Easy and Hard Functions for the Boolean Hidden Shift Problem

Andrew M. Childs[1], Robin Kothari[2], Maris Ozols[3(1,4)], and Martin Roetteler[4]

1 **Department of Combinatorics & Optimization and Institute for Quantum Computing, University of Waterloo**
200 University Avenue West, Waterloo, ON, N2L 3G1, Canada
amchilds@uwaterloo.ca

2 **David R. Cheriton School of Computer Science and Institute for Quantum Computing, University of Waterloo**
200 University Avenue West, Waterloo, ON, N2L 3G1, Canada
rkothari@uwaterloo.ca

3 **IBM TJ Watson Research Center**
1101 Kitchawan Road, Yorktown Heights, NY 10598, USA
marozols@yahoo.com

4 **NEC Laboratories America**
4 Independence Way, Suite 200, Princeton, NJ 08540, USA
mroetteler@nec-labs.com

Abstract

We study the quantum query complexity of the Boolean hidden shift problem. Given oracle access to $f(x+s)$ for a known Boolean function f, the task is to determine the n-bit string s. The quantum query complexity of this problem depends strongly on f. We demonstrate that the easiest instances of this problem correspond to bent functions, in the sense that an exact one-query algorithm exists if and only if the function is bent. We partially characterize the hardest instances, which include delta functions. Moreover, we show that the problem is easy for random functions, since two queries suffice. Our algorithm for random functions is based on performing the pretty good measurement on several copies of a certain state; its analysis relies on the Fourier transform. We also use this approach to improve the quantum rejection sampling approach to the Boolean hidden shift problem.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Boolean hidden shift problem, quantum algorithms, query complexity, Fourier transform, bent functions

Digital Object Identifier 10.4230/LIPIcs.TQC.2013.50

1 Introduction

Many computational problems for which quantum algorithms can achieve superpolynomial speedup over the best known classical algorithms are related to the *hidden subgroup problem* (see for example [1]).

▶ **Problem 1** (Hidden subgroup problem)**.** For any finite group G, say that a function $f\colon G \to X$ *hides* a subgroup H of G if it is constant on cosets of H in G and distinct on different cosets. Given oracle access to such an f, find a generating set for H.

8th Conference on Theory of Quantum Computation, Communication and Cryptography.
Editors: Simone Severini and Fernando Brandao; pp. 50–79


Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Two early examples of algorithms for hidden subgroup problems are the Deutsch–Jozsa algorithm [2] and Simon's algorithm [3]. Inspired by the latter, Shor discovered efficient quantum algorithms for factoring integers and computing discrete logarithms [4]. Kitaev subsequently introduced the Abelian stabilizer problem and derived an efficient quantum algorithm for it that includes Shor's factoring and discrete logarithm algorithms as special cases [5]. Eventually it was observed that all of the above algorithms solve special instances of the hidden subgroup problem [6, 7, 8].

This early success created significant interest in studying various instances of the hidden subgroup problem and led to discovery of many other quantum algorithms. For example, period finding over the reals was used by Hallgren to construct an efficient quantum algorithm for solving Pell's equation [9]. Moreover, the hidden subgroup problem over symmetric and dihedral groups are related to the graph isomorphism problem [10, 11, 12, 13, 14] and certain lattice problems [15], respectively. The possibility of efficient quantum algorithms for these problems remains a major open question. Kuperberg has provided a subexponential-time quantum algorithm for the dihedral subgroup problem [16, 17, 18], which has been used to construct elliptic curve isogenies in quantum subexponential time [19].

The *hidden shift problem* (also known as the *hidden translation problem*) is a natural variant of the hidden subgroup problem.

▶ **Problem 2** (Hidden shift problem)**.** Let G be a finite group. Given oracle access to functions $f_0, f_1 \colon G \to X$ with the promise that $f_0(x) = f_1(x \cdot s)$ for some $s \in G$, determine s.

If G is Abelian and f_0 is injective, this problem is equivalent to the hidden subgroup problem in the semidirect product group $G \rtimes \mathbb{Z}_2$, where the group operation is defined by $(x_1, b_1) \cdot (x_2, b_2) := \left(x_1 \cdot x_2^{(-1)^{b_1}}, b_1 + b_2\right)$ and the hiding function $f \colon G \rtimes \mathbb{Z}_2 \to X$ is defined as $f[(x, b)] := f_b(x)$. One can check that f is constant on cosets of $H := \langle (s, 1) \rangle$ and that injectivity of f_0 implies that f is distinct on different cosets. Thus, f hides the subgroup H in $G \rtimes \mathbb{Z}_2$.

Notice that if $G = \mathbb{Z}_d$ then $G \rtimes \mathbb{Z}_2$ is the dihedral group. Ettinger and Høyer [20] showed that the dihedral hidden subgroup problem reduces to the special case of a subgroup $\langle (s, 1) \rangle$. Thus the hidden shift problem in $\mathbb{Z}_d$ (with f_0 injective) is equivalent to the dihedral hidden subgroup problem, motivating further study of the hidden shift problem for various groups [21, 22, 23, 24, 25, 26].

While the case where f_0 is injective is simply related to the hidden subgroup problem, one can also consider the hidden shift problem without this promise. For example, van Dam, Hallgren, and Ip [21] gave an efficient quantum algorithm to solve the shifted Legendre symbol problem, a non-injective hidden shift problem. Their result breaks a proposed pseudorandom function [27], showing the potential for cryptographic applications of hidden shift problems. Work on hidden shift problems can also inspire new algorithmic techniques, such as quantum rejection sampling [28]. Moreover, negative results could have applications to designing classical cryptosystems that are secure against quantum attacks [15].

For the rest of the paper we restrict our attention to the *Boolean hidden shift problem*, in which the hiding function has the form $f_0 \colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ for some integer $n \geq 1$. For this problem (with $n > 1$), f_0 is necessarily non-injective. This problem has previously been studied in [29, 30, 31, 28, 32].

Notice that to determine the hidden shift of an injective function f_0, it suffices to find x_0 and x_1 such that $f_0(x_0) = f_1(x_1)$. However, this does not hold in the non-injective case, so it is nontrivial to verify a candidate hidden shift (see [28, Appendix B]). In fact, sometimes the hidden shift cannot be uniquely determined in principle (see Sect. D.1). On the other

hand, by considering functions with codomain $\mathbb{Z}_2$, we have more structure than in the hidden subgroup problem or the injective hidden shift problem, where the codomain is arbitrary. We exploit this structure by encoding the values of the function as phases and using the Fourier transform.

More precisely, the main problem studied in this paper, sometimes denoted BHSP_f, is as follows.

▶ **Problem 3** (Boolean hidden shift problem)**.** Given a complete description of a function $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ and access to an oracle for the *shifted function* $f_s(x) := f(x+s)$, determine the *hidden shift* $s \in \mathbb{Z}_2^n$.

Note that in degenerate cases, when the oracle does not contain enough information to completely recover the hidden shift, no algorithm can succeed with certainty.

Let us highlight the main differences between the above problem and other types of hidden shift problem. In the Boolean hidden shift problem,

- the function f is *not* injective, and
- we are given a *complete description* of the unshifted function f instead of having only oracle access to f.

Moreover, we are interested only in the *query complexity* of the problem and do not consider its time complexity. This means that we can pre-process the description of f (which may be exponentially large) at no cost before we start querying the oracle.

This problem has been considered previously, e.g., by [28]. Note that some prior work does not give complete description of f but only oracle access to it [29, 30, 31, 32] (and in some cases [30] also gives oracle access also to $\tilde{f}$, the dual bent function of f).

To address this problem on a quantum computer, we use an oracle that computes the shifted function in the phase. Such an oracle can be implemented using only one query to an oracle that computes the function in a register.

▶ **Definition 1.** The quantum *phase oracle* is $O_{f_s}\colon |x\rangle \mapsto (-1)^{f(x+s)}|x\rangle$.

More generally, one can use a controlled phase oracle $\bar{O}_{f_s}\colon |b,x\rangle \mapsto (-1)^{bf(x+s)}|b,x\rangle$ for $b \in \{0,1\}$, which is equivalent to an oracle that computes the function in the first register up to a Hadamard transform. Some of our algorithms do not make use of this freedom, although our lower bounds always take it into account.

Ultimately, we would like to characterize the classical and quantum query complexities of the hidden shift problem for any Boolean function (or more generally, for any function $f\colon \mathbb{Z}_d^n \to \mathbb{Z}_d$). While we do not resolve this question completely, we make progress by providing a new quantum query algorithm (see Sect. 4) and improving an existing one (see Sect. 5). However, it remains an open problem to better understand both the classical and quantum query complexities of the BHSP for general functions.

While general functions are difficult to handle, the quantum query complexity of the hidden shift problem is known for two extreme classes of Boolean functions:

- If f is a *bent function*, i.e., it has a "flat" Fourier spectrum (see Sect. 3.1), then one quantum query suffices to solve the problem exactly [30].
- If f is a *delta function*, i.e., $f(x) := \delta_{x,x_0}$ for some $x_0 \in \mathbb{Z}_2^n$, then the hidden shift problem for f is equivalent to unstructured search—finding $x_0 + s$ among the 2^n elements of $\mathbb{Z}_2^n$—so the quantum query complexity is $\Theta(\sqrt{2^n})$ [33, 34].

Intuitively, other Boolean functions should lie somewhere between these two extreme cases. In this paper, we give formal evidence for this: we show that the problem can be solved exactly with one query only if f is bent, and we show that it can be solved for any function

with $O(\sqrt{2^n})$ queries, with a lower bound of $\Omega(\sqrt{2^n})$ only if the truth table of f has Hamming weight $\Theta(1)$ or $\Theta(2^n)$. This is similar to the weighing matrix problem considered by van Dam [35], which also interpolates between two extreme cases: the Bernstein-Vazirani problem [36] and Grover search [33].

Aside from delta and bent functions, the Boolean hidden shift problem has previously been considered for several other families of functions. Boolean functions that are quadratic forms or are close to being quadratic are studied in [29]. Random Boolean functions have been considered in [31, 32]. Finally, [28] uses quantum rejection sampling to solve the BHSP for any function, although its performance in general is not well understood.

Apart from algorithms designed specifically for the BHSP, there are generic classical and quantum algorithms for the BHSP derived from learning theory. In particular, the BHSP can be viewed as an instantiation of the problem of exact learning through membership queries. The resulting algorithms are optimal for classical and quantum query complexity up to polynomial factors in n. More precisely, for any learning problem, Servedio and Gortler define a combinatorial parameter γ [37]. For the problem BHSP_f, we denote the parameter as γ_f. From their results it follows that the classical query complexity of BHSP_f is lower bounded by $\Omega(n)$ and $\Omega(1/\gamma_f)$ and upper bounded by $O(n/\gamma_f)$. For quantum algorithms, they show a lower bound of $\Omega(1/\sqrt{\gamma_f})$. Atıcı and Servedio [38] later showed an upper bound of $O(n \log n/\sqrt{\gamma_f})$ queries.

The rest of this paper is organized as follows. In Sect. 2 we briefly review some basic Fourier analysis to establish notation. Next, in Sect. 3 we explore the extreme cases of the BHSP. In Sect. 4 we introduce a new approach to the BHSP based on the pretty good measurement. We analyze its performance for delta, bent, and random Boolean functions in Sect. 4.3. In Sect. 5 we propose an alternative method for boosting the success probability of the quantum rejection sampling algorithm from [28]. Finally, Sect. 6 presents conclusions and open questions.

This paper has several appendices. In Appendix A we show that the easy instances of the BHSP correspond to bent functions. In Appendix B, we show that with one quantum query we can succeed on a constant fraction of all functions, whereas in Appendix C we prove that two quantum queries suffice to solve the BHSP for random functions. Finally, in Appendix D we analyze the structure of zero Fourier coefficients of Boolean functions.

2 Fourier analysis

Our main tool is Fourier analysis of Boolean functions [39]. Here we state the basic definitions and properties of the Fourier transform and convolution. Readers who are familiar with the topic might skip this section, except for Definition 6.

▶ **Definition 2.** The *Hadamard gate* is $H := \frac{1}{\sqrt{2}}\left(\begin{smallmatrix} 1 & 1 \\ 1 & -1 \end{smallmatrix}\right)$.

▶ **Definition 3.** The *Fourier transform* of a function $F\colon \mathbb{Z}_2^n \to \mathbb{R}$ is a function $\hat{F}\colon \mathbb{Z}_2^n \to \mathbb{R}$ defined as $\hat{F}(w) := \langle w|H^{\otimes n}|F\rangle$ where $|F\rangle := \sum_{x\in\mathbb{Z}_2^n} F(x)|x\rangle$. Here $\hat{F}(w)$ is called the *Fourier coefficient* of F at $w \in \mathbb{Z}_2^n$. Explicitly, $\hat{F}(w) = \frac{1}{\sqrt{2^n}} \sum_{x\in\mathbb{Z}_2^n} (-1)^{w\cdot x} F(x)$ where $x \cdot y := \sum_{i=1}^n x_i y_i$. The set $\{\hat{F}(w)\colon w \in \mathbb{Z}_2^n\}$ is called the *Fourier spectrum* of F.

To define the Fourier transform of a Boolean function $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$, we identify f with a real-valued function $F\colon \mathbb{Z}_2^n \to \mathbb{R}$ in a canonical way: $F(x) := (-1)^{f(x)}/\sqrt{2^n}$. Note that F is normalized: $\sum_{x\in\mathbb{Z}_2^n} |F(x)|^2 = 1$. Now we can abuse Definition 3 as follows:

▶ **Definition 4.** The *Fourier transform* of $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ is $\hat{F}(w) = \frac{1}{2^n} \sum_{x\in\mathbb{Z}_2^n} (-1)^{w\cdot x + f(x)}$.

To avoid confusion, we use lower case letters for $\mathbb{Z}_2$-valued functions and capital letters for $\mathbb{R}$-valued functions.

▶ **Definition 5.** The *convolution* of functions $F, G\colon \mathbb{Z}_2^n \to \mathbb{R}$ is a function $(F * G)\colon \mathbb{Z}_2^n \to \mathbb{R}$ defined as $(F * G)(x) := \sum_{y\in\mathbb{Z}_2^n} F(y)G(x-y)$. The *t-fold convolution* of $F\colon \mathbb{Z}_2^n \to \mathbb{R}$ is a function $F^{*t}\colon \mathbb{Z}_2^n \to \mathbb{R}$ defined as

$$F^{*t}(w) := (\underbrace{F * \cdots * F}_{t})(w) = \sum_{y_1,\ldots,y_{t-1}\in\mathbb{Z}_2^n} F(y_1)\cdots F(y_{t-1})F\big(w-(y_1+\cdots+y_{t-1})\big). \tag{1}$$

▶ **Fact.** Let $F, G, H\colon \mathbb{Z}_2^n \to \mathbb{R}$ denote arbitrary functions. The Fourier transform and convolution have the following basic properties:

1. The Fourier transform is linear: $\widehat{F+G} = \hat{F} + \hat{G}$.
2. The Fourier transform is self-inverse: $\hat{\hat{F}} = F$.
3. Since $H^{\otimes n}$ is unitary, the Plancherel identity $\sum_{w\in\mathbb{Z}_2^n}|\hat{F}(w)|^2 = \sum_{x\in\mathbb{Z}_2^n}|F(x)|^2$ holds.
4. Convolution is commutative ($F * G = G * F$) and associative ($(F * G) * H = F * (G * H)$).
5. The Fourier transform and convolution are related through the following identities: $(\hat{F} * \hat{G})/\sqrt{2^n} = \widehat{FG}$ and $\widehat{(F * G)}/\sqrt{2^n} = \hat{F}\hat{G}$, where $FG\colon \mathbb{Z}_2^n \to \mathbb{C}$ is the entry-wise product of functions F and G: $(FG)(x) := F(x)G(x)$.
6. By induction, the t-fold convolution satisfies the identity $\big[\hat{F}/\sqrt{2^n}\big]^{*t} = \widehat{F^t}/\sqrt{2^n}$.

The following t-fold generalization of the Fourier spectrum plays a key role:

▶ **Definition 6.** For $t \geq 1$, the *t-fold Fourier coefficient* of $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ at $w \in \mathbb{Z}_2^n$ is $\mathcal{F}^t(w) := \sqrt{\big[\hat{F}^2\big]^{*t}(w)}$. In particular, for $t = 1$ we have $\mathcal{F}^1(w) = |\hat{F}(w)|$.

We can express $\mathcal{F}^t(w)$ in many equivalent ways using the identities listed above:

$$\big[\mathcal{F}^t(w)\big]^2 = \big[\hat{F}^2\big]^{*t}(w) = \left[\frac{1}{\sqrt{2^n}}\big(\widehat{F * F}\big)\right]^{*t}(w) = \frac{1}{\sqrt{2^n}}\,(\widehat{F * F})^t\,(w). \tag{2}$$

3 Characterization of extreme cases

In this section we explore the set of functions for which the quantum query complexity of the BHSP is extreme. Recall that the BHSP can be solved with one query for bent functions and with $\Theta(\sqrt{2^n})$ queries for delta functions. Here we prove that BHSP_f can be solved exactly with one query only if f is bent, and with $O(\sqrt{2^n})$ queries (with bounded error) for any f.

3.1 Easy functions are bent

In general, the quantum query complexity of the BHSP for an arbitrary function is unknown. However, the problem becomes particularly easy for *bent functions*, where a single query suffices to solve the problem exactly [30]. In fact, bent functions are the only functions with this property, as we show here.

Bent functions can be characterized in many equivalent ways [40, 41]. The standard definition is that bent functions have a "flat" Fourier spectrum:

▶ **Definition 7.** A Boolean function $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ is *bent* if all its Fourier coefficients $\hat{F}(w)$ (see Definition 4) have the same absolute value: $|\hat{F}(w)| = 1/\sqrt{2^n}$ for all $w \in \mathbb{Z}_2^n$.

While many examples of bent functions have been constructed (e.g., see [42, 43, 44]), no complete classification is known. As an example, the *inner product* of two n-bit strings (modulo two) is a bent function [41, 42]: $\mathrm{IP}_n(x_1,\ldots,x_n,y_1,\ldots,y_n) := \sum_{i=1}^n x_i y_i$.

We make a few simple observations about bent functions. Recall from Sect. 2 that the Fourier spectrum of f is normalized as $\sum_{w\in\mathbb{Z}_2^n}|\hat{F}(w)|^2 = 1$, so the spectrum is "flat" only when $|\hat{F}(w)| = 1/\sqrt{2^n}$ for all $w \in \mathbb{Z}_2^n$. Recall from Definition 4 that $\hat{F}(w)$ is always an integer multiple of $1/2^n$. Thus an n-variable function can only be bent if n is even [43, 42]. Moreover, from $|\hat{F}(0)| = 1/\sqrt{2^n}$ we get that $|\sum_{w\in\mathbb{Z}_2^n}(-1)^{f(x)}| = \sqrt{2^n}$, so a bent function f is close to being balanced: $|f| = (2^n \pm \sqrt{2^n})/2$ where $|f| := |\{x \in \mathbb{Z}_2^n \colon f(x) = 1\}|$ is the *Hamming weight* of f.

Our main result regarding bent functions is as follows.

▶ **Theorem 8.** *Let $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ be a Boolean function with $n \geq 2$. A quantum algorithm can solve BHSP_f exactly with a single query to O_{f_s} if and only if f is bent.*

The proof is based on a characterization of an exact one-query quantum algorithm using a system of linear equations. This system can be analyzed in terms of the autocorrelation of f, which in turn characterizes whether f is bent. The proof appears in Appendix A.

3.2 Hard functions

In this section we study hard instances of the BHSP. First, we observe that the quantum query complexity of solving BHSP_f for any function f is $O(\sqrt{2^n})$.

▶ **Theorem 9.** *For any $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$, the bounded-error quantum query complexity of BHSP_f is $O(\sqrt{2^n})$.*

If we view f as a 2^n-bit string indexed by $x \in \mathbb{Z}_2^n$, this is a special case of the oracle identification problem considered by Ambainis et al. [45, Theorem 3], who show the following.

▶ **Theorem 10** (Oracle Identification Problem)**.** *Given oracle access to an unknown N-bit string with the promise that it is one of N known strings, the bounded-error quantum query complexity of identifying the unknown string is $O(\sqrt{N})$.*

In the BHSP, we have $N := 2^n$. By Theorem 9, the hardest functions are those with query complexity $\Omega(\sqrt{N})$. We know that delta functions have this query complexity, but are there any other functions that are as hard? The delta functions have $|f| = 1$ (recall that $|f|$ denotes the Hamming weight of f). Next we show that as $|f|$ increases, the query complexity strictly decreases at first, until $|f| = \Theta(\sqrt{N})$. For example, functions with $|f| = 2$ have strictly smaller query complexity than the delta functions. However, as we approach $|f| = \Omega(N)$, our upper bound is $\Theta(\sqrt{N})$ again. Without loss of generality, we assume that $|f| \leq N/2$; otherwise we can simply negate the function to obtain a function with $|f| \leq N/2$ that has exactly the same query complexity. Formally, we show the following refinement of Theorem 9.

▶ **Theorem 11.** *For any $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ with $1 \leq |f| \leq N/2$, the bounded-error quantum query complexity of BHSP_f is at most $\frac{\pi}{4}\sqrt{N/|f|} + O(\sqrt{|f|})$.*

Proof. The algorithm has two parts. First we look for a "1" in the bit string contained in the oracle, i.e., an x such that $f(x) = 1$. This can be done by a variant of Grover's algorithm that finds a "1" in a string of length N using at most $\frac{\pi}{4}\sqrt{N/|f|}$ queries [46]. Now we have an x such that $f_s(x) = 1$ for some unknown s. Note that there can be at most $|f|$ shifts s

with this property, because each corresponds to a distinct solution to $f(x+s)=1$ and there are only $|f|$ solutions to this equation.

We are now left with $|f|$ candidates for the black-box function. Viewing this as an oracle identification problem, we have oracle access to an N-bit string that could be one of $|f|$ possible candidates. Although the string has length N, there are only $|f|$ potential candidates, so intuitively it seems like we should be able to restrict the strings to length $|f|$ and apply Theorem 10 to obtain the desired result.

Formally, it can be shown that given $k \geq 2$ distinct Boolean strings of length N, there is a subset of indices, S, of size at most $k-1$, such that all the strings are distinct when restricted to S. We show this by induction. The base case is easy: we can choose any index that differentiates the two distinct strings. Now say we have m distinct strings $y_1, y_2, \ldots, y_m$ and a subset of indices S of size at most $m-1$, such that the m strings are distinct on S. We want to add another string y_{m+1} and increase the size of S by at most 1. If y_{m+1} differs with $y_1, y_2, \ldots, y_m$ on S, then we do not need to add any more indices to S and we are done. If y_{m+1} agrees with one of $y_1, y_2, \ldots, y_m$ on all of S, first note that it can only agree with one such string; to differentiate between these two, we add any index at which they differ to S, which must exist since they are distinct. ◀

This shows that a function can be hard—i.e., can have query complexity $\Theta(\sqrt{N})$—only if $|f|$ is $O(1)$ or $\Theta(N)$.

Note that there do exist hard functions with $|f| = \Theta(N)$. For example, consider the following function: $f(x) = 1$ if the first bit of x is 1 or if x is the all-zero string. This essentially embeds a delta function on the last $n-1$ bits, and thus requires $\Theta(\sqrt{N})$ queries. This function has $|f| = N/2 + 1$. However, there are also easy functions with $|f| = \Theta(N)$, namely the bent functions. Thus the Hamming weight does not completely characterize the hardness of the BHSP at high Hamming weight. However, it precisely characterizes the quantum query complexity at low Hamming weight:

▶ **Theorem 12.** *For any $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ with no undetectable shifts, the bounded-error quantum query complexity of* BHSP_f *is* $\Omega(\sqrt{N/|f|})$.

This follows from a simple application of the quantum adversary argument, with the adversary matrix taken to be the all ones matrix with zeroes on the diagonal. It also follows from Theorem 4 of [45].

4 The PGM approach

We now present an approach to the Boolean hidden shift problem based on the pretty good measurement (PGM) [47]. In particular, this approach shows that the Boolean hidden shift problem for random functions has small query complexity (see Sect. 4.3.3).

The main idea of the PGM approach is as follows. We apply the oracle on the uniform superposition and prepare t independent copies of the resulting state (see Sect. 4.1). Then we use knowledge of the function f to perform the PGM in order to extract the hidden shift s (see Sect. 4.2). A similar strategy was used to efficiently solve the hidden subgroup problem for certain semidirect product groups, including the Heisenberg group [48], and was subsequently applied to a hidden polynomial problem [49].

4.1 Performing t queries in parallel

In this section we describe a quantum circuit that prepares a state with $w \cdot s$ encoded in the phase, where s is the hidden shift and w is the label of the corresponding standard basis

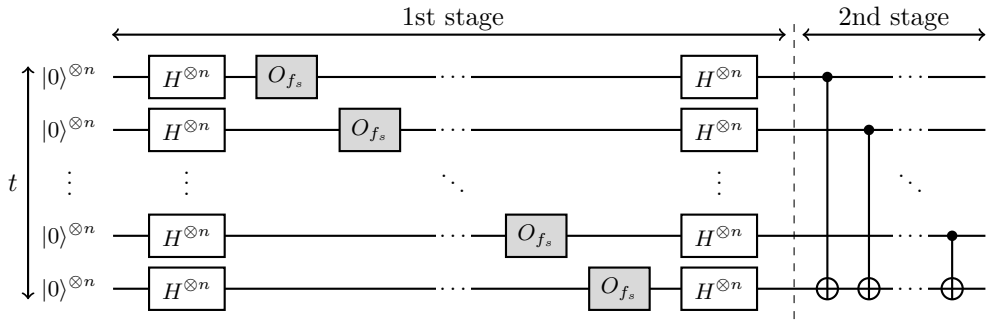

Figure 1 Quantum algorithm for preparing the t-fold Fourier sate $|\Phi^t(s)\rangle$ in Eq. (8). The state on any register at the end of the first stage is given in Eq. (4).

vector. We use this circuit t times in parallel, followed by a sequence of CNOTs, to prepare a certain state $|\Phi^t(s)\rangle$. In the next section we perform a PGM on these states for different values of s.

4.1.1 Circuit

The circuit for preparing $|\Phi^t(s)\rangle$ appears in Fig. 1. It consists of two stages. The first stage prepares t identical copies of the same state by using one oracle call between two quantum Fourier transforms on each register independently. Recall from Definition 1 that the oracle acts on n qubits and encodes the function in the phase: $O_{f_s}\colon |x\rangle \mapsto (-1)^{f(x+s)}|x\rangle$. The second stage entangles the states by applying a sequence of transversal controlled-NOT gates acting as $|x\rangle|y\rangle \mapsto |x\rangle|y+x\rangle$ for $x, y \in \mathbb{Z}_2^n$.

Note that all unitary post-processing after the oracle queries can be omitted since it does not affect the distinguishability of the states. We include it only to simplify the analysis.

4.1.2 Analysis

During the first stage of the circuit, the first register evolves under $H^{\otimes n} O_{f_s} H^{\otimes n}$ (see Fig. 1):

$$|0\rangle^{\otimes n} \mapsto \frac{1}{\sqrt{2^n}} \sum_{x\in\mathbb{Z}_2^n} |x\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{x\in\mathbb{Z}_2^n} (-1)^{f(x+s)}|x\rangle \mapsto \frac{1}{2^n} \sum_{x,y\in\mathbb{Z}_2^n} (-1)^{f(x+s)+x\cdot y}|y\rangle. \tag{3}$$

We can rewrite the resulting state as follows:

$$\sum_{y\in\mathbb{Z}_2^n} (-1)^{s\cdot y} \left(\frac{1}{2^n} \sum_{x\in\mathbb{Z}_2^n} (-1)^{f(x)+x\cdot y} \right) |y\rangle = \sum_{y\in\mathbb{Z}_2^n} (-1)^{s\cdot y} \hat{F}(y)|y\rangle. \tag{4}$$

The overall state after the first stage is just the t-fold tensor product of the above state:

$$\sum_{y_1,\ldots,y_t\in\mathbb{Z}_2^n} (-1)^{s\cdot(y_1+\cdots+y_t)} \bigotimes_{i=1}^{t} \hat{F}(y_i)|y_i\rangle. \tag{5}$$

In the second stage of the algorithm, the controlled-NOT gates transform this state into

$$\sum_{y_1,\ldots,y_t\in\mathbb{Z}_2^n} (-1)^{s\cdot(y_1+\cdots+y_t)} \left[\bigotimes_{i=1}^{t-1} \hat{F}(y_i)|y_i\rangle\right] \hat{F}(y_t)|y_1+\cdots+y_t\rangle \tag{6}$$

$$= \sum_{y_1,\ldots,y_t\in\mathbb{Z}_2^n} (-1)^{s\cdot y_t} \left[\bigotimes_{i=1}^{t-1} \hat{F}(y_i)|y_i\rangle\right] \hat{F}\big(y_t-(y_1+\cdots+y_{t-1})\big)|y_t\rangle. \tag{7}$$

We can rewrite this state as

$$|\Phi^t(s)\rangle := \sum_{w\in\mathbb{Z}_2^n} (-1)^{s\cdot w}|\mathcal{F}_w^t\rangle|w\rangle, \tag{8}$$

where the non-normalized state $|\mathcal{F}_w^t\rangle$ on $(t-1)n$ qubits is given by

$$|\mathcal{F}_w^t\rangle := \sum_{y_1,\ldots,y_{t-1}\in\mathbb{Z}_2^n} \hat{F}(y_1)\cdots\hat{F}(y_{t-1})\hat{F}\big(w-(y_1+\cdots+y_{t-1})\big)|y_1\rangle\cdots|y_{t-1}\rangle. \tag{9}$$

Its norm is just the t-fold Fourier coefficient: $\||\mathcal{F}_w^t\rangle\| = \mathcal{F}^t(w)$ (see Definition 6).

4.2 The pretty good measurement

Let $\{\rho_s^{(t)}\colon s\in\mathbb{Z}_2^n\}$ be a set of mixed states where $\rho_s^{(t)}$ is given with probability p_s. The *pretty good measurement* (PGM) [47] for discriminating these states is a POVM with operators $\{E_s\colon s\in\mathbb{Z}_2^n\}\cup\{E_*\}$ where

$$E_s := E^{-1/2}\, p_s\rho_s^{(t)}\, E^{-1/2}, \qquad E := \sum_{s\in\mathbb{Z}_2^n} p_s\rho_s^{(t)}, \qquad E_* := I - \sum_{s\in\mathbb{Z}_2^n} E_s. \tag{10}$$

In our case, $\rho_s^{(t)} := |\Phi^t(s)\rangle\langle\Phi^t(s)|$ and $p_s := 1/2^n$ where $|\Phi^t(s)\rangle$ is defined in Eq. (8).

To find the operators E_s, we compute

$$E = \sum_{s\in\mathbb{Z}_2^n} \frac{1}{2^n} \sum_{w,w'\in\mathbb{Z}_2^n} (-1)^{(w+w')\cdot s}|\mathcal{F}_w^t\rangle\langle\mathcal{F}_{w'}^t| \otimes |w\rangle\langle w'| \tag{11}$$

$$= \sum_{w\in\mathbb{Z}_2^n} \||\mathcal{F}_w^t\rangle\|^2 \cdot \frac{|\mathcal{F}_w^t\rangle\langle\mathcal{F}_w^t|}{\||\mathcal{F}_w^t\rangle\|^2} \otimes |w\rangle\langle w|. \tag{12}$$

From now on we use the convention that terms with $\||\mathcal{F}_w^t\rangle\| = 0$ are omitted from all sums. As E is a sum of mutually orthogonal rank-1 operators with eigenvalues $\||\mathcal{F}_w^t\rangle\|^2$, we find

$$E^{-1/2} = \sum_{w\in\mathbb{Z}_2^n} \frac{1}{\||\mathcal{F}_w^t\rangle\|} \cdot \frac{|\mathcal{F}_w^t\rangle\langle\mathcal{F}_w^t|}{\||\mathcal{F}_w^t\rangle\|^2} \otimes |w\rangle\langle w|. \tag{13}$$

Note that $E_s = |E_s\rangle\langle E_s|$ where $|E_s\rangle := E^{-1/2}\sqrt{p_s}|\Phi^t(s)\rangle$. We can express $|E_s\rangle$ as follows:

$$|E_s\rangle = \left(\sum_{w\in\mathbb{Z}_2^n} \frac{|\mathcal{F}_w^t\rangle\langle\mathcal{F}_w^t|}{\||\mathcal{F}_w^t\rangle\|^3} \otimes |w\rangle\langle w|\right) \frac{1}{\sqrt{2^n}} \left(\sum_{w\in\mathbb{Z}_2^n} (-1)^{w\cdot s}|\mathcal{F}_w^t\rangle|w\rangle\right) \tag{14}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{w\in\mathbb{Z}_2^n} (-1)^{w\cdot s} \frac{|\mathcal{F}_w^t\rangle}{\||\mathcal{F}_w^t\rangle\|} \otimes |w\rangle. \tag{15}$$

Notice that the vectors $|E_s\rangle$ are orthonormal, so the PGM is just an orthogonal measurement in this basis (with another outcome corresponding to the orthogonal complement). Therefore the measurement is unambiguous: if it outputs a value of s (rather than the inconclusive outcome $*$) then it is definitely correct. The corresponding zero-error algorithm can be summarized as follows:

PGM(f,t)
1. Prepare $|\Phi^t(s)\rangle$ using the circuit shown in Fig. 1.
2. Recover s by performing an orthogonal measurement with projectors $\{|E_s\rangle\langle E_s| : s \in \mathbb{Z}_2^n\} \cup \{E_*\}$.

▶ **Lemma 13.** *The t-query algorithm* **PGM**(f,t) *solves* BHSP$_f$ *with success probability*

$$p_f(t) := \left(\frac{1}{\sqrt{2^n}} \sum_{w\in\mathbb{Z}_2^n} \mathcal{F}^t(w)\right)^2, \tag{16}$$

where $\mathcal{F}^t(w) = \||\mathcal{F}_w^t\rangle\|$ denotes the t-fold Fourier spectrum of $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ (see Definition 6).

Proof. Recall that the PGM for discriminating the states $|\Phi^t(s)\rangle = \sum_{w\in\mathbb{Z}_2^n}(-1)^{s\cdot w}|\mathcal{F}_w^t\rangle|w\rangle$ from Eq. (8) is an orthogonal measurement on $|E_s\rangle$ (defined in Eq. (15)) and the orthogonal complement. Thus, given the state $|\Phi^t(s)\rangle$, the success probability to recover the hidden shift s correctly is $\left|\langle E_s|\Phi^t(s)\rangle\right|^2$. This is equal to the expression in Eq. (16). Moreover, it does not depend on s, so $p_f(t)$ is the success probability even if s is chosen adversarially as in the definition of BHSP$_f$ (Problem 3). Note that the convention of omitting terms with $\||\mathcal{F}_w^t\rangle\| = 0$ is consistent since such terms do not appear in Eq. (16). ◀

We can use Eq. (2) to write the success probability as

$$p_f(t) = \frac{1}{2^n}\left(\sum_{w\in\mathbb{Z}_2^n} \sqrt{\frac{1}{\sqrt{2^n}}\,\widehat{(F*F)^t}\,(w)}\right)^2. \tag{17}$$

Recall from Sect. 2 that $\mathcal{F}^1(w) = |\hat{F}(w)|$, so for $t = 1$ we have

$$p_f(1) = \frac{1}{2^n}\left(\sum_{w\in\mathbb{Z}_2^n} |\hat{F}(w)|\right)^2. \tag{18}$$

4.3 Performance analysis

In this section we analyze the performance of the PGM algorithm described above on several different classes of Boolean functions. For delta functions our algorithm performs worse than Grover's algorithm. On the other hand, for bent and random functions it needs only one and two queries, respectively.

4.3.1 Delta functions

Let us check how our algorithm performs when f is a delta function, i.e., $f(x) = \delta_{x,x_0}$ for some $x_0 \in \mathbb{Z}_2^n$. A simple calculation using the Fourier spectrum of a delta function shows that the success probability of **PGM**(f,t) is

$$p_f(t) = \frac{1}{2^{2n}}\left((2^n-1)\sqrt{1-\left(\frac{2^n-4}{2^n}\right)^t} + \sqrt{1+(2^n-1)\left(\frac{2^n-4}{2^n}\right)^t}\right)^2. \tag{19}$$

Unfortunately, if we choose $t = \sqrt{2^n}$, then the success probability goes to 0 as $n \to \infty$. In fact, the same happens even if $t = c^n$ for any $c < 2$. Only if we take $t = 2^n$ does the success probability approach a positive constant $1 - 1/e^4 \approx 0.98$ as $n \to \infty$. This means that the PGM algorithm does not give us the quadratic speedup of Grover's algorithm. (Indeed, this follows from the more general fact that quantum speedup for unstructured search cannot be parallelized [50].) Thus the PGM algorithm is not optimal in general.

4.3.2 Bent functions

Let f be a Bent function. Recall from Sect. 3.1 that its Fourier spectrum is "flat", i.e., $|\hat{F}(w)| = 1/\sqrt{2^n}$ for all $w \in \mathbb{Z}_2^n$. In this case, Eq. (18) gives $p_f(1) = 1$, so we can find the hidden shift with certainty by measuring $|\Phi^1(s)\rangle$ with the pretty good measurement (recall that preparing $|\Phi^1(s)\rangle$ requires only one query to O_{f_s}), reproducing a result of Rötteler.

▶ **Theorem 14** ([30]). *If f is a bent function then a quantum algorithm can solve* BHSP_f *exactly using a single query to O_{f_s}.*

4.3.3 Random functions

For random Boolean functions, our algorithm performs almost as well as for bent functions. For random f, we are only able to show that the expected success probability of the one-query algorithm $\mathbf{PGM}(f, 1)$ is at least $2/\pi + o(1)$ for large n (see Theorem 19 in Appendix B), so the algorithm only succeeds with constant probability, which cannot easily be boosted. However, the expected success probability of the two-query algorithm $\mathbf{PGM}(f, 2)$ is exponentially close to 1.

▶ **Theorem 15.** *Let f be an n-argument Boolean function chosen uniformly at random and suppose that a hidden shift for f is chosen adversarially. Then* $\mathbf{PGM}(f, 2)$ *solves* BHSP_f *with expected success probability $\bar{p} \geq 1 - \frac{3}{64} \cdot 2^{-n}$.*

The proof uses the second moment method to lower bound the expected success probability. We compute the variance of the 2-fold Fourier spectrum by relating it to the combinatorics of pairings. The proof appears in Appendix C.

Theorem 15 implies that our algorithm can determine the hidden shift with near certainty as $n \to \infty$. This is surprising since some functions, such as delta functions (see Sect. 3.2), require $\Omega(\sqrt{2^n})$ queries. Furthermore, a randomly chosen function could have an undetectable shift (see Sect. D.1), in which case it is not possible in principle to completely determine an adversarially chosen shift with success probability more than 1/2.

At first glance, Theorem 15 may appear to be a strengthening of the main result of [31], which shows that $O(n)$ queries suffice to solve a version of the Boolean hidden shift problem for a random function. However, while our approach uses dramatically fewer queries, the results are not directly comparable: Ref. [31] considers a weaker model in which the unshifted function is given by an oracle rather than being known explicitly. In particular, while the result of [31] gives an average-case exponential separation between classical and quantum query complexity, such a result is not possible in the model where the function is known explicitly. In this model, there cannot be a super-polynomial speedup for quantum computation. This follows from general results from learning theory discussed at the end of Sect. 1. In particular, it follows that if the quantum query complexity of the problem for a function f is Q, then the deterministic classical query complexity of the problem for the same function is at most $O(nQ^2)$ [37].

5 Quantum rejection sampling with parallel queries

In this section we explain a hybrid approach that combines the Quantum Rejection Sampling (QRS) algorithm for the BHSP [28] with the PGM approach. The resulting algorithm does not require an extra amplification step for boosting the success probability, unlike the original QRS algorithm.

5.1 Original quantum rejection sampling approach

▶ **Theorem 16** ([28]). *For a given Boolean function $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$, define unit vectors $\boldsymbol{\pi}, \boldsymbol{\sigma} \in \mathbb{R}^{2^n}$ as $\pi_w := |\hat{F}(w)|$ and $\sigma_w := 1/\sqrt{2^n}$ for $w \in \mathbb{Z}_2^n$. Moreover, let*

$$p_{\min} := (\boldsymbol{\sigma}^{\mathsf{T}} \cdot \boldsymbol{\pi})^2 = \frac{1}{2^n}\Bigg(\sum_{w\in\mathbb{Z}_2^n} |\hat{F}(w)|\Bigg)^2, \quad p_{\max} := \sum_{k\colon \pi_k>0} \sigma_k^2 = \frac{1}{2^n}|\{w\colon \hat{F}(w)\neq 0\}|. \tag{20}$$

For any desired success probability $p \in [p_{\min}, p_{\max}]$, the quantum rejection sampling algorithm solves BHSP_f *with $O(1/\|\boldsymbol{\varepsilon}^p_{\boldsymbol{\pi}\to\boldsymbol{\sigma}}\|)$ queries, where the "water-filling" vector $\boldsymbol{\varepsilon}^p_{\boldsymbol{\pi}\to\boldsymbol{\sigma}} \in \mathbb{R}^{2^n}$ is defined in [28].*

In particular, if $p_{\max} = 1$ then the QRS algorithm can achieve any success probability arbitrarily close to 1 with $O\big(1/(\sqrt{2^n}\hat{F}_{\min})\big)$ queries, where $\hat{F}_{\min} := \min_w |\hat{F}(w)|$. However, if $\hat{F}(w) = 0$ for some w, then from Eq. (20) we see that $p_{\max} < 1$. In this case one needs an additional amplification step to boost the success probability (a method based on SWAP test was proposed in [28]). We show that this step can be avoided by using t parallel queries in the original QRS algorithm for some $t \leq n$.

5.2 Non-degenerate functions with almost vanishing spectrum

Before explaining our hybrid approach, let us verify that there exist non-trivial functions with a large fraction of their Fourier spectrum equal to zero, so the issue discussed above applies.

It is easy to construct degenerate functions with the desired property. For example, if a function is shift-invariant, i.e., $f(x+s) = f(x)$ for some $s \in \mathbb{Z}_2^n$, then at least half of the Fourier spectrum of f is guaranteed to be zero. The same also happens if $f(x+s) = f(x)+1$ (see Lemma 24 in Sect. D.1). However, such examples are not interesting, since a shift-invariant n-argument Boolean function is equivalent to an $(n-1)$-argument Boolean function (see Sect. D.1 for more details).

Instead, we consider Boolean functions defined using decision trees. A *decision tree* is a binary tree whose vertices are labeled by arguments of f and whose leaves contain the values of f. An example of such tree and the rules for evaluating the corresponding function are given in Fig. 2.

Without loss of generality, we can consider only decision trees where on each path from the root to a leaf no argument appears more than once (otherwise some parts of the tree would not be reachable). The length of a longest path from the root to a leaf is the *height* of the tree. If a Boolean function is defined by a decision tree of height h, then all its Fourier coefficients with Hamming weight larger than h are zero (see Lemma 25 in Sect. D.2). This observation can be used to construct non-degenerate Boolean functions with almost vanishing Fourier spectrum.

▶ **Example.** The 10-argument Boolean function f_{10} whose decision tree is shown in Fig. 2 has no shift invariance, yet 928 (out of $2^{10} = 1024$) of its Fourier coefficients are zero.

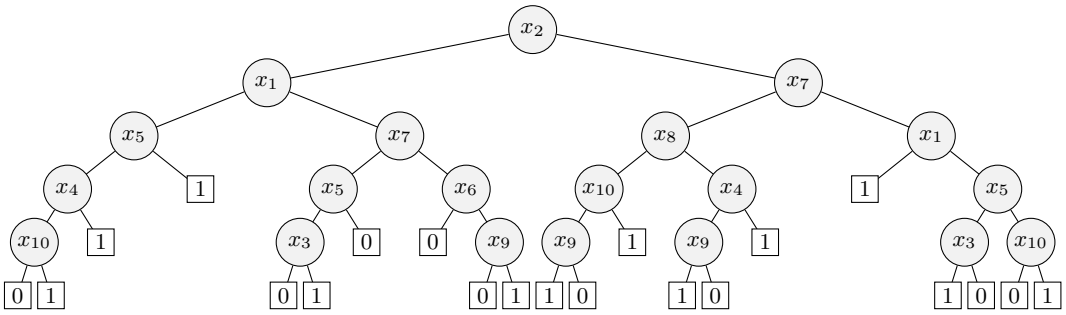

Figure 2 Decision tree for a 10-argument Boolean function f_{10}. To compute the value of the function for given input $x_1, \ldots, x_{10} \in \mathbb{Z}_2^n$, proceed down the tree starting from the root; move left if the corresponding argument is equal to 0 or right if it is equal to 1. Once a leaf is reached, its label is the value of the function for the given input. For example, $f_{10}(x_1, \ldots, x_{10})$ evaluates to zero when $x_2 = x_1 = x_5 = x_4 = x_{10} = 0$, since the leftmost leaf has label zero. This tree has height five.

5.3 The t-fold Fourier spectrum as t increases

Let us now show how to deal with the zero Fourier coefficients. The main idea stems from the following observation: if $S_t := \{w \in \mathbb{Z}_2^n : \mathcal{F}^t(w) \neq 0\}$ then $S_{t+1} = S_t + S_1$ (see Prop. 26 in Sect. D.3). If S_1 spans $\mathbb{Z}_2^n$, we can apply this recursively and eliminate all zeroes from the t-fold Fourier spectrum $\mathcal{F}^t$. In particular, it suffices to take $t \leq n$ (see Lemma 27 in Sect. D.3). For example, for f_{10} the fraction of non-zero values of $\mathcal{F}^t$ for $t = 1, 2, 3, 4$ is 0.09, 0.61, 0.94, 1, respectively. In particular, $\mathcal{F}^4$ is non-zero everywhere.

5.4 Quantum rejection sampling with t-fold queries

We can use quantum rejection sampling with t queries in parallel to solve the BHSP. Suppose we transform the t-fold Fourier state $|\Phi^t(s)\rangle$ from Eq. (8) into the PGM basis vector $|E_s\rangle$ defined in Eq. (15) using QRS. This corresponds to setting $\pi_w = \mathcal{F}^t(w)$ and $\sigma_w = 1/\sqrt{2^n}$. Since the circuit from Fig. 1 can be used to prepare $|\Phi^t(s)\rangle$ with t queries, Theorem 16 still holds if $|\hat{F}(w)|$ is replaced by $\mathcal{F}^t(w)$ and the query complexity is multiplied by t. This observation together with Lemma 27 implies that as long as f is not shift invariant, we can recover the hidden shift s with success probability arbitrarily close to 1 using quantum rejection sampling with some $t \leq n$.

▶ **Theorem 17.** *Let $f : \mathbb{Z}_2^n \to \mathbb{Z}_2$ be a Boolean function and let p be sufficiently large. Then BHSP_f can be solved with success probability p using $O(t/\|\boldsymbol{\varepsilon}^p_{\boldsymbol{\pi}\to\boldsymbol{\sigma}}\|)$ queries for some $t \in \{1, \ldots, n\}$ where $\pi_w := \mathcal{F}^t(w)$, $\sigma_w := 1/\sqrt{2^n}$, and the "water-filling" vector $\boldsymbol{\varepsilon}^p_{\boldsymbol{\pi}\to\boldsymbol{\sigma}} \in \mathbb{R}^{2^n}$ is defined in [28].*

6 Conclusions

A comparison of quantum query complexity bounds for solving the BHSP for different classes of functions is given in Table 1. If the QRS algorithm works for random functions with $O(1)$ queries, then it is optimal up to constant factors in all three cases listed in the table. However, from Sect. 5.1 we know that the basic QRS algorithm without amplification performs poorly when f has many zero Fourier coefficients (which is the case, e.g., for the decision trees considered in Sect. D.2). This suggests that the basic (unamplified) QRS algorithm is likely not optimal in general.

Table 1 Summary of quantum query complexity upper and lower bounds for BHSP. We do not know the query complexity of the QRS algorithm for random functions.

Approach	Functions			Comments
	delta	bent	random	
PGM	$O(2^n)$	1	2	zero error
QRS [28]	$O(\sqrt{2^n})$	1	?	
"Simon" [31]	$O(n\sqrt{2^n})$	$O(n)$	$O(n)$	zero error, black-box f okay
Learning theory [38]	$O(n \log n\sqrt{2^n})$	$O(n \log n)$	$O(n \log n)$	optimal up to log factors $\forall f$
Lower bounds:	$\Omega(\sqrt{2^n})$	1	1	

The "Simon"-type approach due to [31] always has an overhead of a factor $O(n)$, reflecting the fact that at least n linearly independent equations are needed to solve a linear system in n variables. (Note that this approach works in the weaker model where the unshifted function is given by an oracle, so it still provides an upper bound when the function is known explicitly.) The learning theory approach [38] also has logarithmic overhead. Finally, the PGM approach performs very well in the easy cases, the bent and random functions, but fails to provide any speedup for delta functions. As mentioned in Sect. 4.3.1, this can be attributed to the fact that Grover's algorithm is intrinsically sequential.

In summary, none of the algorithms listed in Table 1 is optimal. However, by combining these algorithms and possibly adding some new ideas, one might obtain an algorithm that is optimal for all Boolean functions. In particular, the QRS approach with t-fold queries appears promising.

We conclude by mentioning some open questions regarding the Boolean hidden shift problem:

1. Find a query-optimal quantum algorithm for general functions (recall that the learning theory algorithm is only optimal up to logarithmic factors [37, 38]).
2. Identify natural classes of Boolean functions lying between the two extreme cases of bent and delta functions (say, the decision trees considered in Sect. D.2) and characterize the quantum query complexity of the BHSP for these functions.
3. Determine the number of queries required by the QRS algorithm for random functions.
4. What is the query complexity of verifying a given shift? (A quantum procedure with one-sided error, based on the swap test, was given in [28].)
5. What is the quantum query complexity of extracting one bit of information about the hidden shift?
6. What is the classical query complexity of the Boolean hidden shift problem?
7. Can we say anything non-trivial about the time complexity of the Boolean hidden shift problem, either classically or quantumly?
8. Can the BHSP for random functions be solved with a single query? Our approach based on the PGM only gives a lower bound on the expected success probability that approaches $2/\pi$ for large n (see Theorem 19), whereas we require a success probability that approaches 1 as $n \to \infty$. It might be fruitful to consider querying the oracle with non-uniform amplitudes.

Finally, it might be interesting to consider the generalization of the Boolean hidden shift problem to the case of functions $f\colon \mathbb{Z}_d^n \to \mathbb{Z}_d$.

Acknowledgements. We thank Jérémie Roland for useful discussions and Dmitry Gavinsky for suggesting to use decision trees to construct non-degenerate functions with many zero Fourier coefficients. Part of this work was done while AC and MO were visiting NEC Labs, and during the Quantum Cryptanalysis seminar (No. 11381) at Schloss Dagstuhl. This work was supported in part by NSERC, the Ontario Ministry of Research and Innovation, and the US ARO/DTO. MO acknowledges additional support from the DARPA QUEST program under contract number HR0011-09-C-0047.

References

1 Andrew M. Childs and Wim van Dam. Quantum algorithms for algebraic problems. *Rev. Mod. Phys.*, 82(1):1–52, Jan 2010. arXiv:0812.0380, doi:10.1103/RevModPhys.82.1.

2 David Deutsch and Richard Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 439(1907):553–558, 1992. doi:10.1098/rspa.1992.0167.

3 Daniel R. Simon. On the power of quantum computation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS 1994)*, pages 116–123, Nov 1994. doi:10.1109/SFCS.1994.365701.

4 Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. Earlier version in FOCS 1994, pp. 124–134. arXiv:quant-ph/9508027, doi:10.1137/S0097539795293172.

5 Alexei Kitaev. Quantum measurements and the Abelian Stabilizer Problem. 1995. arXiv:quant-ph/9511026.

6 Richard Jozsa. Quantum algorithms and the Fourier transform. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969):323–337, 1998. arXiv:quant-ph/9707033, doi:10.1098/rspa.1998.0163.

7 Michele Mosca and Artur Ekert. The hidden subgroup problem and eigenvalue estimation on a quantum computer. In *Quantum Computing and Quantum Communications*, volume 1509 of *Lecture Notes in Computer Science*, pages 174–188. Springer, 1999. arXiv:quant-ph/9903071, doi:10.1007/3-540-49208-9_15.

8 Richard Jozsa. Quantum factoring, discrete logarithms, and the hidden subgroup problem. *Computing in Science Engineering*, 3(2):34–43, Mar/Apr 2001. arXiv:quant-ph/0012084, doi:10.1109/5992.909000.

9 Sean Hallgren. Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem. *Journal of the ACM*, 54(1):4:1–4:19, Mar 2007. doi:10.1145/1206035.1206039.

10 Dan Boneh and Richard Lipton. Quantum cryptanalysis of hidden linear functions. In *Advances in Cryptology – CRYPTO 1995*, volume 963 of *Lecture Notes in Computer Science*, pages 424–437. Springer, 1995. doi:10.1007/3-540-44750-4_34.

11 Robert Beals. Quantum computation of Fourier transforms over symmetric groups. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC 1997)*, pages 48–53. ACM, 1997. doi:10.1145/258533.258548.

12 Peter Høyer. Efficient quantum transforms. 1997. arXiv:quant-ph/9702028.

13 Mark Ettinger and Peter Høyer. A quantum observable for the graph isomorphism problem. 1999. arXiv:quant-ph/9901029.

14 Andris Ambainis, Loïck Magnin, Martin Roetteler, and Jérémie Roland. Symmetry-assisted adversaries for quantum state generation. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity (CCC'11)*, pages 167–177. IEEE Computer Society, 2011. arXiv:1012.2112, doi:10.1109/CCC.2011.24.

15 Oded Regev. Quantum computation and lattice problems. *SIAM Journal on Computing*, 33(3):738–760, 2004. arXiv:cs/0304005, doi:10.1137/S0097539703440678.

16 Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing*, 35(1):170–188, 2005. arXiv:quant-ph/0302112, doi:10.1137/S0097539703436345.

17 Oded Regev. A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space. 2004. arXiv:quant-ph/0406151.

18 Greg Kuperberg. Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. 2011. arXiv:1112.3333.

19 Andrew M. Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. 2010. arXiv:1012.4019.

20 Mark Ettinger and Peter Høyer. On quantum algorithms for noncommutative hidden subgroups. *Advances in Applied Mathematics*, 25(3):239–251, 2000. arXiv:quant-ph/9807029, doi:10.1006/aama.2000.0699.

21 Wim van Dam, Sean Hallgren, and Lawrence Ip. Quantum algorithms for some hidden shift problems. *SIAM Journal on Computing*, 36(3):763–778, 2006. arXiv:quant-ph/0211140, doi:10.1137/S009753970343141X.

22 Katalin Friedl, Gábor Ivanyos, Frédéric Magniez, Miklos Santha, and Pranab Sen. Hidden translation and orbit coset in quantum computing. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC 2003)*, pages 1–9. ACM, 2002. arXiv:quant-ph/0211091, doi:10.1145/780542.780544.

23 Cristopher Moore, Daniel Rockmore, Alexander Russell, and Leonard J. Schulman. The power of strong Fourier sampling: Quantum algorithms for affine groups and hidden shifts. *SIAM Journal on Computing*, 37(3):938–958, Jun 2007. arXiv:quant-ph/0503095, doi:10.1137/S0097539705447177.

24 Andrew M. Childs and Pawel Wocjan. On the quantum hardness of solving isomorphism problems as nonabelian hidden shift problems. *Quantum Information and Computation*, 7(5):504–521, Jul 2007. URL: http://www.rintonpress.com/journals/qiconline.html#v7n56, arXiv:quant-ph/0510185.

25 Andrew M. Childs and Wim van Dam. Quantum algorithm for a generalized hidden shift problem. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms (SODA 2007)*, pages 1225–1232. SIAM, 2007. URL: http://dl.acm.org/citation.cfm?id=1283383.1283515, arXiv:quant-ph/0507190.

26 Gábor Ivanyos. On solving systems of random linear disequations. *Quantum Information and Computation*, 8(6&7):579–594, 2008. URL: http://www.rintonpress.com/journals/qiconline.html#v8n67, arXiv:0704.2988.

27 Ivan B. Damgård. On the randomness of Legendre and Jacobi sequences. In *Advances in Cryptology – CRYPTO 1988*, volume 403 of *Lecture Notes in Computer Science*, pages 163–172. Springer, 1990. doi:10.1007/0-387-34799-2_13.

28 Maris Ozols, Martin Roetteler, and Jérémie Roland. Quantum rejection sampling. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS 2012)*, pages 290–308. ACM, 2012. arXiv:1103.2774, doi:10.1145/2090236.2090261.

29 Martin Rötteler. Quantum algorithms to solve the hidden shift problem for quadratics and for functions of large Gowers norm. In *Proceedings of the 34st International Symposium on Mathematical Foundations of Computer Science (MFCS 2009)*, volume 5734 of *Lecture Notes in Computer Science*, pages 663–674. Springer, 2009. arXiv:0911.4724, doi:10.1007/978-3-642-03816-7_56.

30 Martin Rötteler. Quantum algorithms for highly non-linear Boolean functions. In *Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA 2010)*, pages

448–457. SIAM, 2010. URL: http://dl.acm.org/citation.cfm?id=1873601.1873638, arXiv:0811.3208.

31 Dmitry Gavinsky, Martin Roetteler, and Jérémie Roland. Quantum algorithm for the Boolean hidden shift problem. In *Computing and Combinatorics*, volume 6842 of *Lecture Notes in Computer Science*, pages 158–167. Springer, 2011. arXiv:1103.3017, doi:10.1007/978-3-642-22685-4_14.

32 Mirmojtaba Gharibi. Reduction from non-injective hidden shift problem to injective hidden shift problem. *Quantum Information and Computation*, 13(3&4):0221–0230, 2013. URL: http://www.rintonpress.com/journals/qiconline.html#v13n34, arXiv:1207.4537.

33 Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC 1996)*, pages 212–219. ACM, 1996. arXiv:quant-ph/9605043, doi:10.1145/237814.237866.

34 Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997. arXiv:quant-ph/9701001, doi:10.1137/S0097539796300933.

35 Wim van Dam. Quantum algorithms for weighing matrices and quadratic residues. *Algorithmica*, 34(4):413–428, 2008. arXiv:quant-ph/0008059, doi:10.1007/s00453-002-0975-4.

36 Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997. Earlier version in STOC 1993, pp. 11–20. doi:10.1137/S0097539796300921.

37 Rocco A. Servedio and Steven J. Gortler. Equivalences and separations between quantum and classical learnability. *SIAM Journal on Computing*, 33(5):1067–1092, 2004. doi:10.1137/S0097539704412910.

38 Alp Atıcı and Rocco A. Servedio. Improved bounds on quantum learning algorithms. *Quantum Information Processing*, 4(5):355–386, 2005. arXiv:quant-ph/0411140, doi:10.1007/s11128-005-0001-2.

39 Ronald de Wolf. A brief introduction to Fourier analysis on the Boolean cube. *Theory of Computing Library – Graduate Surveys*, 1:1–20, 2008. doi:10.4086/toc.gs.2008.001.

40 Thomas W. Cusick and Pantelimon Stănică. *Cryptographic Boolean Functions and Applications*. Academic Press/Elsevier, 2009. URL: http://books.google.ca/books?id=OAkhkLSxxxMC&pg=PA73.

41 John F. Dillon. A survey of bent functions. *The NSA technical journal*, pages 191–215, 1972.

42 Jessie F. MacWilliams and Neil J.A. Sloane. *The theory of error-correcting codes: Part 2*. North-Holland, 1977. URL: http://books.google.ca/books?id=nv6WCJgcjxcC&pg=PA426.

43 John F. Dillon. Elementary Hadamard difference sets. In *Proceedings of the 6th Southeastern Conference on Combinatorics, Graph Theory, and Computing*, pages 237–249. Utilitas Mathematica Pub., 1975.

44 Hans Dobbertin. Construction of bent functions and balanced Boolean functions with high nonlinearity. In *Fast Software Encryption*, volume 1008 of *Lecture Notes in Computer Science*, pages 61–74. Springer, 1995. doi:10.1007/3-540-60590-8_5.

45 Andris Ambainis, Kazuo Iwama, Akinori Kawachi, Hiroyuki Masuda, Raymond H. Putra, and Shigeru Yamashita. Quantum identification of Boolean oracles. In *Proceedings of the 21st Annual Symposium on Theoretical Aspects of Computer Science (STACS 2004)*, volume 2996 of *Lecture Notes in Computer Science*, pages 105–116. Springer, 2004. arXiv:quant-ph/0403056, doi:10.1007/978-3-540-24749-4_10.

46 Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46(4-5):493–505, 1998. arXiv:quant-ph/9605034, doi:10.1002/(SICI)1521-3978(199806)46:4/5<493::AID-PROP493>3.0.CO;2-P.

47 Paul Hausladen and William K. Wootters. A 'pretty good' measurement for distinguishing quantum states. *Journal of Modern Optics*, 41(12):2385–2390, 1994. doi:10.1080/09500349414552221.

48 Dave Bacon, Andrew M. Childs, and Wim van Dam. From optimal measurement to efficient quantum algorithms for the hidden subgroup problem over semidirect product groups. In *Proceedings of the 46th Annual Symposium on Foundations of Computer Science (FOCS 2005)*, pages 469–478, Oct 2005. arXiv:quant-ph/0504083, doi:10.1109/SFCS.2005.38.

49 Thomas Decker, Jan Draisma, and Pawel Wocjan. Efficient quantum algorithm for identifying hidden polynomials. *Quantum Information and Computation*, 9(3-4):215–254, 2009. URL: http://www.rintonpress.com/journals/qiconline.html#v9n34, arXiv:0706.1219.

50 Christof Zalka. Grover's quantum searching algorithm is optimal. *Physical Review A*, 60:2746–2751, 1999. arXiv:quant-ph/9711070, doi:10.1103/PhysRevA.60.2746.

51 Thomas Koshy. *Catalan Numbers with Applications*. Oxford University Press, 2008. URL: http://books.google.ca/books?id=MqPLSivdBDAC&pg=PA48.

A Converse for bent functions

The goal of this appendix is to prove Theorem 8. First we need an alternative characterization of bent functions.

▶ **Proposition 18.** A Boolean function f is bent if and only if $(F * F)(x) = \delta_{x,0}$.

Proof. If $(F * F)(x) = \delta_{x,0}$, then using identities from Sect. 2, we find

$$\hat{F}^2(w) = \frac{1}{\sqrt{2^n}}\widehat{(F * F)}(w) = \frac{1}{2^n}\sum_{x\in\mathbb{Z}_2^n}(-1)^{w\cdot x}(F * F)(x) = \frac{1}{2^n} \tag{21}$$

so f is bent. Conversely, if f is bent then

$$(F * F)(w) = \sqrt{2^n}\widehat{\hat{F}^2}(w) = \sum_{x\in\mathbb{Z}_2^n}(-1)^{w\cdot x}\hat{F}^2(x) = \sum_{x\in\mathbb{Z}_2^n}(-1)^{w\cdot x}\frac{1}{2^n} = \delta_{w,0} \tag{22}$$

and the result follows. ◀

▶ **Theorem 8.** *Let $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ be a Boolean function with $n \geq 2$. A quantum algorithm can solve* BHSP_f *exactly with a single query to O_{f_s} if and only if f is bent.*

Proof. The most general one-query algorithm for solving BHSP_f using a controlled phase oracle (or equivalently, an oracle that computes the function in a register) performs a query on some superposition of all binary strings $x \in \mathbb{Z}_2^n$ and an extra symbol "$\varnothing$" that allows for the possibility of not querying the oracle. Without loss of generality, the initial state is

$$\alpha_\varnothing|\varnothing\rangle + \sum_{x\in\mathbb{Z}_2^n}\alpha_x|x\rangle \tag{23}$$

for some amplitudes $\alpha_\varnothing \in \mathbb{C}$ and $\alpha_x \in \mathbb{C}$ for $x \in \mathbb{Z}_2^n$ such that $|\alpha_\varnothing|^2 + \sum_{x\in\mathbb{Z}_2^n}|\alpha_x|^2 = 1$. The oracle acts trivially on $|\varnothing\rangle$, so the state after the query is

$$|\phi_s\rangle := \alpha_\varnothing|\varnothing\rangle + \sum_{x\in\mathbb{Z}_2^n}\alpha_x(-1)^{f(x+s)}|x\rangle \tag{24}$$

where $s \in \mathbb{Z}_2^n$ is the hidden shift. For an exact algorithm, we must have

$$\forall\, s \neq s' :\ 0 = \langle \phi_s | \phi_{s'} \rangle = |\alpha_\varnothing|^2 + \sum_{x \in \mathbb{Z}_2^n} |\alpha_x|^2 (-1)^{f(x+s)+f(x+s')}. \tag{25}$$

We can describe Eq. (25) as a linear system of equations. Define $p_\varnothing := |\alpha_\varnothing|^2$ and let p be a sub-normalized probability distribution on $\mathbb{Z}_2^n$ defined by $p_x := |\alpha_x|^2$. Let M be a rectangular matrix with rows labeled by elements of $A := \{(s, s') \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n : s \neq s'\}$ and columns labeled by $x \in \mathbb{Z}_2^n$, with entries

$$M_{ss';x} := (-1)^{f(x+s)+f(x+s')}. \tag{26}$$

Then Eq. (25) is equivalent to

$$Mp = -p_\varnothing u \tag{27}$$

where u is the all-ones vector indexed by elements of A. In other words, there exists an exact one-query quantum algorithm for solving BHSP_f if and only if Eq. (27) holds for some $p_\varnothing$ and p that together form a probability distribution on $\{\varnothing\} \cup \mathbb{Z}_2^n$.

If f is bent, there is an exact one-query quantum algorithm corresponding to $p_\varnothing = 0$ and $p = \mu$, the uniform distribution (i.e., $\mu_x := 1/2^n$ for all $x \in \mathbb{Z}_2^n$). Notice that the entries of the vector $M\mu$ are

$$\begin{aligned}
(M\mu)_{ss'} &= \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} M_{ss';x} && (28)\\
&= \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x+s)+f(x+s')} && (29)\\
&= \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x)+f(x+s+s')} && (30)\\
&= (F * F)(s + s'). && (31)
\end{aligned}$$

Prop. 18 implies that $(F * F)(x) = \delta_{x,0}$, so $(Mp)_{ss'} = 0$ for all $s \neq s'$. Since $p_\varnothing = 0$, Eq. (27) holds and the algorithm is exact.

To prove the converse, assume there is an exact one-query quantum algorithm that solves BHSP_f. Then Eq. (27) holds for some $p_\varnothing$ and p that form a probability distribution on $\{\varnothing\} \cup \mathbb{Z}_2^n$.

First, we claim that without loss of generality, the probabilities p_x can be set equal for all $x \in \mathbb{Z}_2^n$. More precisely, we set $\bar{p} := (1 - p_\varnothing)\mu$ and show that Eq. (27) still holds if we

replace p by $\bar{p}$. Note that $1 - p_\varnothing = \sum_{y \in \mathbb{Z}_2^n} p_{x+y}$ for any $x \in \mathbb{Z}_2^n$, so

$$(M\bar{p})_{ss'} = \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} M_{ss',x} \, (1 - p_\varnothing) \tag{32}$$

$$= \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x+s)+f(x+s')} \sum_{y \in \mathbb{Z}_2^n} p_{x+y} \tag{33}$$

$$= \frac{1}{2^n} \sum_{y \in \mathbb{Z}_2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x+y+s)+f(x+y+s')} p_x \tag{34}$$

$$= \frac{1}{2^n} \sum_{y \in \mathbb{Z}_2^n} \sum_{x \in \mathbb{Z}_2^n} M_{(y+s,y+s'),x} \, p_x \tag{35}$$

$$= \frac{1}{2^n} \sum_{y \in \mathbb{Z}_2^n} (Mp)_{(y+s,y+s')} \tag{36}$$

$$= -p_\varnothing \tag{37}$$

where the last equality follows since p is a solution of Eq. (27). We conclude that $\bar{p}$ is also a solution of Eq. (27), i.e.,

$$(1 - p_\varnothing) M\mu = -p_\varnothing u. \tag{38}$$

Recall from Eqs. (28) to (31) that $(M\mu)_{ss'} = (F * F)(s + s')$, which together with Eq. (38) implies that $(1 - p_\varnothing)(F * F)(s + s') = -p_\varnothing$ for all $s \neq s'$. Clearly, there is no solution with $p_\varnothing = 1$. Thus we have

$$(F * F)(w) = -\frac{p_\varnothing}{1 - p_\varnothing} \leq 0 \tag{39}$$

for any $w \neq 0$. Observe that $(F * F)(w) = \sum_{x \in \mathbb{Z}_2^n} \frac{1}{2^n} (-1)^{f(x)+f(x+w)}$ is an integer multiple of $1/2^n$ and $(F * F)(0) = 1$ for any f. Thus, we can rewrite Eq. (39) as

$$(F * F)(w) = \begin{cases} 1 & \text{if } w = 0, \\ -k/2^n & \text{otherwise} \end{cases} \tag{40}$$

for some integer $k \geq 0$. Therefore

$$\sum_{w \in \mathbb{Z}_2^n} (F * F)(w) = 1 - \frac{2^n - 1}{2^n} k. \tag{41}$$

On the other hand,

$$\sum_{w \in \mathbb{Z}_2^n} (F * F)(w) = \sum_{w \in \mathbb{Z}_2^n} \sum_{x \in \mathbb{Z}_2^n} \frac{1}{2^n} (-1)^{f(x)+f(x+w)} \tag{42}$$

$$= \left[\frac{1}{\sqrt{2^n}} \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x)} \right]^2 \tag{43}$$

$$= \frac{1}{2^n} \left[\sum_{x \in \mathbb{Z}_2^n} \big(1 - 2f(x)\big) \right]^2 \tag{44}$$

$$= \frac{1}{2^n} \big(2^n - 2|f|\big)^2. \tag{45}$$

Putting this together with Eq. (41) gives

$$\left(2^n - 2|f|\right)^2 = 2^n - (2^n - 1)k. \tag{46}$$

This equation has no solutions for $k \geq 2$ since the right-hand side is negative (for $n \geq 2$). Similarly, there are no solutions for $k = 1$ since the left-hand side is even and the right-hand side is odd. Therefore $k = 0$ (and hence $p_\varnothing = 0$), which implies that f is bent by Eq. (40) and Prop. 18. ◀

Note that there is a solution to Eq. (46) with $k = 2$ and $n = 1$, provided $|f| = 1$. This trivial case involves the one-argument Boolean functions $f(x) = x$ and $f(x) = \mathrm{NOT}(x)$. For these functions we can choose $p_\varnothing = 1/2$ and $p_0 = p_1 = 1/4$ to determine the hidden shift exactly with one query. A deterministic classical algorithm can also solve BHSP_f with one query for these functions.

B Success probability of one-query PGM for random functions

In this appendix, we show that for one query, the expected success probability of $\mathbf{PGM}(f, 1)$ approaches a constant less than 1 for large n. This suggests that one query might not be enough to solve the problem with success probability arbitrarily close to 1. However, we do not know if the PGM algorithm has optimal success probability in the one-query case.

▶ **Theorem 19.** *Let f be an n-argument Boolean function chosen uniformly at random and suppose that a hidden shift for f is chosen adversarially. Then $\mathbf{PGM}(f, 1)$ solves BHSP_f with one query to O_{f_s} and expected success probability $\bar{p} \geq 1/2$ over the choice of f. Indeed, $\bar{p} \geq 2/\pi - o(1)$ as $n \to \infty$.*

Proof. Recall from Eq. (16) in Lemma 13 that $\mathbf{PGM}(f, t)$ recovers the hidden shift of f correctly after t queries with success probability $p_f(t)$. If the function f is chosen uniformly at random, then the expected success probability after t queries is

$$\bar{p}(t) := \frac{1}{2^{2^n}} \sum_f p_f(t) = \frac{1}{2^{2^n}} \sum_f \frac{1}{2^n} \left(\sum_{w \in \mathbb{Z}_2^n} \mathcal{F}^t(w) \right)^2. \tag{47}$$

We can obtain a lower bound on $\bar{p}(t)$ using the Cauchy-Schwarz inequality:

$$\bar{p}(t) \geq \frac{1}{2^n} \frac{1}{(2^{2^n})^2} \left(\sum_f \sum_{w \in \mathbb{Z}_2^n} \mathcal{F}^t(w) \right)^2 = 2^n \left(\frac{1}{2^n} \sum_{w \in \mathbb{Z}_2^n} \frac{1}{2^{2^n}} \sum_f \mathcal{F}^t(w) \right)^2 =: \tilde{p}(t). \tag{48}$$

Taking $t = 1$, this gives

$$\bar{p} \geq \frac{1}{2^n} \frac{1}{(2^{2^n})^2} \left(\sum_f \sum_{w \in \mathbb{Z}_2^n} |\hat{F}(w)| \right)^2 \tag{49}$$

$$= \frac{1}{2^n} \left(\frac{1}{2^{2^n}} \sum_{w \in \mathbb{Z}_2^n} \sum_f \left| \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{w \cdot x + f(x)} \right| \right)^2. \tag{50}$$

For each w we can define $f'(x) := w \cdot x + f(x)$ and change the order of summation by summing over f' instead of f. The value of this sum does not depend on w, so we get

$$\bar{p} \geq \frac{1}{2^n} \left(\frac{1}{2^{2^n}} \sum_f \left| \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x)} \right| \right)^2 = \frac{L(2^n)^2}{2^n} \tag{51}$$

where

$$L(N) := \frac{1}{2^N} \sum_{z \in \{1,-1\}^N} \left| \sum_{i=1}^{N} z_i \right| \tag{52}$$

is the expected distance traveled by N steps of a random walk on a line (where each step is of size one and is to the left or the right with equal probability). It remains to lower bound $L(N)$.

Let $N = 2m$ for some integer $m \geq 1$. Using standard identities for sums of binomial coefficients, we compute

$$L(2m) = \frac{1}{2^{2m}} \cdot 2 \sum_{k=0}^{m} (2m - 2k) \binom{2m}{k} \tag{53}$$

$$= \frac{1}{2^{2m}} \cdot 2m \binom{2m}{m}. \tag{54}$$

Since the central binomial coefficient satisfies [51, p. 48]

$$\binom{2m}{m} \geq \frac{4^m}{\sqrt{4m}}, \tag{55}$$

we find

$$L(2m) \geq \sqrt{m}. \tag{56}$$

For $N = 2^n$ this gives $L(2^n) \geq \sqrt{2^n/2}$. We plug this in Eq. (51) and get $\bar{p} \geq 1/2$. In fact, according to Stirling's formula $\binom{2m}{m} \sim 4^m/\sqrt{\pi m}$ as $m \to \infty$. This means that $L(N) \sim \sqrt{2N/\pi}$ as $N \to \infty$ and our lower bound on $\bar{p}$ approaches $2/\pi$ as $n \to \infty$. ◀

C Two queries suffice for random functions

In this appendix we prove the following:

▶ **Theorem 15.** *Let f be an n-argument Boolean function chosen uniformly at random and suppose that a hidden shift for f is chosen adversarially. Then* $\mathbf{PGM}(f, 2)$ *solves* BHSP_f *with expected success probability* $\bar{p} \geq 1 - \frac{3}{64} \cdot 2^{-n}$.

C.1 Strategy

Our goal is lower bound $\tilde{p}(t)$, as defined in Eq. (48). Let us define a random variable X over Boolean functions $f \colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ and binary strings $w \in \mathbb{Z}_2^n$, whose value is

$$X := \left[\mathcal{F}^t(w)\right]^2 = \left[\hat{F}^2\right]^{*t}(w), \tag{57}$$

where f and w are chosen uniformly at random. Notice from Eq. (48) that

$$\tilde{p}(t) = 2^n \left(\mathbb{E}[\sqrt{X}]\right)^2. \tag{58}$$

Clearly, for any $x \geq 0$ we have

$$\mathbb{E}[\sqrt{X}] \geq \sqrt{x} \Pr(X \geq x). \tag{59}$$

Our strategy is to use a one-sided version of Chebyshev's inequality, known as Cantelli's inequality, to lower-bound $\Pr(X \geq x)$, and then choose a value of x that maximizes our lower bound on $\tilde{p}(t)$.

▶ **Fact** (Cantelli's inequality)**.** Let $\mu := \mathbb{E}[X]$ and $\sigma^2 := \mathbb{E}[X^2] - \mu^2$ be the mean and variance of X, respectively. Then $\Pr(X - \mu \geq k\sigma) \geq \frac{1}{1+k^2}$.

Alternatively, if we substitute X by $-X$ and reverse the inequality then

$$\Pr(X \geq \mu - k\sigma) \geq \frac{k^2}{1+k^2}. \tag{60}$$

If we substitute $x := \mu - k\sigma$ in Eq. (59), then according to the above inequality,

$$\mathbb{E}[\sqrt{X}] \geq \sqrt{\mu - k\sigma}\,\frac{k^2}{1+k^2}. \tag{61}$$

Using Eq. (48), Eq. (58), and Eq. (61) gives

$$\bar{p}(t) \geq \tilde{p}(t) = 2^n \big(\mathbb{E}[\sqrt{X}]\big)^2 \geq 2^n(\mu - k\sigma)\left(1 + \frac{1}{k^2}\right)^{-2}. \tag{62}$$

It remains to lower bound μ (Sect. C.2), upper bound σ (Sect. C.3), and make a reasonable choice of the deviation parameter k (Sect. C.4).

C.2 Computing the mean

Let us compute the mean

$$\mu = \mathbb{E}[X] = \frac{1}{2^{2^n}} \sum_f \frac{1}{2^n} \sum_{w \in \mathbb{Z}_2^n} \big[\hat{F}^2\big]^{*t}(w) \tag{63}$$

for any integer $t \geq 1$. Notice that

$$\begin{aligned}
\sum_{w \in \mathbb{Z}_2^n} \big[\hat{F}^2\big]^{*t}(w) &= \sum_{w, y_1, \ldots, y_{t-1} \in \mathbb{Z}_2^n} \hat{F}(y_1)^2 \cdots \hat{F}(y_{t-1})^2 \hat{F}\big(w - (y_1 + \cdots + y_{t-1})\big)^2 && (64)\\
&= \sum_{y_1, \ldots, y_t \in \mathbb{Z}_2^n} \hat{F}(y_1)^2 \cdots \hat{F}(y_{t-1})^2 \hat{F}(y_t)^2 && (65)\\
&= \left(\sum_{y \in \mathbb{Z}_2^n} \hat{F}(y)^2\right)^t && (66)\\
&= 1 && (67)
\end{aligned}$$

by unitarity of the Fourier transform (see Plancherel's identity in Sect. 2). We conclude that

$$\mu = \frac{1}{2^n} \tag{68}$$

independent of t.

C.3 Computing the variance

Next we compute the variance

$$\mathbb{E}[X^2] = \frac{1}{2^{2^n}} \sum_f \frac{1}{2^n} \sum_{w \in \mathbb{Z}_2^n} \Big(\big[\hat{F}^2\big]^{*t}(w)\Big)^2. \tag{69}$$

Note that from Eq. (2) and Plancherel identity we have

$$\sum_{w\in\mathbb{Z}_2^n}\left(\left[\hat{F}^2\right]^{*t}(w)\right)^2 = \sum_{w\in\mathbb{Z}_2^n}\left(\frac{1}{\sqrt{2^n}}\left(\widehat{F*F}\right)^t(w)\right)^2 = \frac{1}{2^n}\sum_{w\in\mathbb{Z}_2^n}(F*F)^{2t}(w). \tag{70}$$

We substitute this in Eq. (69) and get

$$\mathbb{E}[X^2] = \frac{1}{2^{2^n}}\sum_f \frac{1}{2^n}\left(\frac{1}{2^n}\sum_{w\in\mathbb{Z}_2^n}(F*F)^{2t}(w)\right) \tag{71}$$

$$= \frac{1}{2^{2n}}\sum_{w\in\mathbb{Z}_2^n}\frac{1}{2^{2^n}}\sum_f\left(\frac{1}{2^n}\sum_{x\in\mathbb{Z}_2^n}(-1)^{f(x)+f(w+x)}\right)^{2t}. \tag{72}$$

C.3.1 Counting pairings

Let us introduce some combinatorial ideas that will help us to evaluate the sum in Eq. (72).

▶ **Definition 20.** Let S be a finite set and let $l \geq 1$ be an integer. We say that $a_1, a_2, \ldots, a_{2l} \in S$ are *paired* if there exists a permutation π of $\{1, 2, \ldots, 2l\}$ such that $a_{\pi(2i-1)} = a_{\pi(2i)}$ for all $i \in \{1, 2, \ldots, l\}$. Define $\Delta\colon S^{2l} \to \mathbb{Z}_2$ as

$$\Delta(a_1, a_2, \ldots, a_{2l}) := \begin{cases} 1 & \text{if } a_1, a_2, \ldots, a_{2l} \text{ are paired,} \\ 0 & \text{otherwise.} \end{cases} \tag{73}$$

Notice that for $l = 2$ we have $\Delta(a, b, c, d) = \delta_{a,b}\delta_{c,d} + \delta_{a,c}\delta_{b,d} + \delta_{a,d}\delta_{b,c} - 2\delta_{a,b,c,d}$, so the number of ways to pair four elements of S is

$$\sum_{a,b,c,d\in S}\Delta(a,b,c,d) = 3\sum_{a,b,c,d\in S}\delta_{a,b}\delta_{c,d} - 2\sum_{a,b,c,d\in S}\delta_{a,b,c,d} = 3|S|^2 - 2|S|. \tag{74}$$

▶ **Proposition 21.** Let $S = \{0,1\}^n$. Then for any $a_1, a_2, \ldots, a_{2l} \in S$,

$$\frac{1}{2^{2^n}}\sum_f (-1)^{f(a_1)+f(a_2)+\cdots+f(a_{2l})} = \Delta(a_1, a_2, \ldots, a_{2l}) \tag{75}$$

where the sum is over all Boolean functions $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$.

Proof. Clearly, if $a_1, a_2, \ldots, a_{2l}$ are paired, then the exponent of -1 is even and the sum is 1. Otherwise, we can omit the paired arguments, and all remaining a_i are distinct. Since we are averaging over all f and the values that f takes at distinct points are independent, the sum vanishes. ◀

We can use this observation to rewrite Eq. (72) as follows:

$$\mathbb{E}[X^2] = \frac{1}{2^{2(t+1)n}}\sum_{w\in\mathbb{Z}_2^n}\sum_{a_1,\ldots,a_{2t}\in\mathbb{Z}_2^n}\Delta(a_1, a_1+w, a_2, a_2+w, \ldots, a_{2l}, a_{2l}+w). \tag{76}$$

C.3.2 Evaluating the variance at $t = 2$

In general, the variance depends on t. However, we are interested only in the $t = 2$ case, so from now on we will assume that $t = 2$ and do not write the dependence on t explicitly. For $t = 2$, Eq. (76) reads

$$\mathbb{E}[X^2] = \frac{1}{2^{6n}}\sum_{w\in\mathbb{Z}_2^n}\sum_{a,b,c,d\in\mathbb{Z}_2^n}\Delta(a, a+w, b, b+w, c, c+w, d, d+w). \tag{77}$$

We consider two cases. First, when $w = 0$, the eight arguments of Δ are always paired, so the inner sum in Eq. (77) evaluates to

$$\sum_{a,b,c,d \in \mathbb{Z}_2^n} \Delta(a,a,b,b,c,c,d,d) = 2^{4n}. \tag{78}$$

Now suppose $w \neq 0$. Then $w_i = 1$ for some $i \in \{1,\ldots,n\}$ and thus either $a_i = 0$ or $a_i + w_i = 0$ (and similarly for b, c, and d). In total there are $2^4 = 16$ cases. Since Δ is invariant under permutations of arguments, we can substitute a by $a + w$, which effectively swaps the arguments a and $a + w$. By performing a similar operation for b, c, and d, we can ensure that $a_i = b_i = c_i = d_i = 0$. Among the eight arguments of Δ in Eq. (77), arguments a, b, c, and d can be paired only among themselves since $w_i = 1$. Moreover, once a and b are paired, then so are $a + w$ and $b + w$. Thus, we can restrict the ith bit of w to be 1 and ignore the four extra arguments of Δ. Then the inner sum in Eq. (77) becomes

$$16 \sum_{a,b,c,d \in \mathbb{Z}_2^{n-1}} \Delta(a,b,c,d) = 16 \cdot \left(3 \cdot 2^{2n-2} - 2 \cdot 2^{n-1}\right) = 12 \cdot 2^{2n} - 16 \cdot 2^n, \tag{79}$$

where the first equality follows from Eq. (74) with $S = \mathbb{Z}_2^{n-1}$.

By combining Eq. (78) and Eq. (79), we can rewrite Eq. (77) as

$$\mathbb{E}[X^2] = \frac{1}{2^{6n}}\left(2^{4n} + (2^n - 1) \cdot (12 \cdot 2^{2n} - 16 \cdot 2^n)\right) \tag{80}$$

$$= \frac{1}{2^{2n}} + \frac{12}{2^{3n}} - \frac{28}{2^{4n}} + \frac{16}{2^{5n}}. \tag{81}$$

Using the value of μ from Eq. (68), we see that for $n \geq 1$ the variance is

$$\sigma^2 = \mathbb{E}[X^2] - \mu^2 = \frac{12}{2^{3n}} - \frac{28}{2^{4n}} + \frac{16}{2^{5n}} \geq \frac{1}{2^{3n}}. \tag{82}$$

C.4 Choosing the deviation

To complete the lower bound on the success probability, recall from Eq. (62) that

$$\bar{p} \geq 2^n(\mu - k\sigma)\left(1 + \frac{1}{k^2}\right)^{-2}. \tag{83}$$

Substituting the bounds on μ and σ from Eq. (68) and Eq. (82), respectively, gives

$$\bar{p} \geq \left(1 - \frac{k}{\sqrt{2^n}}\right)\left(1 + \frac{1}{k^2}\right)^{-2}. \tag{84}$$

Notice that $\left(1 + \frac{1}{k^2}\right)^{-2} \geq 1 - \frac{2}{k^2}$ for any k, so

$$\bar{p} \geq \left(1 - \frac{k}{\sqrt{2^n}}\right)\left(1 - \frac{2}{k^2}\right) \geq 1 - \frac{k}{\sqrt{2^n}} - \frac{2}{k^2}. \tag{85}$$

It remains to make a good choice for k. Let $\alpha = \sqrt{2^n}$ and $k = \alpha^c$ for some $c > 0$. Then

$$\bar{p} \geq 1 - \alpha^{c-1} - 2\alpha^{-2c}. \tag{86}$$

Choosing $c = 1/3$ (i.e., $k = 2^{n/6}$) gives

$$\bar{p} \geq 1 - \frac{3}{64} \cdot 2^{-n}. \tag{87}$$

This concludes the proof of Theorem 15.

D Zeroes in the Fourier spectrum

D.1 Undetectable shifts and anti-shifts

In some cases the Boolean hidden shift problem cannot be solved exactly in principle. For example, if the function f is invariant under some shift, then the hidden shift cannot be uniquely determined, as the oracle does not contain enough information (an extreme case of this is a constant function which is invariant under all shifts). In this section we consider such degenerate functions and analyze their Fourier spectra.

▶ **Definition 22.** Let $b \in \mathbb{Z}_2$. We say that s is a *b-shift* for a function $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ if f has the following property: $\forall x \in \mathbb{Z}_2^n\colon f(x+s) = f(x) + b$. We refer to 0-shifts as *undetectable shifts* since they cannot be distinguished from the trivial shift $s = 0$. We also refer to 1-shifts as *anti-shifts* since they negate the truth table of f.

The following result provides an alternative characterization of b-shifts. It relates the maximal and minimal autocorrelation value of F to undetectable shifts and anti-shifts of f, respectively (see Definition 5 for the definition of convolution).

▶ **Proposition 23.** The string $s \in \mathbb{Z}_2^n$ is a b-shift for function $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ if and only if $(F * F)(s) = (-1)^b$, where $F(x) := (-1)^{f(x)}/\sqrt{2^n}$ for all $x \in \mathbb{Z}_2^n$.

Proof. Let s be a b-shift of f. Then

$$(F * F)(s) = \sum_{x \in \mathbb{Z}_2^n} F(x)F(x+s) \tag{88}$$

$$= \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{f(x)}(-1)^{f(x)+b} \tag{89}$$

$$= \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{b} \tag{90}$$

$$= (-1)^b. \tag{91}$$

For the converse, note that all terms on the right-hand side of Eq. (88) have absolute value equal to $1/2^n$. In total there are 2^n terms, so $|(F * F)(s)| \leq 1$. If this bound is saturated, then all terms in Eq. (88) must have the same phase. Thus, s is a b-shift for some $b \in \mathbb{Z}_2$. ◀

If s' and s'' are undetectable shifts of f then so is $s'+s''$, since $f(x+s'+s'') = f(x+s') = f(x)$ for any x. Hence the set of all undetectable shifts forms a linear subspace of $\mathbb{Z}_2^n$. Also, if a' and a'' are anti-shifts, then $a' + a''$ is an undetectable shift. In particular, a Boolean function with no undetectable shifts can have at most one anti-shift.

If we want to solve the hidden shift problem for a function f that has an undetectable shift s, we can apply an invertible linear transformation A on the input variables such that $A \cdot 0 \ldots 01 = s$. Thus we simulate the oracle for the function $f'(x) := f(A \cdot x)$ such that $f'(x + 0 \ldots 01) = f'(x)$. Notice that f' is effectively an $(n-1)$-argument function, since it does not depend on the last argument. Similarly, if f has a k-dimensional subspace of undetectable shifts, it is effectively an $(n-k)$-argument function. Solving the hidden shift problem for such a function is equivalent to solving it for the reduced $(n-k)$-argument function f' and picking arbitrary values for the remaining k arguments. In this sense, Boolean functions with undetectable shifts are degenerate and we can consider only functions with no undetectable shifts without loss of generality.

Similarly, if f has an anti-shift, we can use the same construction to show that it is equivalent to a function f' such that $f'(x_1, \ldots, x_{n-1}, x_n) = f''(x_1, \ldots, x_{n-1}) \oplus x_n$ where f'' is an $(n-1)$-argument function. To solve the hidden shift problem for f', we first solve it for f'' and then learn the value of the remaining argument x_n via a single query. In this sense, Boolean functions with anti-shifts are also degenerate. Thus, without loss of generality we can consider the hidden shift problem only for *non-degenerate* functions, i.e., ones that have no b-shifts for any $b \in \mathbb{Z}_2$.

Finally, let us show that Boolean functions with b-shifts have at least half of their Fourier coefficients equal to zero. Let $\mathcal{S}$ be an $(n-1)$-dimensional subspace of $\mathbb{Z}_2^n$, and let us denote the two cosets of $\mathcal{S}$ in $\mathbb{Z}_2^n$ by $\mathcal{S}_b := \mathcal{S} + br$, where $b \in \mathbb{Z}_2$ and $r \in \mathbb{Z}_2^n \setminus \mathcal{S}$ is any representative of the coset for $b = 1$. The following result relates the property of having a b-shift to the property of having zero Fourier coefficients with special structure.

▶ **Lemma 24.** *A function $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ has a non-zero b-shift if and only if there is an $(n-1)$-dimensional subspace $\mathcal{S} \subset \mathbb{Z}_2^n$ such that $\hat{F}(w) = 0$ when $w \notin \mathcal{S}_b$.*

Proof. Assume that s is a b-shift of f. Then

$$\hat{F}(w) = \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{w \cdot x + f(x)} \tag{92}$$

$$= \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{w \cdot (x+s) + f(x+s)} \tag{93}$$

$$= \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{w \cdot (x+s) + f(x) + b} \tag{94}$$

$$= (-1)^{w \cdot s + b} \frac{1}{2^n} \sum_{x \in \mathbb{Z}_2^n} (-1)^{w \cdot x + f(x)} \tag{95}$$

$$= (-1)^{w \cdot s + b} \hat{F}(w). \tag{96}$$

Thus, $\hat{F}(w) = 0$ when $w \cdot s \neq b$. Let $\mathcal{S}$ be the $(n-1)$-dimensional subspace of $\mathbb{Z}_2^n$ orthogonal to s. Then $w \in \mathcal{S}_b \Leftrightarrow w \cdot s = b$ and thus $\hat{F}(w) = 0$ when $w \notin \mathcal{S}_b$.

For the converse, assume that $\mathcal{S}$ is an $(n-1)$-dimensional subspace of $\mathbb{Z}_2^n$ and $\hat{F}(w) = 0$ when $w \notin \mathcal{S}_b$. Let $s \in \mathbb{Z}_2^n$ be the unique non-zero vector orthogonal to $\mathcal{S}$. Then $\mathcal{S}_b = \{w\colon w \cdot s = b\}$ and we have

$$F(x+s) = \hat{\hat{F}}(x+s) \tag{97}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{w \in \mathbb{Z}_2^n} (-1)^{(x+s) \cdot w} \hat{F}(w) \tag{98}$$

$$= \frac{1}{\sqrt{2^n}} \sum_{w \in \mathcal{S}_b} (-1)^{(x+s) \cdot w} \hat{F}(w) \tag{99}$$

$$= (-1)^b \frac{1}{\sqrt{2^n}} \sum_{w \in \mathcal{S}_b} (-1)^{x \cdot w} \hat{F}(w) \tag{100}$$

$$= (-1)^b F(x). \tag{101}$$

Hence $f(x+s) = f(x) + b$ and thus s is a b-shift of f. ◀

D.2 Decision trees

In the previous section we discussed degenerate cases of Boolean functions that have many zero Fourier coefficients. In this section we explain how to construct non-degenerate examples.

▶ **Lemma 25.** *If f is a Boolean function defined by a decision tree of height h then $\hat{F}(w) = 0$ when $|w| > h$.*

Proof. Since the Boolean function f is given by a decision tree, let $\{P_1, \ldots, P_m\}$ be the set of all paths that start at the root of this tree and end at a parent of a leaf labeled by 1. For example, $P_1 = \{x_2, x_1, x_5, x_4, x_{10}\}$ and $P_2 = \{x_2, x_7, x_1\}$ are two such paths for the tree shown in Fig. 2. We can write the disjunctive normal form of f as

$$f(x) = \bigvee_{i=1}^{m} \bigwedge_{j \in P_i} (b_j^{(i)} \oplus x_j) \tag{102}$$

where "$\vee$" and "$\wedge$" represent logical OR and AND functions, respectively, and $b_j^{(i)} \in \mathbb{Z}_2$ is equal to 1 if and only if variable x_j has to be negated on path P_i. For example, x_{10} is negated on P_1, and x_2 and x_7 are negated on P_2.

To prove the desired result about the Fourier coefficients of f, we switch from Boolean functions to (± 1)-valued functions with (± 1)-valued variables. In particular, we replace $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ by a function $\tilde{F}\colon \{1,-1\}^n \to \{1,-1\}$ in variables $X_i \in \{1,-1\}$ such that

$$\tilde{F}\big((-1)^x\big) = (-1)^{f(x)} \tag{103}$$

for all $x \in \mathbb{Z}_2^n$.

Notice that the (± 1)-valued versions of logical NOT, AND, and OR functions are given by the following polynomials:

$$\mathrm{NOT}(X) := -X, \tag{104}$$

$$\mathrm{AND}(X_1, \ldots, X_k) := \quad 1 - 2\prod_{i=1}^{k} \frac{1 - X_i}{2}, \tag{105}$$

$$\mathrm{OR}(X_1, \ldots, X_k) := -1 - 2\prod_{i=1}^{k} \frac{1 + X_i}{2}. \tag{106}$$

We can use these polynomials and Eq. (102) to write $\tilde{F}$ as

$$\tilde{F}(X) = \mathrm{OR}_{i=1}^{m} \mathrm{AND}_{j \in P_i} (-1)^{b_j^{(i)}} X_j, \tag{107}$$

where $\mathrm{OR}_{i=1}^{m} X_i$ stands for $\mathrm{OR}(X_1, \ldots, X_m)$ and a similar convention is used for AND.

When we determine the value of f using a decision tree, each input $x \in \mathbb{Z}_2^n$ leads to a unique leaf of the tree. Thus, when $f(x) = 1$, there is a unique value of i in Eq. (102) for which the corresponding term in the disjunction is satisfied. With this promise we can simplify Eq. (106) to

$$\mathrm{OR}(X_1, \ldots, X_k) := \sum_{i=1}^{k} (X_i - 1) + 1. \tag{108}$$

If we use this in Eq. (107), we get

$$\tilde{F}(X) = \sum_{i=1}^{m} \Big(\mathrm{AND}_{j \in P_i} (-1)^{b_j^{(i)}} X_j - 1 \Big) + 1, \tag{109}$$

$$= 1 - 2\sum_{i=1}^{m} \prod_{j \in P_i} \frac{1 - (-1)^{b_j^{(i)}} X_j}{2}. \tag{110}$$

Notice that this polynomial has degree at most $\max_i |P_i| \leq h$, the height of the tree. On the other hand, the Fourier transform is self-inverse (see Sect. 2), so

$$(-1)^{f(x)} = \sqrt{2^n} F(x) = \sqrt{2^n} \hat{\hat{F}}(x) = \sum_{w \in \mathbb{Z}_2^n} (-1)^{x \cdot w} \hat{F}(w). \tag{111}$$

The (± 1)-valued equivalent of this equation is

$$\tilde{F}(X) = \sum_{w \in \mathbb{Z}_2^n} \hat{F}(w) \prod_{i \colon w_i = 1} X_i. \tag{112}$$

By comparing this with Eq. (110) we conclude that $\hat{F}(w) = 0$ when $|w| > h$. ◀

According to this lemma, we can use the following strategy to construct Boolean functions with a large fraction of their Fourier coefficients equal to zero. We pick a random decision tree with many variables but small height, i.e., large n and small h (notice that $n \leq 2^h - 1$). Then we are guaranteed that the fraction of non-zero Fourier coefficients does not exceed

$$\frac{1}{2^n} \sum_{k=0}^{h} \binom{n}{k} \leq \frac{2^{H(\frac{h}{n})n}}{2^n} = \left(\frac{1}{2^n}\right)^{1 - H(\frac{h}{n})} \tag{113}$$

where $H(p) := -p \log_2 p - (1-p) \log_2 (1-p)$ is the binary entropy function. In particular, if $h \sim \log_2 n$ then this fraction vanishes as n goes to infinity, i.e., $\hat{F}$ is zero almost everywhere.

However, notice that when the number of zero Fourier coefficients is large, it is also more likely to pick a degenerate Boolean function (i.e., one that has a b-shift for some $b \in \mathbb{Z}_2$); we would like to avoid this. Recall from Lemma 24 that f has a b-shift only if all its non-zero Fourier coefficients lie in a coset $\mathcal{S}_b$ of some $(n-1)$-dimensional subspace $\mathcal{S} \subset \mathbb{Z}_2^n$. Unfortunately, we do not know the probability that a random decision tree with n variables and height $\log_2 n$ corresponds to a Boolean function with this property.

D.3 Zeroes in the t-fold Fourier spectrum

In this section we study the fraction of zeroes in the t-fold Fourier spectrum $\mathcal{F}^t$ of f as a function of t. The main observation is Lemma 27, which shows that unless f has an undetectable shift, $\mathcal{F}^t$ becomes non-zero everywhere when t is sufficiently large. This means that even for functions with a high density of zeroes in the Fourier spectrum, one can boost the success probability of the basic quantum rejection sampling approach discussed in Sect. 5.1 by using the t-fold generalization from Sect. 5.4.

▶ **Proposition 26.** Let $S_t := \{w \in \mathbb{Z}_2^n \colon \mathcal{F}^t(w) \neq 0\}$ be the set of strings for which $\mathcal{F}^t$ is non-zero. Then $S_{t+1} = S_t + S_1$ where $A + B := \{a + b \colon a \in A, b \in B\}$.

Proof. Note that $\left[\mathcal{F}^{t+1}\right]^2 = \left[\mathcal{F}^t\right]^2 * \left[\mathcal{F}^1\right]^2$ from Definition 6. Also, $\mathcal{F}^t(w) \geq 0$ for any $t \geq 1$ and $w \in \mathbb{Z}_2^n$. Assume that $w_0 \in S_t$ and $w_1 \in S_1$. Then $\mathcal{F}^t(w_0) > 0$ and $\mathcal{F}^1(w_1) > 0$, so

$$\left[\mathcal{F}^{t+1}\right]^2 (w_0 + w_1) = \sum_{x \in \mathbb{Z}_2^n} \left[\mathcal{F}^t\right]^2 (x) \cdot \left[\mathcal{F}^1\right]^2 (w_0 + w_1 - x) \tag{114}$$

$$\geq \left[\mathcal{F}^t\right]^2 (w_0) \cdot \left[\mathcal{F}^1\right]^2 (w_0 + w_1 - w_0) > 0. \tag{115}$$

Thus $w_0 + w_1 \in S_{t+1}$ and hence $S_t + S_1 \subseteq S_{t+1}$. Conversely, if w cannot be written in the form $w_0 + w_1$ for some $w_0 \in S_t$ and $w_1 \in S_1$ then $\mathcal{F}^{t+1}(w) = 0$, since all terms of the sum in Eq. (114) vanish. ◀

▶ **Lemma 27.** *If $f\colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ does not have an undetectable shift, then there exists $t \in \{1,\ldots,n\}$ such that $\mathcal{F}^t$ is non-zero everywhere.*

Proof. If S_1 spans the whole space $\mathbb{Z}_2^n$, we can inductively apply Prop. 26 to conclude that $S_t = \mathbb{Z}_2^n$ for some sufficiently large t. In particular, it suffices to take $t \leq n$ (say, if S_1 is the standard basis). On the other hand, if S_1 spans only a proper subspace of $\mathbb{Z}_2^n$, then it is contained in some $(n-1)$-dimensional subspace $\mathcal{S}_0$. Since $\mathcal{F}^1 = |\hat{F}|$ vanishes outside of $\mathcal{S}_0$, we conclude by Lemma 24 that f has an undetectable shift. ◀