}_n$ are said to be isomorphic (denoted by $f \cong g$) if there exists a permutation $\pi : [n] \to [n]$ such that $\forall x \in \{0,1\}^n, f(x) = g^\pi(x)$.*

Since in this paper, we are interested in the case when two boolean functions are *close* to being isomorphic, we first define the notion of closeness of boolean functions.

Definition 2. *Two boolean functions f, g are $\frac{1}{2^\ell}$-close if $\Pr_{x\in\{0,1\}^n}\left[f(x) \neq g(x)\right] \leq \frac{1}{2^\ell}$.*

With this definition in hand, we can define when two boolean functions are approximately isomorphic.

Definition 3. *Two boolean functions f, g are $\frac{1}{2^\ell}$-approximate isomorphic if there exists a permutation $\pi : [n] \to [n]$ such that the functions f and g^π are $\frac{1}{2^\ell}$-close.*

3 Main Result

In this section we focus on the problem of computing an approximate isomorphism for two boolean functions $f, g \in \mathcal{AC}_{s,d,n}$. We first recall some Fourier analysis of Boolean functions which is an important ingredient in our algorithm. For Fourier analytic purposes, it is convenient for us to consider boolean functions with domain $\{-1,1\}^n$ and range $\{-1,1\}$. The range $\{-1,1\}$ makes it convenient to define the Fourier basis.

The set $\mathcal{F} = \{f : \{-1,1\}^n \to \mathbb{R}\}$ of real-valued functions forms a 2^n-dimensional vector space over $\mathbb{R}$, where vector addition is defined as $(f+g)(x) = f(x) + g(x)$. The vector space $\mathcal{F}$ forms an inner product space with inner product defined as:

$$\langle f, g\rangle = \mathbb{E}_{x\in\{-1,1\}^n}\left[f(x)g(x)\right] = \frac{1}{2^n}\sum_{x\in\{-1,1\}^n} f(x)g(x).$$

The ℓ_2-norm of a function $f \in \mathcal{F}$ is $\|f\|_2 = \sqrt{\langle f, f\rangle}$. Clearly every Boolean function $f : \{-1,1\}^n \to \{-1,1\}$ has unit norm under this inner product. The *Fourier basis*, $\{\chi_S \big| S \subseteq [n]\}$ is defined as $\chi_S(x) = \prod_{i\in S} x_i$. It is easy to observe that $\mathbb{E}_x[\chi_S(x)] = 0$ for nonempty sets S and $\mathbb{E}_x[\chi_\emptyset(x)] = 1$. Furthermore, $\langle \chi_S, \chi_T\rangle = \mathbb{E}_x[\chi_{S\triangle T}(x)]$. It follows that the Fourier basis is an orthonormal basis with respect to the inner product. Thus, any $f \in \mathcal{F}$ can be written as $f = \sum \widehat{f}_S \chi_S$. This is the *Fourier representation* of f, and the numbers $\widehat{f}_S = \langle f, \chi_S\rangle$ are the *Fourier coefficients* of f. The orthonormality of the Fourier basis yields *Parseval's identity:*

$$\langle f, f\rangle = \sum_{S\subseteq[n]} \widehat{f}(S)^2.$$

In particular, since any Boolean function $f : \{-1,1\}^n \to \{-1.1\}$ has unit norm, we note that $\sum_{S\subseteq[n]} \widehat{f}(S)^2 = 1$.

In the next two propositions we relate the isomorphism of Boolean functions f and g to their Fourier coefficients.

Proposition 1. *Let $\pi : [n] \to [n]$ be any permutation, g any boolean function and $S \subseteq [n]$, then $\widehat{g^\pi}(S) = \widehat{g}(S^\pi)$ where $S^\pi = \{i | \pi(i) \in S\}$.*

Proof. $\widehat{g^\pi}(S) = \frac{1}{2^n}\sum_{x\in\{-1,1\}^n} g^\pi(x)\chi_S(x) = \frac{1}{2^n}\sum_{x\in\{-1,1\}^n} g(x)\chi_{S^\pi}(x) = \widehat{g}(S^\pi)$. ∎

Proposition 2. *Two Boolean functions $f, g : \{-1.1\}^n \to \{-1,1\}$ are isomorphic via permutation π if and only if $\widehat{f}(S) = \widehat{g}(S^\pi)$ for each subset S.*

Proof. Suppose $\pi : [n] \to [n]$ is an isomorphism. I.e. $f(x) = g^\pi(x)$ for all $x \in \{-1,1\}^n$. Consider any subset $S \subseteq [n]$

$$\widehat{f}(S) = \frac{1}{2^n}\sum_{x\in\{-1,1\}^n} f(x)\chi_S(x) = \frac{1}{2^n}\sum_{x\in\{-1,1\}^n} g^\pi(x)\chi_S(x) = \widehat{g^\pi}(S) = \widehat{g}(S^\pi).$$

Conversely, if $\widehat{f}(S) = \widehat{g}(S^\pi)$ for each subset S, by the previous proposition we have $\widehat{f}(S) = \widehat{g^\pi}(S)$ which implies that $f = g^\pi$. ∎

3.1 Approximate Isomorphism for $\mathcal{AC}_{s,d,n}$

Now we turn to isomorphism for Boolean functions from the class $\mathcal{AC}_{s,d,n}$. We first outline our approach to Boolean function isomorphism via Fourier coefficients.

A crucial theorem that we will use is the celebrated result of Linial-Mansour-Nisan [LMN93] which gives the distribution of Fourier coefficients for Boolean functions computed by small depth circuits.

Theorem 1 ([LMN93]). *Let $f : \{-1,1\}^n \to \{-1,1\}$ be computed by an $\mathcal{AC}_{s,d,n}$ circuit. Then for all $t > 0$,*

$$\sum_{\substack{S\subseteq[n]\\|S|>t}} \widehat{f}(S)^2 \le 2s2^{-t^{1/d}/20}.$$

Consequently, for $\widetilde{f} = \sum_{\substack{S\subseteq[n]\\|S|\le t}} \widehat{f}(S)\chi_S$ we have $\|f - \widetilde{f}\|_2^2 \le 2s2^{-t^{1/d}/20}$.

Notice that each $\chi_S = \prod_{i\in S} x_i$ is a monomial, and hence $\widetilde{f}$ is a degree-t polynomial that approximating f. Given boolean functions $f, g \in \mathcal{AC}_{s,d,n}$ as an instance of boolean function isomorphism, our aim is to work with the polynomials $\widetilde{f}$ and $\widetilde{g}$:

$$\widetilde{f} = \sum_{\substack{S\subseteq[n]\\|S|\le t}} \widehat{f}(S)\chi_S \quad \text{and} \quad \widetilde{g} = \sum_{\substack{S\subseteq[n]\\|S|\le t}} \widehat{g}(S)\chi_S. \tag{1}$$

This is because $\widetilde{f}$ and $\widetilde{g}$ are of degree t and have only n^t terms. I.e. we will check if there is an approximate isomorphism between $\widetilde{f}$ and $\widetilde{g}$. Notice that the polynomials $\widetilde{f}, \widetilde{g} : \{-1,1\}^n \to \mathbb{R}$ are not boolean-valued functions. We need an appropriate notion of isomorphism here.

Definition 4. *Let $f', g' : \{-1,1\}^n \to \mathbb{R}$ be two functions from $\mathcal{F}$. We say that f' and g' are $\frac{1}{2^\ell}$-approximate isomorphic witnessed by a permutation $\pi : [n] \to [n]$ if $\|f' - g'^\pi\|_2^2 \le \frac{1}{2^\ell}$.*

We now explain the connection between $\frac{1}{2^\ell}$-approximate isomorphism of two functions and their Fourier coefficients.

Proposition 3. *If $f, g : \{-1,1\}^n \to \{-1,1\}$ are $\frac{1}{2^\ell}$-close, then $\|f-g\|_2^2 \le 4\frac{1}{2^\ell}$*

Proof. Follows from $\|f-g\|^2 = \mathbb{E}[(f-g)^2] = 4\Pr[f \neq g]$. ∎

Lemma 1. *Let f and g be two boolean functions that are $\frac{1}{2^\ell}$-approximate isomorphic via permutation $\pi : [n] \to [n]$. Then $\forall S \subseteq [n]$: $\left|\widehat{f}(S) - \widehat{g^\pi}(S)\right| \le \frac{2}{2^{\ell/2}}$.*

Proof. Notice that $\sum_{S\subseteq[n]}(\widehat{f}(S) - \widehat{g^\pi}(S))^2 = \|f - g^\pi\|_2^2$. Suppose f, g are $\frac{1}{2^\ell}$-approximate isomorphic via permutation π. By Proposition 3 we know that $\sum_{S\subseteq[n]}(\widehat{f}(S) - \widehat{g^\pi}(S))^2 = \|f - g^\pi\|_2^2 \le \frac{4}{2^\ell}$. Hence for each subset $S \subseteq [n]$ we have $\left(\widehat{f}(S) - \widehat{g^\pi}(S)\right)^2 \le \frac{4}{2^\ell}$. ∎

Suppose f and g be two boolean functions that are $\frac{1}{2^\ell}$-approximate isomorphic via permutation $\pi : [n] \to [n]$. By the above proposition $|\widehat{f}(S) - \widehat{g^\pi}(S)|$ is bounded by $\frac{2}{2^{\ell/2}}$. Furthermore, since both $\widehat{f}(S)$ and $\widehat{g^\pi}(S)$ are Fourier coefficients of Boolean functions f and g^π, we have $0 \le |\widehat{f}(S)| \le 1$ and $0 \le |\widehat{g^\pi}(S)| \le 1$. Hence, the bound implies that the $\lfloor \ell/2 \rfloor - 1$ most significant positions in the binary representation of $\widehat{f}(S)$ and $\widehat{g^\pi}(S)$ are identical.

For each subset S, let $\widehat{f}_\ell(S)$ denote the truncation of $\widehat{f}(S)$ to the first $\lfloor \ell/2 \rfloor - 1$ bits. Thus, $|\widehat{f}_\ell(S) - \widehat{f}(S)| \le \frac{1}{2^{\lfloor \ell/2 \rfloor - 1}}$ for each S. Similarly, $\widehat{g}_\ell(S)$ denotes the truncation of $\widehat{g}(S)$ to the first $\lfloor \ell/2 \rfloor - 1$ bits. We define the following two functions f_ℓ and g_ℓ from $\{-1,1\}^n \to \mathbb{R}$:

$$f_\ell = \sum_{S\subseteq[n]} \widehat{f}_\ell(S)\chi_S \quad \text{and} \quad g_\ell = \sum_{S\subseteq[n]} \widehat{g}_\ell(S)\chi_S. \tag{2}$$

The following lemma summarizes the above discussion. It gives us a way to go from approximate isomorphism to exact isomorphism.

Lemma 2. *If f and g be two boolean functions that are $\frac{1}{2^\ell}$-approximate isomorphic via permutation $\pi : [n] \to [n]$ then $f_\ell = g_\ell^\pi$, i.e. the functions f_ℓ and g_ℓ are (exactly) isomorphic via the permutation π.*

Lemma 1 and Proposition 3 yield the following observation.

Lemma 3. *Suppose f, g are two boolean functions that are $\frac{1}{2^\ell}$-approximate isomorphic via permutation π. Then $\|\widetilde{f} - \widetilde{g^\pi}\|_2^2 \le \frac{4}{2^\ell}$. I.e. $\widetilde{f}$ and $\widetilde{g}$ are $\frac{4}{2^\ell}$-approximate isomorphic via the same permutation π. Furthermore, $|\widehat{f}(S) - \widehat{g^\pi}(S)| \le \frac{2}{2^{\ell/2}}$ for all $S : |S| \le t$.*

Proof. By Lemma 1 and Proposition 3 we have $\sum_{S\subseteq[n]}\left(\widehat{f}(S) - \widehat{g^\pi}(S)\right)^2 \le \frac{4}{2^\ell}$, which implies $\|\widetilde{f} - \widetilde{g^\pi}\|_2^2 = \sum_{|S|\le t}\left(\widehat{f}(S) - \widehat{g^\pi}(S)\right)^2 \le \frac{4}{2^\ell}$. It follows that $|\widehat{f}(S) - \widehat{g^\pi}(S)| \le \frac{2}{2^{\ell/2}}$ for all $S : |S| \le t$. ∎

Now, if $|\widehat{f}(S) - \widehat{g^\pi}(S)| \le \frac{2}{2^{\ell/2}}$ for all $S : |S| \le t$, it implies that $\widehat{f}_\ell(S) = \widehat{g_\ell^\pi}(S)$ for all $S : |S| \le t$, where $\widehat{f}_\ell(S)$ and $\widehat{g}_\ell(S)$ are defined in Equation 2. Indeed, if we truncate the coefficients of the polynomials $\widetilde{f}$ and $\widetilde{g}$ also to the first $\lfloor \ell/2 \rfloor - 1$ bits we obtain the polynomials

$$\widetilde{f}_\ell = \sum_{S:|S|\le t} \widehat{f}_\ell(S)\chi_S \quad \text{and} \quad \widetilde{g}_\ell = \sum_{S:|S|\le t} \widehat{g}_\ell(S)\chi_S. \tag{3}$$

It clearly follows that π is an exact isomorphism between $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$. We summarize the above discussion in the following lemma which is crucial for our algorithm.

Lemma 4. *Suppose f, g are two boolean functions that are $\frac{1}{2^\ell}$-approximate isomorphic via permutation π. Then:*

1. *$\|\widetilde{f} - \widetilde{g^\pi}\|_2^2 \leq \frac{4}{2^\ell}$. I.e. $\widetilde{f}$ and $\widetilde{g}$ are $\frac{4}{2^\ell}$-approximate isomorphic via the same permutation π, and hence $|\widehat{f}(S) - \widehat{g^\pi}(S)| \leq \frac{2}{2^{\ell/2}}$ for all $S : |S| \leq t$.*
2. *Consequently, π is an exact isomorphism between $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$.*

We can represent $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ as weighted hypergraphs with hyperedges $S : |S| \leq t$ of weight $\widehat{f}_\ell(S)$ and $\widehat{g}_\ell(S)$ respectively. We can then apply the Babai-Codenotti hypergraph isomorphism algorithm [BC08] to compute an exact isomorphism π between $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$. Now, if π is an exact isomorphism π between $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ what can we infer about π as an approximate isomorphism between f and g? The following lemma quantifies it.

Lemma 5. *Suppose f and g are boolean functions in $\mathcal{AC}_{s,d,n}$ such that π is an exact isomorphism between $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ (where $\widetilde{f}$ and $\widetilde{g}$ are given by Equation 1). Then π is an $(\delta + \varepsilon)^2$-approximate isomorphism between f and g, where $\delta = 2s2^{-t^{1/d}/20}$ and $\varepsilon = \frac{2n^{t/2}}{2^{(\ell-1)/2}}$.*

Proof. Since π is an exact isomorphism between $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ we have $\widetilde{f}_\ell = \widetilde{g^\pi_\ell}$. Now consider $\|f - g^\pi\|_2^2$. By triangle inequality

$$\begin{aligned}\|f - g^\pi\|_2 &\leq \|f - \widetilde{f}\| + \|\widetilde{f} - \widetilde{f}_\ell\| + \|\widetilde{f}_\ell - \widetilde{g^\pi}_\ell\| + \|\widetilde{g^\pi} - \widetilde{g^\pi}_\ell\| + \|g^\pi - \widetilde{g^\pi}\| \\ &= \|f - \widetilde{f}\| + \|\widetilde{f} - \widetilde{f}_\ell\| + \|\widetilde{g^\pi} - \widetilde{g^\pi}_\ell\| + \|g^\pi - \widetilde{g^\pi}\|.\end{aligned}$$

By Theorem 1, both $\|f - \widetilde{f}\|$ and $\|g^\pi - \widetilde{g^\pi}\|$ are bounded by δ. Furthermore,

$$\|\widetilde{f} - \widetilde{f}_\ell\|_2^2 = \sum_{S:|S|\leq t} (\widehat{f}(S) - \widehat{f}_\ell(S))^2 \leq \sum_{S:|S|\leq t} \frac{4}{2^{\ell-1}} \leq \frac{4n^t}{2^{\ell-1}}.$$

Hence, $\|\widetilde{f} - \widetilde{f}_\ell\| \leq \frac{2n^{t/2}}{2^{(\ell-1)/2}}$ and, likewise, $\|\widetilde{g^\pi} - \widetilde{g^\pi}_\ell\| \leq \frac{2n^{t/2}}{2^{(\ell-1)/2}}$. Putting it together with Proposition 3 we get

$$4\Pr[f \neq g^\pi] = \|f - g^\pi\|_2^2 \leq \left(2\delta + \frac{4n^{t/2}}{2^{(\ell-1)/2}}\right)^2.$$

It follows that f and g are $(\delta + \varepsilon)^2$-approximate isomorphic via the permutation π. ∎

Suppose f and g are in $\mathcal{AC}_{s,d,n}$ and are given by circuits C_f and C_g. Our goal now is to design an efficient algorithm that will compute the polynomials $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$, where ℓ will be appropriately chosen in the analysis. In order to compute $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ we need to estimate to $\lfloor \ell/2 \rfloor - 1$ bits of precision the Fourier coefficients $\widehat{f}(S)$ and $\widehat{g}(S)$ for each subset $S : |S| \leq t$. Now, by definition, $\widehat{f}(S)$ is the average of $f(x)\chi_S(x)$ where x is uniformly distributed in $\{-1, 1\}^n$. Hence, following a standard Monte-Carlo sampling procedure, we can estimate $\widehat{f}(S)$ quite accurately from a random sample of inputs from $\{-1, 1\}^n$ and hence with high probability we can exactly compute $f_\ell(S)$ for all $S : |S| \leq t$. We formally explain this in the next lemma.

Lemma 6. *Given $f : \{-1,1\}^n \to \{-1,1\}$ computed by an $\mathcal{AC}_{s,d,n}$ circuit, there is a randomized algorithm $\mathcal{C}$ with running time* $\mathrm{poly}(s, n^t, 2^\ell)$ *that outputs the set $\{f_\ell(S) \mid |S| \leq t\}$ with probability $1 - \frac{1}{2^{\Omega(n)}}$.*

Proof. We use the same technique as [LMN93]to estimate the required Fourier coefficients.

1. For each subset $S \subset [n]$ such that $|S| \leq t$ do the following two steps:
2. Pick $x_i \in_r \{-1,1\}^n$ and compute the value $f(x_i)\chi_S(x_i)$ for $i \in [m]$.
3. Estimate the Fourier coefficient as $\alpha_f(S) = \frac{1}{m}\sum_{i=1}^{m} f(x_i)\chi_S(x_i)$.

Applying Chernoff bounds, for each subset S we have

$$\Pr\left[\left|\widehat{f}(S) - \alpha_f(S)\right| \geq \lambda\right] \leq 2e^{-\lambda^2 m/2}.$$

In our case we set $\lambda = \frac{1}{2^{\lfloor \ell/2 \rfloor - 1}}$. In order to estimate $\widehat{f}(S)$ for each $S : |S| \leq t$ within the prescribed accuracy and with small error probability, we set $m = tn \log n 2^\ell$. The entire procedure runs in $\mathrm{poly}(s, n^t, 2^\ell)$ time. Furthermore, by a simple union bound it follows that with probability $1 - 2^{-\Omega(n)}$ we have $\alpha_f(S) = f_\ell(S)$ for each $S : |S| \leq t$ with probability. Thus, the randomized algorithm computes the polynomial $\widetilde{f_\ell}$ with high probability. ∎

3.2 Exact isomorphism test for low degree polynomials

We now focus on the problem of checking if the polynomials $\widetilde{f_\ell} = \sum_{S:|S|\leq t} \widehat{f_\ell}(S)\chi_S$ and $\widetilde{g_\ell} = \sum_{S:|S|\leq t} \widehat{g_\ell}(S)\chi_S$ are isomorphic, and if so to compute an exact isomorphism π. To this end, we shall encode f_ℓ and g_ℓ as *weighted* hypergraphs G_f and G_g, respectively.

The vertex sets for both graphs is $[n]$. Let E denote the set of all subsets $S \subset [n]$ of size at most t. The weight functions for the edges are w_f and w_g for G_f and G_g defined as follows

$$w_f(S) = \begin{cases} \alpha_f(S) & \forall S \subseteq [n], |S| \leq t \\ 0 & otherwise, \end{cases}$$

$$w_g(S) = \begin{cases} \alpha_g(S) & \forall S \subseteq [n], |S| \leq t \\ 0 & otherwise. \end{cases}$$

The isomorphism problem for the polynomials the f_ℓ and g_ℓ is now the edge-weighted hypergraph isomorphism problem, where G_f and G_g are the two edge-weighted graphs, and the problem is to compute a permutation on $[n]$ that maps edges to edges (preserving edge weights) and non-edges to non-edges. Our aim is to apply the Babai-Codenotti isomorphism algorithm for hypergraphs with hyperedge size bounded by k [BC08]. Their algorithm has running time $2^{\widetilde{O}(k^2\sqrt{n})}$. We need to adapt their algorithm to work for hypergraphs with edge weights. Since the edge weights for the graphs G_f and G_g are essentially $\lfloor \ell/2 \rfloor - 1$ bit strings, we can encode the weights into the hyperedges by introducing new vertices.

More precisely, we create new graphs G'_f and G'_g corresponding to f and g, where the number of vertices is now $n + O(\ell)$. Let the set of new vertices be $\{v_1, \ldots, v_r\}$, where $r = O(\ell)$. Let $S \subset [n]$ be a hyperedge in the original graph G_f. A subset $T \subset \{v_1, \ldots, v_r\}$ encodes an r-bit string via a natural bijection (the j^{th} bit is 1 if and only if $v_j \in T$). Let $T(S) \subset \{v_1, \ldots, v_r\}$ denote the encoding of the number $\widehat{f_\ell}(S)$ for each hyperedge $S \in E$. Similarly, let $T'(S) \subset \{v_1, \ldots, v_r\}$ denote the encoding of the number $\widehat{g_\ell}(S)$. The hyperedge $S \cup T(S)$ encodes S

along with its weight $\widehat{f}_\ell(S)$ for each S in G'_f. Similarly, $S \cup T'(S)$ encodes S along with its weight $\widehat{g}_\ell(S)$ for each S in G'_g. As candidate isomorphisms between G'_f and G'_g we wish to consider only permutations on $[n] \cup \{v_1, \ldots, v_r\}$ that fix each $v_i, 1 \leq i \leq r$. This can be easily ensured by standard tricks like coloring each v_i with a unique color, where the different colors can be implemented by putting directed paths of different lengths, suitably long so that the vertices v_i are forced to be fixed by any isomorphism between G'_f and G'_g.

This will ensure that G'_f and G'_g are isomorphic iff there is a weight preserving isomorphism between G_f and G_g. Now we invoke the algorithm of [BC08] on G'_f and G'_g which will yield an isomorphism ψ between f' and g'. In summary, the algorithm for isomorphism testing f_ℓ and g_ℓ carries out the following steps.

Polynomial Isomorphism Algorithm

1. Construct the hypergraphs G'_f and G'_g as defined above.
2. Run the algorithm of Babai and Codenotti[BC08] on the hypergraphs G'_f and G'_g and output isomorphism ψ or report they are non-isomorphic.

Lemma 7. *The isomorphism of polynomials $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ (defined by Equation 2) can be tested in time $2^{O(\sqrt{n})(\ell+t)^2 \log^{O(1)} n}$, and if the polynomials are isomorphic an exact isomorphism can be computed in the same running time bound.*

3.3 The approximate isomorphism algorithm

We now give an outline of the entire algorithm.

Input: $f, g \in \mathcal{AC}_{s,d,n}$ given by circuits of size s along with parameters t and ℓ.

Step 1. Compute the polynomials $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ using randomized algorithm of Lemma 6.

Step 2. Check if $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ are isomorphic using the polynomial isomorphism algorithm described above. If they are not isomorphic *reject* else **output** the computed exact isomorphism π.

Suppose π is an exact isomorphism between $\widetilde{f}_\ell$ and $\widetilde{g}_\ell$ computed by the above algorithm. By Lemma 5 π is a $(\delta+\varepsilon)^2$-approximate isomorphism between f and g, where $\delta = 2s2^{-t^{1/d}/20}$ and $\varepsilon = \frac{2n^{t/2}}{2^{(\ell-1)/2}}$. From Lemmas 6 and 7 it follows that the overall running time of the algorithm is $\text{poly}(s, n^t, 2^\ell) + 2^{O(\sqrt{n})(\ell+t)^2 \log^{O(1)} n}$ and the error probability, as argued in Lemma 6, is at most $2^{-\Omega(n)}$.

We now set parameters to obtain the main result of the paper. Suppose f and g are $\frac{1}{2^\ell}$-approximate isomorphic, where $\ell = (\log n + \log s)^{kd}$ for a suitably large constant $k > 1$. Then we choose $t = (\log n + \log s)^{O(d)}$ so that $(\delta + \varepsilon)^2$ is bounded by $2^{-(\log n)^{O(1)}}$.

Theorem 2. *Given two boolean functions $f, g \in \mathcal{AC}_{s,d,n}$ which are $\frac{1}{2^{(\log n)^{O(d)}}}$-isomorphic, there is a randomized algorithm running in time $2^{O(\log^{O(d)}(n)\sqrt{n})}$ to compute a permutation π such that f, g are $\frac{1}{2^{(\log n)^{O(1)}}}$-approximate isomorphic with respect to π.*

4 A general approximate isomorphism algorithm

Given two n-variable boolean functions f and g (either by Boolean circuits computing them or just by black-box access) consider the optimization problem of finding a permutation π that minimizes $|\{x \in \{0,1\}^n \mid f(x) \neq g^\pi(x)\}|$. This problem is coNP-hard under Turing reductions. We reduce the coNP-complete problem TAUTOLOGY (checking if a propositional formula is a tautology) to the problem MinBooleanIso of computing a permutation π that minimizes $|\{x \in \{0,1\}^n \mid f(x) \neq g^\pi(x)\}|$.

Lemma 8. TAUTOLOGY *is polynomial-time Turing reducible to* MinBooleanIso.

Proof. Given $f : \{0,1\}^n \to \{0,1\}$ as an n-variable propositional formula, we define functions $g_i : \{0,1\}^n \to \{0,1\}$ for $i \in [n]$ such that $g_i(1^i0^{n-i}) = 0$ and $g_i(x) = 1$ for all $x \neq 1^i0^{n-i}$. Notice that if f is a tautology then for each i $|\{x \in \{0,1\}^n \mid f(x) \neq g_i^\pi(x)\}| = 1$ for all permutations π.

We now describe a polynomial-time algorithm for TAUTOLOGY with MinBooleanIso as oracle. For each g_i, we compute (with a query to the function oracle MinBooleanIso) a permutation π_i that minimizes $|\{x \in \{0,1\}^n \mid f(x) \neq g_i^\pi(x)\}|$. If $f(\pi_i^{-1}(1^i0^{n-i})) = 1$ for each i, the algorithm describing the Turing reduction "accepts" f as a tautology and otherwise it "rejects" f.

We now show the correctness of the reduction. If f is a tautology, then clearly for each π_i we have $f(\pi_i^{-1}(1^i0^{n-i})) = 1$. Conversely, suppose f is not a tautology. Then $f^{-1}(0) = \{x \in \{0,1\}^n \mid f(x) = 0\}$ is nonempty. Let $|f^{-1}(0)| = N$. Then for any permutation π the cardinality $|\{x \in \{0,1\}^n \mid f(x) \neq g_i^\pi(x)\}|$ is either $N+1$ or $N-1$ for each i. Furthermore, suppose $x \in f^{-1}(0)$ has Hamming weight i. Then for any permutation π_i that maps x to 1^i0^{n-i} we have $|\{x \in \{0,1\}^n \mid f(x) \neq g_i^{\pi_i}(x)\}| = N-1$. Hence, $f(\pi_i^{-1}(1^i0^{n-i})) = f(x) = 0$. ∎

A brute-force search that runs in $n!$ time by cycling through all permutations yields a trivial algorithm for the optimization problem MinBooleanIso.

The corresponding *maximization problem* is: Find π that maximizes $|\{x \in \{0,1\}^n \mid f(x) = g^\pi(x)\}|$. Of course computing an optimal solution to this problem is polynomial-time equivalent to MinBooleanIso. In remainder of this section we design a simple approximate isomorphism algorithm for the *maximization* problem. Our simple algorithm is based on the method of conditional probabilities. We first examine how good a random permutation is as an approximate isomorphism. Then we describe a deterministic algorithm for computing a permutation with the same solution quality.

For boolean functions f and g, consider the random variable $|\{x|f(x) = g^\pi(x)\}|$ when the permutation π is picked uniformly at random from S_n.

Let $s_i(f)$ denote the cardinality $|\{x \in \{0,1\}^n| \text{ wt}(x) = i, f(x) = 1\}|$ where $\text{wt}(x)$ is the hamming weight of the boolean string x. Clearly, $s_i(f) \leq \binom{n}{i}$. For each $u \in \{0,1\}^n$ define the 0-1 random variable X_u which takes value 1 if and only if $f(u) = g^\pi(u)$ for $\pi \in S_n$ picked uniformly at random. If $\text{wt}(u) = i$, then

$$\Pr_\pi[X_u = 1] = \frac{s_i(g)}{\binom{n}{i}} f(u) + \frac{\binom{n}{i} - s_i(g)}{\binom{n}{i}} (1 - f(u)).$$

The sum $X = \sum_{u \in \{0,1\}^n} X_u$ is the random variable $|\{x | f(x) = g^\pi(x)\}|$ for a random permutation $\pi \in S_n$. We have

$$\mathbb{E}_\pi[X] = \sum_{i=0}^{n} \sum_{u \,:\, \mathrm{wt}(u)=i} \frac{s_i(g)}{\binom{n}{i}} f(u) + \sum_{i=0}^{n} \sum_{u \,:\, \mathrm{wt}(u)=i} \frac{\binom{n}{i} - s_i(g)}{\binom{n}{i}} (1 - f(u)) \tag{4}$$

$$= \sum_{i=0}^{n} \frac{s_i(g)s_i(f)}{\binom{n}{i}} + \sum_{i=0}^{n} \frac{(\binom{n}{i} - s_i(g))(\binom{n}{i} - s_i(f))}{\binom{n}{i}} \tag{5}$$

$$\geq \max_i \left(\frac{s_i(f)s_i(g)}{\binom{n}{i}}, \frac{(\binom{n}{i} - s_i(g))(\binom{n}{i} - s_i(f))}{\binom{n}{i}} \right). \tag{6}$$

Theorem 3. *There is a deterministic $2^{O(n)}$ time algorithm that takes as input Boolean functions $f, g : \{0,1\}^n \to \{0,1\}$ as input (either by Boolean circuits or by black-box access) and outputs a permutation σ with the following property: If f and g^π are δ-close for some permutation π and constant δ, then*

$$|\{x | f(x) = g^\sigma(x)\}| \geq \Omega(1/\sqrt{n})2^n.$$

Proof. First we show how to compute a permutation σ such that $|\{x | f(x) = g^\sigma(x)\}| \geq \mathbb{E}_\pi[X]$ (see Equation 4 and discussion preceding it for the definition of random variable X). Firsrtly, notice that given a partial permutation σ_i defined on $\{1, 2, \ldots, i\}$, we can define random variables $X_{\sigma_i,u}$ for each $u \in \{0,1\}^n$ and $X_{\sigma_i} = \sum_u X_{\sigma_i,u}$, defined by a uniformly picked random permutation π in S_n that extends σ_i. Similar to Equation 4, we can write an expression for $\mathbb{E}[X_{\sigma_i}]$ and compute it exactly in time $2^{O(n)}$ for a given σ_i. Now, in time $2^{O(n)}$ we can compute an extension σ_{i+1} of σ_i that tries all $(n-i)$ extensions σ_{i+1}, computes $\mathbb{E}[X_{\sigma_{i+1}}]$, and chooses the image of $i+1$ that maximizes $\mathbb{E}[X_{\sigma_{i+1}}]$. In particular, this will satisfy

$$\mathbb{E}[X_{\sigma_{i+1}}] \geq \mathbb{E}[X_{\sigma_i}].$$

Continuing this process until $i = n$ yields $\sigma_n = \sigma$ such that $|\{x \mid f(x) = g^\sigma(x)\}| \geq \mathbb{E}_\pi[X]$, where π is randomly picked from S_n.

Now, consider the expected value $\mathbb{E}_\pi(X)$. It is promised that for some permutation $\tau \in S_n$ the fraction $\delta = \max_{\sigma \in S_n} |\{x | f(x) = g^\sigma(x)\}|/2^n$ is a constant (independent of n). For $0 \leq i \leq n$ let

$$\delta_i = \frac{|\{x \mid f(x) = g^\tau(x), \mathrm{wt}(x) = i\}|}{\binom{n}{i}}.$$

Thus $\sum_{i=0}^{n} \delta_i \binom{n}{i} = \delta 2^n$ which we can write as

$$\sum_{i=0}^{\sqrt{n}} (\delta_i + \delta_{n-\sqrt{n}+i}) \binom{n}{i} + \sum_{i=n/2-\sqrt{n}}^{n/2+\sqrt{n}} \delta_i \binom{n}{i} = \delta 2^n.$$

Since each $\delta_i \leq 1$, $\sum_{i=0}^{\sqrt{n}} (\delta_i + \delta_{n-\sqrt{n}+i}) \binom{n}{i} \leq 2n^{\sqrt{n}+1} \leq 2^{2\sqrt{n}\log n}$ for sufficiently large n.

Let A denote the sum $\sum_{i=n/2-\sqrt{n}}^{n/2+\sqrt{n}} \delta_i \binom{n}{i}$. Then

$$A \geq \delta 2^n \left(1 - \frac{2^{2\sqrt{n}\log n}}{\delta 2^n}\right) \geq \frac{\delta}{2} 2^n.$$

By averaging, there is some hamming weight i in the range $n/2 - \sqrt{n} \leq i \leq n/2 + \sqrt{n}$, such that

$$\delta_i \binom{n}{i} = |\{u \mid \mathrm{wt}(u) = i \text{ and } f(u) = g^\pi(u)\}| \geq \frac{\delta 2^n}{4\sqrt{n}}.$$

We fix this value of i and let S denote the set $\{u \mid \mathrm{wt}(u) = i \text{ and } f(u) = g^\pi(u)\}$. Assume without loss of generality that $|f^{-1}(1) \cap S| > \frac{\delta 2^n}{8\sqrt{n}}$ (Otherwise we consider $f^{-1}(0) \cap S$). Thus, we have $s_i(f) \geq |f^{-1}(1) \cap S| = |(g^\pi)^{-1}(1) \cap S| \geq \frac{\delta 2^n}{8\sqrt{n}}$.

Now, $\{u \mid \mathrm{wt}(u) = i \text{ and } g^\pi(u) = 1\} \supseteq (g^\pi)^{-1}(1) \cap S$. Hence $|(g^\pi)^{-1}(1) \cap S| \leq |\{u \mid \mathrm{wt}(u) = i \text{ and } g^\pi(u) = 1\}| = |\{u \mid \mathrm{wt}(u) = i \text{ and } g(u) = 1\}| = s_i(g)$. Combined with Equation 4 and using the inequality $\binom{n}{i} \leq \frac{2^n}{\sqrt{n}}$ for large enough n, we get the desired lower bound on $\mathbb{E}[X]$:

$$\mathbb{E}[X] \geq \frac{s_i(f)s_i(g)}{\binom{n}{i}} \geq \frac{\delta^2 2^{2n}}{64n\binom{n}{i}} \geq \frac{\delta^2 2^n}{64\sqrt{n}} = \Omega\left(\frac{2^n}{\sqrt{n}}\right).$$

∎

Concluding Remarks. Motivated by the question whether boolean function isomorphism testing has algorithms faster than Luks' $2^{O(n)}$ time algorithm [Luk99], we initiate the study of approximate boolean function isomorphism. As our main result we show a substantially faster algorithm that for boolean functions having small depth and small size circuits computes an approximate isomorphism. Precisely characterizing the approximation threshold for this problem for various boolean function classes is an interesting direction of research.

Acknowledgment. We are grateful to Johannes Köbler and Sebastian Kuhnert for discussions on the topic, especially for their help with Lemma 8.

References

[AT96] Manindra Agrawal and Thomas Thierauf, *The boolean isomorphism problem*, FOCS, 1996, pp. 422–430.

[AB10] Noga Alon and Eric Blais, *Testing Boolean function isomorphism*, APPROX/RANDOM, 2010, pp. 394–405

[BC08] László Babai and Paolo Codenotti, *Isomorphism of hypergraphs of low rank in moderately exponential time*, FOCS, 2008, pp. 667–676.

[BL83] László Babai and Eugene M.Luks, *Canonical labeling of graphs*, *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 171–183, 1983.

[BO10] Eric Blais and Ryan O'Donnell, *Lower Bounds for Testing Function Isomorphism*, CCC, 2010, pp. 235-246.

[Bu00] Bunke: *Graph matching: Theoretical foundations, algorithms, and applications. International Conference on Vision Interface*, Montreal, Quebec, Canada, (2000), pp. 82-88.

[CSM10] Sourav Chakraborty, David García-Soriano, and Arie Matsliah, *Nearly Tight Bounds for Testing Function Isomorphism*, SODA, 2011, pp. 1683-1702

[LMN93] Nathan Linial, Yishay Mansour, and Noam Nisan, *Constant depth circuits, Fourier transform, and learnability*, J. ACM **40** (1993), 607–620.

[Luk99] Eugene M. Luks, *Hypergraph isomorphism and structural equivalence of boolean functions*, STOC, 1999, pp. 652–658.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de