On quantum computation theory

van Dam, W.K.

Citation for published version (APA):
Chapter 5

Quantum Algorithms and Combinatorics

In this chapter we investigate how we can employ the structure of combinatorial objects like Hadamard matrices and weighing matrices to devise new quantum algorithms. We show how the properties of a weighing matrix can be used to construct a problem for which the quantum query complexity is significantly lower than the classical one. It is pointed out that this scheme captures both Bernstein & Vazirani's inner-product protocol, as well as Grover's search algorithm.

In the second part we consider Paley's construction of Hadamard matrices to design a more specific problem that uses the Legendre symbol \( \chi \) (which indicates if an element of a finite field \( \mathbb{F}_p \) is a quadratic residue or not). It is shown how for a shifted Legendre function \( f_s(x) = \chi(x + s) \), the unknown \( s \in \mathbb{F}_p \) can be obtained exactly with only two quantum calls to \( f_s \). This is in sharp contrast with the observation that any classical, probabilistic procedure requires at least \( k \log p \) queries to solve the same problem.

5.1 Combinatorics, Hadamard and Weighing Matrices

The matrix \( H \) associated with the Hadamard transform is—in the context of quantum computation—called the 'Hadamard matrix'. This terminology is perhaps unfortunate because the same term has already been used in combinatorics to cover a much broader concept. (See the 1893 article by Jacques Hadamard[50] for the origin of this term.)

**Definition 7 (Hadamard matrix in combinatorics)** A matrix \( M \in \{-1,+1\}^{n \times n} \) is called a Hadamard matrix if and only if \( M \cdot M^T = n \cdot I_n \), where \( T \) denotes the transpose of a matrix.

Obviously, when \( M \) is a Hadamard matrix, then \( \frac{M}{\sqrt{n}} \in \mathbb{U}(n) \) is a unitary matrix. The following two standard results are easy to verify.

- If \( M \) is a Hadamard matrix, then the dimension of \( M \) will be 1, 2 or divisible by 4.
• If $M_1$ and $M_2$ are Hadamard matrices, then their tensor product $M_1 \otimes M_2$ is a Hadamard matrix as well.

It is a famous open problem whether or not there exists a Hadamard matrix for every dimension $4k$.

The $H^2_n$ matrices, which we encountered before, form only a small subset of all the Hadamard matrices that we know in combinatorics. Instead, the matrices $\sqrt{2^n}H^2_n$ should perhaps be called “Hadamard matrices of the Sylvester kind” after the author who first discussed this specific family of matrices.[96]

The properties of Hadamard matrices (especially the above mentioned $4k$-question) is an intensively studied topic in combinatorics, and its complexity is impressive given the simple definition.[33, 51, 85, 86, 93] In 1933, Raymond Paley proved the existence of two families of Hadamard matrices that are very different from Sylvester’s $2^n$-construction.

**Fact 8 (Paley construction I and II)**

I: For every prime $p$ with $p = 3 \mod 4$ and every integer $k$, there exists a Hadamard matrix of dimension $(p^k + 1) \times (p^k + 1)$. II: For every prime $p$ with $p = 1 \mod 4$ and every integer $k$, there exists a Hadamard matrix of dimension $(2p^k + 2) \times (2p^k + 2)$.

**Proof:** See the original article [76].

For here it suffices to say that Paley’s construction uses the theory of quadratic residues over finite fields $\mathbb{F}_{p^k}$. We will discuss this topic in Section 5.3 in order to acquire the necessary tools for the construction of the quantum algorithm of Theorem 9.

One can extend the notion of Hadamard matrices by allowing three possible matrix elements $\{-1, +1, 0\}$, while still requiring the $M \cdot M^T \propto I_n$ restriction. We thus reach the following definition.

**Definition 8 (Weighing matrix [33, 85])** A matrix $M \in \{-1, 0, +1\}^{n \times n}$ is called a weighing matrix if and only if $M \cdot M^T = k \cdot I_n$ for some $0 \leq k \leq n$. The set of such matrices is denoted by $W(n, k)$.

By looking at a row of a matrix $M \in \{-1, 0, +1\}^{n \times n}$, we see that $M \cdot M^T = k \cdot I_n$ implies that this row has has $n - k$ zeros, and $k$ entries “$+1$” or “$-1$”. As a result, $W(n, n)$ are the Hadamard matrices again, whereas $W(n, n - 1)$ are called conference matrices. The identity matrix $I_n$ is an example of a $W(n, 1)$ matrix. If $M_1 \in W(n_1, k_1)$ and $M_2 \in W(n_2, k_2)$, then their tensor product $M_1 \otimes M_2$ is an element of $W(n_1n_2, k_1k_2)$. This implies that for every weighing matrix $M \in W(n, k)$ we have in fact a whole family of matrices $M^{\otimes t} \in W(n^t, k^t)$, indexed by $t \in \mathbb{N}$.

**Example 1**

$$
\begin{pmatrix}
  +1 & +1 & +1 & 0 \\
  +1 & -1 & 0 & +1 \\
  +1 & 0 & -1 & -1 \\
  0 & +1 & -1 & +1
\end{pmatrix}^{\otimes t}
$$

is a $W(4^t, 3^t)$ weighing matrix.
The observation that for every $M \in W(n, k)$ the matrix $\frac{1}{\sqrt{k}} \cdot M \in U(n)$ is a unitary matrix makes the connection between combinatorics and quantum computation that we explore in this chapter. In the next section we will see how the mutually orthogonal basis of such a matrix can be used for a query efficient quantum algorithm. The classical lower bound for the same problem is proven using standard, decision tree arguments.

5.2 Quantum Algorithms for Weighing Matrices

In this section we will describe a general weighing-matrix-problem and its quantum solution. But before doing so, we first mention the following state-construction lemma which follows directly from earlier results on Grover's search algorithm.

**Lemma 6 (State construction lemma)** Let $f : \{1, \ldots, n\} \to \{-1, 0, +1\}$ be a black-box function. If we know that $k$ of the function values are "+1" or "-1", and the remaining $n - k$ entries are "0", then the preparation of the state

$$|f\rangle = \frac{1}{\sqrt{k}} \sum_{i=1}^{n} f(i)|i\rangle,$$

requires no more than $\left\lceil \frac{n}{k} \cdot \sqrt{k} \right\rceil + 1$ quantum evaluations of the black-box function $f$. When $k = n$, a single query is sufficient.

**Proof:** First, we use the amplitude amplification process of Grover's search algorithm [48] to create, exactly, the state

$$\frac{1}{\sqrt{k}} \sum_{\substack{i=1 \ldots n \\text{such that} \ f(i) \neq 0}} |i\rangle$$

with $\leq \left\lceil \frac{n}{k} \cdot \sqrt{k} \right\rceil$ queries to $f$. (See the article by Boyer et al. [25] for a derivation of this upper bound. Obviously, no queries are required if $k = n$.) After that, following Fact 2, one additional $f$-call is sufficient to insert the proper amplitudes, yielding the desired state $|f\rangle$.

We will now define the central problem of this chapter, which assumes the existence of a weighing matrix.

**Definition 9 (Weighing matrix problem)** Let $M$ be a $W(n,k)$ weighing matrix. Define a set of $n$ functions $f^M_s : \{1, \ldots, n\} \to \{-1, 0, +1\}$ for every $s \in \{1, \ldots, n\}$ by

$$f^M_s(i) = M_{si}.$$

Given a function $f^M_s$ in the form of a black-box, we want to calculate the parameter $s$. The (probabilistic) query complexity of the weighing matrix problem is the minimum number of calls to the function $f$ that is necessary to determine the value $s$ (with high probability).
With the quantum protocol of Lemma 6 we can solve this problem in a straightforward way.

**Theorem 8 (Quantum algorithm for the weighing matrix problem)**  Given a matrix $M \in W(n, k)$ with the corresponding query problem of Definition 9, there exists a quantum algorithm that exactly determines $s$ with $\left\lceil \frac{n}{4} \sqrt{\frac{n}{k}} \right\rceil + 1$ queries to $f_s^M$. (When $n = k$, the problem can be solved with one query to the function.)

**Proof:** First, prepare the state $|f^M_s\rangle = \frac{1}{\sqrt{k}} \sum_{i=1}^{n} f^M_s(i)|i\rangle$ with $\left\lceil \frac{n}{4} \sqrt{\frac{n}{k}} \right\rceil + 1$ queries to the function $f$. Then, measure the state in the basis spanned by the vectors $|f_1^M\rangle, |f_2^M\rangle, \ldots, |f_n^M\rangle$. Because $M$ is a weighing matrix, this basis is orthogonal and hence the outcome of the measurement gives us the value $s$ (via the outcome $f^M_s$) without error.

For every possible weighing matrix, this result establishes a separation between the quantum and the classical query complexity of the problem, as is shown by the following classical lower bound.

**Lemma 7 (Classical lower bounds for the weighing matrix problem)**  Consider the problem of Definition 9 for a weighing matrix $M \in W(n, k)$. Let $d$ be the number of queries used by a classical algorithm that recovers $s$ with an error probability of $\varepsilon$. Then, this query complexity is bounded from below by

$$d \geq \log_3(1 - \varepsilon) + \log_3 n,$$

$$d \geq (1 - \varepsilon) \frac{n}{k} - \frac{1}{k},$$

$$d \geq \log((1 - \varepsilon)n + n - k) - \log(n - k + 1).$$

(For the case where $k = n$, this lower bound equals $d \geq \log(1 - \varepsilon) + \log n$.)

**Proof:** We will prove these bounds by considering the decision trees that describe the possible classical protocols. The procedure starts at the root of the tree and this node contains the first index $i$ that the protocol queries to the function $f$. Depending on the outcome $f(i) \in \{-1, 0, +1\}$, the protocol follows one of the (three) outgoing edges to a new node $x$, which contains the next query index $i_x$. This routine is repeated until the procedure reaches one of the leaves of the tree. At that point, the protocol guesses which function it has been querying. With this representation, the depth of such a tree reflects the number of queries that the protocol uses, while the number of leaves (nodes without outgoing edges) indicates how many different functions the procedure can distinguish.

For a probabilistic algorithm with error probability $\varepsilon$, we need to have decision trees with at least $(1 - \varepsilon)n$ leaves. Because the number of outgoing edges cannot be bigger than 3, a tree with depth $d$ has maximally $3^d$ leaves. This proves the first lower bound via $3^d \geq (1 - \varepsilon)n$.

For the second and third bound we have to analyze the maximum size of the optimal decision tree as it depends on the values $k$ and $n$. We know that for every index $i_x$, there
are only $k$ different functions with $f(i_x) \neq 0$. This implies that at every node $x$ the joint number of leaves of the two subtrees (associated with the outcomes $f(i_x) = -1$ and $+1$) cannot be bigger than $k$. Hence, by considering the path (starting from the root) along the edges that correspond to the answers $f(i_x) = 0$, we see that a decision tree with $d$ queries, can distinguish no more than $dk + 1$ functions. (Consider for example the case where $k = 1$.) Similarly, we can use the observation that there are exactly $n - k$ functions with $f(i_x) = 0$ for every node $x$. This tells us that a tree with depth $d$ has a maximum number of leaves of $2^d + (2^d - 1)(n - k)$.

The above bounds simplify significantly when we express them as functions of (big enough) $n$. This gives us the following table (note that the quantum complexity holds for the exact solution with $\varepsilon = 0$):

<table>
<thead>
<tr>
<th>$k$</th>
<th>quantum upper bound</th>
<th>classical lower bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>$o(n)$</td>
<td>$[\frac{n}{4} \sqrt[4]{\frac{n}{k}}] + 1$</td>
<td>$(1 - \varepsilon)\frac{n}{4} - O(1)$</td>
</tr>
<tr>
<td>$\Theta(n)$</td>
<td>$O(1)$</td>
<td>$\log n + \log(1 - \varepsilon)$</td>
</tr>
<tr>
<td>$n$</td>
<td>$1$</td>
<td>$\log n + \log(1 - \varepsilon)$</td>
</tr>
</tbody>
</table>

Note that the $n$-dimensional identity matrix is a $W(n, 1)$ weighing matrix, and that for this $I_n$, the previous theorem and lemma are just a restating (with $k = 1$) of the results on Grover's search algorithm for exactly one matching entry. The algorithm of Bernstein & Vazirani is also captured by the above as the case where $k$ has the maximum value $k = n$ (with the weighing matrices $(\sqrt{2} \cdot H)^{\otimes t} \in W(2^t, 2^t)$). Hence we can think of those two algorithms as the extreme instances of the more general weighing matrix problem.

As we phrased it, a weighing matrix $M \in W(n, k)$ gives only a input-size specific problem for which there is a classical/quantum separation, but not a problem that is defined for every input size $N$, as is more customary. We know, however, that for every such matrix $M$, the tensor products $M^{\otimes t}$ are also $W(n^t, k^t)$ weighing matrices (for all $t \in \mathbb{N}$). We therefore have the following direct consequence of our results.

**Lemma 8** Every weighing matrix $M \in W(n, k)$ leads—via the set of matrices $M^{\otimes t} \in W(n^t, k^t)$—to a weighing matrix problem for $N = n^t$ and $K = k^t = N^{\log n}$. By defining $\gamma = 1 - \frac{1}{2} \log 3 \approx 0.21$, we have, for every suitable $N$, a quantum algorithm with query complexity $\frac{\pi}{4} \sqrt{N^\gamma}$ for which there is a classical, probabilistic lower bound of $(1 - \varepsilon) \cdot N^\gamma$.

**Example 2** Using the $W(4^t, 3^t)$ weighing matrices of Example 1, we have $\gamma = 1 - \frac{1}{2} \log 3 \approx 0.21$, and hence a quantum algorithm with query complexity $\frac{\pi}{4} N^{0.10\ldots}$. The corresponding classical probabilistic lower bound of this problem is $(1 - \varepsilon) \cdot N^{0.21\ldots}$.

A legitimate objection against the weighing-matrix-problem is that it does not seem to be very useful (besides the known boundary cases $k = 1$ and $k = n$). In order to obtain more natural problems one can try to look into the specific structure that constitutes the weighing matrix or matrices. An example of such an approach will be
given in the next two sections via Paley’s construction of Hadamard matrices. We will see how this leads to the definition of a problem about quadratic residues of finite fields with a quantum solution that is more efficient than any classical protocol.

5.3 Quadratic Residues of Finite Fields

This section describes some standard results about quadratic residues and Legendre symbols over finite fields. Readers familiar with this topic can safely skip the next paragraphs and continue with Section 5.6. For more background information one can look up references like [32] or [57].

5.4 Finite Field Factoids

From now on $p$ denotes an odd prime. It is known that there always exists a generator $\zeta$ for the multiplicative group $\mathbb{F}_{p^k}^* = \mathbb{F}_{p^k} \setminus \{0\}$. [32, 57] This means that the sequence $\zeta, \zeta^2, \zeta^3, \ldots$ will generate all non-zero elements of $\mathbb{F}_{p^k}^*$. As this is a set of size $p^k - 1$, it follows that $\zeta^{p^k} = \zeta$, and hence $\zeta^{(p^k - 1)} = 1$. Hence we have the equality

$$\zeta^i = \zeta^j \quad \text{if and only if} \quad i = j \mod (p^k - 1) \quad (5.1)$$

for every integer $i$ and $j$.

We now turn our attention to the definition of the generalized Legendre symbol.[32]

Definition 10 (Legendre symbol over finite fields) For every finite field $\mathbb{F}_p$, with $p$ an odd prime, the Legendre symbol-function $\chi : \mathbb{F}_p^* \to \{-1, 0, +1\}$ indicates if a number is a quadratic residue or not, and is thus defined by

$$\chi(x) := \begin{cases} 
0 & \text{if } x = 0 \\
+1 & \text{if } \exists y \neq 0 : y^2 = x \\
-1 & \text{if } \forall y : y^2 \neq x.
\end{cases}$$

By Equation 5.1, the quadratic expression $(\zeta^j)^2 = \zeta^{2j} = \zeta^i$ is correct if and only if $2j = i \mod p^k - 1$. As $p$ is odd, $p^k - 1$ will be even, and hence there can only exist a $j$ with $(\zeta^j)^2 = \zeta^i$ when $i$ is even. Obviously, if $i$ is even, then $\zeta^j$ with $j = \frac{i}{2}$ gives a solution to our quadratic equation. This proves that 50% of the elements of $\mathbb{F}_{p^k}^*$ are a quadratic residue with $\chi(x) = +1$, while the other half has $\chi(x) = -1$. In particular, $\chi(\zeta^i) = (-1)^i$, and hence for the total sum of the function values: $\sum_x \chi(x) = 0$.

5.5 Multiplicative Characters over Finite Fields

The rule $\chi(\zeta^i) \cdot \chi(\zeta^j) = \chi(\zeta^{i+j})$, in combination with $\chi(0) = 0$, shows that the Legendre symbol $\chi$ is a multiplicative character with $\chi(x) \cdot \chi(y) = \chi(xy)$ for all $x, y \in \mathbb{F}_{p^k}^*$.
5.5. Multiplicative Characters over Finite Fields

Definition 11 (Multiplicative characters over finite fields) The function \( \chi : \mathbb{F}_{p^k} \to \mathbb{C} \) is a multiplicative character if and only if \( \chi(xy) = \chi(x)\chi(y) \) for all \( x, y \in \mathbb{F}_{p^k} \). The constant function \( \chi(x) = 1 \) is called the trivial character. (We do not consider the other trivial function \( \chi(x) = 0 \).)

See [32, 57] for the usage of multiplicative characters in number theory. They have the following elementary properties, which we present without proof:

- \( \chi(1) = 1 \),
- for all nonzero \( x \), the value \( \chi(x) \) is a \((p^k - 1)\)th root of unity,
- if \( \chi \) is nontrivial, we have \( \chi(0) = 0 \),
- the inverse of nonzero \( x \) obeys \( \chi(x^{-1}) = \chi(x)^{-1} = \chi(x)^* \),
- \( \sum_x \chi(x) = 0 \) for nontrivial \( \chi \).

The remainder of this section is used to prove a 'near orthogonality' property, typical for nontrivial characters, which will be the crucial ingredient of the quantum algorithm of the next section.

Lemma 9 (Near orthogonality of shifted characters) Consider a nontrivial character \( \chi : \mathbb{F}_{p^k} \to \mathbb{C} \). For the 'complex inner product' between two \( \chi \)-s that are shifted by \( s \) and \( r \in \mathbb{F}_{p^k} \) it holds that

\[
\sum_{x \in \mathbb{F}_{p^k}} \chi(x + r)^* \chi(x + s) = \begin{cases} p^k - 1 & \text{if } s = r \\ -1 & \text{if } s \neq r. \end{cases}
\]

Proof: Rewrite

\[
\sum_{x \in \mathbb{F}_{p^k}} \chi(x + r)^* \chi(x + s) = \sum_{x \in \mathbb{F}_{p^k}} \chi(x)^* \chi(x + \Delta)
\]

with \( \Delta = s - r \). If \( s = r \) this sum equals \( p^k - 1 \). Otherwise, we can use the fact that \( \chi(x)^* \chi(x + \Delta) = \chi(1 + x^{-1}\Delta) = \chi(\Delta)\chi(\Delta^{-1} + x^{-1}) \) (for \( x \neq 0 \)) to reach

\[
\sum_{x \in \mathbb{F}_{p^k}} \chi(x)^* \chi(x + \Delta) = \chi(\Delta) \sum_{x \in \mathbb{F}_{p^k}^*} \chi(\Delta^{-1} + x^{-1}).
\]

Earlier we noticed that \( \sum_x \chi(x) = 0 \), and therefore in the above summation (where the value \( x = 0 \) is omitted) we have \( \sum_{x} \chi(x^{-1} + \Delta^{-1}) = -\chi(\Delta^{-1}) \). This confirms that indeed

\[
\chi(\Delta) \sum_{x \in \mathbb{F}_{p^k}^*} \chi(x^{-1} + \Delta^{-1}) = -1,
\]

which finishes the proof. \( \square \)

We will use this lemma in the setting where the character is the earlier described Legendre symbol.
5.6 The shifted Legendre Symbol Problem

Raymond Paley used the near orthogonality property of the Legendre symbol for the construction of his Hadamard matrices.[76] Here we will use the same property to describe a problem that, much like the above weighing matrix problem, has a gap between its quantum and its classical query complexity. In light of Theorem 8 and Lemma 7 the results of this section are probably not very surprising. Rather, we wish to give an example of how we can borrow the ideas behind the construction of combinatorial objects to design new quantum algorithms. In this case this is done by stating a problem that uses the Legendre symbol over finite fields.

**Definition 12 (Shifted Legendre Symbol Problem)** Assume that we have a black-box for a shifted Legendre function \( f_s : \mathbb{F}_p \rightarrow \{-1, 0, +1\} \) that obeys
\[
f_s(x) = \chi(x + s),
\]
with the—form unknown—shift parameter \( s \in \mathbb{F}_p \). (Recall Definition 10 for a description of \( \chi \).) The task is to determine the value \( s \) with a minimum number of calls to the function \( f \).

First we will prove a lower bound for the classical query complexity of this problem. This proof is almost identical to the lower bounds of Lemma 7 for the weighing matrix problem.

**Lemma 10 (Classical lower bound for the SLS problem)** Assume a classical algorithm that tries to solve the shifted Legendre symbol problem over a finite field \( \mathbb{F}_p \). To determine the requested value \( s \) with a maximum error rate \( \varepsilon \), requires more than \( kp \log p + \log(1 - \varepsilon) - 1 \) queries to the function \( f_s \).

**Proof:** For every index \( i \), there is exactly one function with \( f(i) = 0 \). For the decision tree of a classical protocol this implies that every node \( x \) can only have two proper subtrees (corresponding to the answers \( f(i) = 1 \) and \( -1 \)) and one deciding leaf (the case \( f(-i)(i) = 0 \)). Hence, a decision tree of depth \( d \) can distinguish no more than \( 2^{d+1} - 1 \) different functions. In order to be able to differentiate between \((1 - \varepsilon)p^k\) functions, we thus need a depth \( d \) of at least \( \log((1 - \varepsilon)p^k - 1) \).

The next theorem shows us how—with a quantum computer—we can recover \( s \) exactly with only two queries.

**Theorem 9 (Two Query Quantum Algorithm for the SLS Problem)** For any finite field \( \mathbb{F}_p \), the problem of Definition 12 can be solved exactly with two quantum queries to the black-box function \( f_s \).

**Proof:** We exhibit the quantum algorithm in detail. We start with the superposition
\[
|\text{start}\rangle = \frac{1}{\sqrt{p^k + 1}} \left( \sum_{x \in \mathbb{F}_p} |x\rangle |0\rangle \right) + \frac{1}{\sqrt{p^k + 1}} |\text{dummy}\rangle |1\rangle.
\]
5.6. The shifted Legendre Symbol Problem

(The reason for the "dummy" part of state that we use will be clear later in the analysis.)

The first oracle call is used to calculate the different $\chi$ values for the non-dummy states, giving

$$\begin{align*}
|\text{start}\rangle & \rightarrow \frac{1}{\sqrt{p^k + 1}} \left( \sum_{x \in \mathbb{F}_{p^k}} |x\rangle |f_s(x)\rangle \right) + \frac{1}{\sqrt{p^k + 1}} |\text{dummy}\rangle |1\rangle \\
& = \frac{1}{\sqrt{p^k + 1}} \left( \sum_{x \in \mathbb{F}_{p^k}} |x\rangle |\chi(x + s)\rangle \right) + \frac{1}{\sqrt{p^k + 1}} |\text{dummy}\rangle |1\rangle.
\end{align*}$$

At this point, we measure the rightmost register to see if it contains the value "zero". If this is indeed the case (probability $\frac{1}{p^k + 1}$), the state has collapsed to $| - s\rangle |0\rangle$ which directly gives us the desired answer $s$. Otherwise, we continue with the now reduced state

$$\begin{equation}
\frac{1}{\sqrt{p^k}} \left( \sum_{x \in \mathbb{F}_{p^k} \setminus \{-s\}} |x\rangle |\chi(x + s)\rangle \right) + \frac{1}{\sqrt{p^k}} |\text{dummy}\rangle |1\rangle,
\tag{5.2}
\end{equation}$$

on which we apply a conditional phase change (depending on the $\chi$ values in the rightmost register). We finish the computing by 'erasing' this rightmost register with a second call to $f_s$. (For the dummy part, we just reset the value to "zero''.) This gives us the final state $\psi$, depending on $s$, of the form

$$|\psi_s\rangle |0\rangle = \frac{1}{\sqrt{p^k}} \left( \sum_{x \in \mathbb{F}_{p^k}} \chi(x + s) |x\rangle \right) |0\rangle + \frac{1}{\sqrt{p^k}} |\text{dummy}\rangle |0\rangle.$$

(Notice how the $\chi(x + s)$ amplitude is zero for the missing entry $x = -s$ in the summation over $\mathbb{F}_{p^k}$.)

What is left to show is that $\{|\psi_s\rangle |s \in \mathbb{F}_{p^k}\}$ forms a set of orthogonal vectors. Lemma 9 tells us that for the inner product between two states $\psi_s$ and $\psi_r$ it holds that

$$\langle \psi_s | \psi_s \rangle = \frac{1}{p^k} \left( \sum_{x \in \mathbb{F}_{p^k}} \chi(x + r)^* \chi(x + s) \right) + \frac{1}{p^k}$$

$$= \begin{cases} 
1 & \text{if } s = r \\
0 & \text{if } s \neq r.
\end{cases}$$

In other words, the states $\psi_s$ for $s \in \mathbb{F}_{p^k}$ are mutually orthogonal. Hence, by measuring the final state in the $\psi$-basis, we can determine without error the shift factor $s \in \mathbb{F}_{p^k}$ after only two oracle calls to the function $f_s$. \hfill \Box

More recently, Peter Høyer has shown the existence of a one query protocol for the same problem.[private communication]
The above algorithm only reduces the query complexity to $f_s$. The time complexity of the protocol is another matter, as we did not explain how to perform the final measurement along the $\psi$ axes in a time-efficient way. In a recent article [37] it is shown how one can implement the unitary mapping

$$|s\rangle \rightarrow \frac{1}{\sqrt{p^k}} \left( \sum_{x \in \mathbb{F}_p} \chi(x + s)|x\rangle \right) + \frac{1}{\sqrt{p^k}} |\text{dummy}\rangle$$

with an efficient quantum circuit of depth $\text{polylog}(p^k)$.

## 5.7 Conclusion

We have established a connection between the construction of weighing matrices in combinatorics, and the design of new quantum algorithms. It was shown how every weighing matrix leads to a query problem that has a more efficient quantum solution than is possible classically.

Using the structure of quadratic residues over finite fields, we gave an explicit example of a task with constant quantum query complexity, but logarithmic classical query complexity.

The implicit goal of this chapter was to suggest new possibilities for the construction of useful quantum algorithms. Other results on Hadamard matrices that are especially interesting in this context are, for example, the complex Hadamard matrices of Turyn[100] and the Hadamard matrices of the dihedral group type[61, 90].