Analysis of a few Quantum Algorithms and Circuits related to Boolean Functions

A THESIS

submitted in partial fulfillment of the requirements
for the award of the degree of

Doctor of Philosophy

in

Computer Science

by

Suman Dutta

under the supervision of

Prof. Subhamoy Maitra

APPLIED STATISTICS UNIT
INDIAN STATISTICAL INSTITUTE, KOLKATA

JULY 2025

To My Family

Acknowledgments

Over the course of my six-year-long PhD journey, I have crossed paths with hundreds of individuals who, directly or indirectly, have influenced my thoughts, beliefs, motivation, inspirations, philosophies, and indeed, this thesis. Naming each one of them would take up half the thesis itself. Instead, I wish to express my heartfelt gratitude to all of them collectively, without whom this journey would not have been the same, and my learning would have remained incomplete.

Even before formally beginning my PhD, I faced the familiar confusion about whether to pursue it. I still remember a reassuring conversation with a senior from my previous institute who encouraged me to go ahead, and a professor who patiently guided me through transitioning into a PhD in computer science with a mathematics background. They may not remember these moments, but they profoundly influenced the path that brought me here.

In 2019, I was one of twenty research fellows embarking on our doctoral journey in computer science at the prestigious Indian Statistical Institute, Kolkata. The coursework phase taught us the value of collaborative effort, particularly through those crucial assignment deadlines. After that phase, we branched into diverse research areas. Whenever doubts crept in, knowing that some of the brightest minds from across the country were also walking this path was deeply reassuring. Some of them have already earned their degrees; some are still continuing. As I prepare to submit my thesis, I sincerely hope our paths will cross again and that we may collaborate in the future.

Speaking of collaboration, I have had the privilege of working with over twenty coauthors across all my published and preprint papers. Many of them were senior researchers, while some were junior colleagues. Each brought something unique to the collaboration – some inspired me to reach higher, others grounded me with wisdom and perspective; some were exceptionally patient, and others tirelessly reviewed multiple drafts. I have come to appreciate that in research, it's not about having "good" or "bad" collaborators – it's about finding those whose wavelength resonates with yours, the ones who make even looming deadlines feel manageable. I look forward to many such collaborations in the future.

Writing the first couple of papers was particularly daunting. During those times, I was fortunate to be surrounded by supportive colleagues, especially seniors, whose encouragement and support made the process both bearable and deeply enriching. Those frequent tea breaks and long hours of discussions may have occasionally delayed work, but they played a crucial role in my growth and helped me form lasting friendships beyond academic colleagues.

I must also mention the incredible community at the research scholar hostel, where

I spent some of the most memorable moments of my life. From complaining about mess food to cooking meals together, from joking about opening a restaurant someday to spontaneous late-night singing, dancing, and endless conversations – these moments became my emotional anchor. Whether it was venting about paper rejections, research pressure, institute bureaucracy, or just life in general, I found comfort in a group that listened without judgment. They are exactly the kind of people every scholar deserves during the uncertain and trying days of a PhD. I will miss them deeply. Long after we part ways, the memories we created will continue to bring a smile to my face.

I learned early on within the academic community that choosing the right advisor can define your PhD experience. In this regard, I have been truly privileged, not only to have a thesis supervisor but also a friend, philosopher, and lifelong mentor. Whether academic or personal, he was often the first person I turned to in times of trouble. Living on the same campus offered the rare privilege to engage in hours of meaningful discussions, debates, and informal gatherings – all of which deeply enriched my intellectual and personal growth.

They say the journey matters more than the destination, and every journey is made better with the right companion. I have been blessed with a kind, understanding, and beautiful partner who has stood by me gracefully through all the uncertainties of PhD life. During setbacks, she was my refuge; when I was ready again, she gently encouraged me forward. Her presence has been a quiet yet powerful force throughout this journey.

Finally, I owe everything to my family – the most loving and supportive people one could ask for. While they may not fully understand what Boolean functions are or how quantum computing relates to them, every time I got my paper accepted, they were happier than me. To avoid long daily commutes, I stayed in the hostel for most of my PhD. As a result, I couldn't spend as much time with them in recent years. Yet, they never complained. I know they understand, and I know they love me.

As this chapter of my life draws to a close, I carry with me not only a thesis but also years of growth, resilience, friendships, and memories. For all that I've received – guidance, support, laughter, and love – I remain deeply grateful.

Abstract

Boolean functions are fundamental to computation and presently play a crucial role in quantum information processing. This thesis presents two facets of Boolean functions in the context of quantum computing: (I) extending and applying the theoretical framework of Forrelation to cryptographic analysis, and (II) designing efficient quantum circuits for multi-controlled Toffoli gates and Boolean circuit implementations.

Given two Boolean functions f and g, Forrelation, introduced by Aaronson (2010), measures the correlation between the truth table of f and the Walsh-Hadamard transform of g at the corresponding points. Here, we revisit the Forrelation framework to study several cryptographically significant spectra of Boolean functions, namely, the Walsh-Hadamard, the crosscorrelation, and the autocorrelation spectra.

We begin with adapting the 3-fold Forrelation formulation to sample the Walsh transform values, achieving a constant-factor improvement in query complexity over the Deutsch-Jozsa algorithm. This has direct applications in resiliency checking. Building on this, we propose a technique to estimate the crosscorrelation (and thereby the autocorrelation) at any desired point. Furthermore, we present, to the best of our knowledge, the first constant-query algorithm for sampling from the complete spectrum of crosscorrelation (and consequently, autocorrelation) using Forrelation. Next, we introduce nega-Forrelation, a variant based on the nega-Hadamard transform, and develop efficient quantum algorithms to estimate and sample from the nega-Hadamard and nega-crosscorrelation spectra. We further connect the hidden shift finding algorithm for bent functions (Rötteler, 2010) with the Forrelation algorithm and extend it to the case of negabent functions. Later, we generalize these cryptographic spectra and the Forrelation formulations for any natural number m, identifying the Forrelation and nega-Forrelation as special cases, and propose quantum algorithms towards their estimation.

In a related direction, we focus on the efficient implementation of multi-controlled Toffoli (MCT) gates, specifically optimizing the T depth, which is a crucial parameter for reducing circuit latency on near-term quantum devices with limited coherence times. We present an explicit trade-off between the Toffoli depth and the number of clean ancilla qubits, extending the conditionally clean ancilla technique. Further, we prove that the T depth in the Clifford+T decomposition of an n-MCT gate, via Toffoli, is lower bounded by $\lceil \log_2 n \rceil$, achieved through a complete binary tree structure. Later, we generalize the result to show that any arbitrary n-input, m-output Boolean function (specified by its Algebraic Normal Form) can be implemented with a quantum circuit having optimal T depth $\lceil \log_2 k \rceil$, where k represents the algebraic degree of the function, which is upper bounded by the number of input variables n. This result has significant implications in S-box design and quantum implementations of block ciphers, such as AES.

Keywords. Autocorrelation, Boolean function, Crosscorrelation, Forrelation, Nega-Forrelation, Quantum Algorithms, Quantum Circuits, T depth, Toffoli decomposition, Walsh-Hadamard transform

List of Publications

Journal publications included in the thesis

- **S. Dutta**, S. Maitra, and C. S. Mukherjee. *Following Forrelation - Quantum Algorithms in Exploring Boolean Functions' Spectra*. Advances in Mathematics of Communication (AMC), 2024. doi:10.3934/amc.2021067
- **S. Dutta** and S. Maitra. *Introducing Nega-Forrelation: Quantum Algorithms in Analyzing Nega-Hadamard and Nega-crosscorrelation Spectra*. Workshop on Coding and Cryptography (WCC), 2022. (Extended in) Design, Codes and Cryptography (DCC), 2024. doi:10.1007/s10623-023-01346-x
- **S. Dutta**, S. Wang, A. Baksi, A. Chattopadhyay, and S. Maitra. *Exact space-depth trade-offs in multicontrolled Toffoli decomposition*. Physical Review A (PRA), 2025. doi:10.1103/PhysRevA.111.052611

Preprints uploaded in arXiv

- **S. Dutta**, S. Maitra, and P. Stanica. *Extending Forrelation: Quantum Algorithms Related to Generalized Fourier-Correlation*, 2025. arXiv: 2507.07231
- **S. Dutta**, A. Basu Bhaumik, A. Chattopadhyay, and S. Maitra. *Optimal T depth quantum circuits for implementing arbitrary Boolean functions*, 2025. arXiv: 2506.01542

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this thesis, we study several combinatorial and algorithmic issues related to Boolean functions where the computing model is quantum. It is needless to mention that the Boolean functions are the most fundamental components in the domain of computing and communication science. There are analog quantum computers, for example, differentiator or integrator, that are primarily designed with operational amplifiers and based on active components like transistors and passive elements like resistances, capacitors, and inductors. However, the advent of digital circuits based on Boolean algebra revolutionized the world of computation. That is where the Boolean functions contributed at the most fundamental level of digital computers. The zeros and ones became the language of discrete mathematics. The theoretical work of George Boole appeared in the mid-nineteenth century, and the early twentieth century had seen the development of theoretical computer science spearheaded by Alan Turing, John von Neumann, and many other eminent scientists. Since then, we have experienced the tremendous development of computing and communication for more than a century.

The early twentieth century has also seen the fundamental concepts of quantum physics pioneered by Niels Bohr, Erwin Schrödinger, Werner Heisenberg, Max Born, Paul Dirac, and other famous physicists. However, the computational model received a formal treatment in the early eighties only. Then, in quick succession, the power of quantum computation could be understood by the works of Deutsch-Jozsa [27], Simon [88], Shor [87], and Grover [49]. In particular, the work of Shor [87] in the nineties could demonstrate that given a quantum computer, factorization can be efficiently executed in polynomial time. This result had a devastating effect on classical public key cryptosystems that are based on the hardness of factorization or discrete log problems. Fortunately, commercial quantum computers at some reasonable cost are still elusive. However, it is now very clear that one requires a new set of algorithms to be used in the public key paradigm once the Y2Q (the year to achieve a reasonably performing quantum computer) arrives.

It should be noted that the works of Deutsch-Jozsa [27], Simon [88], Shor [87], and

Grover [49] involve fundamental ideas related to Boolean functions. These works have shown the quantum supremacy over the classical model. In this direction, Aaronson et. al. have shown that the "Forrelation" formulation related to a Boolean function can achieve further separation between quantum and classical domains. This concept was first introduced in [1] and then was used to obtain the seminal result [2] related to quantum supremacy. More formally, it has been shown that the Forrelation problem has constant versus exponential query complexity separation in the bounded error quantum and the probabilistic classical model [2]. Our work is motivated by this Forrelation problem, which considers the correlation between the Walsh-Hadamard Transform of a Boolean function and its Truth Table. While we do not get into complexity theoretic treatment, we have looked into the Forrelation-related techniques provided by Aaronson et. al. and studied them in depth towards application in the study of Walsh as well as autocorrelation spectra of Boolean functions. The first three contributory chapters of this thesis consider the problems in this direction, which we will describe in more detail in Section 1.1.

It is well known that the actual implementations of the Boolean functions require logic gates. The most fundamental result in this direction explains how any Boolean function on a large number of variables can be implemented using certain kinds of small gates. For example, the set of two-input one-output AND, OR gates, and one-input one-output NOT gate (required many) can be used to realize any combinational circuit. It is also known that such gates can be implemented only by using two-input, one-output NAND or NOR gates. Consequently, any Boolean function can be implemented using only NAND or NOR gates, and these are known as universal gates.

The bits in the classical domain get generalized to the quantum domain as qubits. We will discuss the qubits in more detail in the background chapter (Chapter 2). In the quantum domain, an n-input n-output quantum gate can be seen as a $2^n \times 2^n$ complex unitary matrix. It is also known by the Solovay-Kitaev theorem (we refer to the famous book [71] for an in-depth understanding of quantum computation and information) that any quantum gate can be approximated with arbitrary precision by a sequence of gates from a universal gate set, and this approximation can be found efficiently. One such example in the quantum domain is the set of T, Hadamard, phase, and CNOT gates. Another popular gate that can be accommodated in the universal set could be the multi-controlled Toffoli, which can be implemented using Clifford+T gates. In this regard, one important question is quantifying the trade-off between the Toffoli depth and the number of clean ancilla qubits. Quantum circuits with less depth are always recommendable as the error propagation will be less in Noisy Intermediate Scale Quantum (NISQ) computing. On the other hand, it is clear that lowering the depth will require a large number of ancillas. We look into the problem of concrete quantification of this trade-off. Towards a further generalization, we study the optimal T depth quantum circuits for evaluating any arbitrary n-input m-output Boolean function of algebraic degree $k \leq n$. That is, our thesis revolves around the concepts of Boolean functions

from two related directions of the quantum paradigm.

To summarize, this thesis explores two closely related directions within the quantum paradigm: the analysis of quantum algorithms, including Forrelation, nega-Forrelation, and generalized Forrelation, and the design of complex quantum circuits, such as MCT decomposition and Boolean circuit implementation. These two directions are unified through the theory of Boolean functions. A schematic overview of the thesis contributions is presented in Figure 1.1.

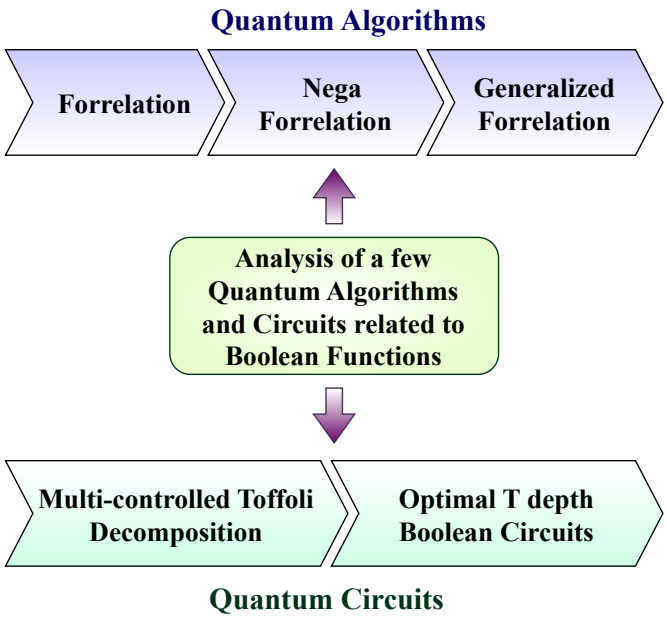


Figure 1.1: Thesis summary.

In the next section, we elaborate on our exact contributions with further details while explaining the organization of the thesis.

1.1 Thesis plan

In the next chapter (Chapter 2), we provide all the necessary background material in detail. However, as we present our contributions throughout this thesis, we will also introduce a few relevant definitions and briefly describe the existing works as required.

We begin with the definition of a Boolean function.

An n-input, m-output Boolean function is a mapping $f : \{0,1\}^n \to \{0,1\}^m$, where f takes an n-bit input and produces an m-bit output.

The set of all n-input, m-output Boolean functions is denoted by $\mathcal{B}_n^m$. For a large part of this thesis, we focus on the case where $m = 1$. It is evident that any n-input, m-output Boolean function can be viewed as m separate n-input, single-output Boolean functions. Since the truth table of an n-input, m-output Boolean function contains 2^n outputs, each of length m bits, the total number of such functions is $2^{m \cdot 2^n}$. For $m = 1$, $\mathcal{B}_n^1 \equiv \mathcal{B}_n$, and the corresponding cardinality is given by 2^{2^n}.

1.1.1 Contribution 1: Quantum algorithms in exploring Boolean functions' spectra using Forrelation

In our first contributory chapter (Chapter 3), we revisit quantum Forrelation algorithms to evaluate some of the well-known, cryptographically significant spectra of Boolean functions, namely the Walsh–Hadamard spectrum, the crosscorrelation spectrum, and the autocorrelation spectrum. Forrelation (short form of Fourier-correlation), introduced by Aaronson in 2010, is formally defined as follows.

Definition 1.1.1. *Given oracle access to $f_1, f_2 \in \mathcal{B}_n$, the (2-fold) Forrelation is a measure of correlation between the truth table of f_1 and the corresponding Walsh-Hadamard spectrum of f_2, formulated as*

$$\Phi_{f_1,f_2} = \frac{1}{2^n} \sum_{\mathbf{x_1} \in \{0,1\}^n} (-1)^{f_1(\mathbf{x_1})} W_{f_2}(\mathbf{x_1}) = \frac{1}{2^{3n/2}} \sum_{\mathbf{x_1},\mathbf{x_2} \in \{0,1\}^n} (-1)^{f_1(\mathbf{x_1})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_2}} (-1)^{f_2(\mathbf{x_2})}.$$

This formulation can be further extended for any $k(> 2) \in \mathbb{N}$ many Boolean functions $f_1, f_2, \ldots, f_k \in \mathcal{B}_n$, known as the k-fold Forrelation.

$$\Phi_{f_1,f_2,\ldots,f_k} = \frac{1}{2^{(k+1)n/2}} \sum_{\mathbf{x_1},\ldots,\mathbf{x_k} \in \{0,1\}^n} (-1)^{f_1(\mathbf{x_1})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_2}} (-1)^{f_2(\mathbf{x_2})} \ldots (-1)^{\mathbf{x_{k-1}} \cdot \mathbf{x_k}} (-1)^{f_k(\mathbf{x_k})}.$$

From the definition, it is straightforward to observe that $-1 \leq \Phi_{f_1,f_2} \leq 1$. Given oracle access to k Boolean functions, Aaronson et al. [2] proposed two efficient quantum algorithms to estimate $\Phi_{f_1,\ldots,f_k}$, one utilizing k sequential queries and the other requiring only $\lceil \frac{k}{2} \rceil$ parallel queries.

In this chapter, we begin with connecting the 2-fold Forrelation formulation with the bent duality-based promise problems as desirable instantiations. Next, we focus on the 3-fold Forrelation formulation, which serves as a unifying framework for evaluating various spectra of Boolean functions, producing the most significant results from this chapter. Specifically, we show that the 3-fold Forrelation for functions $f_1, f_2, f_3 \in \mathcal{B}_n$ can be expressed as

$$\Phi_{f_1,f_2,f_3} = \frac{1}{2^{3n/2}} \sum_{\mathbf{x} \in \{0,1\}^n} (-1)^{f_2(\mathbf{x})} W_{f_1}(\mathbf{x}) W_{f_3}(\mathbf{x}).$$

Given a set of points $S \subseteq \{0,1\}^n$, we fix $f_1 = f_3 = f$ and design f_2 such that $f_2(\mathbf{x}) = 1$ if and only if $\mathbf{x} \in S$, and 0 otherwise. This setup allows us to estimate the Walsh–Hadamard spectrum of f at the points in S using the 3-fold Forrelation algorithm, achieving a higher sampling probability compared to the standard Deutsch–Jozsa algorithm. By employing a similar framework, where $f_2(\mathbf{x}) = 1$ if and only if $wt(\mathbf{x}) \leq m$, and 0 otherwise, we achieve a constant advantage in query complexity over the method proposed in [20] for determining whether f is m-resilient. Although the advantage is constant, the resulting sampling algorithm still outperforms Deutsch–Jozsa while using the same number of queries. This highlights the potential of Forrelation-based approach to provide deeper insights into problems involving Boolean function analysis.

Next, we establish a connection between the crosscorrelation spectrum $C_{f,g}(\mathbf{y})$ of Boolean functions $f_1, f_3 \in \mathcal{B}_n$ and the 3-fold Forrelation framework. We develop a specific quantum algorithm to estimate the crosscorrelation value at a fixed point, employing a strategy analogous to resiliency checking. Borrowing the result from [80, Theorem 3.1] that relates the crosscorrelation at a point $\mathbf{y} \in \{0,1\}^n$ to the 3-fold Forrelation where $f_2(\mathbf{x}) = \mathbf{x} \cdot \mathbf{y}$ is a linear Boolean function, we obtain an efficient estimation technique for the crosscorrelation at $\mathbf{y}$. As an immediate consequence, we derive an algorithm for estimating the autocorrelation spectrum by fixing $f_1 = f_3 = f$ and observed significant improvements over the method proposed in [12].

Finally, we move towards a complete crosscorrelation sampling algorithm by tweaking $\mathcal{A}^{(3,3)}$. Instead of fixing f_2 as a specific linear function, we construct a superposition of all linear functions using an additional n qubits. This allows us to sample directly from the complete crosscorrelation spectrum of f_1 and f_3. As a final contribution, we address the problem of checking whether two functions f_1 and f_3 are uncorrelated up to degree m, i.e., whether $C_{f_1,f_3}(\mathbf{u}) = 0$ for $wt(\mathbf{u}) \leq m$. In this regard, we use Dicke states. For a basic understanding related to the quantum paradigm, we refer to Section 2.2 in the background chapter (Chapter 2).

Definition 1.1.2. *A Dicke state, denoted by $|D_k^n\rangle$, is an n-qubit quantum state with an equal superposition of all $\binom{n}{k}$ basis states with Hamming weight k.*

For example, $|D_1^3\rangle = \frac{1}{\sqrt{3}}(|001\rangle + |010\rangle + |100\rangle)$. From [68], it is known that starting from $|0^n\rangle$, any Dicke state $|D_k^n\rangle$ can be deterministically prepared using $\mathcal{O}(n^2)$ CNOT gates and $\mathcal{O}(n^2)$ many single-qubit gates. Using Dicke states to sample crosscorrelation spectrum further enhance the algorithm and improve the query complexity from $\mathcal{O}(\frac{1}{a})$, where $a^2 = \frac{1}{2^{3n}} \sum_{\mathbf{x}:wt(\mathbf{x})\leq m} C_{f,g}(\mathbf{x})^2$ to $\mathcal{O}(\sum_{i=0}^{m} \frac{1}{b_i})$, with $(b_i)^2 = \frac{1}{\binom{n}{i}2^{2n}} \sum_{\mathbf{x}:wt(\mathbf{x})=i} C_{f,g}(\mathbf{x})^2$. By setting $f_1 = f_3 = f$, the results naturally extend to the sampling of the autocorrelation spectrum. We observe that the autocorrelation sampling using $\mathcal{A}^{(3,2)}$ outperforms the estimation technique presented in [12] in terms of accuracy for a comparable number of queries.

In summary, the main contribution of this chapter is in studying different fundamental and cryptographically relevant spectra of Boolean functions using Forrelation. The

results presented here are either novel or offer improvements over the state-of-the-art methods [12, 20]. As a final note, all the proposed algorithms have been implemented and tested using the IBMQ simulator, and the results are verified with the theoretical expectations.

This chapter is based on the research publication [35].

1.1.2 Contribution 2: Introducing nega-Forrelation and analysis of related quantum algorithms

This is Chapter 4 of the thesis. Motivated by the effectiveness of Forrelation algorithms in sampling different Boolean functions' spectra, a natural question arises whether a similar formulation can be developed for the efficient sampling of nega-Hadamard transforms. The nega-Hadamard transform is defined as follows.

Definition 1.1.3. *Given $f \in \mathcal{B}_n$, the nega-Hadamard transform of f at a point $\boldsymbol{\omega} \in \{0,1\}^n$ is a complex valued function, $N_f : \{0,1\}^n \to \mathbb{C}$, defined as*

$$N_f(\boldsymbol{\omega}) = 2^{-n/2} \sum_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})\oplus \mathbf{x}\cdot\boldsymbol{\omega}} (i)^{wt(\mathbf{x})}.$$

Since the nega-Hadamard transform is complex valued, its complex conjugate is defined as $\overline{N}_f(\boldsymbol{\omega}) = 2^{-n/2} \sum_{\mathbf{x}\in\{0,1\}^n} (-1)^{f(\mathbf{x})\oplus \mathbf{x}\cdot\boldsymbol{\omega}} (-i)^{wt(\mathbf{x})}$. The concept of nega-Hadamard transform was introduced by Riera and Parker [77], who studied generalized bent criteria for Boolean functions exhibiting flat spectra with respect to the nega-Hadamard transform. In this chapter, we extend the Forrelation framework to define the nega-Forrelation, described as follows.

Given oracle access to Boolean functions $f_1, f_2, f_3 \in \mathcal{B}_n$, the (3-fold) nega-Forrelation is defined as a measure of correlation among the truth table of f_1, the nega-Hadamard spectrum of f_2, and the conjugate nega-Hadamard spectrum of f_3. Additionally, we present two efficient quantum algorithms for estimating the nega-Forrelation values, one using 3 sequential queries to the underlying Boolean functions, and the other achieving the same goal using $\lceil \frac{3}{2} \rceil = 2$ parallel queries. In the process, we introduce a novel strategy to sample small values of the nega-Hadamard transform more efficiently than the Extended Deutsch-Jozsa algorithm (due to Sugata et al. [40]), in terms of the required number of queries.

Subsequently, we relate the nega-crosscorrelation spectrum $\hat{C}_{f_1,f_3}(\mathbf{y})$ of Boolean functions $f_1, f_3 \in \mathcal{B}_n$ to the 3-fold nega-Forrelation framework. In this regard, we use the following result from [91, Lemma 4], which connects the nega-crosscorrelation of $f_1, f_2 \in \mathcal{B}_n$ to the product of their nega-Hadamard and conjugate nega-Hadamard transforms (re-

spectively), formulated as

$$\widehat{C}_{f_1,f_2}(\mathbf{y}) = (i)^{wt(\mathbf{y})} \sum_{\mathbf{x}\in\{0,1\}^n} N_{f_1}(\mathbf{x})\overline{N_{f_2}(\mathbf{x})}(-1)^{\mathbf{x}\cdot\mathbf{y}}.$$

Building on this observation, we design quantum algorithms to estimate the nega-crosscorrelation at a fixed point $\mathbf{y} \in \{0,1\}^n$ using the 3-fold nega-Forrelation framework, where the intermediate function is taken to be the linear Boolean function $f_2(\mathbf{x}) = \mathbf{x}\cdot\mathbf{y}$. As a direct application, we propose an algorithm for estimating the nega-autocorrelation spectrum at a specific point by setting $f_1 = f_3 = f$, and present the related results.

We further extend this framework towards a full-spectrum nega-crosscorrelation sampling algorithm by modifying the three-query, three-fold nega-Forrelation algorithm. Here, instead of fixing f_2 as a specific linear function, we construct an equal superposition over all possible linear functions by using an additional n qubits. This modification allows direct sampling from the complete nega-crosscorrelation spectrum of f_1 and f_3. Furthermore, by employing Dicke states, we enable sampling from the crosscorrelation spectrum at points having fixed Hamming weights.

Next, we consider the hidden shift problem of bent functions. Given $f, g \in \mathcal{B}_n$, if there exists a vector $\mathbf{u} \in \{0,1\}^n$ such that $g(\mathbf{x}) = f(\mathbf{x}\oplus\mathbf{u})$ for all $\mathbf{x} \in \{0,1\}^n$, then g is said to be a shifted version of f, with $\mathbf{u}$ as the unknown shift. The problem of finding this hidden shift of a Boolean function with limited oracle access has been a problem of interest since decades. A related version for n-input n-output Boolean functions was addressed in [88], assuming $g = f$.

In [78], Rötteler et al. proposed a polynomial-time quantum algorithm that deterministically recovers the hidden shift $\mathbf{u}$, given oracle access to a bent function g and the dual of another bent function f (denoted by $\tilde{f}$), satisfying $g(\mathbf{x}) = f(\mathbf{x}\oplus\mathbf{u})$ for all $\mathbf{x} \in \{0,1\}^n$. This algorithm requires only a constant number of queries. To the best of our knowledge, it has not been previously observed that this hidden shift finding algorithm is closely related to the 2-fold Forrelation algorithm [2], with some necessary modifications. We further extend the framework for bent functions satisfying $g(\mathbf{x}) = f(\mathbf{x}\oplus\mathbf{b})\oplus\mathbf{c}\cdot\mathbf{x}\oplus d$, and attempt to recover the parameters $\mathbf{b}, \mathbf{c}$ using the exact same algorithm.

Finally, we turn our attention to negabent functions. We show that if a vector $\mathbf{u} \in \{0,1\}^n$ is a hidden shift for two bent functions f and g, satisfying $g(\mathbf{x}) = f(\mathbf{x}\oplus\mathbf{u})$, then the same vector cannot serve as a hidden shift for the corresponding negabent functions $f\oplus s_2$ and $g\oplus s_2$. We define such a vector $\mathbf{u}$ as a nega-shift for the associated negabent functions and present a constant-query, polynomial-time quantum algorithm to recover this hidden nega-shift. Finally, we demonstrate through an example that, by adapting the $\mathcal{O}(n)$-query algorithm, one can recover the standard hidden shift between two negabent functions by generating quantum states orthogonal to the hidden shift $\mathbf{u}$.

This chapter is based on the research publication [34].

1.1.3 Contribution 3: A framework for generalized Forrelation

In Chapter 5, we focus on generalizing the existing frameworks in terms of Boolean functions' spectra. We begin with introducing new unitaries, exploring their implications, and establishing connections to existing ones. Then, we extend the formulation of various fundamental, yet cryptographically significant spectra of Boolean functions, including the Walsh-Hadamard, crosscorrelation, and autocorrelation spectra, to a generalized variation with the m-th primitive root of unity, for any $m \in \mathbb{N}$. In the process, we identify a previously unexamined class of real Hadamard transforms that lies between the Walsh-Hadamard and nega-Hadamard transformations, noting a gap in the existing literature. Additionally, we introduce the most generalized version of the Deutsch-Jozsa algorithm, which extends both the standard Deutsch-Jozsa [27] and its prior extended version [40], incorporating them as special cases. Furthermore, we extend the Forrelation formulation to m-Forrelation and propose new quantum algorithms for estimating them for a given set of Boolean functions. Subsequently, we present various sampling strategies of these newly defined spectra of Boolean functions using the generalized Forrelation algorithms, and present the comparison based on the corresponding sampling probabilities.

This chapter is based on our research work available at [36].

1.1.4 Contribution 4: Exact space-depth trade-offs in multi-controlled Toffoli decomposition

From the mathematical and algorithmic analysis of Boolean functions, we now move towards exact circuit implementation issues in the last two contributory chapters. The quantum gates are the fundamental building blocks of quantum circuits. Unlike classical gates, quantum gates are inherently reversible and are mathematically represented by unitary matrices. The doubly-controlled X-gate, commonly known as the Toffoli gate, is among the most significant quantum gates, with critical applications in arithmetic operations [95, 96, 97], reversible computing [21, 25], and oracle constructions [5, 11, 32, 48]. However, the Toffoli gate's high resource demands, particularly in terms of T count, T depth, and ancilla qubits, can significantly influence the efficiency of fault-tolerant quantum circuits. Consequently, optimizing its implementation is crucial for reducing computational overhead, minimizing error rates, and enhancing scalability, making it indispensable for practical large-scale quantum computations.

In classical computing, (2-input 1-output) NAND and NOR gates are considered universal as they can be used to construct any classical logic circuit. In contrast, quantum computing involves infinite (uncountable) quantum gates, including both single-qubit and multi-qubit ones, making it challenging to define a universal description. However, there exist certain gate sets that can approximate any unitary transformation on a quantum computer to an arbitrary degree of accuracy, known as the universal gate sets. The

Clifford+T gate set is the most widely adopted universal gate set in quantum computing due to its compatibility with fault-tolerant quantum computation.

The Clifford group consists of the Hadamard, phase, and CNOT gates. Augmenting it with the non-Clifford T gate yields a universal set capable of approximating any unitary operation to arbitrary precision. A detailed description of these quantum gates and their internal transformations has been provided in Section 2.2 of the background chapter (Chapter 2).

In quantum computing, the Multi-Controlled Toffoli (MCT) gates also play a crucial role in the design of complex quantum algorithms, including error correction codes and arithmetic operations. Despite its utility, the implementation of MCT gates exploiting the Clifford+T gate set requires careful decomposition to optimize resource usage, such as minimizing the T count, T depth, and the number of ancilla qubits. These optimizations are essential for practical quantum computation, where resource efficiency is critical. The issues related to ancilla qubits, and in particular the idea of conditionally clean ancilla from [59], are also important here, and we describe them in detail.

Towards studying this aspect, in Chapter 6, we consider the optimized implementation of Multi Controlled Toffoli (MCT) using the Clifford+T gate sets. While there are several recent works in this direction, here we explicitly quantify the trade-off (with concrete formulae) between the Toffoli depth (this means the depth using the classical 2-controlled Toffoli) of the n-controlled Toffoli (henceforth denoted as n-MCT) and the number of clean ancilla qubits. Additionally, we achieve a reduced Toffoli depth (and consequently, T depth), which is an extension of the technique introduced by Khattar et al. [59] very recently. In terms of a negative result, we first show that using such conditionally clean ancilla techniques, Toffoli depth can never achieve exactly $\lceil \log_2 n \rceil$, though it remains in the same order. This highlights the limitation of the techniques exploiting conditionally clean ancilla by Nie et al. [70] and Khattar et al. [59]. Then we prove that, in a more general setup, the T depth in the Clifford+T decomposition, via Toffoli gates, is lower bounded by $\lceil \log_2 n \rceil$, and this bound is achieved following the complete binary tree structure. Since the (2-controlled) Toffoli gate can further be decomposed using Clifford+T, various methodologies are explored too in this regard for trade-off related implications.

This chapter is based on the research publication [37].

1.1.5 Contribution 5: Optimal T depth quantum circuits for arbitrary Boolean functions

In Chapter 7, we present a generic construction to obtain an optimal T depth quantum circuit for any arbitrary n-input m-output Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}^m$ having algebraic degree $k \leq n$. Our technique achieves an exact Toffoli (and T) depth of $\lceil \log_2 k \rceil$. This is a broader generalization of the technique used in the previous chapter,

establishing the optimal Toffoli (and consequently T) depth for multi-controlled Toffoli decompositions. We achieve this by inspecting the Algebraic Normal Form (ANF) of a Boolean function.

The Algebraic Normal Form (ANF) of a Boolean function $f \in \mathcal{B}_n$ can be represented by a polynomial over $\{0,1\}$ in n binary variables, given by

$$f(x_1,\ldots,x_n) = a_0 \oplus \bigoplus_{I \subseteq \{1,\ldots,n\}} a_I \prod_{i \in I} x_i$$

where $a_I \in \{0,1\}$. The algebraic degree of a Boolean function $f \in \mathcal{B}_n$, denoted as $\deg(f)$, is given by the maximum cardinality of I, such that $a_I \neq 0$. Clearly, $\deg(f) \leq n$.

Obtaining a benchmark for the minimum T depth of such circuits is of prime importance for the efficient implementation of quantum algorithms by enabling greater parallelism, reducing time complexity, and minimizing circuit latency, making them suitable for near-term quantum devices with limited coherence times. The implications of our results are highlighted, explaining the provable lower bounds on S-box and block cipher implementations, for example, AES.

Our theoretical contribution is that we present optimal Toffoli (consequently T) depth quantum circuits implementation for evaluating any Boolean function that subsumes the idea of the previous chapter [37] and the fundamental observation is to note the XOR ($\mathbb{F}_2$ addition) of AND ($\mathbb{F}_2$ multiplication) in ANF along with the binary tree-based implementation. While our construction achieves optimal T depth, it incurs exponential overhead in ancilla qubits and CNOT gates with respect to the number of input variables. However, we recommend that during any circuit design, first, it must be understood what the benchmark T depth of the circuit should be, and then only other parameters may be optimized; as minimization of T depth is one of the most important criteria in quantum circuit design. That is, the benchmark for optimal T depth for Boolean circuits implemented in quantum algorithms is completely settled with this treatment, and we present the design of block ciphers as concrete examples. While the other resource requirements are high, considering the round structures of block ciphers, we could demonstrate clear benchmarks related to optimal T depth with exact estimates of other resources.

This chapter is based on our research work available at [31].

1.2 Prerequisites

We assume the reader is familiar with undergraduate-level combinatorics, linear and abstract algebra. We will present more details regarding the more involved algebraic and combinatorial structures in Chapter 2. A basic understanding of digital circuits, classical computational models, algorithms, and programming methodologies is neces-

sary to understand the contributions of the thesis. While we explain the basic mathematical structure of quantum computing and Boolean functions in the next chapter, undergraduate-level knowledge in these areas will be an added advantage. There is no requirement to understand quantum information to follow this work. We will develop the background with sufficient details in the following sections and introduce the ideas one by one, as and when required.

1.3 Conclusion

This thesis revolves around Boolean functions, quantum algorithms, and related quantum circuits. We present various novel results in this direction. We first study the famous Forrelation algorithm [1, 2] to see how it can be applied in understanding various spectra of Boolean functions in the quantum domain. Then we extend it to nega-Forrelation and obtain several additional characteristics related to various specific kinds of Boolean functions. Our results also explain interesting applications towards bent and negabent functions. These results are then further extended for a framework of generalized Forrelation that exploits the m-th primitive root of unity. This part of the thesis considers several characteristics of Boolean functions, understanding of Walsh, auto, and cross-correlation spectra, and related quantum algorithms. The properties that we study are important for the Boolean functions that may be exploited as cryptographic primitives. Then we consider the problem of circuit implementation corresponding to certain Boolean functions. In this direction, we first study the exact space-depth trade-offs in multi-controlled Toffoli decomposition and provide results to show how we can optimize Multi Controlled Toffoli (MCT) using the Clifford+T gate sets. Then we note that such a technique can be extended to any Boolean function, considering the Algebraic Normal Form, and in this regard, provide construction results for such circuits with optimal T depth. Finally, we conclude in Chapter 8 with a summary of this thesis and directions towards future research.

Chapter 2

Background

In this chapter, we present a comprehensive overview of Boolean functions, quantum computing and algorithms, and quantum circuit construction. The chapter is organized into three sections. The first section introduces Boolean functions and discusses various spectral representations relevant to their analysis. The second section provides a structured exposition of quantum computing, beginning with foundational concepts and progressing to advanced quantum algorithms. The final section focuses on quantum circuit construction, highlighting how Boolean functions and quantum principles are integrated in the design and optimization of quantum circuits.

2.1 Boolean functions

A Boolean function, which lies at the core of logic and computation, is a simple yet powerful construct that formalizes binary decision-making by mapping input combinations of true and false (or 0 and 1) to structured outputs. The conceptual foundation was laid in the mid-19th century by the English mathematician and logician George Boole, in his seminal work The Laws of Thought (1854) [14]. Boole's abstraction, originally intended to model human reasoning, provided the groundwork for modern symbolic logic and digital computation.

Although Boole's ideas were primarily philosophical and linguistic, their practical significance became evident nearly a century later, when Shannon demonstrated their application in the design and optimization of electrical switching circuits [85]. This insight bridged abstract logic with physical computation, initiating the era of digital computing and influencing foundational areas such as information theory, cryptography, coding theory, and computational complexity.

Let us now formally define a Boolean function in its modern mathematical form.

Let $\mathbb{F}_2 = \{0, 1\}$ denote the finite field of characteristic 2, and let $\mathbb{F}_2^n \equiv \{\mathbf{x} =$

$(x_1, x_2, \ldots, x_n) : x_i \in \mathbb{F}_2, 1 \leq i \leq n\}$ be the n-dimensional vector space over $\mathbb{F}_2$. Throughout this thesis, elements (vectors) of $\mathbb{F}_2^n$ are denoted using bold symbols to distinguish them from scalar elements in $\mathbb{F}_2$.

Definition 2.1.1. *An n-input, m-output* Boolean function *is a mapping $f : \mathbb{F}_2^n \to \mathbb{F}_2^m$.*

The set of all such functions is denoted by $\mathcal{B}_n^m$. Combinatorially, the total number of distinct n-input, m-output Boolean functions is given by $|\mathcal{B}_n^m| = 2^{m \cdot 2^n}$. In particular, the set of all n-input, single-output Boolean functions is denoted by $\mathcal{B}_n^1 \equiv \mathcal{B}_n$, with cardinality $|\mathcal{B}_n| = 2^{2^n}$. Moreover, any n-input, m-output Boolean function can be viewed as a tuple of m single-output Boolean functions, each mapping $\mathbb{F}_2^n$ to $\mathbb{F}_2$. Thus, many of the concepts and results discussed in the context of single-output Boolean functions can be naturally extended to the multi-output case.

There are several standard ways to represent a Boolean function. One of the most widely used is the *truth table* representation, where all possible 2^n input combinations are listed in lexicographic order, paired with their corresponding output bit-pattern(s). The truth tables of 2-variable AND function ($\mathbb{F}_2$-multiplication) and 2-variable XOR function ($\mathbb{F}_2$-addition) are shown below.

x_1	x_0	$x_1 \cdot x_0$
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_0	$x_1 \oplus x_0$
0	0	0
0	1	1
1	0	1
1	1	0

Apart from truth table, another commonly used representation is the *Algebraic Normal Form (ANF).*

Definition 2.1.2. *The ANF of a Boolean function $f \in \mathcal{B}_n$ is a multivariate polynomial over $\mathbb{F}_2$ in n binary variables, expressed as*

$$\begin{aligned} f(x_1, \ldots, x_n) =& a_0 \oplus \bigoplus_{I \subseteq \{1,\ldots,n\}} a_I \prod_{i \in I} x_i \\ =& a_0 \oplus \bigoplus_{i=1}^{n} a_i x_i \oplus \bigoplus_{i<j} a_{ij} x_i x_j \oplus \ldots \oplus a_{12\ldots n} x_1 x_2 \ldots x_n \end{aligned}$$

where $a_I \in \mathbb{F}_2$. Since $I \subseteq \{1, \ldots, n\}$, the maximum number of coefficients a_I required to describe the ANF of an n-variable Boolean function is 2^n.

Remark 2.1.1. *Given the ANF of a Boolean function $f \in \mathcal{B}_n$, the truth table can be computed by evaluating the expression over all input vectors. Conversely, the presence of terms in the ANF can be determined from the truth table using a classical deterministic algorithm. Hence, the truth table representation of a Boolean function is in one-to-one correspondence with its ANF representation.*

The *algebraic degree* of a Boolean function $f \in \mathcal{B}_n$, denoted $\deg(f)$, is the maximum cardinality of I such that $a_I \neq 0$. Clearly, $\deg(f) \leq n$. A Boolean function $f \in \mathcal{B}_n$ is said to be *nonlinear* if $\deg(f) > 1$, *affine* if $\deg(f) = 1$, and *constant* if $\deg(f) = 0$. The ANF of an affine function $f \in \mathcal{B}_n$ is given by $f(x_1, \ldots, x_n) = a_0 \oplus a_1 x_1 \oplus \ldots \oplus a_n x_n$, where $a_i \in \mathbb{F}_2$ for all $0 \leq i \leq n$. Since there are $n+1$ coefficients, each taking one of two possible values, the total number of affine functions is 2^{n+1}. If $a_0 = 0$, the function is *linear*, and there are 2^n linear functions. Additionally, $f = 0$ and $f = 1$ are the two constant Boolean functions.

The *Hamming weight* of a bit-string $\mathbf{x} \in \mathbb{F}_2^n$ is defined as $wt(\mathbf{x}) = \sum_{i=1}^{n} x_i$, i.e., the number of 1's in $\mathbf{x}$. Given $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$, their scalar (or inner) product is given by $\mathbf{x} \cdot \mathbf{y} = \bigoplus_{i=1}^{n} x_i y_i$. Their bit-wise intersection is given by $\mathbf{x} \star \mathbf{y} = (x_1 y_1, \ldots, x_n y_n)$, and the weight of their XOR satisfies: $wt(\mathbf{x} \oplus \mathbf{y}) = wt(\mathbf{x}) + wt(\mathbf{y}) - 2wt(\mathbf{x} \star \mathbf{y})$.

For a complex number $z = a + ib$, where $i^2 = -1$, we denote its real part by $\Re(z) = a$, imaginary part by $\Im(z) = b$, and the squared modulus by $|z|^2 = z \cdot \overline{z} = a^2 + b^2$, where $\overline{z} = a - ib$ denoting the complex conjugate of z.

Definition 2.1.3. *Given $f \in \mathcal{B}_n$, the* Walsh-Hadamard transform *of f at a point $\boldsymbol{\omega} \in \mathbb{F}_2^n$ is an integer valued function, $W_f : \mathbb{F}_2^n \to [-2^{n/2}, 2^{n/2}]$ formulated as*

$$W_f(\boldsymbol{\omega}) = 2^{-n/2} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \boldsymbol{\omega}}.$$

The multi-set $\{W_f(\boldsymbol{\omega}) : \boldsymbol{\omega} \in \mathbb{F}_2^n\}$ is called the *Walsh–Hadamard spectrum* of $f \in \mathcal{B}_n$. The constraint, $\sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} (W_f(\boldsymbol{\omega}))^2 = 2^n$ is known as the *Parseval's identity*. The inverse of Walsh-Hadamard transform is given by

$$(-1)^{f(\mathbf{x})} = 2^{-n/2} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} W_f(\boldsymbol{\omega}) (-1)^{\boldsymbol{\omega} \cdot \mathbf{x}}.$$

Remark 2.1.2. *Given the truth table of a Boolean function $f \in \mathcal{B}_n$, the Walsh-Hadamard spectrum can be computed classically, with a time complexity $\mathcal{O}(n2^n)$. Conversely, the truth table can be recovered from the Walsh-Hadamard spectrum using the inverse Walsh-Hadamard transform. Therefore, the truth table representation and the Walsh-Hadamard spectrum representations of a Boolean function are in one-to-one correspondence.*

Notably, the Walsh-Hadamard transform of $f \in \mathcal{B}_n$ can be interpreted as the difference between the number of points where f is equal to he linear function, $\mathbb{L}_{\boldsymbol{\omega}} = \boldsymbol{\omega} \cdot \mathbf{x}$ and where it does not, with a normalization factor, i.e.,

$$W_f(\boldsymbol{\omega}) = 2^{-n/2} \left[\#\{f = \mathbb{L}_{\boldsymbol{\omega}}\} - \#\{f \neq \mathbb{L}_{\boldsymbol{\omega}}\} \right] = 2^{-n/2} \left[2^n - 2\#\{f \neq \mathbb{L}_{\boldsymbol{\omega}}\} \right].$$

Hence, if for some $\boldsymbol{\omega}$, the function f achieves the minimum (Hamming) distance from $\mathbb{L}_{\boldsymbol{\omega}}$ or $1 \oplus \mathbb{L}_{\boldsymbol{\omega}}$, then the absolute value of $W_f(\boldsymbol{\omega})$ attains its maximum at that point.

Consequently, the nonlinearity of f, in terms of the Walsh-Hadamard transform, is given by

$$nl(f) = 2^{n-1} - 2^{\frac{n}{2}-1} \max_{\boldsymbol{\omega} \in \mathbb{F}_2^n} |W_f(\boldsymbol{\omega})|.$$

The nonlinearity of a Boolean function is maximized when the maximum value of its Walsh-Hadamard transform is minimized, i.e., when the Walsh spectrum is uniformly distributed. By Parseval's identity, this occurs when $|W_f(\boldsymbol{\omega})| = 1$ for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$. In that case, the nonlinearity becomes: $2^{n-1} - 2^{\frac{n}{2}-1}$, and such functions are known as bent functions [18, 28]. Since nonlinearity must be an integer, bent functions exist only for even n. Determining the maximum nonlinearity for Boolean functions with odd n remains an open problem when n is large. It is known that for $n \leq 7$, the maximum nonlinearity is $2^{n-1} - 2^{\frac{n-1}{2}}$. For $n \geq 9$, the maximum nonlinearity is known to exceed $2^{n-1} - 2^{\frac{n-1}{2}}$.

If $f \in \mathcal{B}_n$ is bent, then for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$, $W_f(\boldsymbol{\omega}) = \pm 1 = (-1)^{g(\boldsymbol{\omega})}$, for some $g \in \mathcal{B}_n$. From the inverse Walsh-Hadamard transform of f, we obtain:

$$(-1)^{f(\mathbf{x})} = 2^{-n/2} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} W_f(\boldsymbol{\omega})(-1)^{\boldsymbol{\omega} \cdot \mathbf{x}} = 2^{-n/2} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} (-1)^{g(\boldsymbol{\omega})}(-1)^{\boldsymbol{\omega} \cdot \mathbf{x}} = W_g(\mathbf{x}).$$

Thus, g is also bent, and the functions f and g are said to be *dual*, denoted by $g = \widehat{f}$, and $f = \widehat{g}$. Clearly, $\widehat{\widehat{f}} = \widehat{g} = f$. If $f = \widehat{f}$, then f is called *self-dual*. Further, if $W_f(\boldsymbol{\omega}) = (-1)^{g(\boldsymbol{\omega}) \oplus 1}$, then f and g are *anti-dual*. Similarly, if $f = \widehat{f} \oplus 1$, then f is said to be *anti self-dual*.

Example 2.1.1. *The 4-variable Boolean function, $f(x_0, x_1, x_2, x_3) = x_0 x_1 \oplus x_2 x_3$ is a bent function. The truth table of f is given by: $\{0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0\}$. The Walsh-Hadamard spectrum of f is: $\{1, 1, 1, -1, 1, 1, 1, -1, 1, 1, 1, -1, -1, -1, -1, 1\}$. The nonlinearity of f is given by $2^{4-1} - 2^{2-1} = 6$, while the algebraic degree of f is 2. Moreover, since $W_f(\boldsymbol{\omega}) = (-1)^{f(\boldsymbol{\omega})}$ for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$, f is self-dual.*

For an affine Boolean function $f(\mathbf{x}) = \mathbf{a} \cdot \mathbf{x} \oplus b$, the Walsh-Hadamard transform at the point $\mathbf{a} \in \mathbb{F}_2^n$ is $W_f(\mathbf{a}) = 2^{-n/2} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{\mathbf{a} \cdot \mathbf{x} \oplus b \oplus \mathbf{a} \cdot \mathbf{x}} = (-1)^b 2^{n/2}$, implying $(W_f(\mathbf{a}))^2 = 2^n$. By Parseval's identity, it follows that $W_f(\boldsymbol{\omega}) = 0$ for all $\boldsymbol{\omega} \neq \mathbf{a}$. Consequently, the nonlinearity of any affine Boolean function is: $nl(f) = 2^{n-1} - 2^{n/2-1}(2^{n/2}) = 0$. A Boolean function $f \in \mathcal{B}_n$ is said to be balanced, if it has an equal number of 0's and 1's in its output. For a balanced Boolean function, $W_f(0^n) = 2^{-n/2} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x})} = 0$. We come back to identifying the Walsh-Hadamard spectrum of different class of Boolean functions, when we introduce quantum algorithms in the next section.

We define another property of the Boolean functions, related to Walsh-Hadamard transform values, called the resiliency (see [81] for more details and cryptographic implications). The resiliency of a function f is defined as follows [98].

Definition 2.1.4. *An Boolean function $f \in \mathcal{B}_n$ is called m-resilient (for $m < n$) if and only if $W_f(\boldsymbol{\omega}) = 0$ for all $\boldsymbol{\omega}$ such that $0 \leq wt(\boldsymbol{\omega}) \leq m$.*

If there exists an $\boldsymbol{\omega} \in \mathbb{F}_2^n$ such that $wt(\boldsymbol{\omega}) \leq m$ and $W_f(\boldsymbol{\omega}) \neq 0$, then f is not m-resilient.

In Chapter 3, we propose an improved algorithm for checking resiliency of a Boolean function compared to the state-of-the-art results [20].

We now define the crosscorrelation of two Boolean functions, a widely studied cryptographic property closely related to Shannon's notion of confusion.

Definition 2.1.5. *The crosscorrelation of two Boolean functions $f, g \in \mathcal{B}_n$, at a point $\mathbf{u} \in \mathbb{F}_2^n$ is defined as*

$$C_{f,g}(\mathbf{u}) = \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus g(\mathbf{x} \oplus \mathbf{u})}.$$

Assuming $f = g$, we obtain the autocorrelation of $f \in \mathcal{B}_n$ at a point $\mathbf{u} \in \mathbb{F}_2^n$, defined as

$$C_f(\mathbf{u}) = \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus f(\mathbf{x} \oplus \mathbf{u})}.$$

In this regard, we present the following result, which connects the crosscorrelation (and consequently, autocorrelation) of Boolean functions with the product (or square) of their Walsh–Hadamard transforms.

Theorem 2.1.1 ([80])**.** *Let $f, g \in \mathcal{B}_n$. Then, $C_{f,g}(\mathbf{u}) = \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} W_f(\boldsymbol{\omega}) W_g(\boldsymbol{\omega}) (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}$. For $f = g$, we obtain, $C_f(\mathbf{u}) = \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} (W_f(\boldsymbol{\omega}))^2 (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}$.*

This leads immediately to the following corollary.

Corollary 2.1.1. *Let $f \in \mathcal{B}_n$ be bent, i.e., $|W_f(\boldsymbol{\omega})| = 1$ for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$. Then,*

$$C_f(\mathbf{u}) = \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} (W_f(\boldsymbol{\omega}))^2 (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}} = \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}.$$

Therefore, $C_f(\mathbf{u}) = 0$ for all $\mathbf{u} \neq 0^n$, and $C_f(0^n) = 2^n$.

Remark 2.1.3. *Note that, a Boolean function $f \in \mathcal{B}_n$ can also be represented via its autocorrelation spectrum. While the autocorrelation can be computed from either the truth table or the Walsh–Hadamard spectrum of f, the converse does not hold: since it involves the square of the Walsh–Hadamard coefficients, sign information is lost. Consequently, the original truth table or Walsh–Hadamard spectrum cannot be uniquely recovered from the autocorrelation alone.*

Additionally, we refer to the following terminology related to crosscorrelation.

Definition 2.1.6. *Two Boolean functions $f, g \in \mathcal{B}_n$ are said to be uncorrelated of degree k if $C_{f,g}(\mathbf{u}) = 0$ for all $\mathbf{u}$ such that $0 \leq wt(\mathbf{u}) \leq k$.*

It is desirable that the component functions of a cryptographic system are pairwise as much uncorrelated as possible. In Figure 2.1 we present various standard representation of an n-input, single output Boolean function, and their internal conversion. Clearly, truth table is simplest way to represent a Boolean function, therefore, placed at the center and connect all other representations.

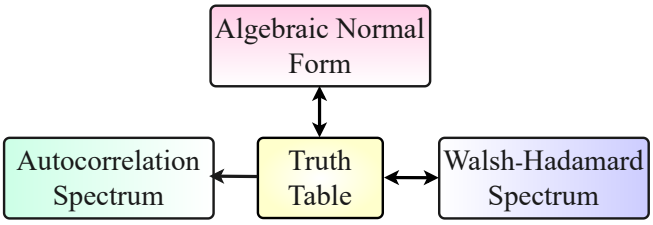


Figure 2.1: Various standard representations of Boolean function.

Definition 2.1.7. *Given $f \in \mathcal{B}_n$, the nega-Hadamard transform of f at a point $\boldsymbol{\omega} \in \mathbb{F}_2^n$ is a complex valued function, $N_f : \mathbb{F}_2^n \to \mathbb{C}$ formulated as*

$$N_f(\boldsymbol{\omega}) = 2^{-n/2} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \boldsymbol{\omega}} (i)^{wt(\mathbf{x})}.$$

The multi-set $\{N_f(\boldsymbol{\omega}) : \boldsymbol{\omega} \in \mathbb{F}_2^n\}$ is called the *nega–Hadamard spectrum* of $f \in \mathcal{B}_n$. Since nega-Hadamard transform is complex valued, the *nega-Parseval's identity* is given by $\sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} |N_f(\boldsymbol{\omega})|^2 = \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} N_f(\boldsymbol{\omega}) \overline{N_f(\boldsymbol{\omega})} = 2^n$. The inverse nega-Hadamard transform is given by

$$(-1)^{f(\mathbf{x})} = 2^{-n/2} (i)^{-wt(\mathbf{x})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} N_f(\boldsymbol{\omega}) (-1)^{\boldsymbol{\omega} \cdot \mathbf{x}}.$$

Remark 2.1.4. *Given the truth table of a Boolean function $f \in \mathcal{B}_n$, the nega-Hadamard spectrum can be computed classically, with a time complexity $\mathcal{O}(n2^n)$. Conversely, the truth table can be recovered from the nega-Hadamard spectrum using the inverse nega-Hadamard transform. Thus, the truth table and the nega-Hadamard spectrum representations of a Boolean function are in one-to-one correspondence.*

For even n, a Boolean function $f \in \mathcal{B}n$ is called negabent if and only if it has a perfectly flat nega-Hadamard spectrum in terms of complex modulus, i.e., $|N_f(\boldsymbol{\omega})| = 1$ for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$. As shown in [77], if f is bent, then $f \oplus s_2$ is negabent, and vice versa, where $s_2(\mathbf{x}) = \bigoplus_{i<j} x_i x_j$ is a quadratic symmetric Boolean function, which itself is bent. A Boolean function is called bent-negabent if it satisfies both bent and negabent

properties. Clearly, if f is bent-negabent, then so is $f \oplus s_2$. Furthermore, if f is bent but not negabent, then $f \oplus s_2$ is negabent but not bent. Otherwise, if $f \oplus s_2$ were also bent, then $f = (f \oplus s_2) \oplus s_2$ would be negabent, leading to a contradiction. As shown in [73, Proposition 1], all affine Boolean functions are negabent. Clearly, affine functions are not bent, and therefore, functions of the form $s_2 \oplus \mathbf{a} \cdot \mathbf{x} \oplus b$, where $\mathbf{a} \in \mathbb{F}_2^n, b \in \mathbb{F}_2$, are always bent but not negabent.

Example 2.1.2. *A 6-variable Boolean function $f(x_0, x_1, x_2, x_3, x_4, x_5) = x_0x_2 \oplus x_0x_3$ is a non-affine negabent function, which is not bent. Clearly,*

$$f \oplus s_2 = x_0x_1 \oplus x_0x_4 \oplus x_0x_5 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_1x_5 \oplus x_2x_3 \oplus x_2x_4 \oplus x_2x_5 \oplus x_3x_4 \oplus x_3x_5 \oplus x_4x_5$$

is a 6-variable bent function, which is not negabent.

Similar to crosscorrelation, let us now define nega-crosscorrelation (and consequently nega-autocorrelation from [91].

Definition 2.1.8. *The nega-crosscorrelation of two Boolean functions $f, g \in \mathcal{B}_n$, at a point $\mathbf{u} \in \mathbb{F}_2^n$ is defined as*

$$\widehat{C}_{f,g}(\mathbf{u}) = \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus g(\mathbf{x} \oplus \mathbf{u})} (-1)^{\mathbf{x} \cdot \mathbf{u}}.$$

Assuming $f = g$, we obtain the nega-autocorrelation of $f \in \mathcal{B}_n$ at a point $\mathbf{u} \in \mathbb{F}_2^n$, defined as

$$\widehat{C}_{f}(\mathbf{u}) = \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus f(\mathbf{x} \oplus \mathbf{u})} (-1)^{\mathbf{x} \cdot \mathbf{u}}.$$

The following result connects the nega-crosscorrelation (and consequently, nega-autocorrelation) of Boolean functions with the product (or square) of their nega–Hadamard transforms.

Theorem 2.1.2 ([90])**.** *Let $f, g \in \mathcal{B}_n$. Then, $\widehat{C}_{f,g}(\mathbf{u}) = (i)^{wt(\mathbf{u})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} N_f(\boldsymbol{\omega}) \overline{N_g(\boldsymbol{\omega})} (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}$. For $f = g$, we obtain, $\widehat{C}_{f}(\mathbf{u}) = (i)^{wt(\mathbf{u})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} |N_f(\boldsymbol{\omega})|^2 (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}$.*

This leads to the following corollary.

Corollary 2.1.2. *Let $f \in \mathcal{B}_n$ be negabent, i.e., $|N_f(\boldsymbol{\omega})| = 1$ for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$. Then,*

$$\widehat{C}_{f}(\mathbf{u}) = (i)^{wt(\mathbf{u})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} |N_f(\boldsymbol{\omega})|^2 (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}} = (i)^{wt(\mathbf{u})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}.$$

Therefore, $\widehat{C}_{f}(\mathbf{u}) = 0$ for all $\mathbf{u} \neq 0^n$, and $\widehat{C}_{f}(0^n) = 2^n$.

Remark 2.1.5. *A Boolean function $f \in \mathcal{B}_n$ can be represented via its nega-autocorrelation spectrum too. While the nega-autocorrelation can be computed from either the truth table or the nega–Hadamard spectrum of f, the converse does not hold: since it depends on the squared nega–Hadamard coefficients, sign information is lost. As a result, neither the original truth table nor the nega–Hadamard spectrum can be uniquely reconstructed from the nega-autocorrelation alone.*

In Chapter 4, we revisit the nega-Hadamard transform and negabent functions, presenting new results and insights. In [89], the discrete Fourier transforms of $f \in \mathcal{B}_n$ have been further extended to 2^k-Hadamard transform, defined as follows.

Definition 2.1.9. *Given $f \in \mathcal{B}_n$, the 2^k-Hadamard transform of f at a point $\boldsymbol{\omega} \in \mathbb{F}_2^n$ is a complex-valued function, $\mathcal{H}_f^{(2^k)} : \mathbb{F}_2^n \to \mathbb{C}$ formulated as*

$$\mathcal{H}_f^{(2^k)}(\boldsymbol{\omega}) = 2^{-n/2} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \boldsymbol{\omega}} \zeta_{2^k}^{wt(\mathbf{x})}.$$

The multi-set $\{\mathcal{H}_f^{(2^k)}(\boldsymbol{\omega}) : \boldsymbol{\omega} \in \mathbb{F}_2^n\}$ is called the 2^k-Hadamard spectrum of $f \in \mathcal{B}_n$. The corresponding 2^k-*Parseval's identity* is given by

$$\sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} |\mathcal{H}_f^{(2^k)}(\boldsymbol{\omega})|^2 = \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \mathcal{H}_f^{(2^k)}(\boldsymbol{\omega}) \overline{\mathcal{H}_f^{(2^k)}(\boldsymbol{\omega})} = 2^n.$$

The inverse 2^k-Hadamard transform is given by

$$(-1)^{f(\mathbf{x})} = 2^{-n/2} \zeta_{2^k}^{-wt(\mathbf{x})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \mathcal{H}_f^{(2^k)}(\boldsymbol{\omega}) (-1)^{\boldsymbol{\omega} \cdot \mathbf{x}}.$$

Clearly, there is an one-to-one correspondence between the truth table representation of a Boolean function, with its 2^k-Hadamard spectrum representation. A Boolean function $f \in \mathcal{B}_n$ satisfying $|\mathcal{H}_f^{(2^k)}(\boldsymbol{\omega})| = 1$ for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$ is called a 2^k-bent function.

In [89], the crosscorrelation and the autocorrelation have been further generalized to 2^k-*crosscorrelation* and 2^k-*autocorrelation*, respectively, defined as follows.

Definition 2.1.10. *The 2^k-crosscorrelation of two Boolean functions $f, g \in \mathcal{B}_n$, at a point $\mathbf{u} \in \mathbb{F}_2^n$ is defined as*

$$C_{f,g}^{(2^k)}(\mathbf{u}) = \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus g(\mathbf{x} \oplus \mathbf{u})} (\zeta_{2^k}^2)^{\mathbf{x} \cdot \mathbf{u}}.$$

Assuming $f = g$, we obtain the 2^k-autocorrelation of $f \in \mathcal{B}_n$ at a point $\mathbf{u} \in \mathbb{F}_2^n$, defined as

$$C_f^{(2^k)}(\mathbf{u}) = \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus f(\mathbf{x} \oplus \mathbf{u})} (\zeta_{2^k}^2)^{\mathbf{x} \cdot \mathbf{u}}.$$

Notably, for $k = 0$ ($\zeta_1 = 1$) and $k = 1$ ($\zeta_2 = -1$) the 2^k-crosscorrelation and 2^k-autocorrelation coincide with the standard crosscorrelation and autocorrelation, respectively. For $k = 2$ ($\zeta_{2^2} = i$), they correspond to the nega-crosscorrelation and nega-autocorrelation.

The following result connects the 2^k-crosscorrelation (and 2^k-autocorrelation) of Boolean functions with the product (or square) of their 2^k–Hadamard transforms.

Theorem 2.1.3 ([89]). *Let $f, g \in \mathcal{B}_n$. Then,*

$$C_{f,g}^{(2^k)}(\mathbf{u}) = \zeta_{2^k}^{wt(\mathbf{u})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \mathcal{H}_f^{(2^k)}(\boldsymbol{\omega}) \overline{\mathcal{H}_g^{(2^k)}(\boldsymbol{\omega})} (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}.$$

For $f = g$, we obtain, $C_f^{(2^k)}(\mathbf{u}) = \zeta_{2^k}^{wt(\mathbf{u})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \left| \mathcal{H}_f^{(2^k)}(\boldsymbol{\omega}) \right|^2 (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}$.

This leads to the following immediate corollary.

Corollary 2.1.3. *Let $f \in \mathcal{B}_n$ be 2^k-bent, i.e., $|\mathcal{H}_f^{(2^k)}(\boldsymbol{\omega})| = 1$ for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$. Then,*

$$C_f^{(2^k)}(\mathbf{u}) = \zeta_{2^k}^{wt(\mathbf{u})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \left| \mathcal{H}_f^{(2^k)}(\boldsymbol{\omega}) \right|^2 (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}} = \zeta_{2^k}^{wt(\mathbf{u})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} (-1)^{\boldsymbol{\omega} \cdot \mathbf{u}}.$$

Therefore, $C_f^{(2^k)}(\mathbf{u}) = 0$ for all $\mathbf{u} \neq 0^n$, and $C_f^{(2^k)}(0^n) = 2^n$.

Remark 2.1.6. *A Boolean function $f \in \mathcal{B}_n$ can be represented via its 2^k-autocorrelation spectrum too. While the 2^k-autocorrelation can be computed from either the truth table or the 2^k–Hadamard spectrum of f, the converse does not hold: since it depends on the squared 2^k–Hadamard coefficients, sign information is lost. As a result, neither the original truth table nor the 2^k–Hadamard spectrum can be uniquely reconstructed from the 2^k-autocorrelation alone. For more details on 2^k-spectra of Boolean functions, the readers can refer to [65] and the references therein.*

In Chapter 5, we revisit the 2^k-Hadamard transform and 2^k-bent functions, presenting further generalizations and new insights. A more generalized version of different Boolean function representation is shown in Figure 2.2. For more on Boolean functions and their cryptographic properties, one may refer to [19, 23].

In the next section, we introduce the basic concepts of quantum computing and quantum algorithms, relevant to proceed with this thesis.

2.2 Basics of quantum computing

The evolution of quantum computing is closely associated with the development of quantum mechanics, a theoretical framework devised to explain phenomena at the atomic

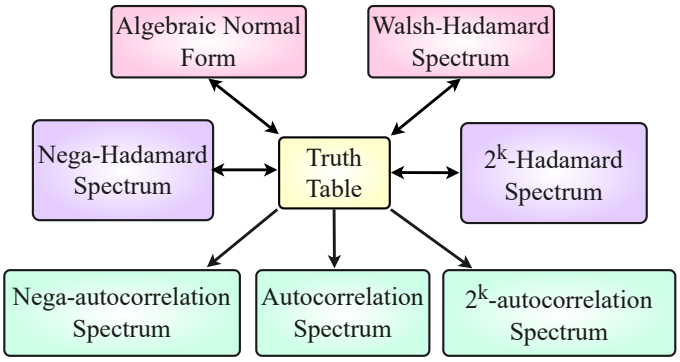


Figure 2.2: Different representations of Boolean functions - a generalized variation.

and subatomic scales. In 1900, Max Planck introduced the concept of energy quantization [74], marking the inception of quantum theory. Later in 1905, Einstein explained the photoelectric effect via the quantization of light [38], reinforcing the theory of quantum mechanics.

The formalization of quantum theory begins in 1920s, through Heisenberg's matrix mechanics [52] and Schrödinger's wave mechanics [83], later unified by Dirac's bra–ket formalism [29], which remains foundational in quantum information theory for representing quantum states and operations.

Core principles such as superposition and entanglement, which distinguish quantum systems from classical ones, eventually formed the basis of quantum computing. However, the notion of quantum computation did not emerge until the 1980s, when Richard Feynman noted the inefficiency of classical computers in simulating quantum systems. In his 1981 lecture Simulating Physics with Computers [39], Feynman proposed the idea of a quantum mechanical computer to address this challenge. This idea was soon formalized by David Deutsch, who proposed the concept of a universal quantum computer analogous to the classical Turing machine [26].

The field advanced rapidly in the 1990s with the introduction of powerful quantum algorithms, such as Deutsch-Jozsa (1992) [27], Simon's (1994) [88], Shor's factoring (1994) [87], and Grover's search (1996) [49], promising significant disruption in the existing security standards. By the early 2000s, attention shifted toward experimental realization, leading to the development of quantum computing architectures based on trapped ions, superconducting qubits, photonics, and neutral atoms. In parallel, major cloud-based platforms such as IBM Qiskit, Microsoft Azure Quantum, and Amazon Braket emerged to facilitate broader access to quantum technologies.

Currently, quantum computing resides in the Noisy Intermediate-Scale Quantum (NISQ) era [75], characterized by devices with tens to hundreds of qubits without full error correction. Though not fault-tolerant, these platforms allow practical experimentation with quantum algorithms and hybrid quantum–classical methods. The field's

trajectory, from theoretical abstraction to accessible experimentation, marks a milestone in interdisciplinary scientific progress, enabling transformative applications across computation, cryptography, and materials science.

In classical computation, the bit is the smallest unit of information. Analogously, the fundamental unit of quantum information is called *qubit* (quantum bit), which, unlike a classical bit, can exist in a superposition of the basis states $|0\rangle$ and $|1\rangle$.

Definition 2.2.1. *A single-qubit state $|\psi_1\rangle$ is a complex linear combination of the computational basis states $|0\rangle$ and $|1\rangle$, expressed as $|\psi_1\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$.*

The states $|0\rangle$ and $|1\rangle$ form an orthonormal basis for a two-dimensional Hilbert space. These correspond to the 2-dimensional column vectors $\binom{1}{0}$ and $\binom{0}{1}$, respectively. Hence, the state $|\psi_1\rangle$ can be represented as the vector $\binom{\alpha}{\beta}$.

Example 2.2.1. *Two commonly used single-qubit states are*

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \text{ and } |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

These form an orthonormal basis, as

$$\langle +|-\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\end{pmatrix}\frac{1}{\sqrt{2}}\begin{pmatrix}1\\-1\end{pmatrix} = 0.$$

The single qubit state $|\psi_1\rangle = \alpha|0\rangle + \beta|1\rangle$ can be expressed in the $\{|+\rangle, |-\rangle\}$ basis as $|\psi_1\rangle = \frac{\alpha+\beta}{\sqrt{2}}|+\rangle + \frac{\alpha-\beta}{\sqrt{2}}|-\rangle$, where the normalization condition $\left|\frac{\alpha+\beta}{\sqrt{2}}\right|^2 + \left|\frac{\alpha-\beta}{\sqrt{2}}\right|^2 = 1$ holds.

Quantum mechanics imposes fundamental limits on information extraction from an unknown quantum state. The *no-cloning theorem* prohibits duplicating an arbitrary quantum state. Moreover, precise determination of the parameters α and β is difficult through measurement, as any observation collapses the superposition state to one of the basis states. This process is known as *projective measurement*. The state $|\psi_1\rangle = \alpha|0\rangle + \beta|1\rangle$, when measured in the computational basis $\{|0\rangle, |1\rangle\}$, yields $|0\rangle$ with probability $|\alpha|^2$, and $|1\rangle$ with probability $|\beta|^2$. Similarly, when $|\psi_1\rangle$ is measured in $\{|+\rangle, |-\rangle\}$ basis, the state $|+\rangle$ appears with probability $\frac{1}{2}|\alpha+\beta|^2$, and $|-\rangle$ appears with probability $\frac{1}{2}|\alpha-\beta|^2$. Other measurement models such as POVMs and syndrome measurements exist, but are beyond the scope of this thesis.

Multi-qubit quantum states are constructed via the tensor product of single-qubit states. An n-qubit state can be written as:

$$|\psi_n\rangle = \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}}|\mathbf{x}\rangle, \quad \text{where } \sum_{\mathbf{x}\in\mathbb{F}_2^n} |\alpha_{\mathbf{x}}|^2 = 1,$$

requiring 2^n complex amplitudes. Alongside superposition, another non-classical property is *entanglement*. An n-qubit state is entangled if it cannot be factored into a tensor product of n single-qubit states.

Example 2.2.2. *The following 2-qubit states are entangled, known as the Bell states.*

$$|\Phi^{\pm}\rangle = \frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle), \quad |\Psi^{\pm}\rangle = \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle).$$

These states are fundamental to protocols including quantum teleportation, superdense coding, remote state preparation, the CHSH game, quantum key distribution, etc. Since these applications are not directly relevant to this thesis, we omit their details and refer readers to [71] for an in-depth discussion.

Quantum gates, unlike classical logic gates, are inherently reversible and represented by unitary matrices. An n-qubit quantum gate corresponds to a $2^n \times 2^n$ unitary matrix U, with inverse $\mathrm{U}^\dagger$ satisfying

$$\mathrm{U}^\dagger(\mathrm{U}\,|\psi\rangle) = \mathrm{U}(\mathrm{U}^\dagger\,|\psi\rangle) = |\psi\rangle\,.$$

Common single-qubit gates (2×2 unitary matrices) include the Pauli gates (X, Y, Z), Hadamard (H), and phase (S), defined as:

$$\mathrm{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \mathrm{Y} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \mathrm{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad \mathrm{H} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \mathrm{S} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}.$$

Among the single qubit quantum gates, the Hadamard gate is particularly significant. When applied to $|0\rangle$, it produces $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, an equal superposition of $|0\rangle$ and $|1\rangle$. Extending this, when two Hadamard gates $\mathrm{H}\otimes\mathrm{H} \equiv \mathrm{H}^{\otimes 2}$ are applied to $|00\rangle$ yields $\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$, an equal superposition over all possible two-qubit basis states. More generally, applying n Hadamard gates to the all-zero state $|0^n\rangle$ results in the state $\frac{1}{\sqrt{2^n}}\sum_{\mathbf{x}\in\mathbb{F}_2^n}|\mathbf{x}\rangle$, a uniform superposition over all possible n-qubit basis states. This phenomenon, known as *quantum parallelism*, is fundamental to the design of most quantum algorithms.

The two-qubit gates ($2^2 \times 2^2$ unitary matrices), such as CNOT, CZ, and SWAP, are given by:

$$\mathrm{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad \mathrm{CZ} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, \quad \mathrm{SWAP} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The CNOT gate flips the second (target) qubit if and only if the first (control) qubit is $|1\rangle$; otherwise, it leaves the target unchanged. Mathematically, $\mathrm{CNOT} : |x\rangle\,|y\rangle \mapsto$

$|x\rangle\,|x \oplus y\rangle$. The CZ gate applies the Z operation to the target qubit conditioned on the control being $|1\rangle$, while the SWAP gate exchanges the two input qubits: SWAP : $|x\rangle\,|y\rangle \mapsto |y\rangle\,|x\rangle$.

The Toffoli gate (also known as the doubly-controlled-NOT or CCNOT gate) is a three-qubit gate where the first two qubits are controls and the third is the target. It flips the target qubit if and only if both control qubits are $|1\rangle$.

$$\text{Toffoli} : |x, y, z\rangle \mapsto |x, y, z \oplus xy\rangle\,.$$

Other doubly-controlled Pauli gates, such as CCZ and CCY, are defined are defined in a similar manner, satisfying: $\text{CCX} = \text{CC(HZH)} = \text{CC(SYS}^\dagger\text{)}$. These doubly-controlled gates can be generalized to multi-controlled Pauli gates. The structural equivalence among them is illustrated in Figure 2.3.

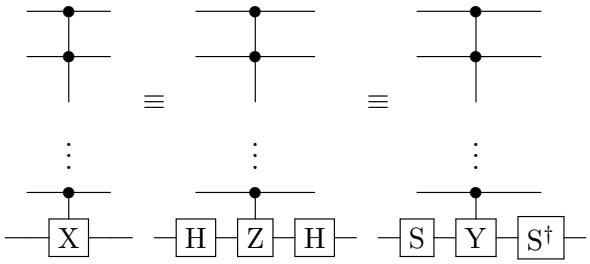


Figure 2.3: Equivalence between multi-controlled Pauli gates.

In classical computing, (2-input 1-output) NAND and NOR gates are considered universal as they can be used to construct any classical logic circuit. In contrast, quantum computing involves infinitely many quantum gates, including both single-qubit and multi-qubit ones, making it challenging to define a universal description. However, there exist certain gate sets that can approximate any unitary transformation on a quantum computer to an arbitrary degree of accuracy, known as the universal gate sets [17]. The most common examples of quantum universal gate sets include the Clifford+T gate set, rotation gates combined with the CNOT gate set, etc.

The Clifford group is generated by three gates: Hadamard (H), phase (S), and CNOT. This set is minimal, as removing any one gate would result in the inability to implement some Clifford operations. The Pauli gates are derived from the Clifford gates as shown in Equation 2.1.

$$\text{I} = \text{H}^{\otimes 2}, \quad \text{X} = \text{HZH}, \quad \text{Y} = \text{S}^\dagger\text{XS}, \quad \text{Z} = \text{S}^2 = \text{HXH}. \tag{2.1}$$

However, the Clifford gates alone do not constitute a universal set of quantum gates because certain gates, for example, the $\text{T} = \sqrt{\text{S}}$ gate, cannot be arbitrarily approximated using only Clifford operations. Therefore, the Clifford group, when augmented with the T gate, forms a universal quantum gate set for quantum computation.

Given a Boolean function $f \in \mathcal{B}_n^m$, the output $f(\mathbf{x}) = \mathbf{a}$ does not uniquely determine the input $\mathbf{x}$, making f inherently irreversible. In contrast, quantum operations are reversible by nature (excluding measurement), and thus the corresponding quantum circuit implementing f must be reversible, known as *quantum oracle*. Such a circuit operates on $n+m$ qubits, where the first n qubits represent the input state $|\mathbf{x}\rangle$, and the remaining m qubits, initialized to $|\mathbf{y}\rangle$, store the functional output. The functioning of U_f is given by:

$$U_f : |\mathbf{x}\rangle |\mathbf{y}\rangle \to |\mathbf{x}\rangle |\mathbf{y} \oplus \mathbf{f}(\mathbf{x})\rangle .$$

A schematic diagram of a quantum oracle is shown in Figure 2.4.

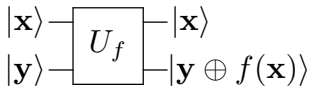


Figure 2.4: Quantum oracle for an unknown Boolean function f.

If $\mathbf{y} = 0^m$, then the oracle simplifies to:

$$U_f : |\mathbf{x}\rangle |0^m\rangle \to |\mathbf{x}\rangle |\mathbf{f}(\mathbf{x})\rangle$$

meaning the second register holds the output of f in its original form. Furthermore, for $f \in \mathcal{B}_n$ and $|y\rangle = |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, we have:

$$U_f : |\mathbf{x}\rangle |-\rangle \to |\mathbf{x}\rangle \frac{|0 \oplus f(\mathbf{x})\rangle - |1 \oplus f(\mathbf{x})\rangle}{\sqrt{2}}.$$

Observe that if $f(\mathbf{x}) = 0$, the resulting state remains $|\mathbf{x}\rangle |-\rangle$, whereas if $f(\mathbf{x}) = 1$, the resulting state becomes $(-1) |\mathbf{x}\rangle |-\rangle$. In other words, the sign of the output is determined by $f(\mathbf{x})$, yielding the transformation:

$$U_f : |\mathbf{x}\rangle |-\rangle \to (-1)^{f(\mathbf{x})} |\mathbf{x}\rangle |-\rangle .$$

This phenomenon is known as phase kickback.

In the *black-box model* of quantum computing, the quantum oracle corresponding to an unknown Boolean function is provided, allowing specific input-output queries, thus also known as the query model. The goal is to determine specific properties of f using the minimum number of oracle queries. Several foundational quantum algorithms, including those by Deutsch-Jozsa [27], Grover [49], Simon [88], and Shor [87], are built on this black-box paradigm, as described in detail in the following subsection.

2.2.1 Quantum algorithms

Here, we discuss a few quantum algorithms that will provide certain ideas in this domain and they will also be relevant for this thesis.

Deutsch-Jozsa algorithm [27]

Suppose, we are given the oracle access of an unknown Boolean function $f \in \mathcal{B}_n$ with the promise that f is either constant or a balanced Boolean function. The goal is to determine which one it is, with the minimum possible number of queries to the oracle U_f. To resolve this problem classically, it might take $2^{n-1}+1$ many queries in the worst case. However, Deutsch-Jozsa algorithm can resolve the problem with certainty with just a single query to the oracle U_f, by producing a superposition state where the amplitude of an individual state is given by the normalized Walsh-Hadamard transform of the function at that point

$$2^{-n} \sum_{\mathbf{x},\mathbf{y} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x}\cdot\mathbf{y}} |\mathbf{y}\rangle = 2^{-n/2} \sum_{\mathbf{y} \in \mathbb{F}_2^n} W_f(\mathbf{y}) |\mathbf{y}\rangle .$$

Essentially, for a constant Boolean function, the complete Walsh spectrum is supported over the all-zero point, makes it certain to observe the state $|0^n\rangle$, upon measurement. However, for a balanced Boolean function, the Walsh transform at the all-zero point is 0, thus the state $|0^n\rangle$ never appears when measured. Therefore, the presence or absence of the all-zero state in the measurement results concludes with certainty whether f is constant or balanced, using only a single query to the oracle U_f. A schematic diagram of the corresponding quantum circuit is shown in Figure 2.5.

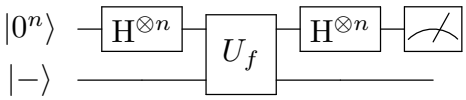


Figure 2.5: Quantum circuit for Deutsch-Jozsa algorithm [27].

In 2017, a modified version of this algorithm, termed the extended Deutsch-Jozsa algorithm [40] was proposed in [40]. There, instead of using all Hadamard gates before measurement, as in [27], a combination of Hadamard and nega-Hadamard gates subject to a bit pattern $\mathbf{c} \in \mathbb{F}_2^n$ was applied. Observe that, for $\mathbf{c} = 0^n$, it becomes the standard Deutsch-Jozsa algorithm, producing the superposition state with the normalized Walsh-Hadamard transform value of the respective states as the individual amplitudes. On the other extreme, when $\mathbf{c} = 1^n$, only the nega-Hadamard gates are applied. As a result, the algorithm produces the superposition where the amplitude of the individual states are given by the normalized nega-Hadamard transform of the respective states,

$$2^{-n} \sum_{\mathbf{x},\mathbf{y} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x}\cdot\mathbf{y}} (i)^{wt(\mathbf{x})} |\mathbf{y}\rangle = 2^{-n/2} \sum_{\mathbf{y} \in \mathbb{F}_2^n} N_f(\mathbf{y}) |\mathbf{y}\rangle .$$

A schematic diagram of the quantum circuit corresponding to $\mathbf{c} = 1^n$ is shown in Figure 2.6. In Chapter 5, we further generalize the Deutsch-Jozsa algorithm, and show the existing variations referred in [27] and [40] as its sub-cases.

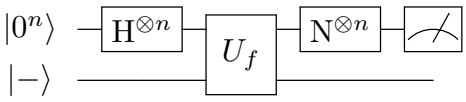


Figure 2.6: Quantum circuit for extended Deutsch-Jozsa algorithm with $\mathbf{c} = 1^n$.

Grover's algorithm [49]

Suppose, we are given oracle access to an unknown Boolean function $f \in \mathcal{B}_n$ with the promise that $f(\mathbf{a}) = 1$, for a unique input $\mathbf{a} \in \mathbb{F}_2^n$, and $f(\mathbf{x}) = 0$ for all $\mathbf{x} \neq \mathbf{a}$. The objective is to identify $\mathbf{a} \in \mathbb{F}_2^n$ using as few queries to the oracle U_f as possible. Classically, this requires $\mathcal{O}(2^n)$ queries in the worst case. However, Grover's search algorithm solves this problem with probability close to 1, using only $\mathcal{O}(\sqrt{2^n})$ queries, leveraging a technique known as amplitude amplification. A schematic of Grover's quantum circuit is shown in Figure 2.7.

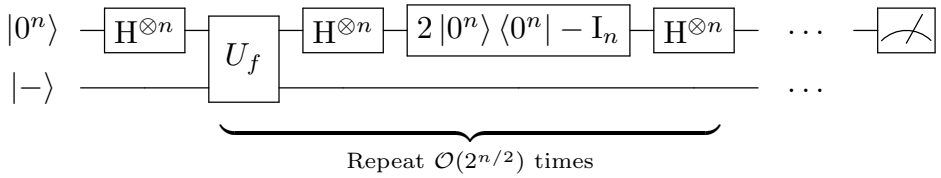


Figure 2.7: Quantum circuit for Grover's search algorithm [49].

The algorithm begins by preparing an equal superposition over all 2^n inputs via Hadamard gates ($\mathrm{H}^{\otimes n}$) applied to $|0^n\rangle$. The oracle U_f then applies a phase flip to the solution state $|\mathbf{a}\rangle$, i.e., multiplies its amplitude by -1 while leaving others unchanged, effectively marking it. Grover's diffusion operator then reflects all the quantum states about the average amplitude, amplifying the marked state's amplitude while suppressing the rest. Repeating this process $\mathcal{O}(\sqrt{2^n})$ times boosts the amplitude of $|\mathbf{a}\rangle$ close to 1, enabling its identification with high probability upon measurement.

Initially, the amplitude of the target state $|\mathbf{a}\rangle$ in a uniform superposition is $1/\sqrt{2^n}$, resulting in a success probability of $1/2^n$ upon direct measurement. Rather than increasing this probability directly, Grover's algorithm amplifies the amplitude of the target state through repeated iterations, thereby achieving a quadratic speedup in query complexity, from $\mathcal{O}(2^n)$ to $\mathcal{O}(\sqrt{2^n})$.

In the context of symmetric key cryptography, for a key of length k, classical brute-force search requires $\mathcal{O}(2^k)$ queries in the worst case. Grover's algorithm reduces this to $\mathcal{O}(2^{k/2})$, halving the effective key length from k to $k/2$. Numerous works have explored Grover-based exhaustive key recovery attacks and associated quantum resource estimations for both block cipher [48] and stream ciphers [11, 32].

Simon's algorithm [88]

Suppose we are given oracle access to an unknown Boolean function $f \in \mathcal{B}_n^m$, with the promise that there exists a k-dimensional subspace $S \subseteq \mathbb{F}_2^n$ such that for all $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$, $f(\mathbf{x}) = f(\mathbf{y})$ if and only if $\mathbf{x} \oplus \mathbf{y} \in S$. It follows that f is a 2^k-to-1 Boolean function, referred to as a k-Simon function. The objective is to find a basis for S using as few queries to U_f as possible. Classically, this requires $\mathcal{O}(\sqrt{2^n})$ queries (equivalent to birthday bound), but Simon's quantum algorithm computes a basis for S using only $\mathcal{O}(n)$ queries to the oracle U_f. A schematic diagram of Simon's quantum circuit is shown in Figure 2.8.

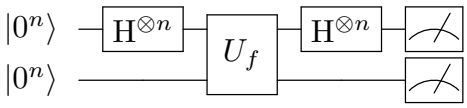


Figure 2.8: Quantum circuit for Simon's hidden-shift finding algorithm [88].

The algorithm begins with two n-qubit registers initialized to $|0^n\rangle |0^n\rangle$. Applying Hadamard gates, $\mathrm{H}^{\otimes n}$ to the first register creates a uniform superposition of all n-bit inputs. This state is then passed through the oracle U_f, and the first register is passed through Hadamard gates, $\mathrm{H}^{\otimes n}$, yielding: $2^{-n} \sum_{\mathbf{y} \in \mathbb{F}_2^n} \left(|\mathbf{y}\rangle \otimes \left(\sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{\mathbf{x}\cdot\mathbf{y}} |f(\mathbf{x})\rangle \right) \right)$. Therefore, the probability of observing a particular bit-pattern $\mathbf{y} \in \mathbb{F}_2^n$ is given by:

$$\mathcal{P}(|\mathbf{y}\rangle) = \left\| \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{\mathbf{x}\cdot\mathbf{y}} |f(\mathbf{x})\rangle \right\|^2 .$$

Let $A \subseteq \mathbb{F}_2^m$ be the set of distinct outputs of f. For each $\mathbf{z} \in A$, there are 2^k pre-images such that $f(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{s_i}) = \mathbf{z}$ for all $\mathbf{s_i} \in S$. Hence, the above expression becomes:

$$\begin{aligned} \mathcal{P}(|\mathbf{y}\rangle) = \left\| \frac{1}{2^n} \sum_{\mathbf{z} \in A} \sum_{\mathbf{s_i} \in S} (-1)^{(\mathbf{x} \oplus \mathbf{s_i})\cdot\mathbf{y}} |\mathbf{z}\rangle \right\|^2 &= \left\| \frac{1}{2^n} \sum_{\mathbf{z} \in A} (-1)^{\mathbf{x}\cdot\mathbf{y}} \sum_{\mathbf{s_i} \in S} (-1)^{\mathbf{s_i}\cdot\mathbf{y}} |\mathbf{z}\rangle \right\|^2 \\ &= \left\| \frac{1}{2^n} \sum_{\mathbf{z} \in A} (-1)^{\mathbf{x}\cdot\mathbf{y}} \prod_{\mathbf{s_i} \in S_b} \left(1 + (-1)^{\mathbf{s_i}\cdot\mathbf{y}}\right) |\mathbf{z}\rangle \right\|^2 . \end{aligned}$$

This probability is non-zero if and only if $s_i \cdot y = 0$ for all $s_i \in S_b$, i.e., $y \in S^\perp$. In this case, the product becomes 2^k, and

$$\mathcal{P}(|\mathbf{y}\rangle) = \left\| \frac{1}{2^n} \sum_{\mathbf{z} \in A} (-1)^{\mathbf{x}\cdot\mathbf{y}} 2^k |\mathbf{z}\rangle \right\|^2 = \left\| \frac{1}{2^{n-k}} \sum_{\mathbf{z} \in A} (-1)^{\mathbf{x}\cdot\mathbf{y}} |\mathbf{z}\rangle \right\|^2 = \frac{|A|}{2^{2(n-k)}} = \frac{1}{2^{n-k}}.$$

Thus, the observed bit-strings $\mathbf{y} \in \mathbb{F}_2^n$ are uniformly distributed over $S^\perp$. If k is known, observing $n-k$ linearly independent such strings $\mathbf{y_1}, \ldots, \mathbf{y_{n-k}}$ satisfying $\mathbf{y_i} \cdot \mathbf{s} = 0$ for all $\mathbf{s} \in S$ suffices to form a basis for S. If k is unknown, one may collect $n-1$ such observations (possibly with dependencies) and determine the null space of the matrix $Y = \begin{pmatrix} \mathbf{y_1} & \mathbf{y_2} & \cdots & \mathbf{y_{n-1}} \end{pmatrix}^T$ by reducing Y to row-echelon form over $\mathbb{F}_2$.

Although Simon's algorithm is deterministic, retrieving the hidden shift space from a noisy quantum device is challenging and may require exponentially many oracle queries. The case of 1-dimensional subspaces is thoroughly analyzed in [64], while the generalization to k-dimensional subspaces is addressed in [33].

Shor's algorithm [87]

Given an odd composite integer N, the objective is to find a non-trivial factor of N. Shor's algorithm achieves this by selecting a random integer $a \in \mathbb{Z}_N^*$ and determining its order r modulo N, i.e., the smallest positive integer such that $a^r \equiv 1 \mod N$. If r is even and $a^{r/2} \not\equiv \pm 1 \mod N$, then $\gcd(a^{r/2} \pm 1, N) \geq 1$ yields a non-trivial factor of N.

The order-finding problem reduces to finding the period of the function $f(x) = a^x \mod N$, which is computed using the quantum Fourier transform (QFT). The algorithm begins with two quantum registers initialized to $|0^n\rangle |0^m\rangle$. Applying Hadamard gates to the first register creates a uniform superposition over all n-bit inputs. The unitary transformation U_f maps $|\mathbf{x}\rangle |0\rangle \mapsto |\mathbf{x}\rangle |a^{\mathbf{x}} \mod N\rangle$. Measuring the second register collapses the system to a superposition over all $\mathbf{x}$ such that $a^{\mathbf{x}} \mod N = \mathbf{y}$ for some fixed $\mathbf{y}$, i.e., over a coset of the period r: $\frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} |x_0 + jr\rangle$, where $M \approx 2^n/r$.

Applying the QFT on the first register transforms this periodic state to:

$$\frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} |x_0 + jr\rangle \to \sum_{k=0}^{2^n-1} c_k |k\rangle ,$$

where the amplitudes c_k are sharply peaked near integer multiples of $2^n/r$. Upon measurement, one obtains an outcome k, and a continued fraction expansion of $k/2^n \approx s/r$ for $s \in \mathbb{N}$, allows recovery of the period r. If r is even and satisfies the non-triviality condition, then $\gcd(a^{r/2} \pm 1, N)$ yields a non-trivial factor of N. A schematic diagram of the quantum circuit for Shor's algorithm is shown in Figure 2.9.

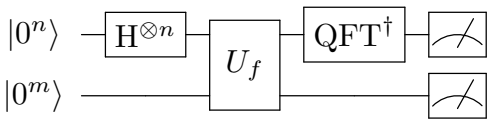


Figure 2.9: Quantum circuit for Shor's factoring algorithm [87].

Shor's algorithm runs in polynomial time, in stark contrast to the best known clas-

sical methods, such as the general number field sieve, which require sub-exponential time $\exp\left((\log N)^{1/3}(\log\log N)^{2/3}\right)$. The quantum advantage arises from evaluating $f(x)$ in superposition and extracting the period via interference using the QFT, which is infeasible classically.

Although the impact of Shor's factorization algorithm on public-key cryptographic standards is potentially devastating, mounting a full-scale attack remains elusive due to the resource constraints of current quantum hardware. However, recent developments have shown that RSA-2048 could be broken using fewer than one million qubits [44].

Forrelation algorithm [1, 2]

The Forrelation problem plays a central role in understanding the separation between bounded-error quantum and classical probabilistic models in the black-box setting. It quantifies the correlation between the truth table of one Boolean function and the Fourier spectrum (Walsh-Hadamard transform) of another, defined as follows.

Definition 2.2.2 ([1])**.** *Let $f_1, f_2 \in \mathcal{B}n$. The (2-fold) Forrelation of f_1 and f_2, denoted $\Phi f_1, f_2$, is formulated as*

$$\Phi_{f_1,f_2} = \frac{1}{2^n}\sum_{\mathbf{x_1}\in\mathbb{F}_2^n}(-1)^{f_1(\mathbf{x_1})}W_{f_2}(\mathbf{x_1}) = \frac{1}{2^{3n/2}}\sum_{\mathbf{x_1},\mathbf{x_2}\in\mathbb{F}_2^n}(-1)^{f_1(\mathbf{x_1})}(-1)^{\mathbf{x_1}\cdot\mathbf{x_2}}(-1)^{f_2(\mathbf{x_2})}.$$

This definition generalizes to $k(>2)$ many Boolean functions $f_1, \ldots, f_k \in \mathcal{B}_n$, yielding the k-fold Forrelation

$$\Phi_{f_1,\ldots,f_k} = \frac{1}{2^{\frac{(k+1)n}{2}}}\sum_{\mathbf{x_1},\ldots,\mathbf{x_k}\in\mathbb{F}_2^n}(-1)^{f_1(\mathbf{x_1})}(-1)^{\mathbf{x_1}\cdot\mathbf{x_2}}(-1)^{f_2(\mathbf{x_2})}\ldots(-1)^{\mathbf{x_{k-1}}\cdot\mathbf{x_k}}(-1)^{f_k(\mathbf{x_k})}.$$

Note that the 2-fold Forrelation can also be viewed as a symmetric bilinear form:

$$\Phi_{f_1,f_2} = \frac{1}{2^{3n/2}}\left[\mathbf{f_1}^T\widehat{\mathrm{H}}_n\mathbf{f_2}\right],$$

where $\mathbf{f_1}$ and $\mathbf{f_2}$ are $2^n \times 1$ column matrices resembling the output columns of the truth table of respective Boolean functions

$$\mathbf{f_1} = \left[(-1)^{f_1(0^n)},\ldots,(-1)^{f_1(1^n)}\right]^T, \quad \mathbf{f_2} = \left[(-1)^{f_2(0^n)},\ldots,(-1)^{f_2(1^n)}\right]^T$$

and $\widehat{\mathrm{H}}_n$ is the $2^n \times 2^n$ classical Sylvester Hadamard matrix.

Suppose we are given oracle access to two unknown Boolean functions $f_1, f_2 \in \mathcal{B}n$, with the promise that either $|\Phi_{f_1,f_2}| \leq \frac{1}{100}$ or $\Phi_{f_1,f_2} \geq \frac{3}{5}$. The objective is to distinguish between these two cases using as few oracle queries as possible. As shown in [2], any

classical randomized algorithm requires at least $\Omega\left(\frac{2^{n/2}}{n}\right)$ queries in the worst case, while a bounded-error quantum algorithm can resolve this using a single query to each oracle.

In [2], two quantum algorithms were proposed for estimating k-fold Forrelation: one using k sequential queries and the other using $\lceil \frac{k}{2} \rceil$ parallel queries. Both assume black-box access to the functions $f_1, \ldots, f_k$.

- **The k-query algorithm:** The initial state $|0^n\rangle |-\rangle$ is evolved through alternating layers of Hadamard transforms and oracle calls:

  $$\mathrm{H}^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n} \to U_{f_2} \to \cdots \to U_{f_k} \to \mathrm{H}^{\otimes n}.$$

  In the final state, the amplitude corresponding to $|0^n\rangle$ equals $\Phi_{f_1,\ldots,f_k}$, so measuring the first n qubits yields all zero state with probability $(\Phi_{f_1,\ldots,f_k})^2$. A schematic for $k = 3$ is shown in Figure 2.10.

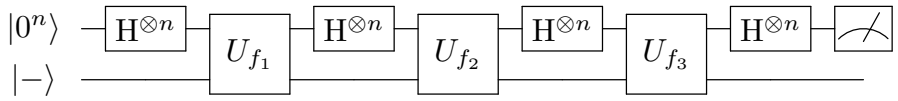


Figure 2.10: Quantum circuit for 3-fold Forrelation using 3 sequential queries [2].

- **The $\lceil \frac{k}{2} \rceil$-query algorithm:** An additional driving qubit, initialized to $|+\rangle$, controls two independent branches applied to the remaining $n+1$ qubits, based on its state. If it is $|0\rangle$, it evolves through $\mathrm{H}^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n} \to \ldots \to U_{f_{\lceil \frac{k}{2} \rceil}} \to \mathrm{H}^{\otimes n}$, and if the driving qubit is $|1\rangle$, the state evolves via $\mathrm{H}^{\otimes n} \to U_{f_k} \to \ldots \to \mathrm{H}^{\otimes n} \to U_{f_{\lceil \frac{k}{2} \rceil + 1}}$. The sequence of controlled Hadamard gates and oracles on each branch produces interference, and the driving qubit is finally measured in the Hadamard basis. The probability of observing outcome $|0\rangle$ is $\frac{1}{2}(1 + \Phi_{f_1,\ldots,f_k})$. A schematic for $k = 3$ is shown in Figure 2.11.

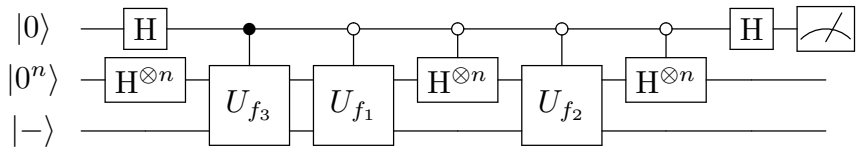


Figure 2.11: Quantum circuit for 3-fold Forrelation using 2 parallel queries [2].

Throughout this thesis, we only focus on 2-fold and 3-fold Forrelation. For $k = 3$, we refer to the k-query and $\lceil \frac{k}{2} \rceil$-query algorithms as $\mathcal{A}^{(3,3)}$ and $\mathcal{A}^{(3,2)}$, respectively.

In the following section, we discuss about quantum circuit construction.

2.3 Quantum circuit construction

Quantum gates are the fundamental building blocks of any quantum circuit. In Section 2.2, we introduced the Toffoli gate, a three-qubit quantum gate where the first two qubits act as control qubits and the third as the target. The Pauli-X gate is applied to the target qubit if and only if both control qubits are in the state 1. Consequently, the gate is also referred to as the doubly-controlled NOT gate or the CCX gate. Its functionality is given by:

$$\text{Toffoli} : |x_1, x_2, y\rangle \rightarrow |x_1, x_2, y \oplus x_1 x_2\rangle \,.$$

This concept generalizes to the multi-controlled Toffoli (MCT) gate, where instead of two, there are multiple (n many) control qubits. In a similar manner, the X gate is applied to the target if an only if all the n control qubits are 1, represented as:

$$n\text{-MCT} : |x_1, x_2, \ldots, x_n, y\rangle \rightarrow |x_1, x_2, \ldots, x_n, y \oplus x_1 x_2 \ldots x_n\rangle \,.$$

It is direct to observe that MCT gates implement a classical AND operation in the quantum setting. As such, they introduce nonlinearity into quantum circuits and are essential for Boolean functions implementation, arithmetic circuit construction, and quantum error correction.

However, MCT gates are not native to existing quantum hardware and must be decomposed into simpler gate sets. Typically, MCT gates are first reduced to circuits containing standard Toffoli gates, which are then further decomposed into gates from a universal gate set. Among these, the Clifford+T gate set is one of the most commonly used in practice.

As mentioned earlier, we are currently in the Noisy Intermediate-Scale Quantum (NISQ) era, where quantum gates are inherently error-prone. This makes efficient decomposition of complex quantum gates into universal gate sets critical, particularly for near-term devices with limited coherence times. Optimization efforts often aim to minimize resource metrics such as circuit depth, gate count, and qubit usage.

The depth of a quantum circuit represents its execution time, while the number of qubits reflects memory requirements. In practice, increasing one of these resources may reduce the other up to a certain extent, leading to a fundamental trade-off between space and depth, as discussed in Chapter 6. Circuit depth is measured by the number of sequential gate layers, assuming gates acting on disjoint qubits can be executed in parallel.

Among all quantum gates, the T gate is assumed to be the most expensive to implement, especially under fault-tolerant quantum computation using surface codes, due to the costly magic state distillation required. Therefore, reducing both T count and T depth has become a key objective. Like circuit depth, T depth refers to the number of

layers of T gates under the assumption of parallel execution on disjoint qubits. Minimizing T depth helps reduce latency and execution time, enhancing the practicality of quantum circuits.

In the last couple of decades, multiple attempts have been made towards optimizing the Clifford+T decomposition of the basic Toffoli gate. While early work focused on reducing T depth with a fixed T count of 7, recent developments use measurement-based uncomputation to achieve Toffoli decompositions with only 4 T gates. Here, we provide a comprehensive overview of existing benchmarks for standard Toffoli decompositions, highlighting the state-of-the-art results in T count, T depth, and ancilla requirements, as summarized in Table 2.1.

In [71, Chapter 4, Section 4.3], the author presents one of the earliest attempts of a Clifford+T decomposition of a standard Toffoli gate using 7 T gates and 9 Clifford gates with a T depth of 6 (Figure 2.12).

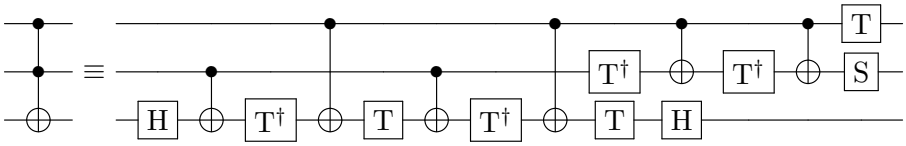


Figure 2.12: Toffoli decomposition with T depth 6, T count 7, without any ancilla [71].

Although, it is direct to observe that with a simple manipulation of gate ordering, the T depth can be reduced to 4 without altering the gate count. In 2013, Amy et al. [4, Section 6] also proposed a Toffoli decomposition (Figure 2.13) with a T depth of 4 that used one fewer Clifford gate and reduced the overall circuit depth from 12 to 8.

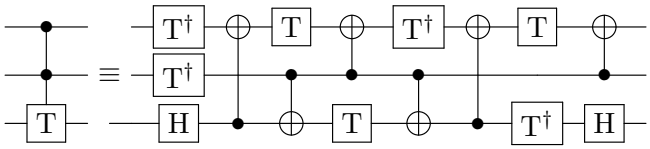


Figure 2.13: Toffoli decomposition with T depth 4, T count 7, without any ancilla [4].

In the same paper, the authors applied a meet-in-the-middle algorithm to present a Toffoli decomposition (Figure 2.14) with a T depth of 3 and an overall depth of 9. For an exact Toffoli decomposition (without measurement-based feedback), this is the lowest T depth that one can achieve without using any ancilla qubit.

Using one ancilla qubit, the authors also proposed a Toffoli decomposition circuit (Figure 2.15) using 7 T gates and 12 Clifford gates, achieving a T depth 2. Later, in the same year, Selinger [84, Section 2] proposed a Toffoli gate decomposition circuit (Figure 2.16) with the lowest possible T depth of 1, using 4 additional ancilla qubits and 18 Clifford gates.

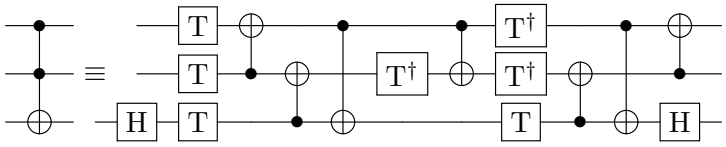


Figure 2.14: Toffoli decomposition with T depth 3, T count 7, without any ancilla [4].

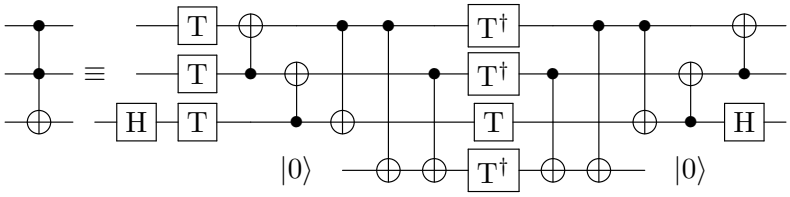


Figure 2.15: Toffoli decomposition with T depth 2, T count 7, using a single ancilla [4].

In [84], Selinger also proposed a doubly-controlled $-iX$ gate, which differs from the Toffoli gate only by a controlled-$S^\dagger$ gate between the two control qubits (shown in Figure 2.17). In the same year, Jones [58] modified Selinger's circuit by employing a measurement-based uncomputation technique to implement an exact Toffoli gate, using a single ancilla qubit with a T count of 4 and achieving a T depth of 1. Later, in 2020, Jaques et al. [56] integrated the designs of Selinger [84] and Jones [58], proposing a modified circuit for single Toffoli decomposition utilizing measurement-based updates, as illustrated in Figure 2.18. This circuit requires one ancilla qubit, uses four T gates with a T depth of 1, and has an overall depth of 8.

Earlier in 2018, Gidney introduced the concept of logical-AND [43] using measurement-based uncomputation, which has a T count of 4 and a T depth of 2, without using any additional ancilla (see Figure 2.19). When multiple logical-AND circuits are employed within the same quantum circuit, all the initial T gates are applied simultaneously at the beginning of the circuit, with a T depth of 1. Consequently, the effective T depth of the

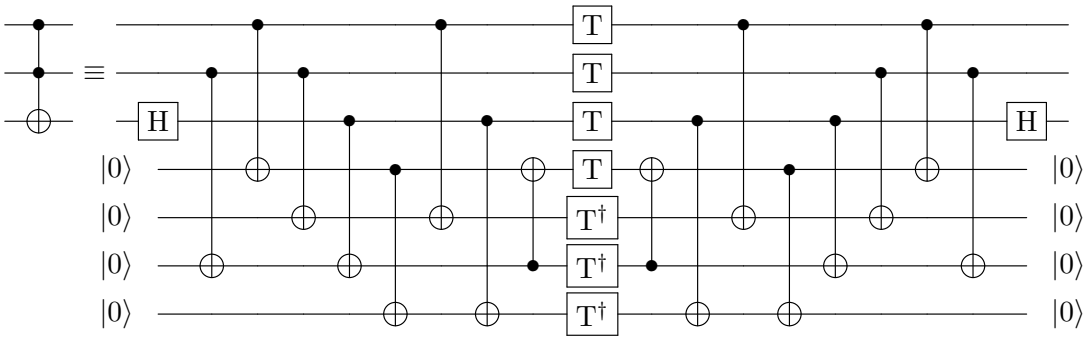


Figure 2.16: Toffoli decomposition with T depth 1, T count 7, using 4 ancilla [84].

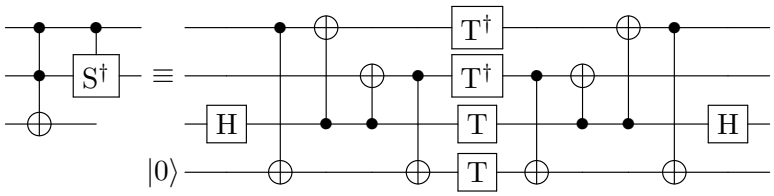


Figure 2.17: Doubly-controlled $-iX$ decomposition with T depth 1 [84].

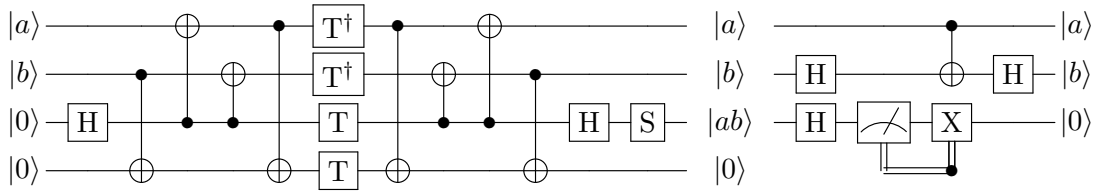


Figure 2.18: Measurement based Toffoli decomposition with T depth 1 [56].

logical-AND decomposition reduces to $1 + \epsilon$. This is marked with (*) in the last row of the Table 2.1. Table 2.1 summarizes the state-of-the-art resource requirements (T count,

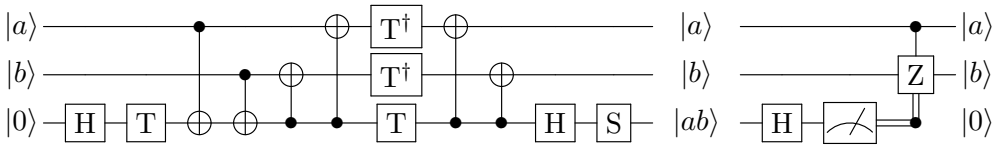


Figure 2.19: Measurement based Toffoli decomposition using logical-AND [43].

T depth, and ancilla count) for the Clifford+T decomposition of a basic (2-controlled) Toffoli gate.

Building upon these optimizations, in Chapter 6 we present exact space–depth trade-offs for the decomposition of multi-controlled Toffoli (MCT) gates, ultimately leading to an optimal T depth MCT construction. For resource estimation, we primarily rely on the measurement-based Toffoli decomposition by Jaques et al. [56], which uses 4 T gates and achieves T depth 1 (see Figure 2.18), and the logical-AND-based decomposition by Gidney [43] (Figure 2.19).

Under a different set-up, the optimal T depth can be reduced to as low as one by employing different constructions, such as using a large number of ancilla qubits and CCZ resource states, and leveraging teleportation-based techniques, as outlined in [46]. However, such explorations are beyond the scope of this thesis. Here, we focus on a specific computational model and establish optimality within that framework, as discussed in Chapter 6.

Given the Algebraic Normal Form (ANF) of a Boolean function $f \in \mathcal{B}_n$, the linear terms are implemented using CNOT gates, with the corresponding input variable as the

References	Ancilla count	T count	T depth	Clifford count	Circuit depth
Amy et al. [4]	0	7	4	8	8
Amy et al. [4]	0	7	3	9	9
Amy et al. [4]	1	7	2	12	11
Selinger [84]	4	7	1	18	8
Jaques et al. [56]	1	4	1	11	8
Gidney [43]*	0	4	$1+1$	9	9

Table 2.1: Summarizing state-of-the-art results related to Toffoli decompositions.

control and the output qubit as the target. Similarly, the non-linear terms are realized using MCT gates, where the relevant input variables serve as control qubits and the output as the target. These MCT gates are first simplified using Clifford and Toffoli gates and then further decomposed into the Clifford+T gate set. We again assume the measurement-based Toffoli decomposition for this step, which avoids additional resources for uncomputation.

To reduce overall circuit depth, ancilla qubits are introduced, making the optimization of space–depth trade-offs a central focus. In Chapter 7, we propose an optimal T depth (via Toffoli) quantum circuit construction technique derived from the ANF of an arbitrary Boolean function.

While recent advances in surface code implementations suggest that T depth may not be the sole bottleneck in quantum circuit design [44, 45], it nonetheless remains a critical metric. The overall cost of fault-tolerant quantum computation involves several factors, including T count, qubit overhead, and scheduling strategies such as resource state distillation and teleportation. However, in the context of near-term quantum devices with limited coherence times, T depth still plays a significant role in determining circuit latency and execution feasibility. In this thesis, we present our results under the common assumption that T depth is among the most resource-intensive parameters, and optimizing it is crucial for practical quantum circuit implementation.

Chapter 3

Quantum algorithms in exploring Boolean functions' spectra using Forrelation

Showing separation between the probabilistic classical model and the bounded-error quantum model has been long-standing problem, recently resolved in [10]. In this context, the Forrelation problem has emerged as a concept of significant interest. First introduced by Aaronson et al. [1] and later formalized in their seminal work [2], Forrelation exhibits a constant versus exponential query complexity separation between the bounded error quantum and the classical probabilistic models. More recently, Tal et al. [93] extended this idea to achieve even greater separations between the two models.

While the problem is easily solvable in the quantum setting, the key contribution of these works lies in proving that no classical probabilistic algorithm can solve it efficiently. Specifically, Aaronson et al. [2] showed that distinguishing whether the Forrelation value is greater than $3/5$ or if its absolute value is less than $1/100$, requires $\Omega\left(\frac{2^{n/2}}{n}\right)$ queries in the classical probabilistic model, where n is the number of input variables of the underlying Boolean functions. In contrast, the same task can be accomplished with a constant number of queries in the bounded-error quantum model. To establish this result, they proposed two quantum algorithms assuming oracle access to all the underlying Boolean functions. Further details on these algorithms and their query complexities are provided in Section 2.2.1.

Although Forrelation was originally introduced to establish a separation between complexity classes, in this chapter (and throughout the thesis), we investigate it from a Boolean function perspective. Specifically, we focus on evaluating various cryptographic spectra of a Boolean function in the black-box model. We present an improved sampling strategy for the Walsh-Hadamard spectrum, offering a constant factor advantage over the Deutsch-Jozsa algorithm, with implications for resiliency checking. Additionally, we

propose a method for estimating crosscorrelation (and hence autocorrelation) values at arbitrary points, improving upon the existing algorithms. Furthermore, by leveraging superpositions over linear functions, we obtain a crosscorrelation sampling technique, which to the best of our knowledge, is the first crosscorrelation sampling algorithm with constant query complexity.

In summary, the main contribution of this paper is in studying different fundamental (as well as cryptographically significant) spectra of Boolean functions using quantum algorithms related to Forrelation. In the process, the results we obtain are new or superior than the existing results [12, 20]. The spectral properties studied here have wide-ranging applications, particularly in cryptology, where the security of a Boolean functions as cryptographic primitives is characterized through various spectral parameters [12, 18, 20, 80, 81, 98].

The organization of this chapter and its section-wise contributions are as follows.

- In Section 3.1, we explore bent-duality based promise problems which can be reduced to different desirable instantiations of the 2-fold Forrelation. We begin with studying the extreme cases, where the Forrelation achieves the maximum value 1, and where it approaches 0, and construct a class of promise problems based on bent function duality. Here, we observe that in certain situations, the standard Deutsch-Jozsa algorithm works as efficiently as the Forrelation set-up, while in the other (majority of the) cases, there is no obvious way of achieving the same using the Deutsch-Jozsa algorithm.

- In Section 3.2, we concentrate on the 3-fold Forrelation which we use as a unifying framework for evaluating different spectra of Boolean functions. We begin with showing that for Boolean functions $f_1, g, f_2 \in \mathcal{B}_n$, the 3-fold Forrelation can be expressed a
$$\Phi_{f_1,g,f_2} = \frac{1}{2^n} \sum_{\mathbf{x}\in\mathbb{F}_2^n} W_{f_1}(\mathbf{x})(-1)^{g(\mathbf{x})} W_{f_2}(\mathbf{x}).$$
By fixing $f_1 = f_3 = f$, the expression reduces to a weighted sum of squared Walsh spectra, with weights determined by g. This formulation enables an efficient sampling of Walsh-Hadamard spectrum and presents improvement in checking resiliency of Boolean functions.

- In Section 3.3, we focus on the crosscorrelation spectrum $C_{f,g}(\mathbf{x})$ of two Boolean functions $f, g \in \mathcal{B}_n$, and connect it to the 3-fold Forrelation set-up. First, we propose an algorithm to estimate the crosscorrelation value at a specific point, using the result from [80, Theorem 3.1], which relates crosscorrelation to the product of Walsh-Hadamard spectra. Then, we extend this to a crosscorrelation sampling algorithm by replacing the fixed function f_2 with a uniform superposition over all linear functions, using n additional qubits. This is enabled by identifying the correspondence between rows of the n-qubit Hadamard matrix, $\mathrm{H}^{\otimes n}$ and the 2^n

linear Boolean functions. Finally, we address the problem of checking if two functions are uncorrelated up to degree m, i.e., whether $C_{f,g}(\mathbf{u}) = 0$ for all $\mathbf{u}$ with $wt(\mathbf{u}) \leq m$, using Dicke states. To the best of our knowledge, this is the first constant query sampling algorithm for the crosscorrelation spectrum, further reinforcing Forrelation as a unifying paradigm for analyzing spectral properties of Boolean functions.

- Section 3.4 concludes the chapter with a brief summary of our contributions with future research possibilities.

3.1 2-fold Forrelation and bent duality

The Forrelation of two functions $f_1, f_2 \in \mathcal{B}_n$ is formally defined as

$$\Phi_{f_1,f_2} = \frac{1}{2^n} \sum_{\mathbf{x_1} \in \mathbb{F}_2^n} (-1)^{f_1(\mathbf{x_1})} W_{f_2}(\mathbf{x_1}) = \frac{1}{2^{3n/2}} \sum_{\mathbf{x_1},\mathbf{x_2} \in \mathbb{F}_2^n} (-1)^{f_1(\mathbf{x_1})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_2}} (-1)^{f_2(\mathbf{x_2})}.$$

Given two Boolean function, $f_1, f_2 \in \mathcal{B}_n$, it is easy to see that $\Phi_{f_1,f_2} = \Phi_{f_2,f_1}$ and $-1 \leq \Phi_{f_1,f_2} \leq 1$.

Given oracle access to the functions $f_1, f_2 \in \mathcal{B}_n$, the algorithm begins with the state $|0^n\rangle |-\rangle$ and traverses through the following sequence of steps:

$$\mathrm{H}^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n} \to U_{f_2} \to \mathrm{H}^{\otimes n}.$$

Finally, after ignoring the last qubit, the amplitude of the all zero state becomes Φ_{f_1,f_2} and thus upon measurement the probability of obtaining the all zero state, $|0^n\rangle$ is given by $(\Phi_{f_1,f_2})^2$ [2]. The quantum structure of the 2-query 2-fold Forrelation algorithm is shown in Figure 3.1.

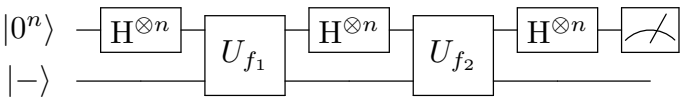


Figure 3.1: Quantum circuit for 2-fold Forrelation using 2 queries [2].

We now present certain desirable instantiations of the Forrelation problem by connecting different properties of Boolean functions. Moreover, we observe how this algorithm varies from the Deutsch-Jozsa algorithm and becomes similar in results depending on the functions f_1 and f_2.

Let us now concentrate on the scenario where the Forrelation of two Boolean functions $f_1, f_2 \in \mathcal{B}_n$ is the maximum, i.e., 1. It is easy to see that this scenario happens when n is even, and f_1 and f_2 are both bent functions and are dual to each other, which we present in the following lemma.

Proposition 3.1.1. *Let $f, g \in \mathcal{B}_n$ be bent. Then $\Phi_{f,g} = 1$ if and only if f and g are dual to each other.*

Proof. If $g = \hat{f}$, from the definition of Forrelation, we obtain

$$\Phi_{f,g} = \Phi_{f,\hat{f}} = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x})} W_{\hat{f}}(\mathbf{x}) = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} W_{\hat{f}}^2(\mathbf{x}) = \frac{2^n}{2^n} = 1.$$

Similarly, for two bent functions f and g, having $\Phi_{f,g} = 1$ necessarily implies that g and $W_f(\boldsymbol{\omega})$ always agree on the signs. Hence, g can be written as $(-1)^{g(\boldsymbol{\omega})} = W_f(\boldsymbol{\omega})$ implying f and g are dual to each other. □

In this direction, we have the following simple corollaries.

Corollary 3.1.1. *Given $f, g \in \mathcal{B}_n$, two bent functions. Then, $\Phi_{f,g} = -1$ if and only if f and g are anti-dual to each other. Similarly, given $f \in \mathcal{B}_n$ a bent function, $\Phi_{f,f} = 1$ if and only if f is self dual and $\Phi_{f,f} = -1$ if and only if f is anti-self-dual.*

Let us now look into the other extreme where the Forrelation of two bent functions has a very low absolute value. Together with Proposition 3.1.1, this gives various instantiations of the desired scenario. The best scenario is obviously the scenario where $\Phi_{f,g} = 0$. This can happen in the following framework.

Proposition 3.1.2. *Let f and g be two bent functions such that $f \oplus g$ is a balanced function. Then the Forrelation $\Phi_{f,\hat{g}} = \Phi_{g,\hat{f}} = 0$ where $\hat{f}$ and $\hat{g}$ are duals of f and g, respectively.*

Proof. From the definition of Forrelation, we obtain

$$\Phi_{f,\hat{g}} = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x})} W_{\hat{g}}(\mathbf{x}) = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x})} (-1)^{g(\mathbf{x})} = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{(f \oplus g)(\mathbf{x})} = 0.$$

The derivation of $\Phi_{g,\hat{f}}$ follows along the same lines. □

Combining the statements of Propositions 3.1.1 and 3.1.2, we have the following promise problem, which can be deterministically resolved using 2-fold Forrelation.

Proposition 3.1.3. *Given oracle access to bent functions $f, g \in \mathcal{B}_n$, the problem of finding whether $g = \hat{f}$ or if $g \oplus \hat{f}$ is balanced can be resolved deterministically by making exactly one query to f and g each.*

Proof. The 2-query Forrelation algorithm is executed with $f_1 = f$ and $f_2 = g$. If $g = \hat{f}$ then $\Phi_{f,g} = 1$, resulting the all zero state upon measurement, with certainty. On the other hand, if $g \oplus \hat{f}$ is balanced then $\Phi_{f,g} = 0$ implying that the all zero output is never appears. Therefore, the presence or absence of the all zero state deterministically resolve the relation between f and g. □

Next we look into another scenario where $\Phi_{f,g} \neq 0$ but has very low value. Then we can not solve the promise problem deterministically, but we can resolve the problem with good probability using only a constant number of queries. In this regard, we consider the Kerdock codes (see [62, 92]) and let us denote this set as $\mathcal{K}$.

Given two bent functions $f_1, f_2 \in \mathcal{K}$, we know that $f_1 \oplus f_2$ is bent. Moreover, we know that in the output of a bent function, the number of ones and zeros differ by $2^{n/2}$. Thus for $f_1, f_2 \in \mathcal{K}$ we have $\sum_{\mathbf{x}\in\mathbb{F}_2^n}(-1)^{f_1(\mathbf{x})\oplus f_2(\mathbf{x})} = \pm 2^{n/2}$. Based on this simple observation we construct a promise problem, which is another desirable instantiation of the 2-fold Forrelation problem, given in [2].

Proposition 3.1.4. *Let $f, g \in \mathcal{B}_n$ be bent such that $f, g, \hat{f} \in \mathcal{K}$, the problem of determining whether $g = \hat{f}$ or not can be efficiently resolved by making a single query each to f and g via 2-query 2-fold Forrelation algorithm proposed in [2].*

Proof. From Proposition 3.1.1, if $g = \hat{f}$, then $\Phi_{f,g} = 1$. Moreover, since g and $\hat{f}$ belong to the same Kerdock code $\mathcal{K}$, we have

$$\Phi_{f,g} = \frac{1}{2^n}\sum_{\mathbf{x}\in\mathbb{F}_2^n}(-1)^{g(\mathbf{x})}W_f(\mathbf{x}) = \frac{1}{2^n}\sum_{\mathbf{x}\in\mathbb{F}_2^n}(-1)^{g(\mathbf{x})}(-1)^{\hat{f}(\mathbf{x})} = \pm\frac{1}{2^n}2^{n/2} = \pm\frac{1}{2^{n/2}}.$$

Therefore, for $n > 14$ we have $|\Phi_{f,g}| = \left|\frac{1}{2^{n/2}}\right| < \frac{1}{100}$ and in all such cases we have the said desirable instantiation following the Forrelation problem [2]. □

3.2 Algorithms related to Walsh-Hadamard spectrum via 3-fold Forrelation

In this section and Section 3.3, we look into the 3-fold Forrelation framework as a unifying tool for analyzing various Boolean function spectra, including the Walsh-Hadamard, crosscorrelation, and autocorrelation spectra. We begin by reviewing two quantum algorithms for 3-fold Forrelation, $\mathcal{A}^{(3,2)}$ and $\mathcal{A}^{(3,3)}$, introduced by [2]. Utilizing this formulation, we develop Deutsch-Jozsa-like algorithms for Walsh spectrum sampling and for testing whether a function is m-resilient. While the 2-query algorithm samples the Walsh-Hadamard transform as efficiently as the Deutsch-Jozsa algorithm, the 3-query version offers a constant-factor improvement. Our key approach involves carefully selecting the functions in the 3-fold Forrelation to reduce the expression to a weighted sum of squared Walsh-Hadamard spectra, enabling the evaluation of spectral properties via $\mathcal{A}^{(3,2)}$ and $\mathcal{A}^{(3,3)}$. This strategy also extends to crosscorrelation estimation in Section 3.3, with appropriate modifications.

3.2.1 A suitable representation of 3-fold Forrelation and the corresponding algorithms

Given $f_1, f_2, f_3 \in \mathcal{B}_n$, the 3-fold Forrelation is defined as

$$\Phi_{f_1,f_2,f_3} = \frac{1}{2^{2n}} \sum_{\mathbf{x_1},\mathbf{x_2},\mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_1(\mathbf{x_1})}(-1)^{\mathbf{x_1}\cdot\mathbf{x_2}}(-1)^{f_2(\mathbf{x_2})}(-1)^{\mathbf{x_2}\cdot\mathbf{x_3}}(-1)^{f_3(\mathbf{x_3})}.$$

Interestingly, we can also write it down in the following manner.

$$\begin{aligned}
\Phi_{f_1,f_2,f_3} =& \frac{1}{2^{2n}} \sum_{\mathbf{x_1},\mathbf{x_2},\mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_1(\mathbf{x_1})}(-1)^{\mathbf{x_1}\cdot\mathbf{x_2}}(-1)^{f_2(\mathbf{x_2})}(-1)^{\mathbf{x_2}\cdot\mathbf{x_3}}(-1)^{f_3(\mathbf{x_3})} \\
=& \frac{1}{2^n} \sum_{\mathbf{x_2} \in \mathbb{F}_2^n} (-1)^{f_2(\mathbf{x_2})} \frac{1}{2^{n/2}} \sum_{\mathbf{x_1} \in \mathbb{F}_2^n} (-1)^{f_1(\mathbf{x_1})}(-1)^{\mathbf{x_2}\cdot\mathbf{x_1}} \frac{1}{2^{n/2}} \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_3(\mathbf{x_3})}(-1)^{\mathbf{x_2}\cdot\mathbf{x_3}} \\
=& \frac{1}{2^n} \sum_{\mathbf{x_2} \in \mathbb{F}_2^n} (-1)^{f_2(\mathbf{x_2})} W_{f_1}(\mathbf{x_2}) W_{f_3}(\mathbf{x_2}).
\end{aligned}$$

Using $f_1 = f_3 = f$ and $f_2 = g$, we obtain:

$$\Phi_{f,g,f} = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{g(\mathbf{x})} W_f(\mathbf{x})^2.$$

Thus, the 3-fold-Forrelation actually allows us to sample different combinations of the squares of the Walsh spectrum values of the function, and when all three functions are different, different combinations of point-wise product of Walsh spectrum values of two functions, where the combinations are decided by the function f_2. In all the algorithms we shall design and describe going ahead, we will modify the functions f_1, f_2 and f_3 along with some additional tweaks to obtain the desirable sampling scenario.

Given oracle access to $f_1, f_2, f_3 \in \mathcal{B}_n$, let us now briefly discuss the 3-query and 2-query quantum algorithms for 3-fold Forrelation, due to [2].

- The 3-query algorithm begins with an $(n+1)$-qubit state $|0^n\rangle|-\rangle$ and applies the oracles U_{f_1}, U_{f_2}, and U_{f_3} interleaved with Hadamard gates. Ignoring the last qubit, the amplitude corresponding to the all zero state becomes Φ_{f_1,f_2,f_3}. The unitary corresponding to the 3-query 3-fold Forrelation algorithm is denoted by $\mathcal{A}^{(3,3)}(f_1, f_2, f_3)$, which assumes oracle access to all three Boolean functions, f_1, f_2, f_3 and results in an n-bit output where the probability of observing the all zero state is given by $(\Phi_{f_1,f_2,f_3})^2$. The quantum circuit structure of $\mathcal{A}^{(3,3)}(f_1, f_2, f_3)$ has been illustrated in Figure 2.10.

- The 2-query algorithm begins with an $(n+2)$ qubit state $|+\rangle|0^n\rangle|-\rangle$, where the first qubit serves as the control (driving) qubit. Conditioned on the control qubit

being $|0\rangle$, we sequentially apply $\mathrm{H}^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n} \to U_{f_2} \to \mathrm{H}^{\otimes n}$ and if the control qubit is $|1\rangle$, the n-qubit Hadamard gate, $\mathrm{H}^{\otimes n}$ is applied first, followed by the oracle U_{f_3}. The oracles act on all qubits except the control. Finally, the control qubit is measured in the Hadamard basis, equivalent to applying a Hadamard gate followed by a computational basis measurement. The probability of observing 0 is given by $\frac{1}{2}(1 + \Phi_{f_1,f_2,f_3})$. We denote the 2-query 3-fold Forrelation algorithm by the unitary $\mathcal{A}^{(3,2)}(f_1, f_2, f_3)$, which assumes oracle access to all three Boolean functions, f_1, f_2, f_3, and outputs 0 and 1 with probabilities $\frac{1}{2}(1 + \Phi_{f_1,f_2,f_3})$ and $\frac{1}{2}(1 - \Phi_{f_1,f_2,f_3})$, respectively. The circuit diagram for $\mathcal{A}^{(3,2)}(f_1, f_2, f_3)$ has been shown in Figure 2.11.

Next, we analyze the Walsh-Hadamard spectrum sampling of a Boolean function f at a given point $\boldsymbol{\omega}$ using the algorithms $\mathcal{A}^{(3,2)}$ and $\mathcal{A}^{(3,3)}$, and compare the sampling probabilities with that of the Deutsch-Jozsa algorithm.

3.2.2 Walsh spectrum sampling using 3-fold Forrelation

Consider $f \in \mathcal{B}_n$ and a set of points $S \subseteq \mathbb{F}_2^n$. The probability of observing one of the states from S in the outcome of the Deutsch-Jozsa algorithm with U_f is given by

$$p := \frac{1}{2^n} \sum_{\mathbf{x} \in S} W_f(\mathbf{x})^2. \tag{3.1}$$

Next we sample the Walsh spectrum values of f on S using 3-fold Forrelation. We define the Forrelation set up for Φ_{f_1,f_2,f_3} as follows. We have $f_1 = f_3 = f$ and the function $f_2 = g$ is designed by us, depending upon the set S such that $g(\mathbf{x}) = 1$ if and only if $\mathbf{x} \in S$ and $g(\mathbf{x}) = 0$, otherwise. Then we have,

$$\Phi_{f,g,f} = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{g(\mathbf{x})} W_f(\mathbf{x})^2 = \frac{1}{2^n} \left(\sum_{\mathbf{x} \notin S} W_f(\mathbf{x})^2 - \sum_{\mathbf{x} \in S} W_f(\mathbf{x})^2 \right).$$

From Parseval's identity, $\sum_{\mathbf{x} \in \mathbb{F}_2^n} W_f(\mathbf{x})^2 = \sum_{\mathbf{x} \notin S} W_f(\mathbf{x})^2 + \sum_{\mathbf{x} \in S} W_f(\mathbf{x})^2 = 2^n$ we obtain,

$$\Phi_{f,g,f} = \frac{1}{2^n} \left(2^n - \sum_{\mathbf{x} \in S} W_f(\mathbf{x})^2 - \sum_{\mathbf{x} \in S} W_f(\mathbf{x})^2 \right) = 1 - \frac{2}{2^n} \sum_{\mathbf{x} \in S} W_f(\mathbf{x})^2.$$

In this regard, we have the following result.

Lemma 3.2.1. *The probability of getting the output* 1 *upon running the algorithm* $\mathcal{A}^{(3,2)}(f, g, f)$ *with* $g(\mathbf{x}) = 1$, *for all* $\mathbf{x} \in S$ *and* $g(\mathbf{x}) = 0$, *otherwise, is same as* p *as in Equation* (3.1).

Proof. From $\mathcal{A}^{(3,2)}(f,g,f)$, the probability of getting the output 1 is

$$\frac{1}{2}\left(1-\Phi_{f,g,f}\right)=\frac{1}{2}\left[1-\left(1-\frac{2}{2^n}\sum_{\mathbf{x}\in S}W_f(\mathbf{x})^2\right)\right]=\frac{1}{2^n}\sum_{\mathbf{x}\in S}W_f(\mathbf{x})^2=p.$$

□

Thus the 2-query algorithm $\mathcal{A}^{(3,2)}(f,g,f)$ makes a single query to f and another query to g, designed based on the set S, behaves equivalent to the Deutsch-Jozsa algorithm in terms of Walsh spectrum sampling. Now we show that $\mathcal{A}^{(3,2)}(f,g,f)$ can be used here to obtain the improvement.

Theorem 3.2.1. *The probability of getting an output with at least one bit being* 1 *upon running the algorithm* $\mathcal{A}^{(3,3)}(f,g,f)$ *for* 3*-fold Forrelation is given by* $4p-4p^2$*, where* p *is as in Equation* (3.1)*.*

Proof. From Lemma 3.2.1, we obtain $\frac{1}{2}\left(1-\Phi_{f,g,f}\right)=p$, which implies $\Phi_{f,g,f}=1-2p$. Now, the probability of getting an output with at least one bit being 1 upon running the algorithm $\mathcal{A}^{(3,3)}(f,g,f)$ is $1-\left(\Phi_{f,g,f}\right)^2=1-(1-2p)^2=4p-4p^2$. □

For $p<0.75$, we have $4p-4p^2>p$. Moreover, if one argues that $\mathcal{A}^{(3,3)}$ makes two queries to f compared one by Deutsch-Jozsa, it is easy to show that the probability of observing any one of the states from S at least once by running the Deutsch-Jozsa algorithm twice is $1-(1-p)^2=2p-p^2$ which is also lower than the probability due to $\mathcal{A}^{(3,3)}$ for small values of p. In fact, we can compare the results due to $\mathcal{A}^{(3,3)}$, sampling from Deutsch-Jozsa once, sampling from Deutsch-Jozsa twice, and sampling using Deutsch-Jozsa followed by one round amplitude amplification for small values of p in the following manner.

1. The probability of observing any one of the states from S after running the Deutsch-Jozsa algorithm is p. This requires one query made to the oracle of f.

2. The probability of observing any one of the states from S at least once, after running the Deutsch-Jozsa algorithm twice is $2p-p^2\approx 2p$. This requires two queries to the oracle of f.

3. If we first apply the Deutsch-Jozsa and then amplify the states from S using a single round of amplitude amplification, then at first we have $\theta=\sin^{-1}p$ and after one round the probability becomes $\sin\left(3\sin^{-1}p\right)$. Since for small values of θ we have $\theta\approx\sin\theta$, the probability of observing any one of the states from S becomes $\approx 3p$. This method also requires two queries to f, one for the initial application of Deutsch-Jozsa and once for designing the inversion about the mean operator.

4. Finally, if we use the algorithm $\mathcal{A}^{(3,3)}(f, g, f)$, where $g(\mathbf{x}) = 1$, for all $\mathbf{x} \in S$ and $g(\mathbf{x}) = 0$ otherwise, then the probability obtaining at-least one bit being 1, is given by $4p - 4p^2 \approx 4p$ which outperforms all the techniques discussed above for small values of p.

Figure 3.2 presents a schematic view of these probability values.

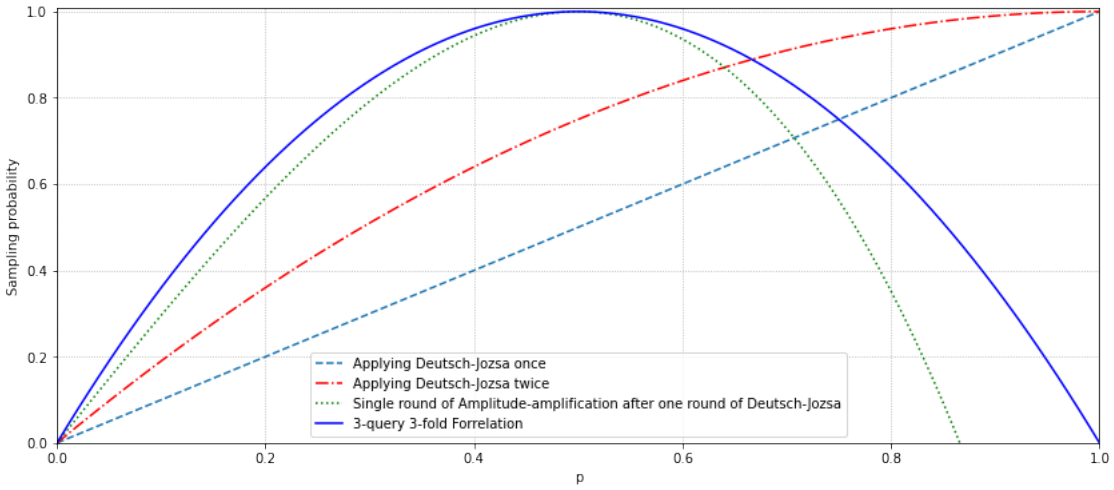


Figure 3.2: Sampling probabilities of Walsh-Hadamard transform using different algorithms

Hence, the 3-query 3-fold Forrelation algorithm, $\mathcal{A}^{(3,3)}(f, g, f)$ samples the Walsh-Hadamard spectrum values f at any given set of points, S more efficiently compared to the Deutsch-Jozsa algorithm. This result underlines how the Forrelation algorithm can efficiently sample the Walsh spectra of a Boolean function at any given point.

Observe that for $|S| = 1$, the 3-fold Forrelation algorithm $\mathcal{A}^{(3,3)}(f, g, f)$ samples the Walsh transform values of f at any given point more efficiently compared to the DJ algorithm. Moreover, if we consider $S = \{\mathbf{x} : wt(\mathbf{x}) \leq m\}$, then using $\mathcal{A}^{(3,3)}(f, g, f)$ we obtain a better sampling of Walsh spectra for the points with Hamming weight less or equal to a particular weight m compared to the Deutsch-Jozsa algorithm. This provides a constant improvement over the state-of-the-art result [20].

3.2.3 Implications to resiliency checking

In [20], Chakraborty et al. present a probabilistic method to determine whether an unknown Boolean function $f \in \mathcal{B}_n$ is m-resilient, assuming oracle access to U_f. At the beginning, the Deutsch–Jozsa algorithm is applied to produce the quantum state $2^{-n/2} \sum_{\mathbf{x} \in \mathbb{F}_2^n} W_f(\mathbf{x}) |\mathbf{x}\rangle$ where W_f denotes the Walsh–Hadamard transform of f. If f

is m-resilient, then $W_f(\mathbf{x}) = 0$ for all $\mathbf{x}$ such that $wt(\mathbf{x}) \leq m$. Thus, if a state with Hamming weight at most m is observed upon measurement, it conclusively implies that f is not m-resilient.

However, if no such state is observed over a polynomial number of trials, one cannot deterministically conclude that f is m-resilient, as nonzero $W_f(\mathbf{x})$ values may exist at low-weight inputs with very small amplitude. To amplify the detection probability of such low-weight states, amplitude amplification is employed over the subspace corresponding to $wt(\mathbf{x}) \leq m$, thereby improving the success probability over classical sampling.

The states $\mathbf{x}$ with Hamming weight $wt(\mathbf{x}) \leq m$ are referred to as undesirable outcomes, whose presence deterministically concludes that f is not m-resilient. Hence, any algorithm should aim to detect whether f is not m-resilient with high confidence, and only declare f to be m-resilient if no undesirable outcome is observed after a specified number of executions.

Here present the following observations.

- The function $4p - 4p^2$ is decreasing for $p > 1/2$. We begin by sampling a constant number of times using the Deutsch-Jozsa algorithm and check for any undesirable outcomes. If $p > 1/2$, such an outcome is likely with high probability. Otherwise, we apply $\mathcal{A}^{(3,3)}(f, g, f)$, which is better suited for sampling low Walsh spectra values than the Deutsch-Jozsa algorithm.

- If p is very low, the probability of observing an undesirable outcome becomes negligible in both the Deutsch-Jozsa and the 3-fold Forrelation setups. To address this, Chakraborty et al. [20] employed the amplitude amplification algorithm [16] atop the Deutsch-Jozsa framework to amplify the amplitude of the undesirable state, thereby increasing its detection probability. A similar amplification can be applied to the outcome of $\mathcal{A}^{(3,3)}$, and the overall circuit complexity remains of the same asymptotic order.

Since the starting state of $\mathcal{A}^{(3,3)}$ makes the sampling probability of an undesirable outcome 4-times more likely compared to Deutsch-Jozsa; therefore, we need to run the amplification to iterate one-fourth of the times to obtain a similar result compared to the result in [20]. However, the application of $\mathcal{A}^{(3,3)}$ requires two oracle queries to f. Therefore if the amplification iterate is run n times for $\mathcal{A}^{(3,3)}$, then it requires a total of $2n$ many queries to f, compared to $4n$ queries in case of [20]. Therefore in total, the query complexity due to $\mathcal{A}^{(3,3)}$ is half compared to [20]. The focus of this chapter is to design the starting states before the sampling algorithms and therefore, further study about the detailed procedure is beyond the scope of this paper.

Finally, in the next section, we discuss how the Forrelation algorithm can also be used to sample crosscorrelation values of two functions by simply replacing the symmetric

function that is f_2's placeholder with a linear function of our choice. We also consider tweaking the quantum algorithm for 3-fold Forrelation to obtain further results.

3.3 Crosscorrelation and 3-fold Forrelation

The crosscorrelation $C_{f,g}$ of two functions $f, g \in \mathcal{B}_n$ is a widely studied cryptographic property, closely related to Shannon's notion of confusion. Unlike the Walsh spectrum, which satisfies Parseval's identity: $2^{-n}\sum_{\mathbf{x}\in\mathbb{F}_2^n} W_f(\mathbf{x})^2 = 1$ enabling probabilistic sampling in the Deutsch-Jozsa algorithm via $\mathcal{P}(\mathbf{x}) = 2^{-n}W_f(\mathbf{x})^2$, the crosscorrelation and autocorrelation spectra lack such a normalization constraint. Specifically, the crosscorrelation spectrum satisfies

$$2^{2n} \leq \sum_{\mathbf{x}\in\mathbb{F}_2^n} C_{f,g}(\mathbf{x})^2 \leq 2^{3n},$$

which prevents the direct application of Deutsch-Jozsa–style sampling techniques to these spectra.

In [12], Bera et al. proposed a quantum algorithm for sampling the autocorrelation spectrum, generating a $2n$-bit state $\mathbf{x}||0^n$ with probability $2^{-3n}C_f(\mathbf{x})^2$, along with estimation algorithms within the quantum framework. Their approach relies on the relationship between the Walsh spectra of derivatives of f and its autocorrelation spectrum, and therefore cannot be directly extended to crosscorrelation. Against this backdrop, here we design crosscorrelation estimation and sampling algorithms (also apply to autocorrelation as special cases), beginning with the crosscorrelation estimation.

We start with a very interesting observation of [80] which leads us to our first algorithm.

Theorem 3.3.1 ([80]). *Given any two functions f and g, we have*

$$[C_{f,g}(000\ldots0),\ldots,C_{f,g}(111\ldots1)]\hat{\mathrm{H}}_n = [W_fW_g(000\ldots0),\ldots,W_fW_g(111\ldots1)]$$

where $\hat{\mathrm{H}}_n = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{\otimes n}$.

Here, observe that $\hat{\mathrm{H}}_n$ is a $2^n \times 2^n$ "non-normalized" Hadamard matrix and thus we have $\left(\hat{\mathrm{H}}_n\right)^2 = 2^n\mathrm{I}_n$ where I_n is the 2^n-dimensional identity matrix. Thus multiplying $\hat{\mathrm{H}}_n$ to both sides of the above identity results in the following equation.

$$[C_{f,g}(00\ldots0),\ldots,C_{f,g}(11\ldots1)]2^n\mathrm{I}_n = [W_fW_g(00\ldots0),\ldots,W_fW_g(11\ldots1)]\hat{\mathrm{H}}_n.$$

We also fix a general notation to denote the 2^n linear functions on n variables, denoting the functions as $\mathbb{L}_{\mathbf{y}}(\mathbf{x}) = (-1)^{\oplus_{i:y_i=1}x_i}$. In this regard, we have the following well-known result.

Fact 3.3.1. *For a given n, the 2^n linear functions have a one-to-one correspondence with the columns of $\hat{\mathrm{H}}_n$. That is, $\hat{\mathrm{H}}_n[\mathbf{i}][\mathbf{j}] = \mathbb{L}_{\mathbf{i}}(\mathbf{j})$, $\mathbf{j} \in \mathbb{F}_2^n$ for all $\mathbf{i} \in \mathbb{F}_2^n$.*

Using these results along with the 3-fold Forrelation formulation, we obtain the following results.

Theorem 3.3.2. *Given oracle access to Boolean functions $f, g \in \mathcal{B}_n$, two algorithms A_1 and A_2 can be designed in a way that they make one query to f and g each and makes another query to a linear function $\mathbb{L}_{\mathbf{y}}$ such that*

1. *In algorithm A_1, upon measuring n predetermined qubits in the computational basis, the probability of obtaining the all zero state is $\frac{\left(C_{f,g}(\mathbf{y})\right)^2}{2^{2n}}$.*
2. *In algorithm A_2, upon measuring one predetermined qubit in the computational basis, the probability of obtaining the output 0 is $\frac{1}{2}\left(1 + \frac{C_{f,g}(\mathbf{y})}{2^n}\right)$.*

Proof. Let A_1 be the algorithm $\mathcal{A}^{(3,3)}(f, \mathbb{L}_{\mathbf{y}}, g)$. Then the probability of obtaining the all zero state upon measurement is given by $\left(\Phi_{f,\mathbb{L}_{\mathbf{y}},g}\right)^2$ where the Forrelation value equates to

$$\Phi_{f,\mathbb{L}_{\mathbf{y}},g} = \frac{1}{2^n}\sum_{\mathbf{x}\in\mathbb{F}_2^n} W_f(\mathbf{x})W_g(\mathbf{x})\mathbb{L}_{\mathbf{y}}(\mathbf{x}) = \frac{1}{2^n}\sum_{\mathbf{x}\in\mathbb{F}_2^n} W_f(\mathbf{x})W_g(\mathbf{x})\hat{\mathrm{H}}_n[\mathbf{y}][\mathbf{x}] = \frac{C_{f,g}(\mathbf{y})}{2^n}.$$

Thus the probability of getting $|0^n\rangle$ as measurement outcome is $\frac{\left(C_{f,g}(\mathbf{y})\right)^2}{2^{2n}}$.

Similarly we can define $A_2 = \mathcal{A}^{(3,2)}(f, \mathbb{L}_{\mathbf{y}}, g)$ and thus we obtain the output 0 with probability $\frac{1}{2}\left(1 + \frac{C_{f,g}(\mathbf{y})}{2^n}\right)$. □

This gives us a constant query algorithm for sampling the crosscorrelation value of any two functions at any given point. Here note that, one needs to design different algorithms (and thus circuits) in order to obtain the crosscorrelation value at different points. Let us first compare these results with the results by [12] on the autocorrelation C_f.

3.3.1 Comparison of autocorrelation results with Bera et al. (INDOCRYPT, 2019)

The work of Bera et al. [12] introduces two quantum algorithms: one for sampling the autocorrelation spectrum of a Boolean function $f \in \mathcal{B}_n$, and another for estimating its value at a specific point $C_f(\mathbf{x})$. Below, we summarize the output states of these algorithms and compare their structure to the Forrelation-based approach presented in Chapter 3.

Bera et al. examine higher-order derivatives of the Walsh spectrum using the Deutsch-Jozsa algorithm and exploit the equivalence between the first-order derivative and the autocorrelation spectrum to sample the later in the following manner.

Theorem 3.3.3 ([12]). *There exists a quantum algorithm A_1 such that $A_1\left(|0\rangle^{2n+1}\right)$ produces the state*

$$|\phi\rangle = |-\rangle \frac{1}{\sqrt{2^n}} \sum_{\mathbf{b}\in\mathbb{F}_2^n} \sum_{\mathbf{y}\in\mathbb{F}_2^n} \widehat{\Delta f_{\mathbf{b}}^{(1)}}(\mathbf{y}) |\mathbf{y}\rangle |\mathbf{b}\rangle ,$$

where $\widehat{\Delta f_{0^n}^{(1)}}(\mathbf{y}) = \frac{C_f(\mathbf{y})}{2^n}$.

Thus, the probability of measuring the $2n$-bit state $|\mathbf{y}\rangle |0^n\rangle$ is $\frac{C_f(\mathbf{y})^2}{2^{3n}}$.

In comparison, the algorithm $\mathcal{A}^{(3,3)}$ designed in Theorem 3.3.2 outputs the n bit state $|0^n\rangle$ with probability $\frac{C_f(\mathbf{y})^2}{2^{2n}}$. Thus, the 3-query 3-fold Forrelation algorithm, $\mathcal{A}^{(3,3)}$ provides the autocorrelation value at a particular point more efficiently, although it does not sample from the autocorrelation spectrum.

Here the following remark is essential.

Remark 3.3.1. *The algorithm in Theorem 3.3.3 is explicitly designed to sample the autocorrelation spectrum of a Boolean function by computing the Walsh spectrum of its derivatives. On the other hand, we derive an algorithm to sample the crosscorrelation spectrum of two Boolean functions f and g, by tweaking the 3-query, 3-fold Forrelation algorithm $\mathcal{A}^{(3,3)}$.*

Next we discuss amplitude-estimation. In [12], the authors further propose a swap-test-based algorithm to facilitate improved estimation of $C_f(\mathbf{y})$:

Lemma 3.3.1 ([12]). *There exists a quantum algorithm $A_2(\mathbf{y})$ that, when applied to $|0\rangle^{3n+2}$, measures 0 on a designated qubit with probability $\frac{1}{2}\left(1 + \frac{C_f(\mathbf{y})^2}{2^{2n}}\right)$.*

This estimation is further refined using a general quantum amplitude estimation technique:

Lemma 3.3.2 ([16]). *Let A be a quantum circuit with no intermediate measurement, and let p denote the probability of observing its output in a particular subspace. Then, there exists a quantum algorithm that makes $\mathcal{O}\left(\frac{\pi}{\epsilon}\log\frac{1}{\delta}\right)$ (controlled) calls to A and returns an estimate $\tilde{p}$ such that*

$$\mathcal{P}[\tilde{p} - \epsilon \leq p \leq \tilde{p} + \epsilon] \geq 1 - \delta,$$

for any $\epsilon \leq 1/4$ and $\delta < 1$.

Combining Lemmas 3.3.1 and 3.3.2, Bera et al. establish the following result:

Theorem 3.3.4 ([12]). *Given a Boolean function f, there exists an algorithm that makes $\mathcal{O}\left(\frac{\pi}{\epsilon}\log\frac{1}{\delta}\right)$ queries and returns an estimate α satisfying*

$$\mathcal{P}[\alpha-\epsilon\leq 2^{-2n}C_f(\mathbf{y})^2\leq\alpha+\epsilon]\geq 1-\delta.$$

Here the algorithm estimates $\frac{C_f(\mathbf{y})^2}{2^{2n}}$ by observing the outcome probability of a specific state in the procedure described in Lemma 3.3.1, which is $\frac{1}{2}\left(1+\frac{C_f(\mathbf{y})^2}{2^{2n}}\right)$. Lemma 3.3.2 is then applied to this result.

Notably, the structure of the algorithm in Lemma 3.3.1 is analogous to the $\mathcal{A}^{(3,2)}$ algorithm for 3-fold Forrelation, as both utilize a control qubit to sequence multiple oracle applications. However, we demonstrate that a suitable instantiation of $\mathcal{A}^{(3,2)}$ yields a more efficient estimation algorithm, especially in terms of improved accuracy, as follows.

Theorem 3.3.5. *Given a Boolean function $f\in\mathcal{B}_n$, there exists an algorithm that makes $\mathcal{O}\left(\frac{\pi}{\epsilon}\log\frac{1}{\delta}\right)$ queries to the oracle of f and returns an estimate α such that*

$$\mathcal{P}[\alpha-\epsilon\leq\frac{C_{f,g}(\mathbf{y})}{2^n}\leq\alpha+\epsilon]\geq 1-\delta.$$

Proof. The algorithm $\mathcal{A}^{(3,2)}$ outputs the state 0 with probability $\hat{P}(f,g)=\frac{1}{2}\left(1+\frac{C_{f,g}(\mathbf{y})}{2^n}\right)$. This enables the estimation of $\hat{P}(f,g)$ using Lemma 3.3.2. It is straightforward to observe that estimating $\frac{1}{2}\left(1+\frac{C_{f,g}(\mathbf{y})}{2^n}\right)$ is equivalent to estimating $\frac{C_{f,g}(\mathbf{y})}{2^n}$, thereby yielding the desired result. Setting $f=g$, this translates into an algorithm for autocorrelation estimation. □

Remark 3.3.2. *The query complexity needed to ϵ-estimate $\frac{C_f(\mathbf{y})}{2^n}$ in Theorem 3.3.5 is the same as the query complexity needed to ϵ-estimate $\left(\frac{C_f(\mathbf{y})}{2^n}\right)^2$ in [12]. This implies for comparative accuracy of $\frac{C_f(\mathbf{y})}{2^n}$, we need square-root of the number of queries, which is again an improvement on [12].*

Finally, we move towards a crosscorrelation sampling algorithm, which is the final contribution of our paper. As a corollary of this result, we can also check if two functions are uncorrelated of degree m for all values of m up to a bound so that the time complexity is better than that of general classical algorithms. For the later part, we use the existence of polynomial-size circuits for Dicke state preparation [68]. Let us now define a Dicke state, denoted by $|D_k^n\rangle$.

Definition 3.3.1. *An n-qubit quantum state with equal superposition of all $\binom{n}{k}$ many basis states of weight k is called a Dicke state.*

3.3.2 A crosscorrelation sampling algorithm

Now we move towards crosscorrelation sampling. In the case of sampling the crosscorrelation value of two functions at a point, we used the 3-fold Forrelation set-up, where the second function is a linear function of our choice.

As a first step, we modify $\mathcal{A}^{(3,3)}$ so that the second function $\mathbb{L}_{\mathbf{y}}$ can be decided by additional inputs. Next we describe a very simple implementation of any linear function in the quantum black-box model. Consider the operation $\mathrm{CNOT}_{\mathsf{b}}^{\mathsf{S}}$ which denotes the multi-controlled NOT operation where S is a set consisting of the control-qubits and the qubit b is the target. Then it is easy to see that any linear function $\mathbb{L}_{\mathbf{y}}$ can be implemented as the series of operations $\mathrm{CNOT}_b^{q_i}$ such that $y_i = 1$. Using this construction, we tweak the algorithm.

Algorithm 1: The Algorithm $\mathbb{A}(\mathrm{C}_n)$ where $\mathrm{C}_n = \mathrm{C}_n^1$.

Input:

1. $\mathbf{u} \in \mathbb{F}_2^n$,
2. $R \otimes Q$ where

   R is an n-qubit register, denoted as $\otimes_{i=1}^n |r_i\rangle$.

   Q is an n-qubit register, denoted as $\otimes_{i=1}^n |q_i\rangle$.
3. Description of an algorithm C_n.

Output: The output $\mathbf{u}||0^n$ with probability $\frac{C_{f,g}(\mathbf{u})}{2^{2n}}$

Apply the following operations in the given sequence:

1. C_n on the register R (In this case defined as $\mathrm{C}_n^1(\mathbf{u})$):
   apply X gate on r_i if $u_i = 1$.
2. $\mathrm{H}^{\otimes n}$ on q_1 to q_n.
3. Oracle of f on $Q|-\rangle$.
4. $\mathrm{H}^{\otimes n}$ on q_1 to q_n.
5. DC defined as: $\mathrm{CNOT}_{\mathsf{q_{n+1}}}^{\mathsf{r_i},\mathsf{q_i}}, \mathsf{1 \leq i \leq n}$.
6. $\mathrm{H}^{\otimes n}$ on q_1 to q_n.
7. Oracle of g on $Q|-\rangle$.
8. $\mathrm{H}^{\otimes n}$ on q_1 to q_n.

Figure 3.3 provides a schematic diagram of the algorithm $\mathbb{A}(\mathrm{C}_n)$.

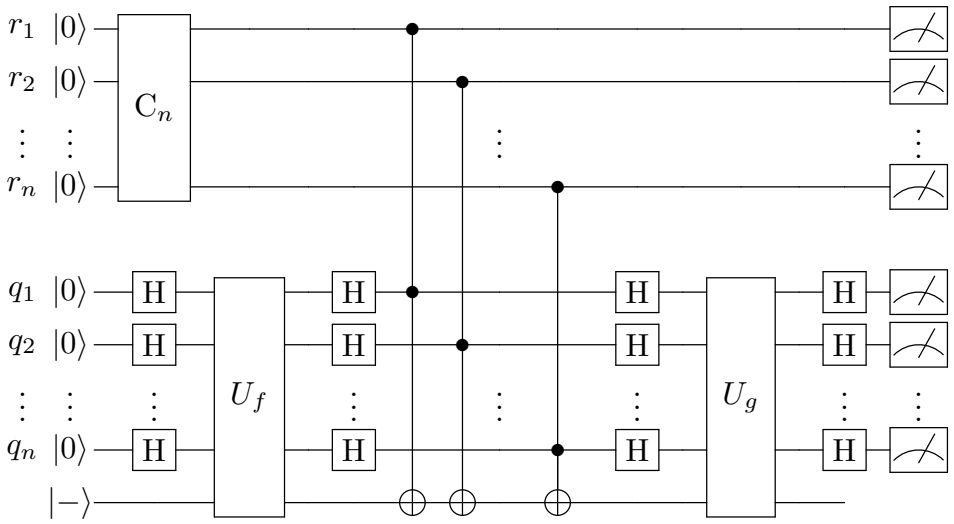


Figure 3.3: Quantum circuit for sampling the complete crosscorrelation spectrum.

Let us now analyze the working of this algorithm. First let us observe a simple fact regarding the operator DC. If $r_i = 1$ for some i, then the operation $\mathrm{CNOT}^{r_i,q_i}_{q_{n+1}}$ works as $\mathrm{CNOT}^{q_i}_{q_{n+1}}$ which is equivalent to the oracle of the function $h(\mathbf{x}) = x_i$. Otherwise if $r_i = 0$ then the operation $\mathrm{CNOT}^{r_i,q_i}_{q_{n+1}}$ works as identity. Which can be formalized as follows.

Proposition 3.3.1. *The operation DC on $R \otimes Q$ can be described as $DC\,|\mathbf{u}\rangle\,|\mathbf{x}\rangle\,|-\rangle = (-1)^{\mathbf{u}\cdot\mathbf{x}}\,|\mathbf{u}\rangle\,|\mathbf{x}\rangle\,|-\rangle$. In essence, the operator DC works as an oracle access to the function $\mathbb{L}_{\mathbf{u}}$ on the register Q when R is in the state $|\mathbf{u}\rangle$.*

Thus when C_n is defined as $C^1_n(\mathbf{u})$, the algorithm effectively becomes same as the 3-fold Forrelation algorithm, $\mathcal{A}^{(3,3)}(f, \mathbb{L}_{\mathbf{u}}, g)$ on register Q and the register R stays in the state $|\mathbf{u}\rangle$ and thus outputs $\mathbf{u}||0^n$ with probability $\frac{C_{f,g}(\mathbf{u})^2}{2^{2n}}$.

The crosscorrelation sampling and checking if two functions are uncorrelated of degree m can be obtained by choosing suitable unitaries C_n, which is the final contribution of the paper. Let us derive the output state for a general C_n operation.

Lemma 3.3.3. *Let C_n be $2^n \times 2^n$ unitary operator so that $\mathrm{C}_n\,|0^n\rangle = \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}}\,|\mathbf{x}\rangle$. Then starting from the state $|0^n\rangle\,|0^n\rangle$ the algorithm $\mathbb{A}(\mathrm{C}_n)$ of Algorithm 1 has the pre-measurement state*

$$\sum_{\mathbf{u}\in\mathbb{F}_2^n} \alpha_{\mathbf{u}}\,|\mathbf{u}\rangle \left(\frac{C_{f,g}(\mathbf{u})}{2^n}\,|0^n\rangle + \beta_{\mathbf{u}}\,|W_{\mathbf{u}}\rangle\right)$$

where $|W_{\mathbf{u}}\rangle$ is an n-qubit superposition state such that the amplitude of the state $|0^n\rangle$ is 0.

The proof of this lemma follows directly from Algorithm 1. Let us now present the following theorem for crosscorrelation sampling using Lemma 3.3.3.

Theorem 3.3.6. *If we fix* $\mathrm{C}_n = \mathrm{H}^{\otimes n}$ *then upon measuring the predetermined* $2n$ *many qubits at the end of* $\mathbb{A}(\mathrm{C}_n)$*, the probability of getting the state* $\mathbf{u}||0^n$ *is* $\frac{C_{f,g}(\mathbf{u})^2}{2^{3n}}$ *for all* $\mathbf{u} \in \mathbb{F}_2^n$.

Proof. If we fix $\mathrm{C}_n = \mathrm{H}^{\otimes n}$, then from Lemma 3.3.3, the pre-measurement state becomes

$$\sum_{\mathbf{u}\in\mathbb{F}_2^n} \frac{1}{2^{\frac{n}{2}}} |\mathbf{u}\rangle \left(\frac{C_{f,g}(\mathbf{u})}{2^n} |0^n\rangle + \beta_{\mathbf{u}} |W_{\mathbf{u}}\rangle \right).$$

Thus the probability of getting the output $\mathbf{u}||0^n$ upon measurement is given by $\frac{C_{f,g}(\mathbf{u})^2}{2^{3n}}$. □

Note that, $2^{2n} \leq \sum_{\mathbf{u}\in\mathbb{F}_2^n} C_{f,g}(\mathbf{u})^2 \leq 2^{3n}$ implies that $\frac{1}{2^{3n}} \sum_{\mathbf{u}\in\mathbb{F}_2^n} C_{f,g}(\mathbf{u})^2 \leq 1$ and this bound is tight for certain choices of f and g. Therefore, it is not possible to have a generalized algorithm that always samples from the crosscorrelation spectrum with better than $\frac{C_{f,g}(\mathbf{u})^2}{2^{3n}}$ probability.

3.3.3 Checking if two functions are uncorrelated of degree m

In this problem we want to check if $C_{f,g}(\mathbf{x}) = 0$ for all $\mathbf{x} \in \mathbb{F}_2^n$ such that $wt(\mathbf{x}) \leq m$. This problem is similar to checking if a function is m-resilient. Thus the same sampling and amplitude amplification algorithm of [20] can be implemented. Suppose there is a quantum algorithm such that the probability of obtaining a good (undesirable) state while calling an algorithm is a^2. In that case, such a state can be obtained with constant probability by running the algorithm $\mathcal{O}\left(\frac{1}{a}\right)$ times without the measurement. Here if we run the crosscorrelation sampling algorithm and if the output is of the form $\mathbf{y}||0^n$ where $wt(\mathbf{y}) \leq m$ then it concludes f and g are not uncorrelated of degree m. Then we have the following result.

Proposition 3.3.2. *There is an algorithm that verifies if two functions* f *and* g *are uncorrelated of degree* m *with constant probability by making* $\mathcal{O}(\frac{1}{a})$ *queries to* f *and* g *each, where*

$$a^2 = \frac{1}{2^{3n}} \sum_{\mathbf{x}:wt(\mathbf{x})\leq m} C_{f,g}(\mathbf{x})^2.$$

Proof. We use the sampling algorithm defined in Theorem 3.3.6. Then let an undesirable state be one that guarantees existence of $\mathbf{u}$ such that $wt(\mathbf{u}) \leq m$ and $C_{f,g}(\mathbf{u}) \neq 0$. This is same as obtaining a state $\mathbf{u}||0^n$ output after measurement of the algorithm in Theorem 3.3.6. The probability of getting such a state is $\frac{1}{2^{3n}} \sum_{\mathbf{x}:wt(\mathbf{x})\leq m} C_{f,g}(\mathbf{x})^2$. Applying the amplitude amplification argument to this yields the result. □

Here note that because of the structure of $\mathbb{A}(\mathrm{C}_n)$ we can control the points over which crosscorrelation is sampled. If C_n is such that the amplitude of some state $|\mathbf{y}\rangle$ in the register R is 0 then $C_{f,g}(\mathbf{y})$ never affects the output of $\mathbb{A}(\mathrm{C}_n)$. Now we can use the fact that the values of $C_{f,g}(\mathbf{x})$ where $wt(\mathbf{x}) > m$ is irrelevant while trying to check if f and g are uncorrelated of degree m. Thus we need a C_n which makes the algorithm efficient by not having states whose outputs are depending on $C_{f,g}(\mathbf{x})$, $wt(\mathbf{x}) > m$. To this end, we use the following result for Dicke state preparation circuits. Dicke states $|D_i^n\rangle$ are the equal superposition state of weight i on some n qubit system.

Theorem 3.3.7 ([68]). *Starting from the state $|0^n\rangle$ any Dicke state $|D_m^n\rangle$ can be deterministically prepared using $\mathcal{O}(n^2)$ CNOT gates and $\mathcal{O}(n^2)$ many single qubit gates.*

We denote the corresponding unitary that prepares the Dicke state $|D_m^n\rangle$ as UD_m^n, so that we have

$$UD_m^n |0^n\rangle = \frac{1}{\sqrt{\binom{n}{m}}} \sum_{\mathbf{x}:wt(\mathbf{x})=m} |\mathbf{x}\rangle .$$

Using this result we design a sampling algorithm in the following manner.

Theorem 3.3.8. *For any $m < n$, there is an algorithm that verifies if two functions f and g are uncorrelated of degree m with constant probability by making $\mathcal{O}(\sum_{i=0}^{m} \frac{1}{a_i})$ queries to f and g each, where*

$$(a_i)^2 = \frac{1}{\binom{n}{i} 2^{2n}} \sum_{\mathbf{x}:wt(\mathbf{x})=i} C_{f,g}(\mathbf{x})^2$$

.

Proof. We proceed in the same way as Proposition 3.3.2 but with a different choice of C_n. Consider the circuit $\mathbb{A}(UD_i^n)$ for some $0 \leq i \leq m$. This algorithm by definition of Algorithm 1 outputs a state $\mathbf{y}||0^n$ with probability $\frac{C_{f,g}(\mathbf{y})^2}{\binom{n}{i}2^{2n}}$ for all values of $\mathbf{y}$ with weight i, and probability of obtaining the state $\mathbf{x}||0^n$ is zero if $wt(\mathbf{x}) \neq i$. Thus any output $\mathbf{y}||0^n$ is an undesirable outcome corresponding to a point of weight i in the crosscorrelation spectrum with nonzero value. We can obtain such an outcome by making $\mathcal{O}(\frac{1}{a_i})$ queries to f and g each, where $(a_i)^2 = \frac{1}{\binom{n}{i}2^{2n}} \sum_{\mathbf{x}:wt(\mathbf{x})=i} C_{f,g}(\mathbf{x})^2$.

Thus, to check if there is any such point for all $1 \leq i \leq m$ we need to run the circuits $\mathbb{A}(UD_i^n), 0 \leq i \leq m$ and apply amplitude amplification on them individually, and thus the total query complexity is $\mathcal{O}(\sum_{i=0}^{m} \frac{1}{a_i})$. □

This is an improvement on the crosscorrelation sampling based result for small values of m. If m is a constant then $\binom{n}{m} < n^m$. Let $M_i = \sum_{\mathbf{x}:wt(\mathbf{x})=i} C_{f,g}(\mathbf{x})^2$. Then the query complexity for checking if f and g are uncorrelated of degree m using the algorithm in

Theorem 3.3.8 would be $2^n \sum_{i=0}^{m} \sqrt{\frac{\binom{n}{i}}{M_i}} \leq 2^n n^{\frac{m+1}{2}} \sum_{i=0}^{m} \frac{1}{\sqrt{M_i}}$. On the other hand if we would have used Proposition 3.3.2 the query complexity would have been $2^{\frac{3n}{2}} \sum_{i=0}^{m} \frac{1}{\sqrt{M_i}}$. For all cases where $\sum_{i=0}^{m} \frac{1}{\sqrt{M_i}}$ is of the form $\frac{1}{poly(n)}$, Theorem 3.3.8 provides polynomial improvement over using the crosscorrelation sampling algorithm.

At this point, one may wonder if a similar Dicke state-oriented approach could be applied for resiliency checking. While such a possibility can not be ruled out, in our proposed algorithmic framework, and also that of [20] it does not help. In m-resiliency checking we want to check if the value $P_1 = \sum_{wt(\mathbf{u})\leq m} W_f(\mathbf{u})^2$ is greater than 0. In this direction [20] designs a quantum algorithm so that the probability of getting a predetermined state is $\frac{P_1}{D_1}$ where $D_1 = 2^{2n}$ and then apply amplitude amplification on it, which gives advantage over known classical algorithms. Here following points are important.

1. In the algorithm due to [20], $D_1 = 2^{2n}$. Thus the denominator in the probability expression solely depends on n.

2. $P_1 \leq 2^{2n}$ and this bound is tight due to Parseval's identity. Thus it is not possible to have $D_1 < 2^{2n}$ if it depends solely on n.

This is important as if the probability of getting a state is $\frac{P}{D}$ where P and D are both integers, then the amplitude amplification algorithm ([16]) has the query complexity of $\mathcal{O}(\frac{\sqrt{D}}{\sqrt{P}})$. Thus with P_1 fixed for resiliency checking, if one could decrease the initial value of D_1 using a different sampling algorithm, it would have resulted in better query and thus time complexity for resiliency checking. However, as we have discussed, that is not possible if D_1 depends only on n.

Similarly for checking if two functions f, g are uncorrelated of degree m we need to determine if $P_2 = \sum_{wt(\mathbf{u})\leq m} C_{f,g}(\mathbf{u})^2$ is greater than 0. If we use the algorithm due to Proposition 3.3.2 then we get a predetermined state with probability $\frac{P_2}{D_2}$ where $D_2 = 2^{3n}$. Now we also know that P_2 can be high as 2^{3n} and thus D_2 cannot be smaller than 2^{3n} if it solely depends on n. At this point we use Dicke states intelligently to make D_2 depend on both n and m and effectively run $m+1$ different algorithms $A_i, 0 \leq i \leq m$ where the probability of getting a predetermined state in A_i is $\frac{P_{2,i}}{\binom{n}{i}2^{2n}}$ where $P_{2,i} = \sum_{wt(\mathbf{u})=i} C_{f,g}(\mathbf{u})^2$. This is what allows us to have advantage in Theorem 3.3.8 for small values of n. If $m = \Omega(n)$ then we essentially have same time complexity due to both Proposition 3.3.2 and Theorem 3.3.8. A similar approach could not be applied for resiliency because the Forrelation set-up for resiliency checking or the set-up in [20] does not give us avenue to make D_1 which depends on both n and m. The central question thus is whether such an improvement can be obtained for resiliency as well, which we highlight in conclusion.

3.4 Conclusion

Forrelation is one of the central problems in the quantum paradigm. This has been studied to demonstrate separation between the bounded error quantum model and the randomized classical model, starting with the seminal paper of Aaronson et al. [2]. The main contribution of [2] concentrated on designing a class of formulations (k-fold Forrelation, that can be defined for any k functions) and provide negative results related to limitation of the classical computation for evaluating such formulations.

In this chapter, we have revisited the Forrelation problem to build a unified framework towards analyzing different Boolean function spectra. First, we studied a desirable instantiation of 2-fold Forrelation set-up to construct promise problems based on bent duality. Then we move to the 3-fold Forrelation set-up and use the existing techniques as well as certain modifications to analyze the Walsh, crosscorrelation, and the autocorrelation spectra of a Boolean function. We first show how Walsh spectrum value estimation can be improved by a constant factor in this framework and use that to revise the resiliency checking algorithm of [20]. Then we move to crosscorrelation estimation and sampling algorithms using 3-fold Forrelation. This also immediately provides results related to autocorrelation (when both the functions used are identical). We present several new results in this domain and also show that the results improve the autocorrelation estimation algorithm due to [12] for similar levels of accuracy. Relating 3-fold Forrelation to crosscorrelation provides a new insight in this domain of research, and to the best of our knowledge, this connection could not be identified earlier. We also exploit the concept of Dicke states to provide a more efficient algorithm when used for checking if two functions f and g are uncorrelated of degree m, giving us a polynomial advantage in some instances.

Chapter 4

Introducing nega-Forrelation and analysis of related quantum algorithms

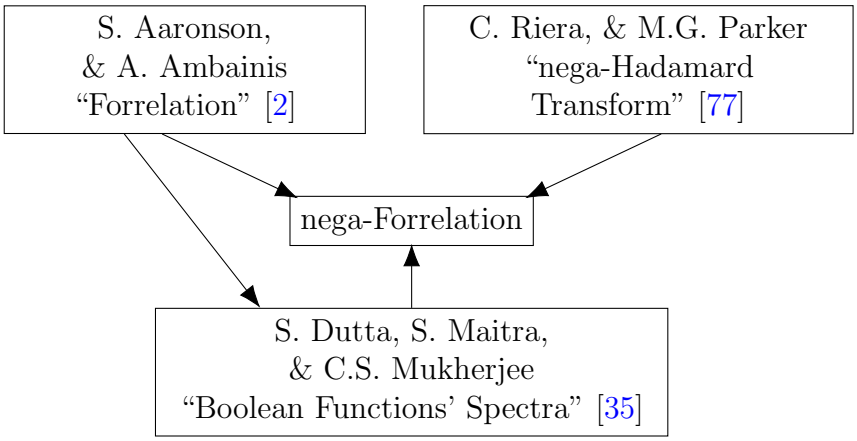
Forrelation, introduced by Aaronson [1], is a measure of correlation between the truth table of a Boolean function f and the Walsh-Hadamard transform of another function g. Although originally proposed to demonstrate a theoretical separation between the bounded-error quantum and randomized classical models [2], in Chapter 3 the Forrelation algorithm was used for sampling various cryptographic spectra of Boolean functions, including Walsh-Hadamard, autocorrelation, and crosscorrelation.

Keeping in mind its effectiveness, a natural question arises, whether a similar quantum framework can be developed for an efficient sampling of the nega-Hadamard spectrum. In this chapter, we address this by introducing nega-Forrelation (denoted η_{f_1,f_2,f_3}), a complex-valued correlation measure between the truth table of a Boolean function f_1 and the nega-Hadamard and conjugate nega-Hadamard transforms of f_2 and f_3, respectively.

Riera and Parker [77] introduced nega-Hadamard transform to study the generalized bent criteria for Boolean functions having flat nega-Hadamard spectrum. They highlighted the relevance of nega-Hadamard transform in quantum information theory, particularly in analyzing stabilizer states, and also hinted at cryptographic implications, such as spectral properties of the AES S-box [77, Section I(C), p. 4145]. Like the Walsh-Hadamard and crosscorrelation spectra, the investigations in the nega-domain offer additional tools for analyzing Boolean functions in cryptographic applications.

Here, we revisit the standard Forrelation algorithm and modify it to construct nega-Forrelation circuits. Furthermore, we adapt the techniques used for efficient sampling of various cryptographic spectra via Forrelation [35], and tailor them judiciously within the nega-Forrelation algorithms to enable efficient sampling of nega-Hadamard transforms.

In this way, the present chapter bridges the contributions of [2], [35], and [77], offering a deeper understanding of the current state-of-the-art.



A key component in this construction is the nega-Hadamard gate [40, 77], defined as

$$\mathrm{N} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & i \\ 1 & -i \end{pmatrix}$$

where $i = \sqrt{-1}$. Applied to a general n-qubit state $|\psi\rangle = \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle$, with $\sum_{\mathbf{x}} |\alpha_{\mathbf{x}}|^2 = 1$, the n-qubit nega-Hadamard gate, $(\mathrm{N}^{\otimes n})$ yields:

$$\mathrm{N}^{\otimes n} \left(\sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle \right) = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} \left(\sum_{\mathbf{y}\in\mathbb{F}_2^n} (-1)^{\mathbf{x}\cdot\mathbf{y}} (i)^{wt(\mathbf{x})} |\mathbf{y}\rangle \right).$$

Similarly, the conjugate nega-Hadamard gate is $\overline{\mathrm{N}} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -i \\ 1 & i \end{pmatrix}$ and the functioning of $\overline{\mathrm{N}}^{\otimes n}$ over the generic quantum state $|\psi\rangle$ is given as follows.

$$\overline{\mathrm{N}}^{\otimes n} \left(\sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle \right) = \frac{1}{\sqrt{2^n}} \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} \left(\sum_{\mathbf{y}\in\mathbb{F}_2^n} (-1)^{\mathbf{x}\cdot\mathbf{y}} (-i)^{wt(\mathbf{x})} |\mathbf{y}\rangle \right).$$

Note that, unlike the Forrelation algorithm, where the n-qubit Hadamard gates are applied at the beginning, in between the oracles and towards the end of the circuit, in nega-Forrelation circuit, we use different combination of H, N and $\overline{\mathrm{N}}$ gates judiciously in order to manipulate the final state and obtain the desired results upon measurement. In this regard, the phase gate, $\mathrm{S} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$ can also be used to introduce the factor $(i)^{wt(\mathbf{x})}$ in the algorithm, defined as follows.

$$\mathrm{S}^{\otimes n} \left(\sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle \right) = \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} (i)^{wt(\mathbf{x})} |\mathbf{x}\rangle.$$

The organization of this chapter and its section-wise contributions are as follows.

- In Section 4.1, we define the 3-fold nega-Forrelation formulation and present two quantum algorithms for estimating its values. The first algorithm uses three sequential queries to compute the nega-Forrelation, while the second estimates its real component using two parallel queries. We conclude the section by introducing a more query-efficient strategy for sampling small values of the nega-Hadamard transform, improving upon the Extended Deutsch-Jozsa algorithm.
- Section 4.2 presents results related to estimation of nega-crosscorrelation and nega-autocorrelation values at any arbitrary point using nega-Forrelation. Additionally, we propose a method to sample from the complete nega-crosscorrelation and nega-autocorrelation spectra at points having a fixed Hamming weight using the Dicke-states.
- Section 4.3 deals with the hidden shift finding algorithms for special classes of Boolean functions such as bent and negabent and their connection with Forrelation.
- Section 4.4 concludes the chapter with a brief summary of our contributions with future research possibilities.

4.1 The (3-fold) nega-Forrelation

Forrelation measures the correlation between the truth table of a Boolean function f and the normalized Walsh spectrum of another Boolean function g. Extending this idea, we introduce nega-Forrelation, which quantifies the correlation of the truth table of a Boolean function with the nega-Hadamard and conjugate nega-Hadamard spectrum of two other Boolean functions, defined as follows.

Definition 4.1.1. *Given Boolean functions $f_1, f_2, f_3 \in \mathcal{B}_n$, the 3-fold nega-Forrelation measures certain kind of correlation between the truth table of the Boolean functions f_1, the nega-Hadamard spectrum of f_2 and the conjugate nega-Hadamard spectrum of f_3, mathematically formulated as*

$$\eta_{f_1,f_2,f_3} = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f_1(\mathbf{x})} N_{f_2}(\mathbf{x}) \overline{N}_{f_3}(\mathbf{x}).$$

Note that, after expanding N_{f_2} and $\overline{N}_{f_3}$, we can also write the expression of 3-fold nega-Forrelation, η_{f_1,f_2,f_3} as follows.

$$\frac{1}{2^{2n}} \sum_{\mathbf{x_1},\mathbf{x_2},\mathbf{x_3}} \left((-1)^{f_2(\mathbf{x_2}) \oplus \langle \mathbf{x_1},\mathbf{x_2} \rangle} (i)^{wt(\mathbf{x_2})}\right) (-1)^{f_1(\mathbf{x_1})} \left((-1)^{f_3(\mathbf{x_3}) \oplus \langle \mathbf{x_1},\mathbf{x_3} \rangle} (-i)^{wt(\mathbf{x_3})}\right).$$

Remark 4.1.1. *In the definition of 3-fold nega-Forrelation, we consider nega-Hadamard transform of one function and the conjugate nega-Hadamard transform of another function. This is due to the fact that nega-Hadamard transform are complex numbers and when $f_2 = f_3 = f$, we obtain the complex-square of the nega-Hadamard transform values for the function, f, where the combinations are decided by the function $f_1 = g$. Further note that for $f_2 = f_3 = f$, the nega-Forrelation values, $\eta_{g,f,f}$ is always a real number.*

This result along with the nega-Parseval's identity provide an efficient sampling strategy for the nega-Hadamard transform compared to the existing result from [40] in terms of the required number of queries. In this direction, we now present two quantum algorithms for estimating the values of 3-fold nega-Forrelation, one using 3 sequential queries and another using 2 parallel queries.

4.1.1 Quantum algorithms for (3-fold) nega-Forrelation

We begin with the 3-query quantum algorithm. Given oracle access to $f_1, f_2, f_3 \in \mathcal{B}_n$, we obtain the 3-fold nega-Forrelation values, η_{f_1,f_2,f_3}, beginning with the state $|0^n\rangle |-\rangle$ and traverse through the following sequence of steps,

$$\mathrm{H}^{\otimes n} \to U_{f_2} \to \mathrm{N}^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n} \to U_{f_3} \to \overline{\mathrm{N}}^{\otimes n}$$

where all the n-qubit gates $\left(\mathrm{H}^{\otimes n}, \mathrm{N}^{\otimes n}, \overline{\mathrm{N}}^{\otimes n}\right)$ are applied to the n query-qubits and the oracles $(U_{f_2}, U_{f_1}, U_{f_3})$ are applied to all the $n+1$ qubits (see Figure 4.1). Ignoring the

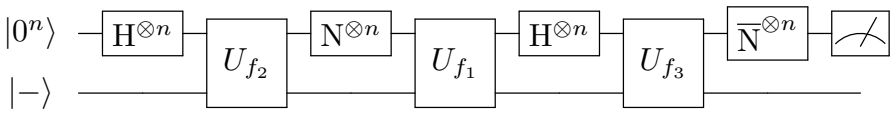


Figure 4.1: Quantum circuit for (3-fold) nega-Forrelation using 3 sequential queries.

last qubit, the amplitude corresponding to the state $|0^n\rangle$ becomes

$$\frac{1}{2^{2n}} \sum_{\mathbf{x_1},\mathbf{x_2},\mathbf{x_3}} (-1)^{f_2(\mathbf{x_2})} (i)^{wt(\mathbf{x_2})} (-1)^{\langle \mathbf{x_1},\mathbf{x_2} \rangle} (-1)^{f_1(\mathbf{x_1})} (-1)^{\langle \mathbf{x_1},\mathbf{x_3} \rangle} (-i)^{wt(\mathbf{x_3})} (-1)^{f_3(\mathbf{x_3})}$$

which is equal to η_{f_1,f_2,f_3}. Since, η_{f_1,f_2,f_3} is a complex number, the probability of observing the all zero state upon measurement is given by $|\eta_{f_1,f_2,f_3}|^2$. Let us denote the 3-query algorithm for estimating the nega-Forrelation value, η_{f_1,f_2,f_3} by $\tilde{A}_n^{3,3}(f_1, f_2, f_3)$.

Remark 4.1.2. *Unlike the Forrelation algorithm where only the Hadamard gates were used in the beginning, in between the oracles, and before the measurements, here we use the nega-Hadamard and conjugate nega-Hadamard gates judiciously in order to obtain the desired formulation. For any given functions $f_1, f_2, f_3 \in \mathcal{B}_n$, the quantum circuit of the algorithm $\tilde{A}_n^{3,3}$ makes 3 sequential queries to the underlying Boolean functions.*

Following the idea of parallel query Forrelation algorithm from [2], we now present the 2-query quantum algorithm for estimating nega-Forrelation.

Given oracle access to $f_1, f_2, f_3 \in \mathcal{B}_n$, we begin with an $(n+2)$-qubit state, $|+\rangle |0^n\rangle |-\rangle$, where the first qubit is called the 'driving qubit' (since, controlled on this qubit, we apply different quantum gates on other qubits) and the next n many qubits are the query-qubits. We first apply the n-qubit Hadamard gate, $\mathrm{H}^{\otimes n}$ to all the query qubits, and distribute the state as follows:

$$|+\rangle |0^n\rangle |-\rangle \xrightarrow{\mathrm{H}^{\otimes n}} \frac{1}{\sqrt{2^{n+1}}} \left(\sum_{\mathbf{x_2} \in \mathbb{F}_2^n} |0\rangle |\mathbf{x_2}\rangle |-\rangle + \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} |1\rangle |\mathbf{x_3}\rangle |-\rangle \right).$$

Then controlled on the driving qubit being in the $|0\rangle$ state we sequentially apply $U_{f_2} \to \mathrm{N}^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n}$ and obtain the state

$$\frac{|0\rangle}{\sqrt{2^{3n+1}}} \sum_{\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_2(\mathbf{x_2})} (i)^{wt(\mathbf{x_2})} (-1)^{\langle \mathbf{x_1}, \mathbf{x_2} \rangle} (-1)^{f_1(\mathbf{x_1})} (-1)^{\langle \mathbf{x_1}, \mathbf{x_3} \rangle} |\mathbf{x_3}\rangle |-\rangle .$$

Similarly, controlled on the driving qubit being in the $|1\rangle$ state, we sequentially apply: $\mathrm{S}^{\otimes n} \to U_{f_3}$ and obtain the state

$$\frac{|1\rangle}{\sqrt{2^{n+1}}} \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_3(\mathbf{x_3})} (i)^{wt(\mathbf{x_3})} |\mathbf{x_3}\rangle |-\rangle .$$

After ignoring the last qubit and assuming the following notations:

$$\alpha_{\mathbf{x_3}} = \left(\frac{1}{\sqrt{2^{3n+1}}} \sum_{\mathbf{x_1}, \mathbf{x_2} \in \mathbb{F}_2^n} (-1)^{f_2(\mathbf{x_2})} (i)^{wt(\mathbf{x_2})} (-1)^{\langle \mathbf{x_1}, \mathbf{x_2} \rangle} (-1)^{f_1(\mathbf{x_1})} (-1)^{\langle \mathbf{x_1}, \mathbf{x_3} \rangle} \right)$$

$$\text{and } \beta_{\mathbf{x_3}} = \frac{1}{\sqrt{2^{n+1}}} (-1)^{f_3(\mathbf{x_3})} (i)^{wt(\mathbf{x_3})},$$

we obtain the final state, (say) $|\psi\rangle = \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (\alpha_{\mathbf{x_3}} |0\rangle |\mathbf{x_3}\rangle + \beta_{\mathbf{x_3}} |1\rangle |\mathbf{x_3}\rangle)$. Finally, we measure the driving qubit in Hadamard basis, which is equivalent to applying a Hadamard gate, followed by the measurement in the $\{|0\rangle, |1\rangle\}$ basis. The final state becomes

$$\frac{1}{\sqrt{2}} \left(\sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (\alpha_{\mathbf{x_3}} + \beta_{\mathbf{x_3}}) |0\rangle |\mathbf{x_3}\rangle + \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (\alpha_{\mathbf{x_3}} - \beta_{\mathbf{x_3}}) |1\rangle |\mathbf{x_3}\rangle \right)$$

and thus the probability of obtaining $|0\rangle |\mathbf{x_3}\rangle$, where $\mathbf{x_3} \in \mathbb{F}_2^n$, is given by

$$\frac{1}{2} \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} |\alpha_{\mathbf{x_3}} + \beta_{\mathbf{x_3}}|^2 = \frac{1}{2} \left[\sum_{\mathbf{x_3} \in \mathbb{F}_2^n} \left(|\alpha_{\mathbf{x_3}}|^2 + |\beta_{\mathbf{x_3}}|^2 \right) + \Re\left(\alpha_{\mathbf{x_3}} \overline{\beta}_{\mathbf{x_3}} \right) \right]$$

where $\Re(z)$ denotes the real part of the complex number, z. Since, $\sum_{\mathbf{x_3}\in\mathbb{F}_2^n} |\alpha_{\mathbf{x_3}}|^2 + |\beta_{\mathbf{x_3}}|^2$ denotes the sum of squares of all the amplitudes for the state $|\psi\rangle$, it is equal to 1. Moreover, check that $\alpha_{\mathbf{x_3}}\overline{\beta}_{\mathbf{x_3}} = \eta_{f_1,f_2,f_3}$. Therefore, the probability of observing the $|0\rangle$ state upon measuring the driving qubit is given by $\frac{1}{2}\left(1 + \Re\left(\eta_{f_1,f_2,f_3}\right)\right)$. We denote the 2-query nega-Forrelation algorithm by $\tilde{A}_n^{2,3}$. Figure 4.2 provides a schematic diagram of the quantum circuit for $\tilde{A}_n^{2,3}(f_1, f_2, f_3)$.

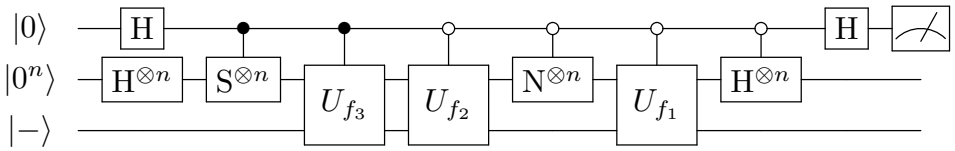


Figure 4.2: Quantum circuit for (3-fold) nega-Forrelation using 2 parallel queries.

Given $f_1, f_2, f_3 \in \mathcal{B}_n$, the quantum algorithm $\tilde{A}_n^{2,3}(f_1, f_2, f_3)$ makes one parallel query to U_{f_2} and U_{f_3}, each followed by a single query to U_{f_1}. Since the nega-Forrelation value, η_{f_1,f_2,f_3} can be a complex number, using the 2-query algorithm, $\tilde{A}_n^{2,3}(f_1, f_2, f_3)$ we can only estimate the real part of η_{f_1,f_2,f_3} and not the complete nega-Forrelation values. Next we present the strategies for sampling the nega-Hadamard transform values for a given set of points, using $\tilde{A}_n^{3,3}$ and $\tilde{A}_n^{2,3}$.

4.1.2 Sampling nega-Hadamard transform values using nega-Forrelation

Given $f \in \mathcal{B}_n$, and a set of points $S \subseteq \mathbb{F}_2^n$, the objective is to determine the total nega-Hadamard transform values of f at all the points in S. In this regard, we define $g \in \mathcal{B}_n$, using S as its support such that $g(\mathbf{x}) = 1$ for all $\mathbf{x} \in S$, and $g(\mathbf{x}) = 0$ otherwise.

Recall (from Section 2.2.1) that using the extended Deutsch-Jozsa algorithm, we can sample the nega-Hadamard transform of f at $S \subseteq \mathbb{F}_2^n$ with probability

$$\frac{1}{2^n}\sum_{\mathbf{x}\in S} |N_f(\mathbf{x})|^2 =: p \text{ (say)}$$

where p is a real number (probability) lies between 0 and 1. Here we present a strategy for sampling the nega-Hadamard transform of f using the 3-fold nega-Forrelation algorithms.

From the definition of nega-Forrelation, using $f_2 = f_3 = f$ and $f_1 = g$, we have

$$\eta_{g,f,f} = \frac{1}{2^n}\sum_{\mathbf{x}\in\mathbb{F}_2^n} (-1)^{g(\mathbf{x})} |N_f(\mathbf{x})|^2 = \frac{1}{2^n}\left(\sum_{\mathbf{x}\notin S} |N_f(\mathbf{x})|^2 - \sum_{\mathbf{x}\in S} |N_f(\mathbf{x})|^2\right).$$

From the nega-Parseval's identity, $\sum_{\mathbf{x}\in\mathbb{F}_2^n} |N_f(\mathbf{x})|^2 = 2^n$ we obtain,

$$\eta_{g,f,f} = \frac{1}{2^n}\left(2^n - \sum_{\mathbf{x}\in S} |N_f(\mathbf{x})|^2 - \sum_{\mathbf{x}\in S} |N_f(\mathbf{x})|^2\right) = 1 - \frac{2}{2^n}\left(\sum_{\mathbf{x}\in S} |N_f(\mathbf{x})|^2\right) = 1 - 2p.$$

This implies, $p = \frac{1}{2}(1 - \eta_{g,f,f})$ which is same as the probability of observing 1 upon measuring the driving qubit from running the algorithm $\tilde{A}^{2,3}(g, f, f)$. Therefore, we have the following proposition.

Proposition 4.1.1. *Given $f, g \in \mathcal{B}_n$ and a set of points $S \subseteq \mathbb{F}_2^n$ such that $g(\mathbf{x}) = 1$ for all $\mathbf{x} \in S$ and $g(\mathbf{x}) = 0$ otherwise, the probability of obtaining 1 upon measuring the driving qubit, from running the algorithm $\tilde{A}^{2,3}(g, f, f)$ is given by p.*

Remark 4.1.3. *Since $\eta_{g,f,f}$ is a real number, we have $\Re(\eta_{g,f,f}) = \eta_{g,f,f}$. Therefore, the 2-query nega-Forrelation algorithm $\tilde{A}^{(2,3)}(g, f, f)$ makes a single query to f and another query to g, designed based on the set S, behaves exactly equivalent to the extended Deutsch-Jozsa algorithm in terms of the number of queries required to sample the nega-Hadamard transform.*

Here, we show that the 3-query 3-fold nega-Forrelation algorithm, $\tilde{A}^{(3,3)}(g, f, f)$ can be used for an improvement. Observe that, upon running the algorithm $\tilde{A}^{3,3}(g, f, f)$, the probability of observing a state with at-least one many 1 in the output bit-pattern is given by $1 - \eta_{g,f,f}^2$. Using $\eta_{g,f,f} = 1 - 2p$ we obtain,

$$1 - \eta_{g,f,f}^2 = 1 - (1 - 2p)^2 = 4p - 4p^2.$$

Therefore, we have the following theorem.

Theorem 4.1.1. *Given $f, g \in \mathcal{B}_n$ and a set of points $S \subseteq \mathbb{F}_2^n$ such that $g(\mathbf{x}) = 1$ for all $\mathbf{x} \in S$ and $g(\mathbf{x}) = 0$ otherwise, the probability of one of the measurement outcomes being 1 from running the 3-query nega-Forrelation algorithm $\tilde{A}^{3,3}(g, f, f)$ is given by $4p - 4p^2$.*

From Theorem 4.1.1, it is evident that for $p < 0.75$ (i.e., when $p < 4p - 4p^2$), the sampling probability obtained from 3-query 3-fold nega-Forrelation algorithm $\left(\tilde{A}^{3,3}(g, f, f)\right)$ outperforms the sampling probability due to extended Deutsch-Jozsa algorithm, in terms of the number of queries required.

In simpler terms, for any $f \in \mathcal{B}_n$, the 3-query nega-Forrelation algorithm $\tilde{A}^{3,3}(g, f, f)$ samples the smaller values of nega-Hadamard transform more efficiently than the extended Deutsch-Jozsa algorithm [40]. Conversely, for larger values of p (where $p > 0.75$), the extended Deutsch-Jozsa algorithm alone is sufficient to estimate the nega-Hadamard transform values.

Here, one might argue that the extended Deutsch-Jozsa algorithm requires to make a single query to the oracle of the concerned Boolean function, whereas the 3-query

3-fold nega-Forrelation algorithm is making two queries to f. However, to counter that, even from running the extended Deutsch-Jozsa twice (two queries to f), one can sample the nega-Hadamard transform values with probability $1-(1-p)^2 = 2p - p^2$ which is also lower compared to sampling probability obtained from $\tilde{A}^{3,3}$. We can also verify that the sampling probability from $\tilde{A}^{3,3}$ is better compared to applying the extended Deutsch-Jozsa once followed by a single round of amplitude amplification, which also requires two queries to the oracle U_f. For a graphical comparison between the sampling probabilities from different algorithms, refer to Figure 4.3, which is visually similar to the graph as in Figure 3.2 except for the fact that the value of p is different.

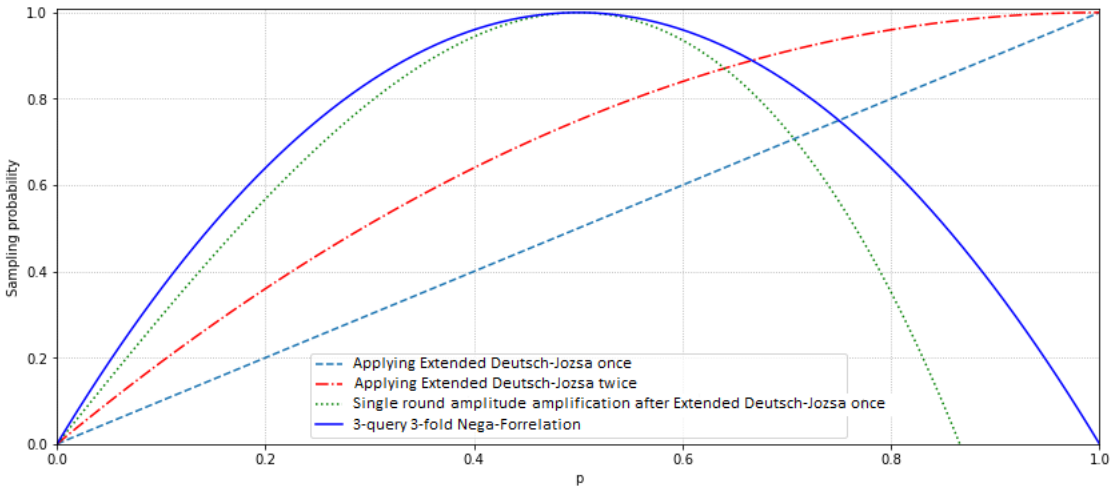


Figure 4.3: Sampling probabilities of nega-Hadamard transform using different algorithms.

Note that the 3-query nega-Forrelation algorithms samples better only for the small values of p when $p < 0.75$. However, for the larger values of p, one can sample the nega-Hadamard transform using the extended Deutsch-Jozsa algorithm, more efficiently compared to $\tilde{A}^{3,3}$, in terms of the number of queries to the oracle U_f.

4.2 Sampling nega-crosscorrelation values using nega-Forrelation

In this section, we provide an efficient sampling of the nega-crosscorrelation spectra, and consequently the nega-autocorrelation spectra, using the nega-Forrelation algorithms, with some necessary tricks and tweaks. In this regard, we use the following observation due to [91, Lemma 4], which connects the nega-crosscorrelation of two Boolean functions, $f_1, f_2 \in \mathcal{B}_n$ with the product of nega-Hadamard transforms of the corresponding

functions.

Lemma 4.2.1 ([90]). *If $f_1, f_2 \in \mathcal{B}_n$, then the nega-crosscorrelation*

$$\widehat{C}_{f_1,f_2}(\mathbf{y}) = (i)^{wt(\mathbf{y})} \sum_{\mathbf{x}\in\mathbb{F}_2^n} N_{f_1}(\mathbf{x})\overline{N_{f_2}(\mathbf{x})}(-1)^{\mathbf{x}\cdot\mathbf{y}}.$$

For the proof of this lemma, one may refer to [90, Lemma 4]. Given this lemma, we now provide an efficient strategy of sampling the nega-crosscorrelation value at any given point using algorithms, $\tilde{A}_n^{3,3}$ and $\tilde{A}_n^{2,3}$. The efficient sampling of the nega-autocorrelation value at any given point follows as an immediate corollary.

Theorem 4.2.1. *Given oracle access of $f_1, f_2 \in \mathcal{B}_n$, the algorithm $\tilde{A}_n^{3,3}$ can estimate the nega-crosscorrelation value at a point $\mathbf{y} \in \mathbb{F}_2^n$ where the probability of observing the all zero state, $|0^n\rangle$ upon measurement is given by*

$$\mathcal{P}|0^n\rangle = \frac{1}{2^{2n}}\left|\widehat{C}_{f_1,f_2}(\mathbf{y})\right|^2.$$

Moreover, the algorithm $\tilde{A}_n^{2,3}$ estimates the real part of the nega-crosscorrelation value at any given point $\mathbf{y} \in \mathbb{F}_2^n$, where the probability of observing $|0\rangle$ upon measuring the driving qubit, is given by

$$\mathcal{P}|0\rangle = \frac{1}{2}\left(1 + \Re\left(\frac{(-i)^{wt(\mathbf{y})}\widehat{C}_{f_1,f_2}(\mathbf{y})}{2^n}\right)\right).$$

Proof. Suppose, $h(\mathbf{x}) \in \mathcal{B}_n$ such that $h(\mathbf{x}) = \mathbf{x}\cdot\mathbf{y}$. Then, following Lemma 4.2.1, we can write $\widehat{C}_{f_1,f_2}(\mathbf{y})$ as

$$\widehat{C}_{f_1,f_2}(\mathbf{y}) = (i)^{wt(\mathbf{y})} \sum_{\mathbf{x}\in\mathbb{F}_2^n} (-1)^{h(\mathbf{x})} N_{f_1}(\mathbf{x})\overline{N_{f_2}(\mathbf{x})} = (i)^{wt(\mathbf{y})} 2^n \cdot \eta_{h,f_1,f_2},$$

which implies $\eta_{h,f_1,f_2} = 2^{-n}(-i)^{wt(\mathbf{y})}\widehat{C}_{f_1,f_2}(\mathbf{y})$. Since, we can estimate η_{h,f_1,f_2} from $\tilde{A}_n^{3,3}$ with $\tilde{A}_n^{2,3}$, the rest of the proof follows directly. □

This gives us a constant query algorithm for sampling the nega-crosscorrelation value for any two functions, $f_1, f_2 \in \mathcal{B}_n$ at any given point, $\mathbf{y} \in \mathbb{F}_2^n$. For $f_1 = f_2 = f$, we obtain a constant query sampling strategy for nega-autocorrelation ($\widehat{C}_f$) as an immediate corollary.

Corollary 4.2.1. *Given oracle access of $f \in \mathcal{B}_n$, the algorithm $\tilde{A}_n^{3,3}$ estimates the nega-autocorrelation value at a point $\mathbf{y} \in \mathbb{F}_2^n$ where the probability of observing the all zero state, $|0^n\rangle$ upon measurement is given by $\mathcal{P}|0^n\rangle = \frac{|\widehat{C}_f(\mathbf{y})|^2}{2^{2n}}$.*

Moreover, the algorithm $\tilde{A}_n^{2,3}$ estimates the real part of the nega-autocorrelation value at any given point $\mathbf{y} \in \mathbb{F}_2^n$, where the probability of observing $|0\rangle$ upon measuring the driving qubit, is given by $\mathcal{P}|0\rangle = \frac{1}{2}\left[1 + \Re\left(2^{-n}(-i)^{wt(\mathbf{y})}\widehat{C}_f(\mathbf{y})\right)\right]$.

Further, we have another direct corollary from the following lemma.

Lemma 4.2.2 ([90]). *A Boolean function $f \in \mathcal{B}_n$ is negabent if and only if $\widehat{C}_f(\mathbf{y}) = 0$ for all $\mathbf{y} \in \mathbb{F}_2^n \setminus \{0^n\}$.*

The corollary is as follows.

Corollary 4.2.2. *Given $f \in \mathcal{B}_n$ is negabent and $g \in \mathcal{B}_n$ is a linear function, then the presence or absence of the all zero state in the measurement outcome of the 3-fold nega-Forrelation algorithm $\tilde{A}_n^{3,3}(g, f, f)$ is determined by whether g is a constant Boolean function such that $g(\mathbf{x}) = 0$ for all $\mathbf{x} \in \mathbb{F}_2^n$ or not, respectively.*

The proof of this corollary is direct from using $f_1 = f_2 = f$ and taking $g(\mathbf{x}) = \mathbf{x} \cdot \mathbf{y}$ for all $\mathbf{x} \in \mathbb{F}_2^n$ in Lemma 4.2.1, followed by Lemma 4.2.2.

The above results sample the nega-crosscorrelation and nega-autocorrelation value at a single point $\mathbf{y} \in \mathbb{F}_2^n$ by choosing the linear function $h(\mathbf{x}) = \mathbf{x} \cdot \mathbf{y}$, that depends upon $\mathbf{y}$. In the coming endeavor, we attempt to sample the complete spectrum of nega-crosscorrelation (and thus the nega-autocorrelation) by placing a superposition of all possible linear functions, in place of a single linear function (shown in Figure 4.4).

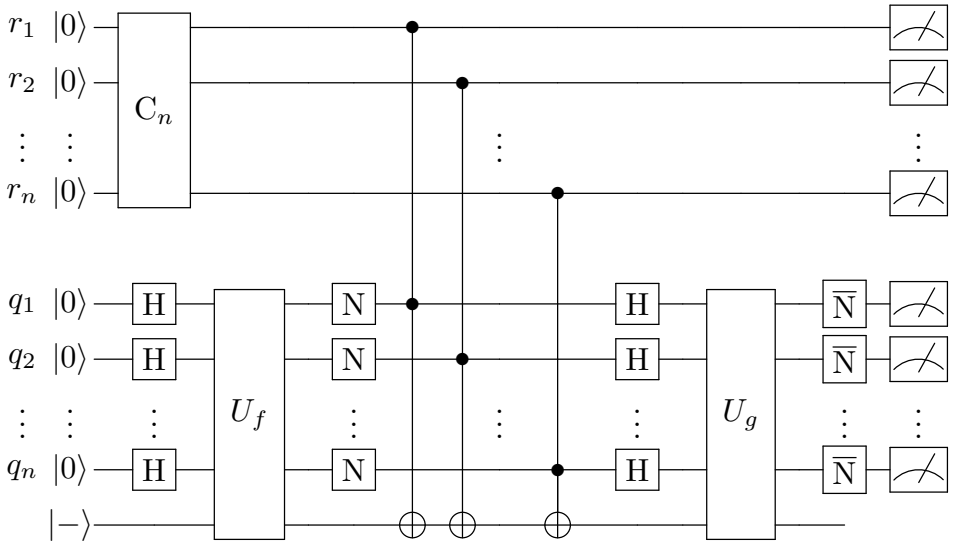


Figure 4.4: Quantum circuit for sampling the complete nega-crosscorrelation spectrum.

Suppose that C_n is a $2^n \times 2^n$ unitary operator such that $\mathrm{C}_n |0^n\rangle = \sum_{\mathbf{x} \in \mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle$. Then, starting from the all zero state, the algorithm $\tilde{\mathbb{A}}(\mathrm{C}_n)$ as in Figure 4.4 has the pre-measurement state

$$\sum_{\mathbf{y} \in \mathbb{F}_2^n} \alpha_{\mathbf{y}} |\mathbf{y}\rangle \left(\frac{\widehat{C}_{f,g}(\mathbf{y})}{2^n} |0^n\rangle + \beta_{\mathbf{y}} |W_{\mathbf{y}}\rangle \right)$$

where $|W_{\mathbf{y}}\rangle$ is an n-qubit superposition state such that the amplitude of the state $|0^n\rangle$ is 0. In the next theorem, we show how the specific choice of C_n helps in sampling the complete spectrum of nega-crosscorrelation.

Theorem 4.2.2. *Fixing $\mathrm{C}_n = \mathrm{H}^{\otimes n}$, the probability of observing the bit-string $|\mathbf{y}\rangle|0^n\rangle$ upon measuring the first $2n$ qubits from Algorithm $\tilde{\mathbb{A}}$ is given by $\frac{\widehat{C}_{f,g}(\mathbf{y})^2}{2^{3n}}$ for all $\mathbf{y} \in \mathbb{F}_2^n$.*

Proof. Fixing $\mathrm{C}_n = \mathrm{H}^{\otimes n}$, the pre-measurement state becomes

$$\sum_{\mathbf{y}\in\mathbb{F}_2^n} \frac{1}{2^{\frac{n}{2}}} |\mathbf{y}\rangle \left(\frac{\widehat{C}_{f,g}(\mathbf{y})}{2^n} |0^n\rangle + \beta_{\mathbf{y}} |W_{\mathbf{y}}\rangle \right).$$

Thus the probability of observing the state $\mathbf{y}||0^n$ upon measuring the first $2n$ many qubits is given by $\frac{\widehat{C}_{f,g}(\mathbf{y})^2}{2^{3n}}$. □

Corollary 4.2.3. *Let $f \in \mathcal{B}_n$ be a negabent function. Then, from running the Algorithm $\tilde{\mathbb{A}}(\mathrm{H}^{\otimes n})$ with $f_1 = f_2$, the probability of observing the state $|0^n\rangle|0^n\rangle$ upon measuring all the qubits, is given by $\frac{1}{2^n}$. Moreover, the probability of observing a state $|\mathbf{x}\rangle|0^n\rangle$ is 0 for all $\mathbf{x} \in \mathbb{F}_2^n \setminus \{0^n\}$.*

The proof is direct from Corollary 4.2.2, from the fact that the nega-autocorrelation for a negabent function at any non-zero point is 0.

In this regard, let us assume that the $2^n \times 2^n$ unitary matrix UD_m^n prepares all the Dicke-state $|D_m^n\rangle$ of weight m such that

$$UD_m^n |0^n\rangle = \frac{1}{\sqrt{\binom{n}{m}}} \sum_{\mathbf{x}:wt(\mathbf{x})=m} |\mathbf{x}\rangle.$$

Thus, if we replace $\mathrm{C}_n = UD_m^n$ with $m < n$, the probability of observing the n-length bit-string $\mathbf{y}\|0^n$ becomes $\frac{\widehat{C}_{f,g}(\mathbf{y})^2}{\binom{n}{m}2^{2n}}$, where the Hamming weight of $\mathbf{y}$ is given by m and the probability of obtaining the bit-string $\mathbf{y}\|0^n$ is zero if $wt(\mathbf{y}) \neq m$. In this way, we can sample the nega-crosscorrelation (and hence the nega-autocorrelation) values at all the points having the Hamming weight m.

4.3 Finding the hidden-shift for bent and negabent functions

As discussed in Section 2.1, bent functions are a special class of Boolean functions, that are maximally distant (in terms of the Hamming distance between the corresponding truth tables) from the class of linear Boolean functions. Because of the nonlinearity, bent

functions plays an important role in the design of cryptographic primitives. Moreover, the Walsh-Hadamard transform of a bent function is either $+1$ or -1, i.e., the absolute value of the Walsh transform is constant, and distributed uniformly over the entire space. Given $f \in \mathcal{B}_n$, an n-variable Boolean function, $\widehat{f}$ that mimics the Walsh spectrum of f in terms of the equation $W_f(\mathbf{x}) = (-1)^{\widehat{f}(\mathbf{x})}$ is called the dual of f (see Section 2.1).

Let f and g be bent functions such that $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$ holds for all $\mathbf{x} \in \mathbb{F}_2^n$. Now, given the oracle access of g and the dual of f, $U_{\widehat{f}}$, [78] proposed a polynomial time quantum algorithm that deterministically computes the hidden shift $\mathbf{u} \in \mathbb{F}_2^n$ using just a single query to each of the oracles, U_g and $U_{\widehat{f}}$. A schematic diagram for the corresponding circuit is shown in Figure 4.5 where the measurement outcome is the unknown shift $\mathbf{u}$. For detail derivation, one may refer to [78, Theorem 4.1].

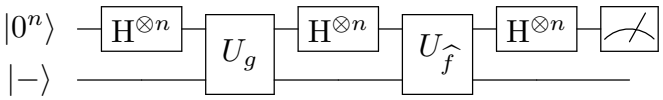


Figure 4.5: Quantum circuit for finding the hidden shift of a bent function.

The circuit diagram (Figure 4.5) reveals a clear similarity between the quantum hidden shift finding algorithm [78] and the 2-query, 2-fold Forrelation algorithm [2], with certain modifications. However, to the best of our knowledge, the connection between the hidden shift problem and Forrelation has not been established prior to this work. Accordingly, we can reinterpret the hidden shift finding algorithm within the framework of Forrelation as follows.

Proposition 4.3.1. *Given f, g are bent functions, such that $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$ for all $\mathbf{x} \in \mathbb{F}_2^n$, then the 2-fold Forrelation, $\Phi_{g,\widehat{f}} = 1$ if $\mathbf{u} = 0^n$, and $\Phi_{g,\widehat{f}} = 0$ if $\mathbf{u} \neq 0^n$.*

The proof of this proposition is direct from [78, Theorem 4.1] and [35, Proposition 1]. Additionally, it can be observed from the fact that the term-by-term product of the Walsh-Hadamard spectrum of the individual functions (f and g) is equal to the Walsh-Hadamard transform of their convolution ($f * g$), as follows.

The 2-fold Forrelation ($\Phi_{g,\widehat{f}}$) as defined in Proposition 4.3.1 (with $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$) is equivalent to the normalized autocorrelation coefficient of f at $\mathbf{u} \in \mathbb{F}_2^n$:

$$\begin{aligned}
\Phi_{g,\widehat{f}} &= \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{g(\mathbf{x})} W_{\widehat{f}}(\mathbf{x}) \\
&= \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{g(\mathbf{x}) \oplus f(\mathbf{x})} \\
&= \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x} \oplus \mathbf{u}) \oplus f(\mathbf{x})} = \frac{1}{2^n} C_f(\mathbf{u}) \text{ (From Section 2.1).}
\end{aligned}$$

Moreover, from [80, Corollary 3.1], we have the following connection between the auto-correlation of a Boolean function (C_f) and the square of Walsh-Hadamard transforms (W_f):

$$\big[C_f(0^n),\ldots,C_f(1^n)\big]\hat{\mathrm{H}}_n = 2^n\big[W_f^2(0^n),\ldots,W_f^2(1^n)\big]$$

where $\hat{\mathrm{H}}_n$ represents the $2^n\times 2^n$ classical Sylvester Hadamard matrix (equivalent to the n-qubit Hadamard gate without the normalizing factor). The term 2^n is multiplied in the RHS to address the normalization in our definition of Walsh-Hadamard transform (Section 2.1). As f is a bent function, $(W_f(\mathbf{u}))^2 = 1$ for all $\mathbf{u}\in\mathbb{F}_2^n$. By multiplying $\hat{\mathrm{H}}_n$ on both sides, we obtain:

$$2^n\big[C_f(0^n),\ldots,C_f(1^n)\big] = 2^n\big[1,\ldots,1\big]\hat{\mathrm{H}}_n.$$

Since all columns of $\hat{\mathrm{H}}_n$ sum to 0, except the first one (which sums to 2^n), the RHS simplifies to $2^n\big[2^n,0,\ldots,0\big]$. Canceling 2^n both side, we obtain

$$\big[C_f(0^n),\ldots,C_f(1^n)\big] = \big[2^n,0,\ldots,0\big].$$

Hence, from the fact that $\Phi_{g,\widehat{f}} = \frac{1}{2^n}C_f(\mathbf{u})$, we conclude:

$$\big[\Phi_{g,\widehat{f}}\big|_{\mathbf{u}=0^n},\ldots,\Phi_{g,\widehat{f}}\big|_{\mathbf{u}=1^n}\big] = \big[1,0,\ldots,0\big]$$

affirming Proposition 4.3.1.

Next, we extend this approach to analyze the affine properties of bent functions, in a more general setup.

Theorem 4.3.1. *Let $f,g\in\mathcal{B}_n$ be bent functions such that $g(\mathbf{x}) = f(\mathbf{x}\oplus\mathbf{b})\oplus\mathbf{c}\cdot\mathbf{x}\oplus d$, where $\mathbf{b},\mathbf{c}\in\mathbb{F}_2^n$ and $d\in\mathbb{F}_2$. Then, running the 2-fold Forrelation algorithm with U_g as the first oracle and the dual of f, $U_{\widehat{f}}$, as the second (see Figure 4.5), the state before measurement is given by $(-1)^{\mathbf{b}\cdot\mathbf{c}\oplus d}2^{-n}\sum_{\mathbf{y},\mathbf{z}}(-1)^{\widehat{f}(\mathbf{y}\oplus\mathbf{c})\oplus\widehat{f}(\mathbf{y})\oplus\mathbf{y}\cdot(\mathbf{z}\oplus\mathbf{b})}|\mathbf{z}\rangle$, and the probability of observing any state $\mathbf{z}\in\mathbb{F}_2^n$ is*

$$2^{-2n}\Big|\sum_{\mathbf{y}}(-1)^{\widehat{f}(\mathbf{y}\oplus\mathbf{c})\oplus\widehat{f}(\mathbf{y})\oplus\mathbf{y}\cdot(\mathbf{z}\oplus\mathbf{b})}\Big|^2.$$

Proof. We follow the 2-fold Forrelation circuit from [78], illustrated in Figure 4.5. The

starting state $|0^n\rangle$ evolves as:

$$
\begin{aligned}
|0^n\rangle \xrightarrow{\mathrm{H}^{\otimes n}} \frac{1}{2^{n/2}} \sum_{\mathbf{x}\in\mathbb{F}_2^n} |\mathbf{x}\rangle \xrightarrow{U_g} \frac{1}{2^{n/2}} \sum_{\mathbf{x}\in\mathbb{F}_2^n} (-1)^{f(\mathbf{x}\oplus\mathbf{b})\oplus\mathbf{c}\cdot\mathbf{x}\oplus d} |x\rangle \\
&\xrightarrow{\mathrm{H}^{\otimes n}} \frac{(-1)^d}{2^n} \sum_{\mathbf{x},\mathbf{y}\in\mathbb{F}_2^n} (-1)^{f(\mathbf{x}\oplus\mathbf{b})\oplus\mathbf{c}\cdot\mathbf{x}\oplus\mathbf{x}\cdot\mathbf{y}} |\mathbf{y}\rangle \\
&= \frac{(-1)^{\mathbf{b}\cdot\mathbf{c}\oplus d}}{2^n} \sum_{\mathbf{x},\mathbf{y}\in\mathbb{F}_2^n} (-1)^{\mathbf{b}\cdot\mathbf{y}} (-1)^{f(\mathbf{x}\oplus\mathbf{b})\oplus(\mathbf{x}\oplus\mathbf{b})\cdot(\mathbf{y}\oplus\mathbf{c})} |\mathbf{y}\rangle \\
&= \frac{(-1)^{\mathbf{b}\cdot\mathbf{c}\oplus d}}{2^{n/2}} \sum_{\mathbf{y}\in\mathbb{F}_2^n} (-1)^{\mathbf{b}\cdot\mathbf{y}} (-1)^{\widehat{f}(\mathbf{y}\oplus\mathbf{c})} |\mathbf{y}\rangle \\
&\xrightarrow{U_{\widehat{f}}} \frac{(-1)^{\mathbf{b}\cdot\mathbf{c}\oplus d}}{2^{n/2}} \sum_{\mathbf{y}\in\mathbb{F}_2^n} (-1)^{\mathbf{b}\cdot\mathbf{y}} (-1)^{\widehat{f}(\mathbf{y}\oplus\mathbf{c})\oplus\widehat{f}(\mathbf{y})} |\mathbf{y}\rangle \\
&\xrightarrow{\mathrm{H}^{\otimes n}} \frac{(-1)^{\mathbf{b}\cdot\mathbf{c}\oplus d}}{2^n} \sum_{\mathbf{y},\mathbf{z}\in\mathbb{F}_2^n} (-1)^{\mathbf{b}\cdot\mathbf{y}} (-1)^{\widehat{f}(\mathbf{y}\oplus\mathbf{c})\oplus\widehat{f}(\mathbf{y})} (-1)^{\mathbf{y}\cdot\mathbf{z}} |\mathbf{z}\rangle\,.
\end{aligned}
$$

Hence, the final amplitude of any basis state $\mathbf{z} \in \mathbb{F}_2^n$ is

$$\frac{(-1)^{\mathbf{b}\cdot\mathbf{c}\oplus d}}{2^n} \sum_{\mathbf{y}} (-1)^{\widehat{f}(\mathbf{y}\oplus\mathbf{c})\oplus\widehat{f}(\mathbf{y})} (-1)^{\mathbf{y}\cdot(\mathbf{b}\oplus\mathbf{z})},$$

and the corresponding probability is given by $\frac{1}{2^{2n}} \left| \sum_{\mathbf{y}} (-1)^{\widehat{f}(\mathbf{y}\oplus\mathbf{c})\oplus\widehat{f}(\mathbf{y})\oplus\mathbf{y}\cdot(\mathbf{b}\oplus\mathbf{z})} \right|^2$. □

Clearly, from the above theorem, the value of $d \in \mathbb{F}_2$ cannot be estimated. Furthermore, if $\mathbf{c} = 0^n$, the state $\mathbf{b}$ is observed with probability 1, as shown in [78]. When $\mathbf{b} = 0^n$, the state $|0^n\rangle$ is never observed. In general, with high probability, the observed state corresponds to the best affine approximation of the balanced Boolean function $\widehat{f}(\mathbf{y} \oplus \mathbf{c}) \oplus \widehat{f}(\mathbf{y})$. Moreover, if $\mathbf{b} \neq 0^n \neq \mathbf{c}$, then the state $\mathbf{b}$ is never observed. Instead, the most probable outcome is the state $\mathbf{z}$, where the linear function $(\mathbf{b} \oplus \mathbf{z}) \cdot \mathbf{y}$ best approximates $\widehat{f}(\mathbf{y} \oplus \mathbf{c}) \oplus \widehat{f}(\mathbf{y})$.

Summary of observations:

1. If the algorithm yields only the state $|0^n\rangle$, then either $|f(\mathbf{x})| = |g(\mathbf{x})|$ for all $\mathbf{x} \in \mathbb{F}_2^n$, or $\mathbf{b} \neq 0^n \neq \mathbf{c}$, with $\mathbf{b} \in \mathbb{F}_2^n$ being the coefficient of the best linear approximation to $\widehat{f}(\mathbf{y} \oplus \mathbf{c}) \oplus \widehat{f}(\mathbf{y})$.

2. If a fixed nonzero state $|\mathbf{z}\rangle$ is observed with probability 1, then either $\mathbf{c} = 0^n$ and the observed state is $\mathbf{b} \in \mathbb{F}_2^n$, or $\mathbf{c} \neq 0^n$ and $\widehat{f}(\mathbf{y} \oplus \mathbf{c}) \oplus \widehat{f}(\mathbf{y}) = (\mathbf{b} \oplus \mathbf{z}) \cdot y$.

3. If all the states are observed except for a fixed nonzero state, then $\mathbf{c} \neq 0^n$, and the missing state corresponds to the hidden shift $\mathbf{b} \in \mathbb{F}_2^n$.

Remark 4.3.1. *Note that to study the affine properties of bent functions, we do not modify the algorithm structurally; instead, we provide an alternative analysis based on affine transformations of the underlying bent functions.*

Next, we study the shift properties of negabent functions and explore how the existing hidden shift algorithm for bent functions can be adapted to recover the hidden shift of negabent functions. In this context, we would like to point out an error in our published result [34, Theorem 5], which claims that if $f, g \in \mathcal{B}_n$ are negabent functions, then there does not exist any $\mathbf{u} \in \mathbb{F}_2^n$ such that $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$ for all $\mathbf{x} \in \mathbb{F}_2^n$. This statement is incorrect, as there do exist negabent functions for which certain shifts result in another negabent function. In fact, [34, Theorem 5] should be replaced with Theorem 4.3.2, which provides an accurate formulation.

Theorem 4.3.2. *Let $f, g \in \mathcal{B}_n$ be bent functions such that $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$ for some $\mathbf{u} \in \mathbb{F}_2^n$, and let $f' = f \oplus s_2$, $g' = g \oplus s_2$ be the corresponding negabent functions. Then, the same shift vector $\mathbf{u}$ can not satisfy $g'(\mathbf{x}) = f'(\mathbf{x} \oplus \mathbf{u})$ for all $\mathbf{x} \in \mathbb{F}_2^n$.*

Proof. We prove this by contradiction. Suppose $g'(\mathbf{x}) = f'(\mathbf{x} \oplus \mathbf{u})$ holds for the same shift vector $\mathbf{u} \in \mathbb{F}_2^n$. Then,

$$g(\mathbf{x}) \oplus s_2(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u}) \oplus s_2(\mathbf{x}) \oplus s_2(\mathbf{u}) \oplus \bigoplus_{i=1}^{n} x_i \left(\oplus_{j=1, j\neq i}^{n} u_i\right).$$

Since $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$, we obtain $\bigoplus_{i=1}^{n} x_i \left(\oplus_{j=1, j\neq i}^{n} u_i\right) = s_2(\mathbf{u})$, i.e., the linear function $\mathbb{L}_{\mathbf{u}}(\mathbf{x}) = \bigoplus_{i=1}^{n} x_i \left(\oplus_{j=1, j\neq i}^{n} u_i\right)$ must be a constant, which implies $\mathbf{u} = 0^n$. Hence, no non-trivial $\mathbf{u}$ can satisfy the said condition. □

While the result works for all the bent function, in particular, this should be interpreted for the ones which are not negabent. Moreover, we have the following proposition.

Proposition 4.3.2. *Let $f, g \in \mathcal{B}_n$ be two bent functions such that $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$. Then $\mathbf{u}$ is a period of the negabent functions $g \oplus s_2$ and $(f \oplus s_2 \oplus \mathbb{L}_{\mathbf{u}} \oplus s_2(\mathbf{u}))$, satisfying $(g \oplus s_2)(\mathbf{x}) = (f \oplus s_2 \oplus \mathbb{L}_{\mathbf{u}} \oplus s_2(\mathbf{u}))(\mathbf{x} \oplus \mathbf{u})$. Moreover, their duals $\widehat{f}$ and $\widehat{g}$ differ by the linear function $\mathbf{u} \cdot \mathbf{x}$, i.e., $\widehat{g}(\mathbf{x}) = \widehat{f}(\mathbf{x}) \oplus \mathbf{u} \cdot \mathbf{x}$.*

Proof. We begin with $(f \oplus s_2 \oplus \mathbb{L}_{\mathbf{u}} \oplus s_2(\mathbf{u}))$:

$$\begin{aligned}(f \oplus s_2 \oplus \mathbb{L}_{\mathbf{u}} \oplus s_2(\mathbf{u}))(\mathbf{x} \oplus \mathbf{u}) =& f(\mathbf{x} \oplus \mathbf{u}) \oplus s_2(\mathbf{x} \oplus \mathbf{u}) \oplus \mathbb{L}_{\mathbf{u}}(\mathbf{x} \oplus \mathbf{u}) \oplus s_2(\mathbf{u}) \\ =& g(\mathbf{x}) \oplus s_2(\mathbf{x}) \oplus \mathbb{L}_{\mathbf{u}}(\mathbf{x}) \oplus s_2(\mathbf{u}) \oplus \mathbb{L}_{\mathbf{u}}(\mathbf{x}) \oplus s_2(\mathbf{u}) \\ =& (g \oplus s_2)(\mathbf{x}).\end{aligned}$$

Furthermore,

$$(-1)^{\widehat{g}(\mathbf{x})} = W_g(\mathbf{x}) = W_f(\mathbf{x})(-1)^{\mathbf{u}\cdot\mathbf{x}} = (-1)^{\widehat{f}(\mathbf{x})}(-1)^{\mathbf{u}\cdot\mathbf{x}} = (-1)^{\widehat{f}(\mathbf{x})\oplus\mathbf{u}\cdot\mathbf{x}}.$$

Hence, $\widehat{f}$ and $\widehat{g}$ differ by the linear function $\mathbf{u} \cdot \mathbf{x}$. □

Additionally, we have the following direct corollary.

Corollary 4.3.1. *Let $f, g \in \mathcal{B}_n$ be bent functions such that $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$ for some $\mathbf{u} \in \mathbb{F}_2^n$, and let $f' = f \oplus s_2$, $g' = g \oplus s_2$ be the corresponding negabent functions. Then*

$$g'(\mathbf{x}) = f'(\mathbf{x} \oplus \mathbf{u}) \oplus \mathbb{L}_{\mathbf{u}}(\mathbf{x}) \oplus c,$$

where $\mathbb{L}_{\mathbf{u}}(\mathbf{x}) = \bigoplus_{i=1}^{n} x_i \left(\oplus_{j=1, j\neq i}^{n} u_j\right)$ is a linear Boolean function and $c = \binom{wt(\mathbf{u})}{2} \mod (2)$ is a constant.

In this backdrop, we define a non-identical shift for the negabent functions; termed as nega-shift, formulated in the following way.

Definition 4.3.1. *Let $f, g \in \mathcal{B}_n$ be two negabent functions and $\mathbf{u} \in \mathbb{F}_2^n$ be an unknown bit string such that for all $\mathbf{x} \in \mathbb{F}_2^n$, $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u}) \oplus \mathbb{L}_{\mathbf{u}}(\mathbf{x}) \oplus c$ where $\mathbb{L}_{\mathbf{u}}(\mathbf{x}) = \bigoplus_{i=1}^{n} x_i \left(\oplus_{j=1, j\neq i}^{n} u_j\right)$ is a linear Boolean function and $c = \binom{wt(\mathbf{u})}{2} \mod (2)$ is a constant. Then $\mathbf{u}$ is called the nega-shift of the negabent function f.*

As discussed, if $\mathbf{u} \in \mathbb{F}_2^n$ is a nega-shift to the negabent functions $f, g \in \mathcal{B}_n$, then the same vector $\mathbf{u}$ becomes the shift vector for the corresponding bent functions, $f' = f \oplus s_2$, $g' = g \oplus s_2$. Consequently, using the constant query deterministic quantum algorithm proposed in [78], we can find the hidden nega-shift for the given negabent function, with some necessary modifications.

Theorem 4.3.3. *Let $f, g \in \mathcal{B}_n$ be negabent, and there exists an unknown bit-string $\mathbf{u} \in \mathbb{F}_2^n$ such that $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u}) \oplus \mathbb{L}(\mathbf{x}) \oplus c$ for all $\mathbf{x} \in \mathbb{F}_2^n$ (with $\mathbb{L}(\mathbf{x})$ and c as above). Then there exist a polynomial time quantum algorithm that deterministically computes the hidden nega-shift $\mathbf{u}$ using a single query to each of the following oracles, U_g, U_{s_2} and $U_{\widehat{f'}}$, where $\widehat{f'}$ is the dual of the bent function $f' = f \oplus s_2$.*

Proof. Given $\mathbf{u}$ being a nega-shift for the negabent functions $f, g \in \mathcal{B}_n$, implies that $\mathbf{u}$ is also a shift for the corresponding bent functions $f' = f \oplus s_2$ and $g' = g \oplus s_2$, satisfying $g'(\mathbf{x}) = f'(\mathbf{x} \oplus \mathbf{u})$. Thus, the problem of finding the hidden nega-shift between negabent functions f and g reduces to the hidden shift problem for the bent functions f' and g'. Therefore, using the constant-query hidden shift finding algorithm ($\mathcal{A}_1$) for bent functions ([78]), we can recover the shift $\mathbf{u}$ with probability 1, given oracle access to U_g, U_{s_2} and $U_{\widehat{f'}}$. □

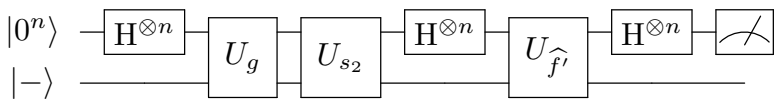


Figure 4.6: Quantum circuit for constant-query hidden nega-shift finding algorithm.

In the same direction, we have an immediate corollary using the probabilistic hidden-shift finding algorithm ($\mathcal{A}_2$) as in [78, Theorem 4.2].

Corollary 4.3.2. *Let $f, g \in \mathcal{B}_n$ be negabent, and there exists an unknown bit-string $\mathbf{u} \in \mathbb{F}_2^n$ such that $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u}) \oplus \mathbb{L}(\mathbf{x}) \oplus c$ for all $\mathbf{x} \in \mathbb{F}_2^n$. Then, there exists a polynomial time quantum algorithm that computes the hidden nega-shift $\mathbf{u}$ with a constant probability of success and makes $\mathcal{O}(n)$ many queries to U_f, U_g and U_{s_2}.*

The proof of the above corollary is direct from Theorem 4.3.3 and [78, Theorem 4.2]. The corresponding circuit diagram is shown in Figure 4.7.

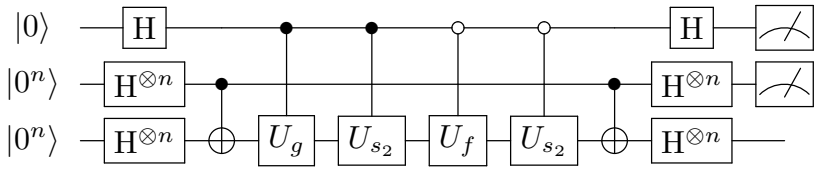


Figure 4.7: Quantum circuit for $\mathcal{O}(n)$-query hidden nega-shift finding algorithm.

The additional oracles U_{s_2} has been added in the quantum circuit to make the corresponding negabent functions bent so that the hidden shift finding algorithm ($\mathcal{A}_2$) can computes the hidden nega-shift $\mathbf{u}$.

Given two bent functions $f, g \in \mathcal{B}_n$ satisfying $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$, one can compute the hidden shift $\mathbf{u} \in \mathbb{F}_2^n$ from [78, Theorem 4.2] with constant success probability, by making $\mathcal{O}(n)$ many queries to f and g. Upon measurement, the algorithm yields the states $|b, \mathbf{z}\rangle$ which are orthogonal to the hidden shift $1, \mathbf{u}$, i.e., $(b, \mathbf{z}) \cdot (1, \mathbf{u}) = b \oplus \mathbf{u} \cdot \mathbf{z} = 0$. From the observed states, $\mathbf{u}$ can be efficiently recovered via Gaussian elimination. Here, we observe that the same algorithm can be adapted to recover the hidden shift when f and g are negabent functions.

To demonstrate this, we apply the hidden shift finding algorithm to the non-trivial 6-variable negabent function $f(x_1, x_2, x_3, x_4, x_5, x_6) = x_1x_3 \oplus x_1x_4$, which is not bent, along with its shifted version $g(\mathbf{x}) = f(\mathbf{x} \oplus 100001) = x_1x_3 \oplus x_1x_4 \oplus x_3 \oplus x_4$ (see Figure 4.8). The observed output states $|0000000\rangle$, $|1100000\rangle$, $|0001100\rangle$, and $|1101100\rangle$ appear with equal probability (see Figure 4.9), indicating that the hidden shift $\mathbf{u} = u_1u_2u_3u_4u_5u_6$ satisfies the constraints $u_1 = 1$, $u_3 = u_4$, and $u_2, u_3, u_5, u_6 \in \mathbb{F}_2$. In fact, there are exactly 2^4 such valid shifts $\mathbf{u} \in \mathbb{F}_2^6$ satisfying $g(\mathbf{x}) = f(\mathbf{x} \oplus \mathbf{u})$ where $f(\mathbf{x}) = x_1x_3 \oplus x_1x_4$ and $g(\mathbf{x}) = x_1x_3 \oplus x_1x_4 \oplus x_3 \oplus x_4$, under the conditions $u_1 = 1$ and $u_3 = u_4$.

Such techniques may be applied for any pair of Boolean functions, that may be exploited in differential cryptanalysis of symmetric ciphers. In this section, we have only concentrated on bent and negabent functions. However, its generalization might be an interesting research exercise.

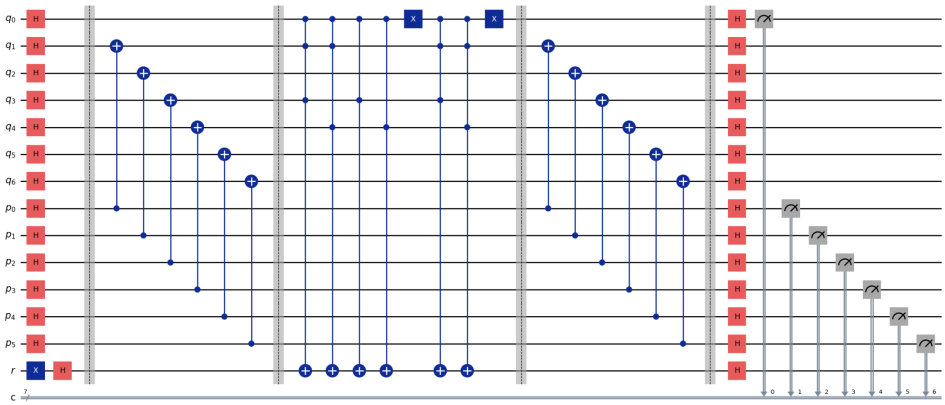


Figure 4.8: Quantum circuit for finding hidden shift from the negabent function.

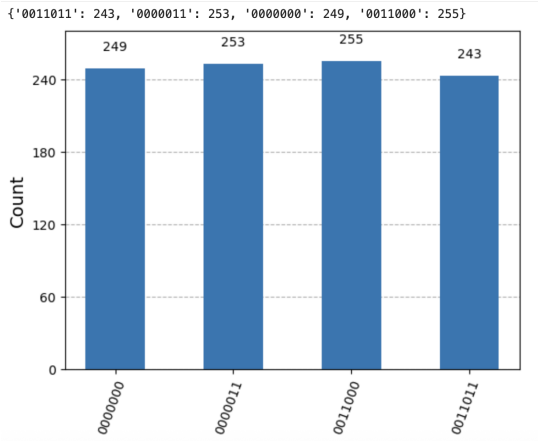


Figure 4.9: Histogram for finding hidden shift from the negabent function.

4.4 Conclusion

In this chapter, we connected Forrelation [1, 2] with the nega-Hadamard transform [77], and defined nega-Forrelation to capture certain kinds of correlation between the truth table of a Boolean function with the nega-Hadamard and conjugate-nega-Hadamard spectra of other Boolean functions. Using this formulation, we proposed an efficient sampling strategy for the nega-Hadamard transform, surpassing the sampling probability due to the extended Deutsch-Jozsa algorithm [40]. Additionally, we designed algorithms for efficient sampling of the nega-crosscorrelation (and consequently the nega-autocorrelation) spectra with minor modifications. Finally, we connected the hidden shift finding algorithm for bent functions [78] to Forrelation and extended the results for negabent functions through necessary adaptations to the existing algorithm. Implementation of the algorithms, considered in the chapter, in a noisy quantum environment could be a future research possibility in this direction.

Chapter 5

A framework for generalized Forrelation

In Chapter 3, we exploited the Forrelation algorithms to enhance sampling probabilities for various cryptographically significant spectra of Boolean functions, including the Walsh-Hadamard spectrum, crosscorrelation, and autocorrelation. Chapter 4 extended these results to the nega-domain and examined the shift properties of bent and negabent functions via Forrelation and nega-Forrelation.

In a related work, Stănică [89] generalized the Walsh-Hadamard transform to the 2^k-Hadamard transform (Definition 2.1.9), by incorporating the weight factor of 2^k-th primitive root of unity, denoted ζ_{2^k}. The notions of crosscorrelation and autocorrelation were also extended within this framework, along with the bent criterion, leading to the definition of 2^k-bent functions and the classification into strong and weak bent functions. Subsequently, Tang [94] investigated the theory and applications of 2^k-bent functions in the context of relative difference sets.

In this chapter, we further generalize these spectral frameworks from their 2^k variants to a more comprehensive formulation by incorporating the m-th primitive root of unity, ζ_m, for any $m \in \mathbb{N}$. Additionally, we generalize the Deutsch-Jozsa algorithm beyond its extended variants [40], identifying both the standard Deutsch-Jozsa [27] and its extended form [40] as special instances. Furthermore, we develop the m-Forrelation framework and propose novel quantum algorithms, with some tweaks to the existing ones, for estimating these newly defined generalized spectra.

The organization of this chapter and its section-wise contributions are as follows.

- In Section 5.1, we focus on generalizing the existing frameworks in terms of Boolean function properties. We begin with introducing new unitaries, exploring their implications, and establishing connections to existing ones. Then, we extend the formulation of various fundamental concepts including the Walsh-Hadamard, crosscorrelation, and autocorrelation spectra, to a generalized variation for any $m \in \mathbb{N}$.

In the process, we identify a previously unexamined class of real Hadamard transforms that lies between the Walsh-Hadamard and nega-Hadamard transformations, addressing a gap in the literature. Additionally, we introduce the most generalized version of the Deutsch-Jozsa algorithm, which extends both the standard Deutsch-Jozsa and its prior extended version, incorporating them as special cases. Furthermore, we extend the Forrelation formulation to m-Forrelation and propose new quantum algorithms for estimating them for a given set of Boolean functions.

- In Section 5.2, we present various sampling strategies of these newly defined spectra of Boolean functions using the generalized Forrelation algorithms, and present the comparison graph based on the corresponding sampling probabilities.
- Section 5.3 presents our final contribution, where we study affine transformations of generalized bent functions.
- Section 5.4 concludes the paper with a brief summary of our major contributions and outlines potential future research directions in this area.

5.1 Generalization of Boolean functions' spectra

In this section, we generalize various cryptographically significant spectra of Boolean functions, including the Walsh-Hadamard, crosscorrelation, and autocorrelation spectra, and then present the most generalized version of a Deutsch-Jozsa-like quantum algorithm, building upon the work of [27, 40]. Finally, we introduce the most comprehensive form of Forrelation, that encompasses both the standard Forrelation [2] and nega-Forrelation [34] as its special cases, and provide the necessary quantum algorithm for estimating it.

In this direction, let us introduce a single qubit unitary operator, Ω_m, with parameter $m \in \mathbb{N}$, defined as $\Omega_m = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & \zeta_m \\ 1 & -\zeta_m \end{pmatrix}$, where $\zeta_m = e^{2\pi i/m}$ denotes the primitive m-th complex root of unity, i.e., $(\zeta_m)^m = 1$. This gate can be derived as a special case from the well known and generalized $U_3(\theta, \phi, \lambda) = \begin{pmatrix} \cos\frac{\theta}{2} & -e^{i\lambda}\sin\frac{\theta}{2} \\ e^{i\phi}\sin\frac{\theta}{2} & e^{i(\phi+\lambda)}\cos\frac{\theta}{2} \end{pmatrix}$ structure. The conjugate of ζ_m is given by $\overline{\zeta_m} = e^{-2\pi i/m} = \zeta_m^{-1}$. Moreover, the inverse of the unitary operator Ω_m is given by $\Omega_m^\dagger = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ \zeta_m^{-1} & -\zeta_m^{-1} \end{pmatrix}$ such that $\Omega_m \Omega_m^\dagger = \Omega_m^\dagger \Omega_m = \mathrm{I}_2$, where I_2 is the 2×2 identity matrix $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. The action of Ω_m on the basis states is given by:

$$|0\rangle \xrightarrow{\Omega_m} \frac{1}{\sqrt{2}} \left(|0\rangle + |1\rangle\right), |1\rangle \xrightarrow{\Omega_m} \frac{1}{\sqrt{2}} \left(\zeta_m |0\rangle - \zeta_m |1\rangle\right).$$

In general, when n-many Ω_m apply on an n-qubit state $|\psi\rangle = \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle$ with $\sum_{\mathbf{x}\in\mathbb{F}_2^n} |\alpha_{\mathbf{x}}|^2 = 1$, the resultant state is given by:

$$\Omega_m^{\otimes n} \left(\sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle \right) = \frac{1}{2^{n/2}} \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} \left(\sum_{\mathbf{y}\in\mathbb{F}_2^n} (-1)^{\mathbf{x}\cdot\mathbf{y}} \zeta_m^{wt(\mathbf{x})} |\mathbf{y}\rangle \right).$$

Remark 5.1.1. *For $m = 1$, the newly defined unitary operator reduces to the Hadamard gate, $\Omega_1 = \mathrm{H}$. Similarly, for $m = 4$, the unitary operator Ω_m corresponds to the nega-Hadamard gate N, as described in [34]. Moreover, for $m = 2$, there exists a real unitary matrix, defined as $\Omega_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$, such that $\Omega_2 (|0\rangle) = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)$ and $\Omega_2 (|1\rangle) = \frac{1}{\sqrt{2}} (-|0\rangle + |1\rangle)$.*

In a similar manner, we define another single-qubit unitary by conjugating the elements of Ω_m, denoted as $\overline{\Omega}_m$, which is given by $\overline{\Omega}_m = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & \overline{\zeta_m} \\ 1 & -\overline{\zeta_m} \end{pmatrix}$. The inverse of $\overline{\Omega}_m$ is given by $\overline{\Omega}_m^\dagger = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ \zeta_m & -\zeta_m \end{pmatrix}$, such that $\overline{\Omega}_m \overline{\Omega}_m^\dagger = \overline{\Omega}_m^\dagger \overline{\Omega}_m = \mathrm{I}_2$. The action of n-many $\overline{\Omega}_m$ on an n-qubit state $|\psi\rangle = \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle$ is given by

$$\overline{\Omega}_m^{\otimes n} \left(\sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle \right) = \frac{1}{2^{n/2}} \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} \left(\sum_{\mathbf{y}\in\mathbb{F}_2^n} (-1)^{\mathbf{x}\cdot\mathbf{y}} (\overline{\zeta_m})^{wt(\mathbf{x})} |\mathbf{y}\rangle \right).$$

Furthermore, for $m = 4$, the newly defined $\overline{\Omega}_m$ coincides with the conjugate nega-Hadamard gate, $\overline{\mathrm{N}}$, introduced in [34].

Additionally, we define another single-qubit gate, S_m, which generalizes the phase gate (S), given by $\mathrm{S}_m = \begin{pmatrix} 1 & 0 \\ 0 & \zeta_m \end{pmatrix}$. The inverse of S_m is given by $\mathrm{S}_m^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & \zeta_m^{-1} \end{pmatrix}$, and so $\mathrm{S}_m \mathrm{S}_m^\dagger = \mathrm{S}_m^\dagger \mathrm{S}_m = \mathrm{I}$. Notably, for $m = 1$, S_m reduces to the identity gate. When $m = 2$, we have $\zeta_2 = -1$, making S_2 equivalent to the Pauli Z gate. Similarly, for $m = 4$, S_4 corresponds to the standard phase gate S, and for $m = 8$, S_8 is equivalent to the well-known T gate. The existing special cases of this newly defined unitary S_m are summarized as follows:

$$\mathrm{S}_1 = \mathrm{I}, \quad \mathrm{S}_2 = \mathrm{Z}, \quad \mathrm{S}_4 = \mathrm{S}, \quad \mathrm{S}_8 = \mathrm{T}.$$

More generally, when n instances of S_m are applied to an n-qubit state $|\psi\rangle = \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle$, the resultant state acquires a multiplicative phase factor of $\zeta_m^{wt(\mathbf{x})}$, that is,

$$\mathrm{S}_m^{\otimes n} \left(\sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle \right) = \sum_{\mathbf{x}\in\mathbb{F}_2^n} \alpha_{\mathbf{x}} \zeta_m^{wt(\mathbf{x})} |\mathbf{x}\rangle.$$

Next, we generalize 2^k-Hadamard transform, 2^k-crosscorrelation, 2^k-autocorrelation and the related results for any $m \in \mathbb{N}$, establishing the existing results as specific cases within this generalized framework. We begin with extending the definition of 2^k-Hadamard transform [89] to define the most generalized version of the Hadamard transform, and connect it with the newly defined unitary operator Ω_m.

Definition 5.1.1 (m-Hadamard transform)**.** *Given $f \in \mathcal{B}_n$, the m-Hadamard transform ($m \in \mathbb{N}$) of f at $\boldsymbol{\omega} \in \mathbb{F}_2^n$ is a complex valued function, $\mathcal{H}_f^{(m)} : \mathbb{F}_2^n \to \mathbb{C}$ defined as*

$$\mathcal{H}_f^{(m)}(\boldsymbol{\omega}) = \frac{1}{2^{n/2}} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \boldsymbol{\omega}} \zeta_m^{wt(\mathbf{x})}.$$

Remark 5.1.2. *For $m = 1$, the m-Hadamard transform reduces to the standard Walsh-Hadamard transform, i.e., $\mathcal{H}_f^{(1)} = W_f$. Similarly, for $m = 4$, it corresponds to the nega-Hadamard transform, $\mathcal{H}_f^{(4)} = N_f$. More generally, for $m = 2^k$, the transform is expressed as $\mathcal{H}_f^{(m)} = \mathcal{H}_f^{(2^k)}$, representing the 2^k-Hadamard transform.*

The multi-set $\{\mathcal{H}_f^{(m)}(\boldsymbol{\omega}) : \boldsymbol{\omega} \in \mathbb{F}_2^n\}$ is called the *m–Hadamard spectrum* of $f \in \mathcal{B}_n$. Since the m-Hadamard transform is a complex valued function, we can define the conjugate m-Hadamard transform as follows:

$$\overline{\mathcal{H}_f^{(m)}(\boldsymbol{\omega})} = \frac{1}{2^{n/2}} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \boldsymbol{\omega}} \left(\overline{\zeta_m}\right)^{wt(\mathbf{x})}.$$

In this regard, the constraint $\sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} |\mathcal{H}_f^{(m)}(\boldsymbol{\omega})|^2 = \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \mathcal{H}_f^{(m)}(\boldsymbol{\omega}) \overline{\mathcal{H}_f^{(m)}(\boldsymbol{\omega})} = 2^n$ is known as the m-Parseval's identity. The proof follows from Corollary 5.1.1.

Similar to the bent-negabent classifications, a Boolean function $f \in \mathcal{B}_n$ is called *m-bent* if its m-Hadamard transform is flat in complex modulus, i.e., $\left|\mathcal{H}_f^{(m)}(\boldsymbol{\omega})\right| = 1$, for all $\boldsymbol{\omega} \in \mathbb{F}_2^n$. Furthermore, any Boolean function $f \in \mathcal{B}_n$ can be represented as a sum of its m-Hadamard transforms, as follows.

Lemma 5.1.1. *Let $f \in \mathcal{B}_n$. Then the m-Hadamard transform is invertible, i.e.,*

$$(-1)^{f(\mathbf{y})} = 2^{-n/2} \zeta_m^{-wt(\mathbf{y})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \mathcal{H}_f^{(m)}(\boldsymbol{\omega}) (-1)^{\boldsymbol{\omega} \cdot \mathbf{y}}.$$

Proof. Let us begin with the right hand side (RHS)

$$\begin{aligned}
2^{-n/2} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \mathcal{H}_f^{(m)}(\boldsymbol{\omega}) (-1)^{\boldsymbol{\omega} \cdot \mathbf{y}} &= 2^{-n} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \boldsymbol{\omega}} \zeta_m^{wt(\mathbf{x})} (-1)^{\boldsymbol{\omega} \cdot \mathbf{y}} \\
&= 2^{-n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x})} \zeta_m^{wt(\mathbf{x})} \sum_{\boldsymbol{\omega} \in \mathbb{F}_2^n} (-1)^{\boldsymbol{\omega} \cdot (\mathbf{x} \oplus \mathbf{y})} \\
&= 2^{-n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x})} \zeta_m^{wt(\mathbf{x})} 2^n \delta_0(\mathbf{x} \oplus \mathbf{y}) = (-1)^{f(\mathbf{y})} \zeta_m^{wt(\mathbf{y})},
\end{aligned}$$

and the claim is shown. □

Next, we compute the m-Hadamard transform for various combinations of Boolean functions (as in [89, 91]).

Lemma 5.1.2. *Let $f, g, h \in \mathcal{B}_n$ and $\zeta_m = e^{2\pi i/m}$. Then, the following holds.*

(a) If $g(\mathbf{x}) = f(\mathbf{x}) \oplus \mathbf{c} \cdot \mathbf{x} \oplus d$ with $\mathbf{c} \in \mathbb{F}_2^n$, $d \in \mathbb{F}_2$, then $\mathcal{H}_g^{(m)}(\mathbf{u}) = (-1)^d \mathcal{H}_f^{(m)}(\mathbf{u} \oplus \mathbf{c})$. Moreover, if $\mathbb{L}(\mathbf{x}) = \mathbf{c} \cdot \mathbf{x} \oplus d$, then

$$\mathcal{H}_{\mathbb{L}}^{(m)}(\mathbf{u}) = (-1)^d 2^{n/2} \left(\cos(\pi/m)\right)^n \left(-i \tan(\pi/m)\right)^{wt(\mathbf{u} \oplus \mathbf{c})} \zeta_m^{n/2}.$$

(b) If $f(\mathbf{x}) = g(\mathbf{x}) \oplus h(\mathbf{x})$, then

$$\mathcal{H}_f^{(m)}(\mathbf{u}) = \frac{1}{2^{n/2}} \sum_{\mathbf{v}} \mathcal{H}_g^{(m)}(\mathbf{v}) W_h(\mathbf{u} \oplus \mathbf{v}) = \frac{1}{2^{n/2}} \sum_{\mathbf{v}} W_g(\mathbf{v}) \mathcal{H}_h^{(m)}(\mathbf{u} \oplus \mathbf{v}).$$

(c) If $g(\mathbf{x}) = f(A\mathbf{x})$, where A is an $n \times n$ orthogonal matrix over $\mathbb{F}_2$, then

$$\mathcal{H}_g^{(m)}(\mathbf{u}) = \mathcal{H}_f^{(m)}(A\mathbf{u}).$$

(d) If $h(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) \oplus g(\mathbf{y})$ with $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$, then $\mathcal{H}_h^{(m)}(\mathbf{u}, \mathbf{v}) = \mathcal{H}_f^{(m)}(\mathbf{u}) \mathcal{H}_g^{(m)}(\mathbf{v})$.

Proof. The proof follows from [89, Theorem 2], with some necessary tricks and tweaks.

(a) If $g(\mathbf{x}) = f(\mathbf{x}) \oplus \mathbf{c} \cdot \mathbf{x} \oplus d$, from definition

$$\mathcal{H}_g^{(m)}(\mathbf{u}) = \frac{1}{2^{n/2}} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus (\mathbf{u} \oplus \mathbf{c}) \cdot \mathbf{x} \oplus d} \zeta_m^{wt(\mathbf{x})} = (-1)^d \mathcal{H}_f^{(m)}(\mathbf{u} \oplus \mathbf{c}).$$

Moreover,

$$\begin{aligned} 1 + \zeta_m &= 1 + \cos(2\pi/m) + i \sin(2\pi/m) = 2 \cos(\pi/m) e^{i\pi/m}, \\ 1 - \zeta_m &= 1 - \cos(2\pi/m) - i \sin(2\pi/m) = -2i \sin(\pi/m) e^{i\pi/m}. \end{aligned}$$

For $\mathbf{u} = u_1, \ldots u_n$ and $\mathbf{c} = c_1, \ldots c_n$, set $b_k = u_k \oplus c_k$. Then,

$$\begin{aligned} \mathcal{H}_{\mathbb{L}}^{(m)}(\mathbf{u}) =& \frac{(-1)^d}{2^{n/2}} \sum_{\mathbf{x} \in \mathbb{F}_2^n} (-1)^{\mathbf{x} \cdot (\mathbf{u} \oplus \mathbf{c})} \zeta_m^{wt(\mathbf{x})} \\ =& \frac{(-1)^d}{2^{n/2}} \prod_{k=1}^{n} \left(1 + \zeta_m (-1)^{b_k}\right) = \frac{(-1)^d}{2^{n/2}} \prod_{b_k=0} (1 + \zeta_m) \prod_{b_k=1} (1 - \zeta_m) \\ =& \frac{(-1)^d}{2^{n/2}} \left(2 \cos(\pi/m) e^{i\pi/m}\right)^{n - wt(\mathbf{u} \oplus \mathbf{c})} \left(-2i \sin(\pi/m) e^{i\pi/m}\right)^{wt(\mathbf{u} \oplus \mathbf{c})} \\ =& (-1)^d 2^{n/2} \left(\cos(\pi/m)\right)^n \left(-i \tan(\pi/m)\right)^{wt(\mathbf{u} \oplus \mathbf{c})} \left(e^{i\pi n/m}\right). \end{aligned}$$

(b) We prove the first identity by evaluating the RHS; the second follows similarly.

$$\begin{aligned}\frac{1}{2^{n/2}}\sum_{\mathbf{v}}\mathcal{H}_g^{(m)}(\mathbf{v})W_h(\mathbf{u}\oplus\mathbf{v}) =& \frac{1}{2^{3n/2}}\sum_{\mathbf{v},\mathbf{x},\mathbf{y}}(-1)^{g(\mathbf{x})\oplus h(\mathbf{y})\oplus\mathbf{x}\cdot\mathbf{v}\oplus\mathbf{y}\cdot(\mathbf{u}\oplus\mathbf{v})}\zeta_m^{wt(\mathbf{x})}\\ =& \frac{1}{2^{3n/2}}\sum_{\mathbf{x},\mathbf{y}}(-1)^{g(\mathbf{x})\oplus h(\mathbf{y})\oplus\mathbf{u}\cdot\mathbf{y}}\zeta_m^{wt(\mathbf{x})}\sum_{\mathbf{v}}(-1)^{\mathbf{v}\cdot(\mathbf{x}\oplus\mathbf{y})}\\ =& \frac{1}{2^{n/2}}\sum_{\mathbf{x}}(-1)^{(g\oplus h)(\mathbf{x})\oplus\mathbf{u}\cdot\mathbf{x}}\zeta_m^{wt(\mathbf{x})} = \mathcal{H}_{g\oplus h}^{(m)}(\mathbf{u}) = \mathcal{H}_f^{(m)}(\mathbf{u}).\end{aligned}$$

(c) If $g(\mathbf{x}) = f(A\mathbf{x})$, from definition, $\mathcal{H}_g^{(m)}(\mathbf{u}) = \frac{1}{2^{n/2}}\sum_{\mathbf{x}}(-1)^{f(A\mathbf{x})\oplus\mathbf{x}\cdot\mathbf{u}}\zeta_m^{wt(\mathbf{x})}$. Suppose, $A\mathbf{x} = \mathbf{y}$, then $\mathbf{x} = A^T\mathbf{y}$. Thus,

$$\begin{aligned}\mathcal{H}_g^{(m)}(\mathbf{u}) =& \frac{1}{2^{n/2}}\sum_{\mathbf{y}}(-1)^{f(\mathbf{y})\oplus(A^T\mathbf{y})\cdot\mathbf{u}}\zeta_m^{wt(A^T\mathbf{y})}\\ =& \frac{1}{2^{n/2}}\sum_{\mathbf{y}}(-1)^{f(\mathbf{y})\oplus\mathbf{y}\cdot A\mathbf{u}}\zeta_m^{wt(\mathbf{y})} = \mathcal{H}_f^{(m)}(A\mathbf{u}).\end{aligned}$$

(d) Starting with the RHS, we obtain

$$\begin{aligned}\mathcal{H}_f^{(m)}(\mathbf{u})\mathcal{H}_g^{(m)}(\mathbf{v}) =& \frac{1}{2^n}\sum_{\mathbf{x},\mathbf{y}}(-1)^{f(\mathbf{x})\oplus g(\mathbf{y})\oplus\mathbf{x}\cdot\mathbf{u}\oplus\mathbf{y}\cdot\mathbf{v}}\zeta_m^{wt(\mathbf{x})+wt(\mathbf{y})}\\ =& \frac{1}{2^n}\sum_{\mathbf{x},\mathbf{y}}(-1)^{h(\mathbf{x},\mathbf{y})\oplus(\mathbf{x},\mathbf{y})\cdot(\mathbf{u},\mathbf{v})}\zeta_m^{wt(\mathbf{x},\mathbf{y})} = \mathcal{H}_h^{(m)}(\mathbf{u},\mathbf{v}).\end{aligned}$$

The claims are shown. □

We now define the most generalized form of crosscorrelation and autocorrelation using the m-th root of unity, ζ_m, as follows.

Definition 5.1.2 (*m-crosscorrelation*)**.** *Given $f, g \in \mathcal{B}_n$, the m-crosscorrelation of f and g at a point $\mathbf{y} \in \mathbb{F}_2^n$ is described as*

$$C_{f,g}^{(m)}(\mathbf{y}) = \sum_{\mathbf{x}\in\mathbb{F}_2^n}(-1)^{f(\mathbf{x})\oplus g(\mathbf{x}\oplus\mathbf{y})}\left(\zeta_m^2\right)^{\mathbf{x}\odot\mathbf{y}}.$$

Taking $f = g$ in the above formulation, we obtain the m-autocorrelation, defined as

$$C_f^{(m)}(\mathbf{y}) = \sum_{\mathbf{x}}(-1)^{f(\mathbf{x})\oplus f(\mathbf{x}\oplus\mathbf{y})}\left(\zeta_m^2\right)^{\mathbf{x}\odot\mathbf{y}},$$

where $\mathbf{x}\odot\mathbf{y}$ denotes the inner product in $\mathbb{C}\times\mathbb{C}$.

Remark 5.1.3. *For $m = 1$, the m-crosscorrelation (m-autocorrelation) reduces to the standard crosscorrelation (autocorrelation). Additionally, for $m = 4$, where $\zeta_m = i$, the m-crosscorrelation and m-autocorrelation correspond to the nega-crosscorrelation and the nega-autocorrelation, respectively.*

The m-crosscorrelation of two Boolean functions is related with the product of their m-Hadamard transforms, as follows.

Theorem 5.1.1. *Let $f, g \in \mathcal{B}_n$. Then $C_{f,g}^{(m)}(\mathbf{z}) = \zeta_m^{wt(\mathbf{z})} \sum_{\mathbf{u}\in\mathbb{F}_2^n} \mathcal{H}_f^{(m)}(\mathbf{u})\overline{\mathcal{H}_g^{(m)}(\mathbf{u})}(-1)^{\mathbf{u}\cdot\mathbf{z}}$.*

Proof. Let us begin with the RHS:

$$\begin{aligned}
&\zeta_m^{wt(\mathbf{z})} \sum_{\mathbf{u}\in\mathbb{F}_2^n} \mathcal{H}_f^{(m)}(\mathbf{u})\overline{\mathcal{H}_g^{(m)}(\mathbf{u})}(-1)^{\mathbf{u}\cdot\mathbf{z}} \\
=&\frac{1}{2^n}\zeta_m^{wt(\mathbf{z})} \sum_{\mathbf{u}} \sum_{\mathbf{x}} (-1)^{f(\mathbf{x})\oplus\mathbf{x}\cdot\mathbf{u}} \zeta_m^{wt(\mathbf{x})} \sum_{\mathbf{y}} (-1)^{g(\mathbf{y})\oplus\mathbf{y}\cdot\mathbf{u}} \zeta_m^{-wt(\mathbf{y})} (-1)^{\mathbf{u}\cdot\mathbf{z}} \\
=&\frac{1}{2^n} \sum_{\mathbf{x},\mathbf{y}} (-1)^{f(\mathbf{x})\oplus g(\mathbf{y})} \zeta_m^{wt(\mathbf{x})+wt(\mathbf{z})-wt(\mathbf{y})} \sum_{\mathbf{u}} (-1)^{\mathbf{x}\cdot\mathbf{u}\oplus\mathbf{y}\cdot\mathbf{u}\oplus\mathbf{u}\cdot\mathbf{z}}.
\end{aligned}$$

Now taking $\mathbf{y} = \mathbf{x} \oplus \mathbf{z}$ in the above expression, we obtain

$$\begin{aligned}
&\frac{1}{2^n} \sum_{\mathbf{x}} (-1)^{f(\mathbf{x})\oplus g(\mathbf{x}\oplus\mathbf{z})} \zeta_m^{wt(\mathbf{x})+wt(\mathbf{z})-wt(\mathbf{x}\oplus\mathbf{z})} \sum_{\mathbf{u}} (-1)^{\mathbf{u}\cdot\mathbf{y}\oplus\mathbf{u}\cdot(\mathbf{x}\oplus\mathbf{z})} \\
=&\sum_{\mathbf{x}} (-1)^{f(\mathbf{x})\oplus g(\mathbf{x}\oplus\mathbf{z})} \left(\zeta_m^2\right)^{\mathbf{x}\odot\mathbf{z}} = C_{f,g}^{(m)}(\mathbf{z}).
\end{aligned}$$

□

Corollary 5.1.1. *Let $f \in \mathcal{B}_n$. Then $C_f^{(m)}(\mathbf{z}) = \zeta_m^{wt(\mathbf{z})} \sum_{\mathbf{u}\in\mathbb{F}_2^n} |\mathcal{H}_f^{(m)}(\mathbf{u})|^2 (-1)^{\mathbf{u}\cdot\mathbf{z}}$.*

Assuming $\mathbf{z} = 0^n$ in the above corollary, we obtain $\sum_{\mathbf{u}\in\mathbb{F}_2^n} |\mathcal{H}_f^{(m)}(\mathbf{u})|^2 = C_f^{(m)}(0^n) = 2^n$, establishing the m-Parseval's identity. Moreover, the m-bent criterion of a Boolean function is related to its m-autocorrelation as follows.

Theorem 5.1.2. *A Boolean function $f \in \mathcal{B}_n$ is m-bent if and only if $C_f^{(m)}(\mathbf{z}) = 0$ for all $\mathbf{z} \in \mathbb{F}_2^n \setminus \{0^n\}$, and $C_f^{(m)}(\mathbf{z}) = 2^n$, for $\mathbf{z} = 0^n$.*

Proof. If $f \in \mathcal{B}_n$ is m-bent, i.e., $|\mathcal{H}_f^{(m)}(\mathbf{u})| = 1$ for all $\mathbf{u} \in \mathbb{F}_2^n$, then $C_f^{(m)}(\mathbf{z}) = \zeta_m^{wt(\mathbf{z})} \sum_{\mathbf{u}\in\mathbb{F}_2^n} (-1)^{\mathbf{u}\cdot\mathbf{z}}$. Now, $\sum_{\mathbf{u}\in\mathbb{F}_2^n} (-1)^{\mathbf{u}\cdot\mathbf{z}} = 0$ for all $\mathbf{z} \in \mathbb{F}_2^n \setminus \{0^n\}$, and 2^n, if $\mathbf{z} = 0^n$. Similarly, the converse follows. □

Next, we present the most generalized version of a Deutsch-Jozsa-like quantum algorithm, building upon the existing variations as shown in [27, 40].

5.1.1 Generalized Deutsch-Jozsa algorithm

Suppose, we are given the oracle access to an unknown Boolean function $f \in \mathcal{B}_n$. Similar to the standard Deutsch-Jozsa algorithm, we start with an $(n+1)$-qubit quantum circuit initialized to $|0^n\rangle |-\rangle$. We then apply n-many Hadamard gates to the first n qubits, creating an equal superposition of all possible n-bit strings, $2^{-n/2} \sum_{\mathbf{x} \in \mathbb{F}_2^n} |\mathbf{x}\rangle$, which is fed into the quantum oracle U_f, thereby achieving quantum parallelism.

After the oracle query, instead of only applying the Hadamard gates to the first n qubits, as in the standard Deutsch-Jozsa algorithm, we apply a sequence of omega-gates determined by a sequence of natural numbers $\mathbf{d} = (d_1, \ldots, d_n)$ where $d_i \in \mathbb{N}$ dictates the application of Ω_{d_i} gate at the i-th qubit. Finally, we measure the first n qubits in the $\{|0\rangle, |1\rangle\}$ basis, and the final state before measurement is given by

$$\frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n,\, d_i \in \mathbb{N}} \sum_{\mathbf{y} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \mathbf{y}} \zeta_{d_i}^{wt(\mathbf{x})} |\mathbf{y}\rangle .$$

Therefore, the probability of observing any particular state $\mathbf{y} \in \mathbb{F}_2^n$ becomes

$$\frac{1}{2^{2n}} \left| \sum_{\mathbf{x} \in \mathbb{F}_2^n,\, d_i \in \mathbb{N}} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \mathbf{y}} \zeta_{d_i}^{wt(\mathbf{x})} \right|^2 .$$

A schematic diagram of the generalized Deutsch-Jozsa algorithm is presented in Figure 5.1, where $U_i = \Omega_{d_i}$ and $d_i \in \mathbb{N}$. If $d_i = m$ for all $1 \leq i \leq n$ where m is a fixed natural number, then we denote the generalized Deutsch-Jozsa algorithm by DJ_m.

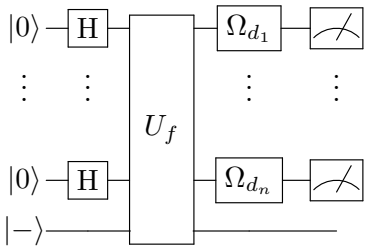


Figure 5.1: Quantum circuit for the most generalized Deutsch-Jozsa algorithm.

Remark 5.1.4. *Depending on the exact sequence of natural numbers,* $\mathbf{d}$*, we have the following observations.*

1. *If* $d_i = 1$ *for all* $1 \leq i \leq n$*, i.e.,* $\Omega_{d_i} = \mathrm{H}$ *for all* $1 \leq i \leq n$*, then the algorithm* DJ_1 *corresponds to the standard Deutsch-Jozsa algorithm [27], where the final state before measurement becomes* $2^{-n} \sum_{\mathbf{x},\mathbf{y} \in \mathbb{F}_2^n} (-1)^{f(\mathbf{x}) \oplus \mathbf{x} \cdot \mathbf{y}} |\mathbf{y}\rangle = 2^{-n/2} \sum_{\mathbf{y}} W_f(\mathbf{y}) |\mathbf{y}\rangle$*, which is the normalized Walsh-Hadamard transform of* f*. As a result, the probability of observing any particular state* $\mathbf{y} \in \mathbb{F}_2^n$ *becomes* $2^{-n} |W_f(\mathbf{y})|^2$*.*

2. *When $d_i \in \{1,4\}$, i.e., $\Omega_{d_i} \in \{\mathrm{H},\mathrm{N}\}$ for all $1 \leq i \leq n$, then depending upon the choice of $\mathbf{d} \in \{1,4\}^n$, a combination of the Hadamard and the nega-Hadamard gates is applied, resulting in the extended Deutsch-Jozsa algorithm [40]. Recall that in the original extended Deutsch-Jozsa algorithm, the choice between Hadamard and nega-Hadamard gates was determined by a binary sequence $\mathbf{c} = (c_j)_{1\leq j\leq n} \in \mathbb{F}_2^n$, where $c_j = 0$ corresponds to the Hadamard gate and $c_j = 1$ corresponds to the nega-Hadamard gate. Similarly, in this context, $d_j = 1$ implies $c_j = 0$, while $d_j = 4$ corresponds to $c_i = 1$. Additionally, fixing $d_j = 4$ for all $1 \leq j \leq n$ (algorithm DJ_4) the final state before measurement becomes $2^{-n}\sum_{\mathbf{x},\mathbf{y}\in\mathbb{F}_2^n}(-1)^{f(\mathbf{x})\oplus\mathbf{x}\cdot\mathbf{y}}i^{wt(\mathbf{x})}\left|\mathbf{y}\right\rangle = 2^{-n/2}\sum_{\mathbf{y}\in\mathbb{F}_2^n}N_f(\mathbf{y})\left|\mathbf{y}\right\rangle$, which is the normalized nega-Hadamard transform of f. Consequently, the probability of observing any particular state $\mathbf{y} \in \mathbb{F}_2^n$ is given by $2^{-n}\left|N_f(\mathbf{y})\right|^2$.*

3. *Similarly, if $d_j = 2^k$ for all $1 \leq j \leq n$ and some fixed $k \in \mathbb{N}$ (corresponds to DJ_{2^k}), then the final state before measurement becomes the normalized 2^k-Hadamard transform, and the corresponding probability of observing any particular state $\mathbf{y} \in \mathbb{F}_2^n$ becomes $2^{-n}\left|\mathcal{H}_f^{(2^k)}\right|^2$.*

4. *Finally, in algorithm DJ_m, where $d_j = m$ for all $1 \leq j \leq n$ and some fixed $m \in \mathbb{N}$, the final pre-measurement state becomes the normalized m-Hadamard transform, and the probability of observing any particular state $\mathbf{y} \in \mathbb{F}_2^n$ is given by $2^{-n}\left|\mathcal{H}_f^{(m)}\right|^2$.*

5.1.2 Generalized Forrelation

We now present the most generalized form of Forrelation, called m-Forrelation, which encompasses both the standard (3-fold) Forrelation [2] and the nega-Forrelation [34] as specific instances.

Definition 5.1.3 (m-Forrelation)**.** *Given oracle access to $f_1, f_2, f_3 \in \mathcal{B}_n$, the (3-fold) m-Forrelation is a measure of correlation between the Boolean function f_1, the m-Hadamard transform of f_2 and the conjugate m-Hadamard transform of f_3, mathematically defined as*

$$\Phi_{f_1,f_2,f_3}^{(m)} = \frac{1}{2^n}\sum_{\mathbf{x_1}\in\mathbb{F}_2^n}(-1)^{f_1(\mathbf{x_1})}\mathcal{H}_{f_2}^{(m)}(\mathbf{x_1})\overline{\mathcal{H}_{f_3}^{(m)}(\mathbf{x_1})},$$

which can be further decomposed to

$$\frac{1}{2^{2n}}\sum_{\mathbf{x_1},\mathbf{x_2},\mathbf{x_3}\in\mathbb{F}_2^n}(-1)^{f_1(\mathbf{x_1})}\left((-1)^{f_2(\mathbf{x_2})\oplus\mathbf{x_1}\cdot\mathbf{x_2}}\zeta_m^{wt(\mathbf{x_2})}\right)\left((-1)^{f_3(\mathbf{x_3})\oplus\mathbf{x_1}\cdot\mathbf{x_3}}(\overline{\zeta_m})^{wt(\mathbf{x_3})}\right).$$

Remark 5.1.5. *From the definition of m-Forrelation, we have the following remarks:*

1. *The (3-fold) m-Forrelation, $\Phi^{(m)}_{f_1,f_2,f_3}$ is complex-valued and not symmetric, meaning its values depend on the order of the underlying Boolean functions.*

2. *When $f_2 = f_3$, the product of the m-Hadamard transform and its conjugate equals the complex square of the m-Hadamard transform of f. Consequently, $\Phi^{(m)}_{f_1,f_2,f_2}$ is always a real number.*

3. *For $m = 1$, $\Phi^{(m)}_{f_1,f_2,f_3} = \Phi_{f_1,f_2,f_3}$, the standard (3-fold) Forrelation as provided in [2]. For $m = 4$, $\Phi^{(m)}_{f_1,f_2,f_3} = \eta_{f_1,f_2,f_3}$, the 3-fold nega-Forrelation as introduced in [34].*

Additionally, note that, similar to the Forrelation, the m-Forrelation formulation can be extended to accommodate k many Boolean functions, $f_1, \ldots, f_k \in \mathcal{B}_n$, referred to as k-fold m-Forrelation. However, in this paper, we focus mainly on the 3-fold variation and simply use the term m-Forrelation to refer to the 3-fold m-Forrelation. Following the approach in [2, 34], we present two quantum algorithms for estimating the m-Forrelation values, one utilizing three sequential queries and another using two parallel queries.

We begin with the 3-query quantum algorithm (see Figure 5.2). Given oracle access to the Boolean functions $f_1, f_2, f_3 \in \mathcal{B}_n$, we begin with the state $|0^n\rangle|-\rangle$ and traverse through the following sequence of steps,

$$\mathrm{H}^{\otimes n} \to U_{f_2} \to \Omega_m^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n} \to U_{f_3} \to \overline{\Omega_m}^{\otimes n}.$$

Ignoring the last qubit, the pre-measurement amplitude corresponding to $|0^n\rangle$ becomes

$$\frac{1}{2^{2n}} \sum_{\mathbf{x_1},\mathbf{x_2},\mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_2(\mathbf{x_2})} \zeta_m^{wt(\mathbf{x_2})} (-1)^{\mathbf{x_1}\cdot\mathbf{x_2}} (-1)^{f_1(\mathbf{x_1})} (-1)^{\mathbf{x_1}\cdot\mathbf{x_3}} (\overline{\zeta_m})^{wt(\mathbf{x_3})} (-1)^{f_3(\mathbf{x_3})},$$

which is equal to $\Phi^{(m)}_{f_1,f_2,f_3}$. Since, $\Phi^{(m)}_{f_1,f_2,f_3}$ is a complex number, the probability of observing the all-zero state upon measurement is given by the complex modulus square, $\left|\Phi^{(m)}_{f_1,f_2,f_3}\right|^2$. Let us denote the 3-query m-Forrelation algorithm by $A_n^{(m)3,3}$. Figure 5.2 provides a schematic diagram of the quantum circuit for $A_n^{(m)3,3}(f_1, f_2, f_3)$.

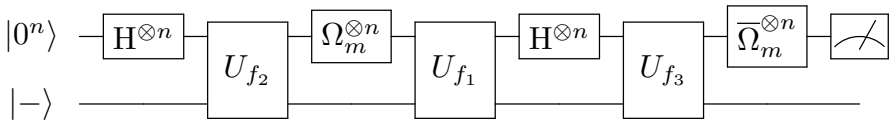


Figure 5.2: Quantum circuit for (3-fold) m-Forrelation using 3 sequential queries.

Next, we present the 2-query quantum algorithm for estimating the m-Forrelation (Figure 5.3). Given oracle access to $f_1, f_2, f_3 \in \mathcal{B}_n$, we begin with an $(n+2)$-qubit state, $|+\rangle|0^n\rangle|-\rangle$, where the first qubit is termed as the 'driving qubit' and the next n many as

the query-qubits. In the beginning, n many Hadamard gates are applied to the n-query qubits, and distribute the state as follows:

$$|+\rangle |0^n\rangle |-\rangle \xrightarrow{\mathrm{H}^{\otimes n}} \frac{1}{\sqrt{2^{n+1}}} \left(\sum_{\mathbf{x_2} \in \mathbb{F}_2^n} |0\rangle |\mathbf{x_2}\rangle + \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} |1\rangle |\mathbf{x_3}\rangle \right) |-\rangle .$$

Then controlled on the driving qubit being $|0\rangle$, we sequentially apply $U_{f_2} \to \Omega_m^{\otimes n} \to U_{f_1} \to \mathrm{H}^{\otimes n}$ and obtain

$$\frac{|0\rangle}{\sqrt{2^{3n+1}}} \sum_{\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_2(\mathbf{x_2})} \zeta_m^{wt(\mathbf{x_2})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_2}} (-1)^{f_1(\mathbf{x_1})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_3}} |\mathbf{x_3}\rangle |-\rangle .$$

Similarly, controlled on the driving qubit being $|1\rangle$, we sequentially apply: $\mathrm{S}_m^{\otimes n} \to U_{f_3}$ and obtain

$$\frac{|1\rangle}{\sqrt{2^{n+1}}} \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_3(\mathbf{x_3})} \zeta_m^{wt(\mathbf{x_3})} |\mathbf{x_3}\rangle |-\rangle .$$

Combining both the scenario, we obtain $|\psi\rangle = \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (\alpha_{\mathbf{x_3}} |0\rangle + \beta_{\mathbf{x_3}} |1\rangle) |\mathbf{x_3}\rangle |-\rangle$, where

$$\alpha_{\mathbf{x_3}} = \left(\frac{1}{\sqrt{2^{3n+1}}} \sum_{\mathbf{x_1}, \mathbf{x_2} \in \mathbb{F}_2^n} (-1)^{f_2(\mathbf{x_2})} \zeta_m^{wt(\mathbf{x_2})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_2}} (-1)^{f_1(\mathbf{x_1})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_3}} \right) \text{ and}$$

$$\beta_{\mathbf{x_3}} = \frac{1}{\sqrt{2^{n+1}}} (-1)^{f_3(\mathbf{x_3})} \zeta_m^{wt(\mathbf{x_3})}.$$

Next, we apply a Hadamard gate to the 'driving qubit', and obtain

$$\frac{1}{\sqrt{2}} \left(\sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (\alpha_{\mathbf{x_3}} + \beta_{\mathbf{x_3}}) |0\rangle + \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} (\alpha_{\mathbf{x_3}} - \beta_{\mathbf{x_3}}) |1\rangle \right) |\mathbf{x_3}\rangle |-\rangle .$$

Finally, we measure the 'driving qubit' in the computational basis. The probability of observing $|0\rangle$ is given by

$$\frac{1}{2} \sum_{\mathbf{x_3} \in \mathbb{F}_2^n} |\alpha_{\mathbf{x_3}} + \beta_{\mathbf{x_3}}|^2 = \frac{1}{2} \left[\sum_{\mathbf{x_3} \in \mathbb{F}_2^n} \left(|\alpha_{\mathbf{x_3}}|^2 + |\beta_{\mathbf{x_3}}|^2 \right) + 2\Re\left(\alpha_{\mathbf{x_3}} \overline{\beta}_{\mathbf{x_3}} \right) \right],$$

where $\Re(z)$ denotes the real part of the complex number z. Note that the expression $\sum_{\mathbf{x_3} \in \mathbb{F}_2^n} |\alpha_{\mathbf{x_3}}|^2 + |\beta_{\mathbf{x_3}}|^2$ represents the sum of the squared amplitudes of the quantum state $|\psi\rangle$, which is equal to 1. Moreover, $\sum_{\mathbf{x_3} \in \mathbb{F}_2^n} 2\Re\left(\alpha_{\mathbf{x_3}} \overline{\beta}_{\mathbf{x_3}} \right) = 2\Re\left(\sum_{\mathbf{x_3} \in \mathbb{F}_2^n} \alpha_{\mathbf{x_3}} \overline{\beta}_{\mathbf{x_3}} \right)$. Expanding $\alpha_{\mathbf{x_3}}$ and $\overline{\beta}_{\mathbf{x_3}}$, we obtain:

$$\Re\left(\frac{1}{2^{2n}} \sum_{\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3} \in \mathbb{F}_2^n} (-1)^{f_2(\mathbf{x_2})} \zeta_m^{wt(\mathbf{x_2})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_2}} (-1)^{f_1(\mathbf{x_1})} (-1)^{\mathbf{x_1} \cdot \mathbf{x_3}} (\overline{\zeta_m})^{wt(\mathbf{x_3})} (-1)^{f_3(\mathbf{x_3})} \right),$$

which is nothing but $\Re\left(\Phi^{(m)}_{f_1,f_2,f_3}\right)$. Therefore, the probability of observing $|0\rangle$ upon measuring the driving qubit is given by $\frac{1}{2}\left(1+\Re\left(\Phi^{(m)}_{f_1,f_2,f_3}\right)\right)$. Let us denote the 2-query m-Forrelation algorithm as $A_n^{(m)2,3}$. In $A_n^{(m)2,3}$, since U_{f_3} is applied in parallel with U_{f_2} and U_{f_1}, the effective number of query remains 2, and hence called the 2-query algorithm. Figure 5.3 provides a schematic diagram of the quantum circuit for $A_n^{(m)2,3}(f_1,f_2,f_3)$.

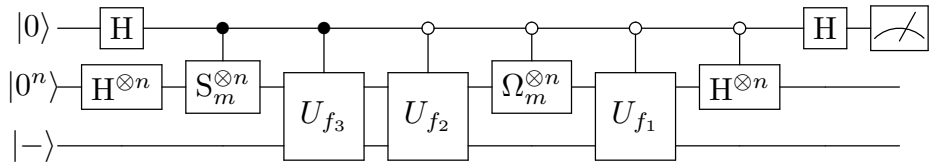


Figure 5.3: Quantum circuit for (3-fold) m-Forrelation using 2 parallel queries.

Remark 5.1.6. *In the 3-query 3-fold m-Forrelation circuit (Figure 5.2), the second and the fourth layers employ the Ω_m and $\overline{\Omega}_m$ gates, respectively, in place of the nega-Hadamard and conjugate nega-Hadamard gates used in nega-Forrelation, or the Hadamard layers used in standard Forrelation algorithms.*

Similarly, in the 2-query 3-fold m-Forrelation circuit (Figure 5.3), a series of Ω_m gates is applied between the oracles for f_1 and f_2, replacing the nega-Hadamard gates found in nega-Forrelation. These modifications are essential to ensure that the measurement probabilities reflect the m-Forrelation spectra.

Additionally, we identify the limitation of our 2-query m-Forrelation algorithm, as follows.

Remark 5.1.7. *Since the m-Forrelation value, $\Phi^{(m)}_{f_1,f_2,f_3}$ is a complex number for $m>2$, using the 2-query algorithm ($A_n^{(m)2,3}$), we can estimate only the real part of $\Phi^{(m)}_{f_1,f_2,f_3}$ and not the complete m-Forrelation value. Furthermore, since $\Phi^{(m)}_{f_1,f_2,f_3}$ is real when $f_2=f_3$, the probability of observing $|0\rangle$ upon measuring the driving qubit after executing $A_n^{(m)2,3}(f_1,f_2,f_2)$ is given by $\frac{1}{2}\left(1+\Phi^{(m)}_{f_1,f_2,f_2}\right)$. Consequently, the probability of observing $|1\rangle$ is $\frac{1}{2}\left(1-\Phi^{(m)}_{f_1,f_2,f_2}\right)$.*

In the next section, we present different strategies for sampling the m-Hadamard spectrum and m-crosscorrelation (consequently m-autocorrelation) values using the m-Forrelation algorithms, $A_n^{(m)3,3}$ and $A_n^{(m)2,3}$.

5.2 Sampling of generalized spectra using generalized Forrelation

Given $f \in \mathcal{B}_n$, and a set of points $S \subseteq \mathbb{F}_2^n$, the goal is to estimate the m-Hadamard transform values of f at all the points in S. Recall that using the generalized Deutsch-Jozsa algorithm DJ_m, one can sample the m-Hadamard transform of f at $S \subseteq \mathbb{F}_2^n$ with probability $\frac{1}{2^n}\sum_{\mathbf{x}\in S}\left|\mathcal{H}_f^{(m)}(\mathbf{x})\right|^2 =: p$ (say), where $0 \leq p \leq 1$. We now present the strategies for sampling the m-Hadamard transforms of f using the m-Forrelation algorithms, $A_n^{(m)3,3}$ and $A_n^{(m)2,3}$.

Suppose, $g \in \mathcal{B}_n$ such that $g(\mathbf{x}) = 1$ for all $\mathbf{x} \in S$, and $g(\mathbf{x}) = 0$ otherwise. From the definition of $\Phi^{(m)}_{f_1,f_2,f_3}$, with $f_2 = f_3 = f$ and $f_1 = g$, we have

$$\Phi^{(m)}_{g,f,f} = \frac{1}{2^n}\sum_{\mathbf{x}\in\mathbb{F}_2^n}(-1)^{g(\mathbf{x})}\left|\mathcal{H}_f^{(m)}(\mathbf{x})\right|^2 = \frac{1}{2^n}\left(\sum_{\mathbf{x}\notin S}\left|\mathcal{H}_f^{(m)}(\mathbf{x})\right|^2 - \sum_{\mathbf{x}\in S}\left|\mathcal{H}_f^{(m)}(\mathbf{x})\right|^2\right).$$

Using the m-Parseval's identity, $\sum_{\mathbf{x}\in\mathbb{F}_2^n}\left|\mathcal{H}_f^{(m)}(\mathbf{x})\right|^2 = 2^n$ we obtain,

$$\Phi^{(m)}_{g,f,f} = \frac{1}{2^n}\left(2^n - \sum_{\mathbf{x}\in S}\left|\mathcal{H}_f^{(m)}(\mathbf{x})\right|^2 - \sum_{\mathbf{x}\in S}\left|\mathcal{H}_f^{(m)}(\mathbf{x})\right|^2\right) = 1 - \frac{2}{2^n}\left(\sum_{\mathbf{x}\in S}\left|\mathcal{H}_f^{(m)}(\mathbf{x})\right|^2\right) = 1 - 2p.$$

This implies, $p = \frac{1}{2}\left(1 - \Phi^{(m)}_{g,f,f}\right)$, which is same as the probability of observing $|1\rangle$ upon measuring the 'driving qubit' from running the algorithm $A_n^{(m)2,3}(g, f, f)$.

Proposition 5.2.1. *Given $f, g \in \mathcal{B}_n$ and a set of points $S \subseteq \mathbb{F}_2^n$ such that $g(\mathbf{x}) = 1$ for all $\mathbf{x} \in S$ and $g(\mathbf{x}) = 0$ otherwise, the probability of observing $|1\rangle$ upon measuring the driving qubit, from executing $A_n^{(m)2,3}(g, f, f)$ is given by p.*

Therefore, the 2-query m-Forrelation algorithm $A_n^{(m)2,3}$ samples the m-Hadamard transform with a probability exactly equal to the sampling probability of Algorithm DJ_m.

Next, we compute the sampling probability of the m-Hadamard transform using the 3-query algorithm, $A_n^{(m)3,3}$. Upon measurement, the probability of observing the all-zero state is given by $\left(\Phi^{(m)}_{g,f,f}\right)^2$. Consequently, the probability of observing a state with at least one $|1\rangle$ in the output is $1 - \left(\Phi^{(m)}_{g,f,f}\right)^2$. Substituting $\Phi^{(m)}_{g,f,f} = 1 - 2p$, we obtain:

$$1 - \left(\Phi^{(m)}_{g,f,f}\right)^2 = 1 - (1-2p)^2 = 4p - 4p^2 \approx 4p.$$

Therefore, we have the following theorem.

Theorem 5.2.1. *Given $f, g \in \mathcal{B}_n$ and a set of points $S \subseteq \mathbb{F}_2^n$ such that $g(\mathbf{x}) = 1$ for all $\mathbf{x} \in S$ and $g(\mathbf{x}) = 0$ otherwise, the probability of one of the states in the measurement outcomes being $|1\rangle$ from executing the 3-query quantum algorithm $A_n^{(m)3,3}(g, f, f)$ is given by $4p - 4p^2$.*

From Theorem 5.2.1, it is evident that when $p < 0.75$ (i.e., when $p < 4p - 4p^2$), the sampling probability obtained from $A_n^{(m)3,3}$ surpasses that of algorithm DJ_m, in terms of the required number of queries. More specifically, for any $f \in \mathcal{B}_n$, the algorithm $A_n^{(m)3,3}$ samples the smaller values of the m-Hadamard transform more efficiently than the algorithm DJ_m. Conversely, for $p > 0.75$, algorithm DJ_m alone is sufficient for estimating the m-Hadamard transform values.

Here, one might argue that the algorithm DJ_m requires only a single query to the oracle U_f, whereas the algorithm $A_n^{(m)3,3}$ necessitates two queries to U_f. However, even after executing DJ_m twice (equivalent to two queries to U_f), the resulting sampling probability, $1-(1-p)^2 = 2p-p^2 \approx 2p$, remains lower than that achieved by $A_n^{(m)3,3}$. Furthermore, the sampling probability obtained from $A_n^{(m)3,3}$ also exceeds that of performing DJ_m once, followed by a single round of amplitude amplification, which likewise requires two queries to U_f, and the corresponding sampling probability is given by $\sin\left(3\sin^{-1} p\right) \approx 3p$. For a graphical comparison of the sampling probabilities across different approaches, refer to Figure 5.4, which naturally resembles [34, Figure 3] and [35, Figure 4].

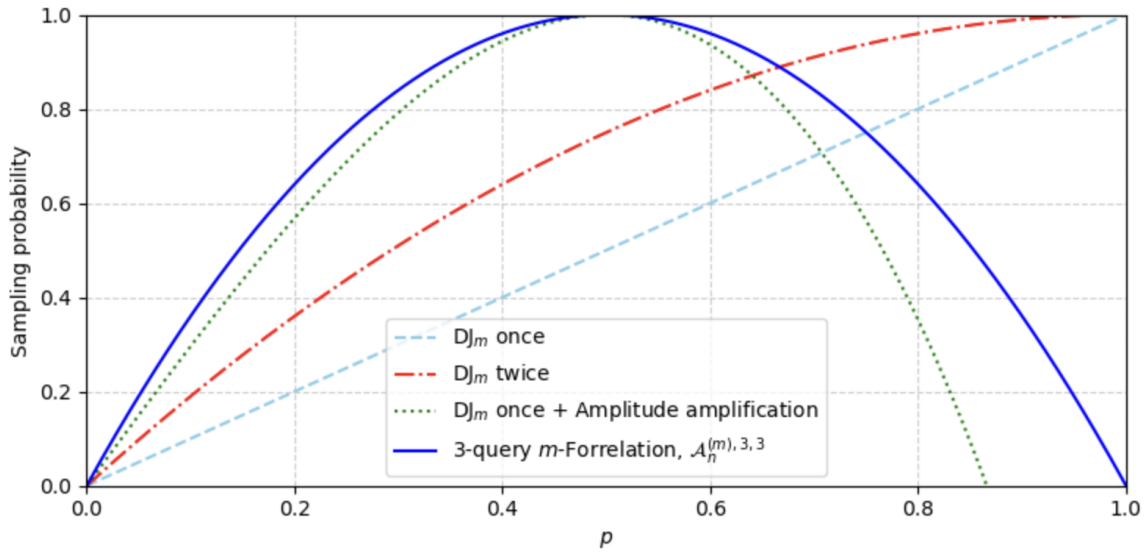


Figure 5.4: Sampling probabilities of m-Hadamard transform using different algorithms.

Next, we provide an efficient sampling of the m-crosscorrelation and m-autocorrelation, using the m-Forrelation algorithms, $A_n^{(m)3,3}$ and $A_n^{(m)2,3}$.

Theorem 5.2.2. *Given oracle access to $f, g \in \mathcal{B}_n$, the 3-query m-Forrelation algorithm, $A_n^{(m)3,3}$, estimates the m-crosscorrelation value of f and g at any given point $\mathbf{y} \in \mathbb{F}_2^n$ by evaluating the probability of measuring the all-zero state, which is given by*

$$\mathcal{P}(|0^n\rangle) = \frac{\left|C_{f,g}^{(m)}(\mathbf{y})\right|^2}{2^{2n}}.$$

Further, the 2-query m-Forrelation algorithm, $A_n^{(m)2,3}$, estimates the real part of the m-crosscorrelation value of f and g at any given point $\mathbf{y} \in \mathbb{F}_2^n$ by evaluating the probability of measuring the 'driving qubit' in the $|0\rangle$ state, which is given by

$$\mathcal{P}(|0\rangle) = \frac{1}{2}\left(1 + \Re\left(2^{-n}\zeta_m^{-wt(\mathbf{y})}C_{f,g}^{(m)}(\mathbf{y})\right)\right).$$

Proof. Suppose, $h(\mathbf{x}) \in \mathcal{B}_n$ such that $h(\mathbf{x}) = \mathbf{x} \cdot \mathbf{y}$. Then, from Theorem 5.1.1, the m-crosscorrelation value at $\mathbf{y} \in \mathbb{F}_2^n$ can be written as

$$C_{f,g}^{(m)}(\mathbf{y}) = \zeta_m^{wt(\mathbf{y})} \sum_{\mathbf{x}\in\mathbb{F}_2^n} \mathcal{H}_f^{(m)}(\mathbf{x})\overline{\mathcal{H}_g^{(m)}(\mathbf{x})}(-1)^{h(\mathbf{x})} = \zeta_m^{wt(\mathbf{y})} 2^n \cdot \Phi_{h,f,g}^{(m)}.$$

This implies $\Phi_{h,f,g}^{(m)} = 2^{-n}\zeta_m^{-wt(\mathbf{y})}C_{f,g}^{(m)}(\mathbf{y})$. Since the m-Forrelation values $\Phi_{h,f,g}^{(m)}$ can be estimated from the algorithms $A_n^{(m)3,3}$ and $A_n^{(m)2,3}$, the rest of the proof follows. □

This gives us a constant query algorithm for sampling the m-crosscorrelation values of any two Boolean functions, $f, g \in \mathcal{B}_n$ at any given point, $\mathbf{y} \in \mathbb{F}_2^n$. For $g = f$, we obtain a constant query sampling of m-autocorrelation, $C_f^{(m)}$ as an immediate corollary.

Corollary 5.2.1. *Given oracle access to $f \in \mathcal{B}_n$, the 3-query m-Forrelation algorithm, $A_n^{(m)3,3}$, estimates the m-autocorrelation value of f at any given point $\mathbf{y} \in \mathbb{F}_2^n$ by evaluating the probability of measuring the all-zero state, $|0^n\rangle$ which is given by*

$$\mathcal{P}(|0^n\rangle) = \frac{|C_f^{(m)}(\mathbf{y})|^2}{2^{2n}}.$$

Furthermore, the 2-query m-Forrelation algorithm, $A_n^{(m)2,3}$, estimates the real part of the m-autocorrelation value of f at any given point $\mathbf{y} \in \mathbb{F}_2^n$ by evaluating the probability of measuring the 'driving qubit' in the $|0\rangle$ state, which is given by

$$\mathcal{P}(|0\rangle) = \frac{1}{2}\left[1 + \Re\left(2^{-n}\zeta_m^{-wt(\mathbf{y})}C_f^{(m)}(\mathbf{y})\right)\right].$$

Moreover, following Theorem 5.1.2, we have another direct corollary as follows.

Corollary 5.2.2. *Let $f \in \mathcal{B}_n$ be an m-bent function, and let $h \in \mathcal{B}_n$ be a linear Boolean function (i.e., either constant or balanced). Then the presence or absence of the all-zero state in the measurement outcome of the 3-query algorithm $A_n^{(m)2,3}(h, f, f)$ determines whether h is constant, i.e., $h(\mathbf{x}) = 0$, for all $\mathbf{x} \in \mathbb{F}_2^n$, or balanced, respectively.*

Proof. From Corollary 5.2.1, $\mathcal{P}(|0^n\rangle) = 2^{-2n}|C_f^{(m)}(\mathbf{y})|^2$. Since, $f \in \mathcal{B}_n$ be an m-bent, from Theorem 5.1.2, $\mathcal{P}(|0^n\rangle) \neq 0$ if $\mathbf{y} = 0^n$, i.e., $h(\mathbf{x}) = \mathbf{x} \cdot \mathbf{y} = 0$, and $\mathcal{P}(|0^n\rangle) = 0$ if $\mathbf{y} \neq 0^n$, i.e., $h(\mathbf{x}) = \mathbf{x} \cdot \mathbf{y}$, a balanced Boolean function. □

Theorem 5.2.2 and Corollary 5.2.1 estimate the m-crosscorrelation and consequently the m-autocorrelation values, at any specified point $\mathbf{y} \in \mathbb{F}_2^n$, by choosing the linear function $h(\mathbf{x}) = \mathbf{x} \cdot \mathbf{y}$, which is dependent upon $\mathbf{y}$. Next, we attempt to sample from the complete spectrum of the m-crosscorrelation (and thus the m-autocorrelation) by placing a superposition of all possible linear Boolean functions, in place of h (shown in Figure 5.5).

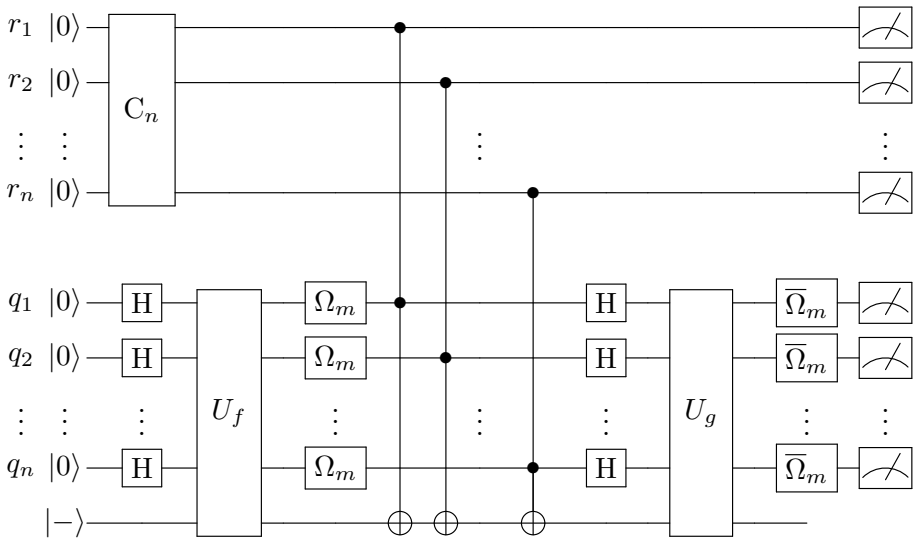


Figure 5.5: Quantum circuit for sampling the complete m-crosscorrelation spectrum.

Let C_n be a $2^n \times 2^n$ unitary operator such that $\mathrm{C}_n(|0^n\rangle) = \sum_{\mathbf{x} \in \mathbb{F}_2^n} \alpha_{\mathbf{x}} |\mathbf{x}\rangle$ with $\alpha_{\mathbf{x}} \in \mathbb{C}$, for all $\mathbf{x} \in \mathbb{F}_2^n$ satisfying $\sum_{\mathbf{x} \in \mathbb{F}_2^n} |\alpha_{\mathbf{x}}|^2 = 1$. Then, starting from the all-zero state, the algorithm $\mathbb{A}^{(m)}(\mathrm{C}_n)$ as in Figure 5.5 has the pre-measurement state

$$\sum_{\mathbf{y} \in \mathbb{F}_2^n} \alpha_{\mathbf{y}} |\mathbf{y}\rangle \left(\frac{C_{f,g}^{(m)}(\mathbf{y})}{2^n} |0^n\rangle + \beta_{\mathbf{y}} |W_{\mathbf{y}}\rangle \right),$$

where $|W_{\mathbf{y}}\rangle$ is an n-qubit superposition state such that the amplitude of the state $|0^n\rangle$ is 0. Theorem 5.2.3 shows that the specific choice of C_n helps in sampling the complete spectrum of m-crosscorrelation, as follows.

Theorem 5.2.3. *Fixing $\mathrm{C}_n = \mathrm{H}^{\otimes n}$, the probability of observing $|\mathbf{y}\rangle |0^n\rangle$ upon measuring the first $2n$ qubits from Algorithm $\mathbb{A}^{(m)}$ is given by $2^{-3n} \left|C_{f,g}^{(m)}(\mathbf{y})\right|^2$ for all $\mathbf{y} \in \mathbb{F}_2^n$.*

Proof. Fixing $\mathrm{C}_n = \mathrm{H}^{\otimes n}$, the pre-measurement state becomes

$$\sum_{\mathbf{y}\in\mathbb{F}_2^n} \frac{1}{2^{n/2}} |\mathbf{y}\rangle \left(\frac{C_{f,g}^{(m)}(\mathbf{y})}{2^n} |0^n\rangle + \beta_{\mathbf{y}} |W_{\mathbf{y}}\rangle \right).$$

Thus the probability of observing the state $|\mathbf{y}\rangle |0^n\rangle$ upon measuring the first $2n$ many qubits is given by $\frac{\left|C_{f,g}^{(m)}(\mathbf{y})\right|^2}{2^{3n}}$. □

The algorithm $\mathbb{A}^{(m)}$ is a generalization, where $m = 1$ corresponds to sampling the complete spectrum of crosscorrelation, as presented in [35, Algorithm 1]. For $m = 4$, it enables the sampling of the complete spectrum of nega-crosscorrelation, as described in [34, Algorithm 1]. Furthermore, for $m = 2^k$, the algorithm can also be used for sampling the complete spectrum of 2^k-crosscorrelation. In this context, we present the following corollary.

Corollary 5.2.3. *Let $f \in \mathcal{B}_n$ be a m-bent function. Then, from executing the algorithm $\mathbb{A}^{(m)}(\mathrm{H}^{\otimes n})$ with $g = f$, the probability of observing the all-zero state, $|0^n\rangle |0^n\rangle$, upon measuring all the $2n$ qubits, is given by 2^{-n}. Moreover, the probability of observing a state $|\mathbf{x}\rangle |0^n\rangle$, where $\mathbf{x} \neq 0^n$, is 0.*

The proof follows from Corollary 5.2.2, using the fact that the m-autocorrelation value of an m-bent function is 0 at any nonzero point.

Let UD_k^n be the $2^n \times 2^n$ unitary operator that prepares the Dicke-state $|D_k^n\rangle$ of weight k such that

$$\mathrm{UD}_k^n |0^n\rangle = \frac{1}{\sqrt{\binom{n}{k}}} \sum_{\mathbf{x}:wt(\mathbf{x})=k} |\mathbf{x}\rangle .$$

Consequently, if we replace $\mathrm{C}_n = \mathrm{UD}_k^n$ with $k < n$, the probability of observing $|\mathbf{y}\rangle |0^n\rangle$ is given by $\binom{n}{k}^{-1} 2^{-2n} \left|C_{f,g}^{(m)}(\mathbf{y})\right|^2$, where the Hamming weight of $\mathbf{y}$ is k and the probability of observing $|\mathbf{y}\rangle |0^n\rangle$ is zero if $wt(\mathbf{y}) \neq k$. In this manner, one can sample the m-crosscorrelation (and hence the m-autocorrelation) values at all the points having an equal Hamming weight k.

5.3 On affine transformation of (generalized) bent functions

In this section, we study the affine transformations of (generalized) bent functions, beginning with an interesting observation on the affine properties of m-bent ones.

Theorem 5.3.1. *Let $f, g \in \mathcal{B}_n$ be two m-bent function such that $g(\mathbf{x}) = f(A\mathbf{x} \oplus \mathbf{b}) \oplus \mathbf{c} \cdot \mathbf{x} \oplus d$, where A is an n-dimensional orthogonal matrix, $\mathbf{b}, \mathbf{c} \in \mathbb{F}_2^n$, $d \in \mathbb{F}_2$. Then, for $\mathbf{b} = 0^n$, if f is m-bent, so is g.*

Proof. Combining Lemma 5.1.2 (a) and Lemma 5.1.2 (c), we obtain:

$$\mathcal{H}_g^{(m)}(\boldsymbol{\omega}) = (-1)^d 2^{-n/2} \sum_{\mathbf{x}} (-1)^{f(\mathbf{y}) \oplus A(\mathbf{c} \oplus \boldsymbol{\omega}) \cdot \mathbf{y}} \zeta_m^{wt(\mathbf{y})} = (-1)^d \mathcal{H}_f^{(m)}(A(\mathbf{c} \oplus \boldsymbol{\omega})).$$

Therefore, $\left|\mathcal{H}_g^{(m)}(\boldsymbol{\omega})\right| = \left|(-1)^d \mathcal{H}_f^{(m)}(A(\mathbf{c} \oplus \boldsymbol{\omega}))\right| = \left|\mathcal{H}_f^{(m)}(A(\mathbf{c} \oplus \boldsymbol{\omega}))\right|$. Hence, if f is m-bent, then so is g. □

Notably, the similar results are already known for specific subcases such as standard bent functions, negabent functions, and k-bent functions. Consider two Boolean functions $f, g \in \mathcal{B}_n$ related by the affine transformation $g(\mathbf{x}) = f(A\mathbf{x} \oplus \mathbf{b}) \oplus \mathbf{c} \cdot \mathbf{x} \oplus d$, where A is an $n \times n$ orthogonal matrix over $\mathbb{F}_2$ (i.e., $AA^T = A^T A = I_n$), $\mathbf{b}, \mathbf{c} \in \mathbb{F}_2^n$, and $d \in \mathbb{F}_2$. Then, the Walsh-Hadamard transforms of f and g satisfy

$$W_g(\boldsymbol{\omega}) = (-1)^{d \oplus A(\boldsymbol{\omega} \oplus \mathbf{c}) \cdot b} W_f(A(\boldsymbol{\omega} \oplus \mathbf{c})),$$

which implies that if g is bent, then f is also bent [82, Theorem 2]. Similarly, for the nega-Hadamard transforms:

$$N_g(\boldsymbol{\omega}) = (-1)^{d \oplus A(\mathbf{c} \oplus \boldsymbol{\omega}) \cdot b} (i)^{wt(\mathbf{b})} N_f(A(\mathbf{c} \oplus \boldsymbol{\omega}) \oplus \mathbf{b}),$$

which shows that if g is negabent, then f is also negabent [91, Theorem 3(d)]. A similar argument applies to the 2^k-Hadamard transforms, yielding:

$$\mathcal{H}_g^{(2^k)}(\boldsymbol{\omega}) = (-1)^d \mathcal{H}_f^{(2^k)}(A(\mathbf{c} \oplus \boldsymbol{\omega})).$$

Studying isomorphisms among Boolean functions is an interesting area of research from computational points of view (see [41] and the references therein). In this initiative, while we establish the theoretical associativity, it is not immediate how a quantum algorithm can efficiently recover the hidden parameters of the affine transformations given oracle access to the m-bent functions. Designing such algorithms remains an open and promising research direction for future work.

5.4 Conclusion

In this chapter, we generalized various cryptographically significant spectra of Boolean functions, including the Walsh-Hadamard, crosscorrelation, and autocorrelation spectra, extending previous formulations to any $m \in \mathbb{N}$. We demonstrated that existing variants, such as the standard version, the nega-variant, and the 2^k-variant—are special cases of this more general framework. Additionally, we identified a previously unexamined class of real Hadamard transforms that lies between the Walsh-Hadamard and nega-Hadamard transformations, filling a gap in the existing literature. Furthermore, we introduced the most generalized version of the Deutsch-Jozsa algorithm, which extends both the standard Deutsch-Jozsa and its prior extended version, thereby encompassing them as special cases. We established that the pre-measurement state in the generalized Deutsch-Jozsa algorithm corresponds to a superposition of the normalized m-Hadamard transforms evaluated at all points. In addition, we extended the Forrelation formulation to m-Forrelation, and presented new quantum algorithms for estimating these newly defined generalized spectra. We believe these techniques may be of use to understand the interactions among Boolean functions, various spectra and relevant quantum algorithms.

Chapter 6

Exact space-depth trade-offs in multi-controlled Toffoli decomposition

Quantum gates are the fundamental building blocks of quantum circuits. Unlike classical gates, quantum gates are inherently reversible and are represented by unitary matrices. An n-controlled Toffoli gate, or n-MCT, is a generalization of the standard (2-controlled) Toffoli gate, where n control qubits determine the state of a single target qubit. Multi-controlled Toffoli (MCT) decompositions play a critical role in quantum arithmetic [95, 96, 97], reversible computing [21, 25], and oracle constructions [5, 11, 32, 48]. For instance, in Grover's algorithm [49], the implementation of an n-bit AND function requires an n-MCT gate.

Since MCT gates cannot be implemented natively on current quantum hardware, they must be decomposed into Clifford plus Toffoli gates, and eventually into the Clifford+T gate set. Optimizing such decomposition is essential for reducing computational overhead, minimizing error rates, and improving scalability, making it a key component in practical large-scale quantum computation.

It is well known that decomposing or designing larger circuits with smaller components often requires additional qubits, known as ancilla qubits, to reduce circuit depth. Optimizing the number of gates, circuit depth, and ancillary qubits has long been a central concern in quantum circuit design, beginning with the seminal work of Moore and Nilsson [67]. A more recent and comprehensive discussion is provided in [57], which poses a fundamental question: "Can we characterize the relationship between the number of ancilla and the possible optimal depth?" While their focus is on CNOT circuits, our work addresses similar questions for Toffoli-based constructions.

Over the past decades, numerous efforts have been made to reduce the resource requirements for implementing multi-controlled Toffoli (MCT) gates [6, 8, 22, 63, 66, 72,

79]. More recent approaches [59, 70] introduced the conditionally clean ancilla technique, which significantly reduces both the Toffoli depth and ancilla count in MCT decompositions. This technique has since gained considerable attention, including its application in quantum adders [76].

Here, we focus on optimizing three key resource metrics: Toffoli count, Toffoli depth, and ancilla count. The Toffoli count and Toffoli depth are further refined with the T count and T depth as one may refer to Table 2.1.

In this chapter, we explore how the Toffoli depth (and consequently, T depth) can be reduced by increasing the number of clean ancilla qubits, utilizing the conditionally clean ancilla technique. We also establish the limitation of this method in terms of the lower bound of the Toffoli depth. Additionally, we show that, in a more general setting, the exact Toffoli depth (which can be further reduced to exact T depth) of an n-MCT decomposition cannot be reduced beyond $\lceil \log_2 n \rceil$.

The organization of this chapter and its section-wise contributions are outlined as follows.

- Section 6.1 is a preparatory section where we revisit the recent developments of MCT decompositions describing the existing best results in terms of Toffoli count, Toffoli depth, and ancilla, which can be subsequently decomposed into T count and T depth, as summarized in Table 6.1. Given that there have been several developments in very recent times, this section provides a holistic view of the existing results. Based on this, we present our findings.

- In Section 6.2, we explore the trade-off between the (clean) ancilla and the Toffoli depth using the existing techniques related to conditionally clean ones. In Section 6.2.1, we take a different look at viewing the MCT circuit decomposition using the conditionally clean ancilla technique of [59] and enumerate their exact Toffoli count. Following the trade-off, we show the reduction in Toffoli depth, and therefore in T depth (Construction 6.2.1, Example 6.2.1) compared to the recent work of Khattar and Gidney [59], by introducing additional clean ancillas into the circuit, while keeping the Toffoli count constant. These results are shown in Section 6.2.2. Then, in Section 6.2.3, we identify the limitation of this technique in Theorem 6.3.1, showing that this direction cannot reduce the Toffoli depth to $\lceil \log_2 n \rceil$, though it is of order $\mathcal{O}(\log_2 n)$.

- In Section 6.3, we prove within a general framework that the exact Toffoli depth in the (2-controlled) Toffoli decomposition of an n-MCT gate cannot be less than $\lceil \log_2 n \rceil$, regardless of the number of ancilla qubits used. In fact, this lower bound can be achieved for T depth as well, by the construction of [56]. More specifically, using the technique of [56], we can obtain an n-MCT gate by further decomposing into the Clifford+T gate set with an exact T depth of $\lceil \log_2 n \rceil$ too, using $2n-2$ ancilla, and a T count of $4(n-1)$. Moreover, using the logical-AND circuit by

Gidney [43], the ancilla count can be reduced to $n-2$, while keeping the T count constant. However, the exact T depth in the case of [43] becomes one more, i.e., $\lceil \log_2 n \rceil + 1$. To highlight our contribution, we are looking at the exact counts and depth instead of their complexity order.

- Section 6.4 concludes the paper with a brief summary of our work and outlines the open problems in this direction.

6.1 Preparation: A consolidated view on the recent developments in MCT decomposition

As we have already discussed, the decomposition of complex quantum gates into simpler and more practical quantum gate sets has been a topic of interest since the inception of quantum computing. More specifically, there have been substantial developments in the direction of multi-controlled Pauli (or, more precisely, MCT) decomposition in the last decades. This is a warm-up section where we provide a comprehensive overview of existing benchmarks for multi-controlled Toffoli gate decompositions. It emphasizes the state-of-the-art optimized results, to the best of our knowledge, in terms of T count, T depth, and ancilla requirements, as summarized in Table 6.1.

From Section 2.2, it is known that the multi-controlled Pauli gates (X, Y, Z) can be transformed into one another using a constant number of Clifford gates, such as the Hadamard or the phase gates, as described in Figure 2.3.

$$\mathrm{C}^n\mathrm{X} \equiv \mathrm{C}^n\left(\mathrm{HZH}\right) \equiv \mathrm{C}^n(\mathrm{SYS}^\dagger).$$

As our primary focus here is to minimize the implementation cost of multi-controlled Toffoli (MCT) gates in terms of Toffoli count, Toffoli depth, and the ancilla count, the inclusion of additional Clifford gates does not affect the resource estimation. Therefore, the resource estimation for any of the aforementioned multi-controlled Pauli gates can be directly translated to others without requiring modification.

In 2021, Gidney and Jones [47] presented a construction (Figure 6.1) of a 3-controlled Z gate using 6 T gates having a T depth of 6. Additionally, they proposed that their design can be used for the construction of an n-controlled Pauli gate, with a T count of $4n-6$, using $n-2$ logical-AND gates.

Proposition 6.1.1. *Following the CCCZ circuit of [47], an n-MCT gate can be constructed using $n-3$ (2-controlled) Toffoli, and a single 3-controlled Toffoli (CCCX) gate with $n-2$ ancilla qubits, resulting in a Toffoli depth of $\lceil \log_2 \frac{n}{3} \rceil + 1^*$, where 1^* represents the depth of the CCCX gate. Replacing the Toffoli gates with logical-AND (Figure 2.19) yields a complete T depth of $\lceil \log_2 \frac{n}{3} \rceil + 6$. Additionally, the total Clifford count of the circuit is $9n-16$.*

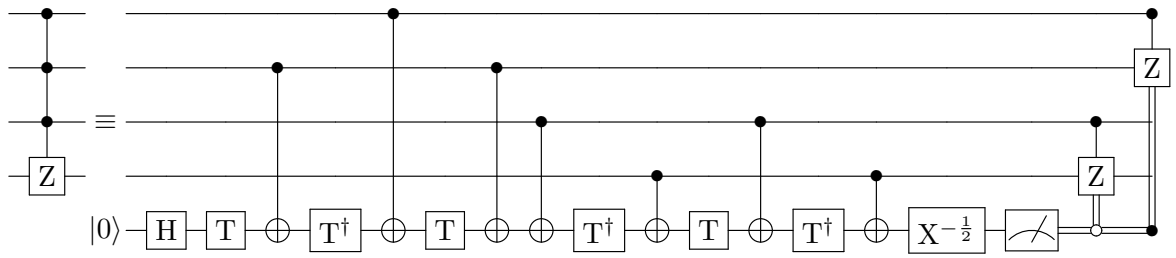


Figure 6.1: Quantum circuit for CCCZ using 6 T gates, with T depth 6 [47].

Proof. As specified in [47], out of the $n-2$ AND gates, the final AND gate and Toffoli gate can be merged to implement a CCCX gate, making the Toffoli count $(n-3)+1^*$, i.e., $n-3$ many (2-controlled) Toffoli and a single 3-controlled Toffoli gate. Consider a tree data structure with the root node having 3-child nodes, and each of them forms a complete binary tree with a total of n leaf nodes. Each of these binary trees has a depth $\lceil \log_2 \frac{n}{3} \rceil$, and the root node, along with its three child nodes, contributes a depth of 1. Consequently, the T depth becomes $\lceil \log_2 \frac{n}{3} \rceil + 6$, where the CCCX gate contributes to the T depth of 6. Since the Clifford count of each logical-AND is 9, and that of the CCCX gate 11, the total Clifford count becomes $9n-16$. □

In Oct 2024, Nakanishi et al. [69] proposed a modified CCCZ circuit (Figure 6.2), reducing the T depth to 2 by utilizing an additional ancilla qubit while keeping a Clifford count of 14. Consequently, the T depth of the n-MCT decomposition is reduced to

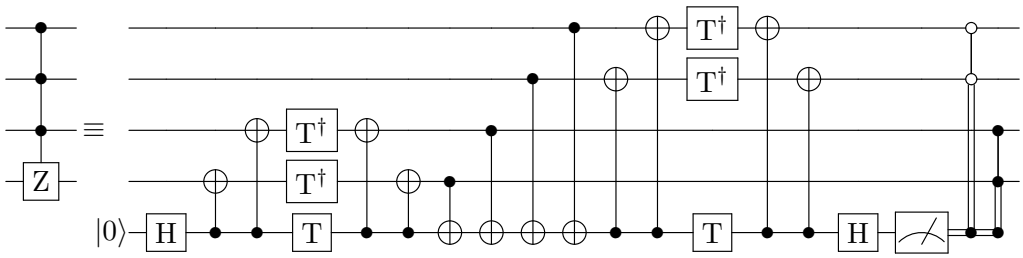


Figure 6.2: Quantum circuit for CCCZ using 6 T gates, with T depth 2 [69].

$\lceil \log_2 \frac{n}{3} \rceil + 2$, with the corresponding Clifford count given by $9n-15$ (see the second row of Table 6.1).

In Feb 2024, Nie et al. [70] first used the notion of conditionally clean ancilla qubits derived from an existing (clean or dirty) ancilla and proposed a novel circuit decomposition (Figure 6.3) for the n-controlled Pauli gates using $\mathcal{O}(n)$ Toffoli gates. The outer layer of their MCT decomposition follows from [42], and the inner layer has been parallelized to obtain an overall Toffoli depth of $\mathcal{O}(\log_2 n)$, compared to the $\mathcal{O}(\log_2 n)$ Toffoli depth in [42]. Additionally, by improving the design of a quantum incrementer, they developed an MCT circuit with a Toffoli count of $\mathcal{O}(n)$, and a Toffoli depth of

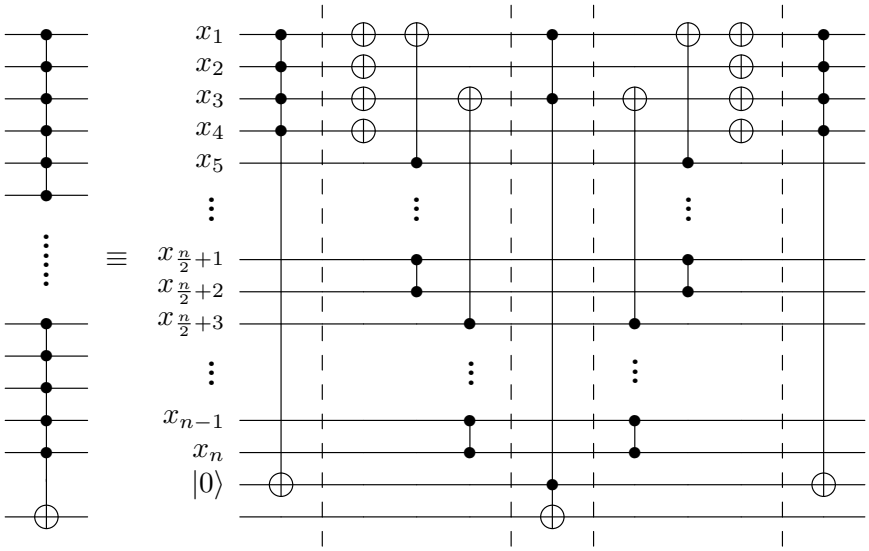


Figure 6.3: MCT decomposition circuit using conditionally clean ancilla due to [70].

$\mathcal{O}(\log^2 n)$ without requiring any additional ancilla qubits. In this design, although the MCT decomposition does not use any additional ancilla, implementing the quantum incrementer requires one ancilla. Since our primary focus here is to reduce Toffoli depth while increasing the clean ancilla count, we ignore the zero-ancilla implementation here.

Proposition 6.1.2. *The MCT circuit decomposition proposed by [70] using a single clean ancilla has a minimum Toffoli count of* $4n + 4$ *and an exact Toffoli depth of* $20 \log_2 n$.

Proof. Step I and Step V implement 4-MCT gates following [42], each requiring $4(4-2) = 8$ Toffoli gates, all applied sequentially, resulting in a Toffoli depth of 16. Similarly, the 3-MCT implemented in Step III has a Toffoli count of 4 and a Toffoli depth of 4. Consequently, Steps I, III, and V together require a total of 20 Toffoli gates, with an overall Toffoli depth of 20, which is referred to in the paper as a Toffoli depth of $\mathcal{O}(1)$.

Additionally, in Step II, an $(n-4)$-MCT is decomposed into two $\left(\frac{n}{2}-2\right)$-MCT, each requiring a minimum of $2\left(\frac{n}{2}-2\right) = n-4$ Toffoli gates. Thus, the Toffoli count for Step II is $2n-8$. Similarly, Step IV also has a Toffoli count of $2n-8$. Therefore, the total Toffoli count for the entire process is given by $2(2n-8)+20 = 4n+4$. Moreover, from [70], we have $D(n) = D(n/2) + \mathcal{O}(n)$, and we estimated $\mathcal{O}(1) = 20$, therefore, the exact Toffoli depth of the MCT circuit is $20 \log_2 n$. □

In July 2024, Khattar and Gidney [59] proposed an optimized implementation of multi-controlled Toffoli (MCT) circuits using the conditionally clean ancilla technique,

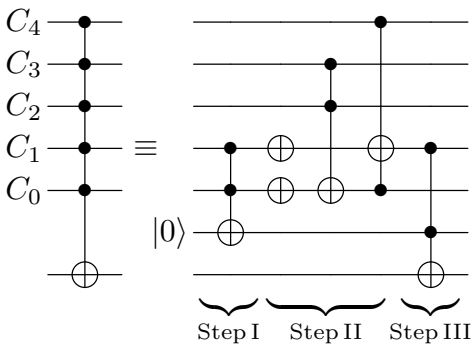


Figure 6.4: Understanding the conditionally clean ancilla technique.

significantly reducing Toffoli depth while maintaining a fixed ancilla count of one or two. Before proceeding, we briefly describe this technique (see Figure 6.4).

The method begins with a clean ancilla qubit, which is used as the target of a Toffoli gate. If the ancilla is flipped, it indicates that both control qubits were in the $|1\rangle$ state. Consequently, applying X gates to the control qubits sets them to $|0\rangle$, allowing them to be reused as conditionally clean ancilla in subsequent rounds. This process is iteratively applied additional Toffoli gates target the new conditionally clean ancilla, and X gates continue to recycle the control qubits for further use.

Once the information from all control qubits is compressed into a few qubits, a final, smaller MCT gate is applied using these compressed controls (and the initial clean ancilla) to control the original target. This ensures that if the ancilla remained unchanged in the first step, the target qubit also remains unaffected.

Let us now analyze Figure 6.4 from [59] by distributing the possible scenarios in two mutually exclusive and exhaustive cases:

Case I: $C_0 = C_1 = C_2 = C_3 = C_4 = |1\rangle$. Then, after Step I, ancilla $= |1\rangle$. Applying X-gate on C_0, C_1 will make, $C_0 = C_1 = |0\rangle$, thus can be used as ancilla for Step II. Since, $C_2 = C_3 = |1\rangle$, the Toffoli gate will make, $C_0 = |1\rangle$. Similarly, $C_0 = |1\rangle$, and $C_4 = |1\rangle$, imply that $C_1 = |1\rangle$. Finally, $C_1 = |1\rangle$, and ancilla $= |1\rangle$, will reverse the target qubit.

Case II: At least one of $C_i = |0\rangle$, for some $i \in [0, 4]$. If one of C_0 or C_1 is $|0\rangle$, then ancilla $=|0\rangle$, and the target will not be flipped in Step III. If $C_0 = C_1 = |1\rangle$, but one of C_2, C_3, C_4 is $|0\rangle$, then the ancilla becomes $|1\rangle$, and after applying X-gate on C_0, C_1 will make, $C_0 = C_1 = |0\rangle$. If one of C_2, C_3 is $|0\rangle$, then C_0 and consequently, C_1 also remains at $|0\rangle$ state. Similarly, if $C_4 = |0\rangle$, then also C_1 remains $|0\rangle$, and in both scenario, the target will not be flipped in Step III.

In [59], the authors first proposed an n-MCT circuit utilizing a single clean ancilla (Figure 6.5), achieving a Toffoli count of $2n - 3$ and a T count of $8n - 12$. Since none of the Toffoli gates are applied simultaneously, the resulting Toffoli depth is $2n - 3$, while

the T depth is $2n-3$, following the T depth 1 Toffoli implementation by [56], requiring one more reusable ancilla.

Furthermore, when the availability of clean ancilla qubit increases to 2 (Figure 6.6), the Toffoli depth reduces to $\mathcal{O}(\log_2 n)$ while maintaining the Toffoli count constant. Additionally, if the clean ancilla is replaced with the dirty ancilla, the Toffoli count increases to $16n-32$, and the Toffoli depth doubles under both scenarios. As we focus here on the conditionally clean ancilla derived solely from clean ancilla qubits, we do not delve into an exact analysis of the Toffoli depth for the dirty ancilla circuit implementation. In Section 6.2.2, we modify the design by introducing additional ancilla qubits in Step I and subsequently creating more conditionally clean ancilla to begin with, which eventually reduces the overall Toffoli depth of the complete circuit.

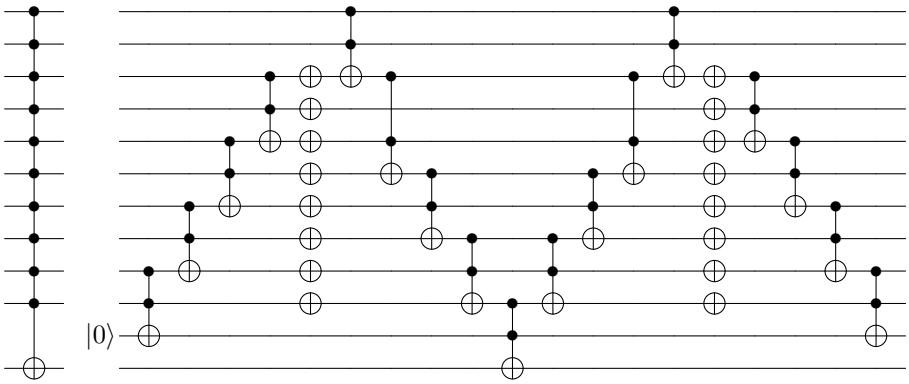


Figure 6.5: Quantum circuit decomposition of 10-MCT, using 17 Toffoli gates and a single clean ancilla, with Toffoli depth 17 [59].

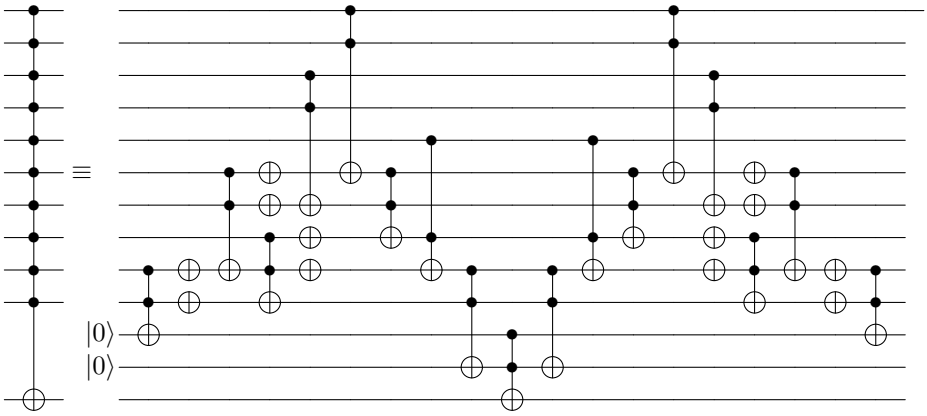


Figure 6.6: Quantum circuit decomposition of 10-MCT, using 17 Toffoli gates and 2 clean ancilla, with Toffoli depth 13 [59].

In this section, we provided the exact enumerations of Toffoli count and Toffoli

References	Ancilla	Toffoli count	Toffoli depth	T count	T depth
Gidney et al. [47]	$n-2$	$(n-3)+1^*$	$\lceil \log_2 \frac{n}{3} \rceil + 1^*$	$4n-6$	$\lceil \log_2 \frac{n}{3} \rceil + 6$
Nakanishi et al. [69]	$n-1$	$(n-3)+1^*$	$\lceil \log_2 \frac{n}{3} \rceil + 1^*$	$4n-6$	$\lceil \log_2 \frac{n}{3} \rceil + 2$
Nie et al. [70]	1	$\geq 4n+4$	$20\log_2 n$	$16n+16$	$20\log_2 n$
Khattar et al. [59]	1	$2n-3$	$2n-3$	$8n-12$	$2n-3$
Khattar et al. [59]	2	$2n-3$	$\approx 4\log_2 n$	$8n-12$	$\approx 4\log_2 n$
This work† [37]	$m_1 (\ll n)+2$	$2n-m_1-3$	δ	$4(2n-m_1-3)$	δ
This work‡ [37]	$n-2$	$n-1$	$\lceil \log_2 n \rceil$	$4(2n-m_1-3)$	$\lceil \log_2 n \rceil$

Table 6.1: Summarizing state-of-the-art results related to multi-controlled Toffoli decompositions.

depth of MCT circuit decomposition across various state-of-the-art results. For the exact enumeration of the Toffoli depth (and T depth), following [59], we propose a novel approach of viewing the MCT decomposition with conditionally clean ancilla, which is a direct extension of [59], thereby presented separately in Proposition 6.2.1 of Section 6.2.1. Later, the approach has also been used to demonstrate the trade-off between Toffoli depth and the clean ancilla count.

The state-of-the-art optimized results corresponding to n-controlled Toffoli decompositions have been summarized in Table 6.1, where the 1^* symbol in the first two rows indicates the presence of a 3-controlled Toffoli gate. The sixth row (marked with †) presents our results on the space–depth trade-off using the conditionally clean ancilla technique. The Toffoli (consequently T) depth δ is defined as

$$\delta = \begin{cases} 2\lfloor \log_2 m_1 \rfloor + 4k, \text{for } n \in \left[\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^{k-1} + k - 1, m_1 2^k + k - 2 \right] \\ 2\lfloor \log_2 m_1 \rfloor + 4k + 2, \text{for } n \in \left[m_1 2^k + k - 1, \left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^k + k - 1 \right] \end{cases}$$

where $k \in \mathbb{N}$, with $k \geq 2$. The result has been proved later in Theorem 6.2.1. Finally, the last row of Table 6.1 (marked with ‡) presents our optimal Toffoli (consequently T) depth MCT decomposition result.

6.2 Further reduction of Toffoli depth

In this section, we adopt the MCT decomposition technique using the conditionally clean ancilla, as introduced in [59, 70], and propose a trade-off between the Toffoli depth with the clean ancilla, showing improvement over the recent work of Khattar and Gidney [59]. The authors proposed the MCT circuit decompositions techniques, achieving a Toffoli depth of $\mathcal{O}(n)$ with a single clean ancilla and $\mathcal{O}(\log_2 n)$ with two clean ancilla, where in both cases, the Toffoli count becomes $2n-3$. Here, we revisit the MCT decompositions by [59] and provide an exact enumeration of the Toffoli depth (consequently T depth) that was not explicitly presented in [59].

6.2.1 Exact enumeration of Toffoli depth proposed by Khattar et al. (Quantum, 2025) in a different lens

In this subsection, we enumerate the exact Toffoli depth of an n-MCT circuit following [59]. The n-MCT decomposition utilizing a single ancilla requires $2n-3$ Toffoli gates, all applied sequentially, resulting in a Toffoli depth of $2n-3$. Subsequently, by employing the measurement-based uncomputation technique for Toffoli decomposition, as illustrated in Figure 2.18, the T depth is again the same, i.e., $2n-3$.

The n-MCT decomposition circuit [59] using two ancilla qubits consists of five steps. Step I and Step V combined have a Toffoli depth of 1 (using logical-AND). In Step II, information from the remaining $n-2$ control qubits is accumulated into $k(\ll n)$ qubits using the conditionally cleaned ancilla technique. Next, in Step III, a $(k+1)$-MCT is implemented using the unused ancilla, following the single-ancilla technique, having a Toffoli depth of $2(k+1)-3 = 2k-1$. In Step IV, the control qubits are returned to their original state through uncomputation, thereby having the same Toffoli count and depth as in Step II.

For an exact enumeration of the Toffoli depth, we introduce a new way of viewing Step I and Step II of the MCT circuit decomposition, as follows.

$$
\begin{array}{llllllll}
2 \rightarrow 1 & & & & & & & \\
\overset{2}{\underset{+1}{\rightarrow}} & \overset{1}{\underset{+1}{\rightarrow}} & 1 & & & & & \\
 & \overset{4}{\underset{+1}{\rightarrow}} & \overset{2}{\underset{+1}{\rightarrow}} & \overset{1}{\underset{+1}{\rightarrow}} & 1 & & & \\
 & & \overset{8}{\underset{+1}{\rightarrow}} & \overset{4}{\underset{+1}{\rightarrow}} & \overset{2}{\underset{+1}{\rightarrow}} & \overset{1}{\underset{+1}{\rightarrow}} & 1 & \\
 & & & \overset{12}{\underset{+1}{\rightarrow}} & \overset{6}{\underset{+1}{\rightarrow}} & \overset{3}{\underset{+1}{\rightarrow}} & 2 \rightarrow & 1
\end{array}
$$

where the first row corresponds to Step I, and the remaining rows correspond to Step II. Each layer of horizontal arrows represents a Toffoli depth of 1, and a row, $a \rightarrow \lceil a/2 \rceil \rightarrow \ldots \rightarrow 1$ implies that the information from a many qubits has been accumulated into a single qubit, with a Toffoli depth of $\lceil \log_2 a \rceil$. Consequently, the first element of each row sums to n. Additionally, the number of layers (k) determines the size of the MCT gate required for Step III.

In the above example, Step I and Step II of a 32-MCT decomposition have been shown using the conditionally cleaned ancilla technique. The Step I has a Toffoli depth of 1, and Step II has a Toffoli depth of 7. Further, in Step III, a 5-MCT needs to be applied to hit the target qubit, having a Toffoli depth $2(5)-3 = 7$. However, a part of the smaller MCT is applied simultaneously with Step II and Step IV, thereby having an effective Toffoli depth of 3 or 5. Clearly, the uncomputation (Step IV) requires an additional Toffoli depth of 8, i.e., the Toffoli depth of the complete circuit is 19.

Using this new approach, the 5-MCT circuit using a conditionally clean ancilla tech-

nique, presented in Figure 6.4, can be seen as follows.

$$\begin{array}{lll} 2 \rightarrow 1 & & \\ \quad {}^{2}_{+1} \rightarrow & {}^{1}_{+1} \rightarrow & 1, \end{array}$$

where the Toffoli depth for Step I and Step II combined is 3, which can also be verified from the actual circuit from Figure 6.4. Moreover, a 2-controlled Toffoli needs to be implemented in Step III to modify the target.

Remark 6.2.1. *The motivation for viewing the MCT decomposition using this new representation because it allows more efficient and accurate estimation of the Toffoli depth, compared to the standard circuit representation. Additionally, our approach provides insight into the number of simultaneous operations and the size of the MCT gate required for implementation in Step III. In the subsequent section, we present our MCT circuit design with additional ancilla qubits using the same representation.*

Further, note that introducing additional rows in Step II, without increasing the overall Toffoli depth will increase the size of the MCT gate required in Step III. Consequently, the overall Toffoli depth of the circuit remains unchanged.

In the direction of enumerating the exact Toffoli depth of an n-MCT decomposition due to [59], using 2 clean ancilla, we have the following lemma.

Lemma 6.2.1. *The exact Toffoli depth, δ for Step II or Step IV of an n-MCT circuit decomposition, as proposed by [59], using two clean ancilla qubits, varies with n as follows.*

$$\delta = \begin{cases} 2k-3, & \text{for } n \in \left[3 \cdot 2^{k-2} + k - 1,\ 2^k + k - 2\right], \\ 2k-2, & \text{for } n \in \left[2^k + k - 1,\ 3 \cdot 2^{k-1} + k - 1\right] \end{cases}$$

where $k \in \mathbb{N}$, with $k \geq 2$.

Lemma 6.2.2. *The number of control qubits, σ, in the MCT gate implemented in Step III of an n-MCT circuit decomposition, as proposed by [59], using two clean ancilla qubits, varies with n as follows.*

$$\sigma = \begin{cases} k & \text{for } n \in \left[3 \cdot 2^{k-2} + k - 1,\ 2^k + k - 1\right], \\ k+1 & \text{for } n \in \left(2^k + k - 1,\ 3 \cdot 2^{k-1} + k - 1\right]. \end{cases}$$

where $k \in \mathbb{N}$, with $k \geq 2$. The corresponding Toffoli depths are $2k-3$, and $2(k+1) - 3 = 2k - 1$, respectively. However, a significant part of the smaller MCT is applied simultaneously with Step II and Step IV, making the effective Toffoli depth of Step III, either 3 or 5.

From the above lemma, we can now estimate the exact Toffoli depth of the n-MCT circuit decomposition using 2 ancilla qubits, as proposed in [59].

Proposition 6.2.1. *Following [59], the exact Toffoli depth of an n-MCT decomposition using 2 clean ancilla is lower bounded by $3\log_2 n$.*

Proof. From the above lemma, the exact Toffoli depth, δ, of the complete n-MCT circuit decomposition is given by

$$\delta = \begin{cases} 1 + 2(2k-3) + 3 = 4k-2 & \text{for } n \in \left[3\cdot 2^{k-2} + k - 1,\, 2^k + k - 1\right], \\ 1 + 2(2k-2) + 3 = 4k & \text{for } n \in \left(2^k + k - 1,\, 3\cdot 2^{k-1} + k - 1\right] \end{cases}$$

where $k \in \mathbb{N}$, with $k \geq 2$. Since here, we are interested in showing the lower bound, we consider n assumes the highest value in the range and still show that $3\log_2 n$ is strictly less than the corresponding depths, as follows.

$$3\log_2(2^k + k - 1) < 3\log_2(2^{k+1}) = 3(k+1),$$

which is less or equal to the depth $4k-2$, for $k \geq 5$.

$$3\log_2(3\cdot 2^{k-1} + k - 1) < 3\log_2(4\cdot 2^{k-1}) = 3(k+1),$$

which is less or equal to the depth $4k$ for $k \geq 3$. □

Although the above proposition establishes that the exact Toffoli depth using two clean ancilla, as per [59], is lower bounded by $3\log_2 n$, this bound is not tight and is closer to $4\log_2 n$. In the following section, we analyze the trade-off between Toffoli depth and clean ancilla, demonstrating that with additional ancilla qubit, the exact Toffoli depth can be reduced to $2\log_2 n$.

6.2.2 Exact trade-off between Toffoli depth and clean ancilla qubits

In this subsection, we explore the ancilla-Toffoli depth trade-off using the concept of conditionally clean ancilla and demonstrate improvements in the Toffoli depth compared to [59] by introducing additional ancilla qubits into the circuit while keeping the Toffoli count constant. Furthermore, for an n-controlled Toffoli gate with m ancilla qubits, we propose an algorithm that determines the Toffoli depth (and consequently the T depth) for the MCT circuit decomposition and presents a graph illustrating the trade-off.

Construction 6.2.1. *Given m ancilla qubits to implement an n-controlled Toffoli gate, we distribute m as $m_1 + m_2$, where m_1 ancilla qubits are allocated for Step I, and m_2 ancilla qubits are reserved for Step III. For $m = 3$, we set $m_1 = 2$ and $m_2 = 1$. Furthermore, for $m \geq 4$, we assume $m_2 = 2$ to implement the smaller MCT in Step III using the 2-clean ancilla technique described in [59].*

The circuit design is as follows.

- *In Step I, the m_1 Toffoli gates are applied to collect information from $2m_1$ control qubits and store them in m_1 ancilla qubits. We implement them using the logical-AND circuit to avoid resource requirements for uncomputation in the final step.*
- *In Step II, information from the remaining control qubits is accumulated into $k(< n)$ qubits using the conditionally cleaned ancilla.*
- *In Step III, a $(k + m_1)$-controlled Toffoli gate is implemented using the remaining m_2 many ancilla, following the 2-ancilla technique, and the target qubit is modified.*
- *In Step IV, uncomputation is performed to return the control qubits to their original state. As earlier, the Toffoli count and Toffoli depth of Step IV are identical to those in Step II.*
- *Finally, in Step V, the first m_1 many Toffoli gates are uncomputed to clean up the ancilla, using logical-AND based uncomputation. This step reduces the overall Toffoli count by m_1 compared to the circuit presented in [59].*

The schematic diagram of the MCT decomposition using $m = m_1 + m_2$ many ancilla is similar to the diagram we used for the exact Toffoli depth enumeration in Section 6.2.1.

$$
\begin{array}{llllllll}
2m_1 \rightarrow m_1 & & & & & & & \\
\quad \overset{2m_1}{+1} \rightarrow & \overset{m_1}{+1} \rightarrow & \ldots \rightarrow & 1 & & & & \\
 & \overset{4m_1}{+1} \rightarrow & \overset{2m_1}{+1} \rightarrow & \overset{m_1}{+1} \rightarrow & \ldots \rightarrow & 1 & & \\
 & & \overset{8m_1}{+1} \rightarrow & \overset{4m_1}{+1} \rightarrow & \overset{2m_1}{+1} \rightarrow & \ldots \rightarrow & 1 & \\
 & & & \vdots \rightarrow & \vdots \rightarrow & \vdots \rightarrow & \vdots \rightarrow & \ldots \rightarrow \quad 1
\end{array}
$$

□

Lemma 6.2.3. *For the proof of correctness, the circuit decomposition presented in Construction 6.2.1 is an n-MCT.*

Proof. In the above MCT decomposition, if any of the first $2m_1$ qubits is in the $|0\rangle$ state, one of the m_1 ancilla qubits will also remain in $|0\rangle$, preventing the target flip in Step III. Similarly, if any of the $n - 2m_1$ qubits is $|0\rangle$, the corresponding row's end qubit remains in $|0\rangle$, ensuring the target is not flipped in Step III. This proves that the designed circuit correctly implements an n-MCT gate. □

Example 6.2.1. *For $m_1 = 3$, the 32-controlled Toffoli can be viewed as:*

$$
\begin{array}{llllll}
6 \rightarrow 3 & & & & & \\
\quad \overset{6}{+1} \rightarrow & \overset{3}{+1} \rightarrow & 2 \rightarrow & 1 & & \\
 & 12 \rightarrow & 6 \rightarrow & 3 \rightarrow & \overset{1}{+1} \rightarrow & 1 \\
 & & \overset{6}{+1} \rightarrow & \overset{3}{+1} \rightarrow & 2 \rightarrow & 1
\end{array}
$$

where the Toffoli depth for Step I is 1*, and the Toffoli depth for Step II (and Step IV) are* 5 *each. Additionally, in Step III, we need to implement a* 6*-controlled Toffoli using* $m_2 = 2$ *ancilla, which requires an additional Toffoli depth of* 3*. Consequently, the Clifford plus Toffoli decomposition of the* 32*-controlled Toffoli gate, utilizing* $(3+2) = 5$ *ancilla qubits, achieves a total Toffoli depth of* $1 + 2(5) + 3 = 14$*. In contrast, the* 32*-controlled Toffoli decomposition with* 2 *ancilla qubits, as presented in [59], results in a Toffoli depth of* 19*.*

Similarly, for $m_1 = 4$*, the* 32*-controlled Toffoli can be viewed as:*

$$
\begin{array}{llllll}
8 \to 4 & & & & & \\
\quad {\scriptstyle 8 \atop +1} \to & {\scriptstyle 4 \atop +1} \to & {\scriptstyle 2 \atop +1} \to & {\scriptstyle 1 \atop +1} \to & 1 & \\
 & {\scriptstyle 14 \atop +1} \to & {\scriptstyle 7 \atop +1} \to & 4 \to & 2 \to & 1
\end{array}
$$

where the Toffoli depth remains 14*. Thus, it can be observed that beyond a certain point, further increases in the number of ancilla qubits do not necessarily lead to a reduction in the Toffoli depth when using conditionally clean ancilla.*

The Toffoli depth of the MCT decomposition using $m = m_1 + m_2$ many ancilla can be estimated from the following lemma.

Lemma 6.2.4. *The exact Toffoli depth of Step II or Step IV of an* n*-MCT circuit decomposition, implemented above, using* $m = m_1 + m_2$ *clean ancilla following the conditionally clean ancilla technique, varies with* n *as follows.*

$$
\delta = \begin{cases} \lfloor \log_2 m_1 \rfloor + 2k - 3, \text{ for } n \in \left[\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^{k-1} + k - 1, m_1 2^k + k - 2 \right] \\ \lfloor \log_2 m_1 \rfloor + 2k - 2, \text{ for } n \in \left[m_1 2^k + k - 1, \left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^k + k - 1 \right] \end{cases}
$$

where $k \in \mathbb{N}$*, with* $k \geq 2$*.*

Lemma 6.2.5. *The number of control qubits,* σ*, in Step III of an* n*-MCT circuit decomposition, implemented above using* $m = m_1 + m_2$ *ancilla qubits, varies with* n *as follows.*

$$
\sigma = \begin{cases} k + m_1 - 1, \text{ for } n \in \left[\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^{k-1} + k - 1, m_1 2^k + k - 1 \right] \\ k + m_1, \text{ for } n \in \left(m_1 2^k + k - 1, \left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^k + k - 1 \right] \end{cases}
$$

where $k \in \mathbb{N}$*, with* $k \geq 2$*. The corresponding Toffoli depths are* $4 \log_2(k + m_1 - 1)$*, and* $4 \log_2(k + m_1)$*, respectively. However, a significant part of the smaller MCT is applied simultaneously with Step II and Step IV, making the effective Toffoli depth of Step III, either* 3 *or* 5*.*

From the above lemma, we can now estimate the exact Toffoli depth of the n-MCT circuit decomposition using $m = m_1 + m_2$ ancilla qubits, as presented above.

Theorem 6.2.1. *The exact Toffoli depth, δ, of the complete n-MCT circuit decomposition using $m = m_1 + m_2$ clean ancilla is given by*

$$\delta = \begin{cases} 2\lfloor \log_2 m_1 \rfloor + 4k, \text{for } n \in \left[\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^{k-1} + k - 1, m_1 2^k + k - 2\right] \\ 2\lfloor \log_2 m_1 \rfloor + 4k + 2, \text{for } n \in \left[m_1 2^k + k - 1, \left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^k + k - 1\right] \end{cases}$$

where $k \in \mathbb{N}$, with $k \geq 2$.

Proof. From Lemma 6.2.4, and 6.2.5, the exact Toffoli depth of an n-MCT, from Step I-IV, for $n \in \left[\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^{k-1} + k - 1, m_1 2^k + k - 2\right]$ is

$$1 + 2\left(\lfloor \log_2 m_1 \rfloor + 2k - 3\right) + 5 = 2\lfloor \log_2 m_1 \rfloor + 4k.$$

For $n \in \left[m_1 2^k + k - 1, \left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^k + k - 1\right]$, the exact Toffoli depth of the n-MCT becomes

$$1 + 2\left(\lfloor \log_2 m_1 \rfloor + 2k - 2\right) + 5 = 2\lfloor \log_2 m_1 \rfloor + 4k + 2.$$

□

It is now understood that the depth of n-MCT can be achieved in $\mathcal{O}(\log_2 n)$, which can be explicitly written $c \log_2 n$, where c needs to be properly estimated, for exact trade-offs which is the main motivation in this paper. In Proposition 6.2.1, we have shown that in the MCT decomposition using two clean ancilla with the technique from [59], c is strictly greater than 3; in fact, it is around 4. However, using the ancilla - Toffoli depth trade-off, c can be further reduced to a certain extent. In the following subsection, we demonstrate that with the conditionally clean ancilla technique, c always remains strictly greater than 1, regardless of the number of ancilla qubits used.

6.2.3 Proving the lower bound on Toffoli depth using conditionally clean ancilla

In this subsection, we show that the Toffoli depth of an n-MCT using the conditionally clean ancilla technique can never be reduced to $\lceil \log_2 n \rceil$.

Theorem 6.2.2. *This is in reference to Construction 6.2.1 for n-MCT.*

1. *Assuming that the control states do not need to be restored to their original values, using the conditionally clean ancilla technique, the exact Toffoli depth must be strictly greater than $\lceil \log_2 n \rceil$, irrespective of the number of available ancilla.*

2. *When the control qubits are required to be returned to their original state upon completion, the Toffoli depth becomes strictly greater than $2\lceil \log_2 n \rceil$.*

Proof. From Proposition 6.2.1, it is evident that the Toffoli depth remains constant over a range of values for n. Since here, we are interested in showing the lower bound, we consider n assumes the highest value in the range and still show that $\log_2 n$ is strictly less than the corresponding depth, as follows.

Let us first proof the item 1. As the control states are not required to be returned to their original state, we consider the Toffoli depth due to Step I, Step II, and half of Step III only, which is

$$\delta' = \begin{cases} \lfloor \log_2 m_1 \rfloor + 2k + 1, \text{if } n \in \left[\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^{k-1} + k - 1, m_1 2^k + k - 2 \right] \\ \lfloor \log_2 m_1 \rfloor + 2k + 2, \text{if } n \in \left[m_1 2^k + k - 1, \left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^k + k - 1 \right] \end{cases}$$

Thus, we need to show the following two cases:

- Case I: $\lceil \log_2 \left(m_1 2^k + k - 2 \right) \rceil \leq \lfloor \log_2 m_1 \rfloor + 2k + 1$,
- Case II: $\lceil \log_2 \left(\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^k + k - 1 \right) \rceil \leq \lfloor \log_2 m_1 \rfloor + 2k + 2$.

Case I: Showing $\lceil \log_2 \left(m_1 2^k + k - 2 \right) \rceil \leq \lfloor \log_2 m_1 \rfloor + 2k + 1$ is equivalent to showing $m_1 2^k + k - 2 \leq 2^{\lfloor \log_2 m_1 \rfloor + 2k}$. We prove this by induction on $k \geq 2$, for $k \in \mathbb{N}$.

Base case: For $k = 2$, RHS $= 2^{\lfloor \log_2 m_1 \rfloor + 4} > 2^{\log_2 m_1 - 1 + 4} = 8m_1$, which is strictly greater than $4m_1$, LHS for $k = 2$.

Induction hypothesis: Suppose the result holds for $k = k_1 \in \mathbb{N}$, i.e.,

$$m_1 2^{k_1} + k_1 - 2 \leq 2^{\lfloor \log_2 m_1 \rfloor + 2k_1}.$$

Inductive step: We need to show that the result holds for $k = k_1 + 1$, i.e.,

$$m_1 2^{k_1 + 1} + k_1 - 1 \leq 2^{\lfloor \log_2 m_1 \rfloor + 2(k_1 + 1)}.$$

LHS, $m_1 2^{k_1+1} + k_1 - 1 = 2 \left(m_1 2^{k_1} + k_1 - 2 \right) - k_1 + 3 \leq 2 \left(2^{\lfloor \log_2 m_1 \rfloor + 2k_1} \right) - k_1 + 3$, which is strictly less than $4 \left(2^{\lfloor \log_2 m_1 \rfloor + 2k_1} \right) = 2^{\lfloor \log_2 m_1 \rfloor + 2(k_1+1)}$, RHS.

Therefore, Case I holds for all $k \geq 2$ for some $k \in \mathbb{N}$.

Case II: Showing $\lceil \log_2 \left(\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^k + k - 1 \right) \rceil \leq \lfloor \log_2 m_1 \rfloor + 2k + 2$ is equivalent to showing

$$\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) 2^k + k - 1 \leq 2^{\lfloor \log_2 m_1 \rfloor + 2k + 1}.$$

We again prove this by induction on $k \geq 2$, for $k \in \mathbb{N}$.

Base case: For $k = 2$, LHS $= 4 \left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1} \right) + 1 = 4m_1 + 1 + 2^{\lfloor \log_2 m_1 \rfloor + 1}$. Similarly, for $k = 2$, RHS $= 2^{\lfloor \log_2 m_1 \rfloor + 4 + 1} = 16 \cdot 2^{\lfloor \log_2 m_1 \rfloor + 1}$, which can be distribute into $15 \cdot 2^{\lfloor \log_2 m_1 \rfloor + 1} + \cdot 2^{\lfloor \log_2 m_1 \rfloor + 1}$.

Now, $15 \cdot 2^{\lfloor \log_2 m_1 \rfloor + 1} > 15 \cdot 2^{\log_2 m_1 - 1 + 1} = 15m_1 > 4m_1 + 1$. Thus, the base case holds.

Induction hypothesis: Suppose the result holds for $k = k_1 \in \mathbb{N}$, i.e.,

$$\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^{k_1} + k_1 - 1 \leq 2^{\lfloor \log_2 m_1 \rfloor + 2k_1 + 1}.$$

Inductive step: We need to show that the result holds for $k = k_1 + 1$, i.e.,

$$\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^{k_1+1} + k_1 \leq 2^{\lfloor \log_2 m_1 \rfloor + 2k_1 + 3}.$$

$$\begin{aligned}
\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^{k_1+1} + k_1 =& 2\left[\left(m_1 + 2^{\lfloor \log_2 m_1 \rfloor - 1}\right) 2^{k_1} + k_1 - 1\right] - k_1 + 2 \\
\leq& 2\left(2^{\lfloor \log_2 m_1 \rfloor + 2k_1 + 1}\right) - k_1 + 2 \\
<& 4\left(2^{\lfloor \log_2 m_1 \rfloor + 2k_1 + 1}\right) = 2^{\lfloor \log_2 m_1 \rfloor + 2k_1 + 3}.
\end{aligned}$$

Therefore, Case II holds for all $k \geq 2$ for some $k \in \mathbb{N}$. In conclusion, using the conditionally clean ancilla technique, the exact Toffoli depth of an n-controlled Toffoli decomposition can not be reduced to $\lceil \log_2 n \rceil$, even when the control qubits are not restored to their original states.

Item 2, where the control qubits are restored to their original state, can be proved inductively in a similar manner. The factor of 2 needs to be multiplied in this case because the full depth, δ, from Proposition 6.2.1, is related to δ' as $\delta = 2\delta' - 2$, i.e., almost twice the depth considered in the first part. □

The above theorem shows that using the conditionally clean ancilla, the constant factor c can never be reduced to 1. In the next section, we demonstrate that the exact Toffoli depth in the Clifford plus Toffoli decomposition of an n-MCT is lower bounded by $\lceil \log_2 n \rceil$, regardless of the technique or the number of ancilla qubits used. This bound is achievable through complete binary tree decomposition of n-controlled Toffoli gates.

6.3 Tight lower bound on Toffoli depth

In this section, we show in a more general framework that the exact Toffoli depth in the Clifford plus Toffoli decomposition of an n-controlled Toffoli gate is lower bounded by $\lceil \log_2 n \rceil$, which is exactly $\log_2 n$ when $n = 2^k$ for some $k \in \mathbb{N}$. Additionally, the exact T depth also becomes $\lceil \log_2 n \rceil$, utilizing $2n - 2$ ancilla qubits and a T count of $4(n - 1)$, provided by [58]. Alternatively, following Gidney's logical-AND circuit [43], the ancilla count can be reduced to $n - 2$, maintaining the same T count, while the T depth increases to $\lceil \log_2 n \rceil + 1$.

Theorem 6.3.1. *The exact Toffoli depth in the Clifford plus Toffoli decomposition of an n-MCT is lower bounded by $\lceil \log_2 n \rceil$, given any technique or any number of ancilla qubits used.*

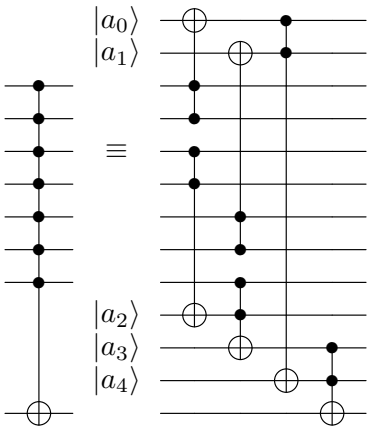


Figure 6.7: Quantum circuit decomposition of 7-MCT with Toffoli depth 3.

Proof. Each 2-controlled Toffoli gate encodes the information of two control qubits into one target qubit. This process can be visualized as a binary tree, where the leaf nodes represent the n control qubits, and their information is successively accumulated in internal parent nodes.

To implement an n-MCT, we begin with n leaf nodes, where each pair of control qubits is combined using a Toffoli gate, reducing the number of active qubits in each round. At each stage, either two parent nodes or one parent node, along with a leftover node from the previous round, are further combined into a new parent node. This process continues iteratively until the final Toffoli gate transfers the accumulated information to the root node (the target qubit).

Since each Toffoli gate reduces two qubits into one, the number of required levels in the binary tree is given by the height of a binary tree with n leaf nodes. The depth of such a binary tree is at least $\lceil \log_2 n \rceil$. Hence, the Toffoli depth of the n-MCT is also at least $\lceil \log_2 n \rceil$, proving the claim. □

A complete binary tree structure, having n leaf nodes, has a height $\log_2 n$, i.e., when n is not a power of 2. When $n = 2^k$, for some $k \in \mathbb{N}$, then it becomes a perfect binary tree, having the depth exactly $\log_2 n = k$. In both cases, the number of internal nodes, i.e., ancilla qubit, is $n - 2$. Additionally, the Toffoli count for both these cases becomes $n - 1$.

For example, a 32-MCT can be implemented using 30 ancilla qubits, with 33 Toffoli gates, having a minimum Toffoli depth of 5. Similarly, all n-MCT decomposition, with $17 \leq n \leq 32$, can be implemented with a minimum Toffoli depth of 5. Figure 6.7 provides a schematic diagram of the 7-MCT circuit decomposition using 5 ancilla qubits, and 6 Toffoli gates, with a Toffoli depth of $\lceil \log_2 7 \rceil = 3$. In the diagram, the first four Toffoli gates in the first two columns are implemented simultaneously, having depth 1. The next two Toffoli gates are applied sequentially, having a Toffoli depth of 1 each.

In this context, we present the following corollaries concerning the lower bound on the T depth of an n-MCT circuit implementation via Clifford plus Toffoli decomposition, regardless of the number of ancilla qubits used. These results can be obtained by first constructing an $\lceil \log_2 n \rceil$ Toffoli depth circuit for the n-MCT decomposition, followed by further decomposing the Toffoli gates into Clifford+T gates as outlined in Table 2.1.

Corollary 6.3.1. *Using the measurement-based Toffoli decomposition circuit proposed by [56], the T depth of the Clifford+T decomposition of an n-MCT gate, implemented via Clifford plus Toffoli decomposition, is lower bounded by* $\lceil \log_2 n \rceil$. *Furthermore, the circuit requires* $2n-2$ *ancilla qubits, and the T count is* $4n-4$.

Corollary 6.3.2. *Using the logical-AND circuit proposed by [43], an n-MCT gate can be implemented with a Clifford+T decomposition utilizing* $n-2$ *ancilla qubits. The resulting circuit has a T depth of* $\lceil \log_2 n \rceil + 1$ *and a T count of* $4n-4$.

Towards, the conclusion, let us outline a generalized approach that may provide a theoretical understanding. Given the Algebraic Normal Form (ANF) of a Boolean function $f \in \mathcal{B}_n^m$, any function of degree at most n can be implemented with a Toffoli (consequently T) depth of $\lceil \log_2 n \rceil$, using parallel MCT gates and an exponential number of ancilla qubits. However, for functions with polynomial many monomials in the ANF, which is common in practice, the ancilla overhead becomes polynomial. This motivates a broader study of space–depth trade-offs for arbitrary Boolean functions, which we explore in Chapter 7.

6.4 Conclusion

In this chapter, we revisited the n-controlled Toffoli decomposition using the conditionally clean ancilla technique described by Khattar and Gidney [59] and proposed an exact trade-off between Toffoli depth and the availability of clean ancilla qubits. By leveraging additional ancilla qubits, we achieved a lower Toffoli depth compared to their approach. Furthermore, we demonstrated that the conditionally clean ancilla technique cannot reduce the Toffoli depth strictly to $\lceil \log_2 n \rceil$, irrespective of unlimited availability of clean ancilla.

Additionally, we established that, regardless of the decomposition technique or available ancilla, the Toffoli depth of an n-MCT circuit is fundamentally lower-bounded by $\lceil \log_2 n \rceil$, with the optimal depth achieved through binary tree-based MCT decomposition. Finally, by incorporating the measurement-based uncomputation technique of Jaques et al. [56] for Toffoli decomposition, we extended this lower bound to T depth as well.

Chapter 7

Optimal T depth quantum circuits for arbitrary Boolean functions

The efficient implementation of quantum algorithms often rely on the efficient encoding of classical Boolean functions into quantum circuits through a process commonly known as oracle construction. An oracle is a reversible quantum circuit that implements an unknown Boolean function and is integral to several landmark algorithms, including Deutsch-Jozsa [27], Grover's search [49], Simon's hidden shift finding [88], and Shor's factoring algorithm [87].

In fault-tolerant quantum computing [58], constructing such oracles poses significant challenges due to the high resource overhead associated with multi-controlled Toffoli (MCT) gates. These gates must be decomposed into a universal gate set, such as Clifford+T [17], impacting key resource metrics like gate count, circuit depth, and qubit usage. Optimizing these parameters is essential for enhancing algorithmic efficiency and maintaining fidelity in large-scale quantum computations.

In the previous chapter, we emphasized on the importance of precise resource estimation for implementing arbitrary n-input, m-output Boolean functions, generalizing beyond simple multivariate AND functions. Building upon that foundation, this chapter presents the first generic construction of optimal T depth quantum circuits for arbitrary Boolean functions $f \in \mathcal{B}_n^m$. Our method generalizes the techniques introduced in Chapter 6 by leveraging the Algebraic Normal Form (ANF) of Boolean functions to minimize the Toffoli, and thus T, depth of the resulting circuits.

Obtaining such a benchmark for the minimum T depth is crucial for the efficient implementation of quantum algorithms by enabling greater parallelism, reducing time complexity, and minimizing circuit latency, making them suitable for near-term quantum devices with limited coherence times.

Although our construction introduces considerable qubit overhead, it offers a compelling trade-off by substantially reducing T depth. This makes our approach particu-

larly useful for oracle construction, reversible circuit synthesis, and quantum implementations of cryptographic primitives, such as S-boxes. As a case study, we demonstrate the practical utility of our approach by applying it to the quantum circuit design of AES, a widely used and structurally complex block cipher.

The organization of this chapter and its section-wise contributions are outlined as follows.

- Section 7.1 is the prime contributory section in terms of theoretical result, which provides a worst-case resource bound for optimal T depth quantum circuit implementation of any arbitrary n-input, m-output Boolean function. We illustrate the result with examples and highlight its implications for S-box design, including a summary of optimal T depth resource estimates for standard S-boxes in Table 7.1.
- In Section 7.2, to explain with an example, we focus on AES, providing a step-by-step resource estimation for an optimal T depth quantum implementation, and discuss the broader impact for a large class of block ciphers executed over multiple rounds.
- Section 7.3 explores the cryptanalytic implications of our results, particularly in the context of Grover's algorithm, and determines the optimal T depth for a full-round Grover's search for a large class of block ciphers.
- Finally, Section 7.4 concludes the paper with a brief summary of our contributions and future research directions.

7.1 Optimal T depth quantum resource estimation

In this section, we present a generic construction of optimal T depth quantum circuits for implementing arbitrary n-input, m-output Boolean functions, thereby extending the work from the previous chapter. Our approach builds on and generalizes the method of [37] by utilizing the ANF framework, particularly the XOR ($\mathbb{F}_2$-addition) of AND ($\mathbb{F}_2$-multiplication) components, in conjunction with tree-based circuit synthesis strategy to achieve minimal Toffoli (and consequently T) depth.

As discussed earlier, implementing higher-degree terms in the ANF of a Boolean function requires multi-controlled Toffoli (MCT) gates, where the variables from the monomials act as control qubits and the qubit storing the corresponding output serves as the target. Specifically, a k-degree monomial necessitates a k-MCT gate. According to Corollary 6.3.1 from the previous chapter, a k-MCT gate can be decomposed into the Clifford+T gate set via Toffoli decomposition, resulting in a T depth of $\lceil \log_2 k \rceil$. This is the optimal T depth for implementing a k-MCT, assuming the decomposition is via Clifford plus Toffoli gates. We refer to this as MCT-Toffoli-T optimality. Accordingly,

using a binary tree structure, the implementation of the highest-degree term $x_1x_2\cdots x_n$ incurs an MCT-Toffoli-T optimal T depth of $\lceil \log_2 n \rceil$. Throughout this chapter, whenever we refer to optimal T depth, we imply that the optimality achieved via Clifford plus Toffoli decomposition, as established in Corollary 6.3.1 and in our paper work [37, Corollary 1].

Notably, the ANF of an arbitrary n-input, m-output Boolean function $f \in \mathcal{B}_n^m$ can contain up to $2^n - (n+1)$ unique non-linear terms. More precisely, it may include at most $\binom{n}{k}$ unique degree-k monomials, each requiring a k-MCT gate for its quantum implementation. If all k-MCT gates are applied in parallel, the MCT depth becomes 1, with the largest being an n-MCT contributing to a T depth of $\lceil \log_2 n \rceil$. Once implemented, the nonlinear monomials can be XOR-ed to the appropriate output qubit using CNOT gates, without increasing the T depth. This implies that the complete ANF of $f \in \mathcal{B}_n^m$ can be implemented in quantum with an optimal T depth of $\lceil \log_2 n \rceil$.

Clearly, to implement all the MCT gates in parallel, multiple copies of the input variables are required, along with distinct ancilla qubits to store intermediate outputs. These copies can be generated using CNOT gates, which, being Clifford operations, do not affect the T depth of the circuit. We summarize the results in the following theorem.

Theorem 7.1.1. *Let $f \in \mathcal{B}_n^m$ be an n-input, m-output Boolean function. Then, the Clifford+T decomposition of the quantum circuit implementing f can be realized with an optimal T depth $\lceil \log_2 n \rceil$, using the following resources:*

- *at most $2^{n-1}(3n-2) - 3n + m + 1$ reusable ancilla qubits,*
- *a total of $2^{n+1}(n-2) + 4$ T gates,*
- *a maximum of $2^{n-1}(11n + 2m - 18) - 4n - m + 9$ CNOT gates, with*
- *a CNOT depth $2^n + 2n + 9\lceil \log_2 n \rceil - 3$.*

Proof. There are $\binom{n}{k}$ degree-k monomials, each involving k variables. Computing all such nonlinear terms in parallel requires a total $\sum_{k=2}^{n} k\binom{n}{k}$ copies of the input variables. Since one copy of each input is already available, this entails an additional $\sum_{k=2}^{n} k\binom{n}{k} - n$ ancilla qubits and an equal number of CNOT gates for both computation and uncomputation. These introduce a CNOT depth of $2\lceil \log_2(\sum_{k=2}^{n} \binom{n-1}{k-1} - n) \rceil = 2(n-1)$. Additionally, $\sum_{k=2}^{n} \binom{n}{k}$ ancilla qubits are needed to store the outputs of these MCT gates.

After realizing all the unique nonlinear monomials from the ANF, any m-output Boolean function can be constructed by copying up to $\sum_{k=2}^{n} \binom{n}{k} + n$ monomials per function to m ancilla qubits. This requires at most $m(\sum_{k=2}^{n} \binom{n}{k} + n)$ CNOT gates and adds a maximum CNOT depth of $2^n - 1$.

Finally, by [37, Corollary 1], each k-MCT gate in its optimal T depth decomposition uses $2(k-1)$ ancilla qubits and $9(k-1)$ CNOT gates. Ancilla qubits are made reusable

via uncomputation. Since each T depth layer corresponds to a CNOT depth of 9, the overall CNOT depth contribution is $9\lceil \log_2 n \rceil$. Hence, the overall resource requirement is given by:

- **#Ancilla qubits:**

$$\left[\sum_{k=2}^{n} k\binom{n}{k} - n\right] + \sum_{k=2}^{n}\binom{n}{k} + m + \sum_{k=2}^{n} 2(k-1)\binom{n}{k} = 2^{n-1}(3n-2) - 3n + m + 1,$$

- **#T gates:**

$$\sum_{k=2}^{n} 4(k-1)\binom{n}{k} = 2^{n+1}(n-2) + 4,$$

- **#CNOT gates:**

$$\begin{aligned} &2\left[\sum_{k=2}^{n} k\binom{n}{k} - n\right] + m\left[\sum_{k=2}^{n}\binom{n}{k} + n\right] + \sum_{k=2}^{n} 9(k-1)\binom{n}{k} \\ =&2^{n-1}(11n + 2m - 18) - 4n - m + 9. \end{aligned}$$

- **CNOT depth:**

$$2(n-1) + (2^n - 1) + 9\lceil \log_2 n \rceil = 2^n + 2n + 9\lceil \log_2 n \rceil - 3.$$

□

Theorem 7.1.1 directly applies to the quantum circuit synthesis of S-boxes, where an n-bit S-box is modeled as an n-input, n-output Boolean function. The resulting circuit requires at most $2^{n-1}(3n-2) - 2n + 1$ reusable ancilla qubits, $2^{n+1}(n-2) + 4$ T gates, and up to $2^{n-1}(13n - 18) - 5n + 9$ CNOT gates. The CNOT depth is bounded by $2^n + 2n + 9\lceil \log_2 n \rceil - 3$, while achieving an optimal T depth of $\lceil \log_2 n \rceil$. To illustrate the construction, we present the 3-bit S-box used in LowMC [3] as an example.

Example 7.1.1. *The coordinate Boolean functions are given by:* $f_0 = x_0 \oplus x_1x_2$, $f_1 = x_0 \oplus x_1 \oplus x_0x_2$, *and* $f_2 = x_0 \oplus x_1 \oplus x_2 \oplus x_0x_1$. *Since the maximum algebraic degree is 2, the corresponding quantum circuit (see Figure 7.1) achieves the optimal T depth* $\lceil \log_2 2 \rceil = 1$, *using 9 ancilla qubits, 12 T gates, and at most 33 CNOT gates. Notably, as the ANFs of these Boolean functions do not contain all possible monomials, the actual resource requirements are significantly lower than the worst-case estimates provided in Theorem 7.1.1. In practice, such sparsity often results in substantially reduced overheads compared to theoretical upper bounds.*

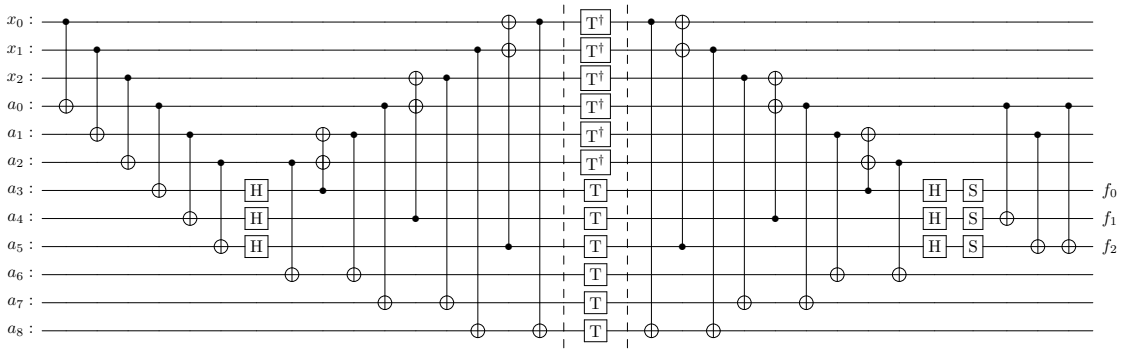


Figure 7.1: Quantum circuit for a 3-bit S-box (used in LowMC) with T depth 1.

S-box	Variables	Ancilla	CNOT count	CNOT depth	T count	T depth
LowMC [3]	3	9	39	13	12	1
DEFAULT [7]	4	17	79	27	24	2
GIFT [9]	4	14	76	27	24	2
PRESENT [13]	4	19	105	30	32	2
PRINCE [15]	4	24	128	28	40	2
ASCON [30]	5	27	120	22	32	1
AES [24]*	8	596	3647	184	984	3

Table 7.1: Optimal T depth quantum circuit synthesis of various standard S-boxes with corresponding resource estimates.

Table 7.1 summarizes the quantum resource requirements for synthesizing various standard S-boxes with optimal T depth. A detailed analysis for the AES S-box (marked with *) is provided in Section 7.2.

From Theorem 7.1.1, the optimal T depth Clifford+T decomposition of an n-input, single-output Boolean function $f \in \mathcal{B}_n$ requires at most $2^{n-1}(3n-2)-3n+2$ reusable ancilla qubits, $2^{n+1}(n-2)+4$ T gates, and up to $2^{n-1}(11n-16)-4n+8$ CNOT gates, with a CNOT depth of $2^n+2n+9\lceil\log_2 n\rceil-3$. We illustrate the circuit construction with a representative example.

Example 7.1.2. *Let $f = x_0x_2 \oplus x_1x_3 \oplus x_0x_1x_2x_3$ be a Boolean function consisting of three nonlinear terms. Since $\deg(f) = 4$, its Clifford+T decomposition yields a T depth of $\lceil\log_2 4\rceil = 2$. The resulting quantum circuit requires 12 ancilla qubits, 20 T gates, and 46 CNOT gates, with a CNOT depth of 12 (see Figure 7.2). Notably, as f does not contain all possible monomials, the resource overhead is significantly lower than the theoretical worst-case bound.*

While our construction achieves optimal T depth, it incurs an exponential overhead in ancilla qubits and CNOT gates with respect to the number of variables. A more practical

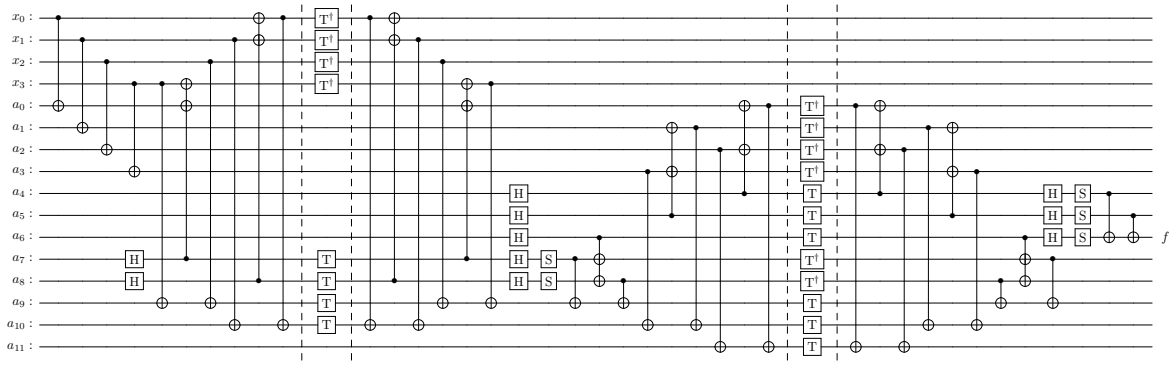


Figure 7.2: Quantum circuit for $f(x_0, x_1, x_2, x_3) = x_0x_2 \oplus x_1x_3 \oplus x_0x_1x_2x_3$ with T depth 2.

alternative is to allow a slight increase in T depth in exchange for a substantial reduction in ancilla and CNOT counts. For instance, replacing the T depth 1 Toffoli decomposition from [56] (Figure 2.18) with the logical-AND-based decomposition from [43] (Figure 2.19) in Theorem 7.1.1 reduces the ancilla count by $2^{n-1}(n-2)+1$ and the CNOT count by $3 \cdot 2^{n-1}(n-2)+3$, while marginally increasing the T depth from $\lceil \log_2 n \rceil$ to $\lceil \log_2 n \rceil + 1$. This trade-off highlights a promising direction for future research, where our construction can serve as a T depth benchmark while optimizing other quantum resources, even at the cost of slight T depth sub-optimality, rather than prioritizing depth reduction alone.

This observation also clarifies the optimal T depth for ANF-based implementations. In particular, for quantum circuits designed for cryptanalytic purposes, this optimal depth, viewed as a benchmark, has not been systematically explored as a performance metric. In the next section, we illustrate this gap through concrete examples, focusing on the widely studied AES block cipher. The idea naturally extends to any standard block ciphers in general.

7.2 Optimal T depth quantum circuit of AES

This section presents an optimal T depth quantum circuit for the AES algorithm. We first construct an optimal T depth implementation of the AES S-box and subsequently extend the construction to the full AES algorithm for key sizes of 128, 192, and 256 bits, corresponding to 10, 12, and 14 rounds, respectively.

AES is a symmetric-key block cipher standardized by NIST, operating on 128-bit data blocks. The internal state is represented as a 4×4 matrix $S \in \mathbb{F}_{2^8}^{4\times 4}$, where each element $S_{i,j} \in \mathbb{F}_{2^8}$ corresponds to a byte.

The AES algorithm can be abstracted as a Boolean function $f : \mathbb{F}_2^{m+k} \to \mathbb{F}_2^c$, where $m = 128$ is the message length, $k \in 128, 192, 256$ is the key length, and $c = 128$ is the

ciphertext length. From Theorem 7.1.1, the optimal T depth for AES, when viewed as a monolithic Boolean circuit, is given by $\lceil \log_2 256 \rceil = 8$, $\lceil \log_2 320 \rceil = 9$, and $\lceil \log_2 384 \rceil = 9$, depending on the key size. However, modeling AES as a single combinational Boolean circuit is both impractical and analytically intractable due to its iterative structure. While these T depth values offer theoretical lower bounds, they do not serve as practical benchmarks for circuit design.

In practice, AES is implemented in a round-wise. Hence, a more feasible approach is to design the quantum circuit for each round individually and estimate the overall T depth based on actual implementations. Each AES round (except the final one) consists of four transformations, described as follows.

- **SubBytes**: Applies the AES S-box to each element $S_{i,j} \in \mathbb{F}_{2^8}$ of the internal state. This is the only nonlinear operation and the primary contributor to the T depth, requiring 16 parallel S-box evaluations: $S_{i,j} \leftarrow \text{S-box}(S_{i,j})$.
- **ShiftRows**: Performs a cyclic left shift on each row of the state matrix. For row i, the transformation is defined as $S_{i,j} \leftarrow S_{i,(j+i) \bmod 4}$. As this is essentially a SWAP operation, it is implemented via rewiring in the quantum circuit and does not incur additional quantum resources.
- **MixColumns**: Applies a linear transformation to each column $\mathbf{c} \in \mathbb{F}_{2^8}^4$ of the internal state S using an MDS matrix $M \in \mathbb{F}_{2^8}^{4\times 4}$: $\mathbf{c} \leftarrow M \cdot \mathbf{c}$. Since this transformation is linear, it can be realized using only CNOT gates. Note that the final round of AES omits the MixColumns operation.
- **AddRoundKey**: The AES key schedule expands the primary key K into round keys $K_0, K_1, \ldots, K_r \in \mathbb{F}_{2^8}^{4\times 4}$, each used in a specific round. In this step, the internal state S is XORed with the current round key K_i, defined as $S \leftarrow S \oplus K_i$.

Since only the SubBytes transformation introduces nonlinearity via S-boxes, specifically through multi-controlled Toffoli (MCT) gates contributing to the T depth, the overall T depth of an AES implementation is dominated by the S-box operations. As 16 S-boxes are evaluated in parallel per round, the total T depth of the AES circuit can be estimated as r times the T depth of a single S-box, where r denotes the number of rounds. We now present a detailed quantum resource estimation for the AES S-box with optimal T depth, as derived from Theorem 7.1.1.

The coordinate Boolean functions of the AES S-box have a maximum algebraic degree of 7. By Theorem 7.1.1, an optimal T depth quantum circuit for the AES S-box can thus be constructed with a T depth of $\lceil \log_2 7 \rceil = 3$. Notably, the ANF of all coordinate functions collectively includes all monomials up to degree 7. We construct these using three layers of parallel Toffoli gates.

- **First Layer:** We begin by constructing all $\binom{8}{2} = 28$ degree-2 monomials. This requires 7 copies of each input variable, 56 copies in total, 8 of which already exist.

Thus, 48 ancilla qubits and 48 CNOT gates are needed to copy the inputs, with a CNOT depth of 3. Storing the 28 quadratic terms requires 28 additional ancilla and 28 parallel Toffoli gates, contributing a Toffoli depth of 1. In total, the first layer uses $48 + 28 = 76$ ancilla qubits, 48 CNOT gates (CNOT depth 3), and 28 Toffoli gates.

- **Second Layer:** This layer constructs all $\binom{8}{3} = 56$ degree-3 and $\binom{8}{4} = 70$ degree-4 monomials by combining previously computed terms. To create 56 cubic terms, we combine degree-2 terms with single-variable terms. As we have 56 single-variable qubits and 28 degree-2 terms, we need to duplicate the quadratic terms using 28 ancilla and 28 CNOT gates (depth 1). Storing the 56 cubic terms adds 56 ancilla. To construct 70 quartic terms, we need five additional copies of all degree-2 monomials (140 ancilla and 140 CNOT gates), and 70 more ancilla to store the outputs. The second layer thus requires a maximum of $(28 + 56 + 140 + 70) = 294$ ancilla, $(28 + 140) = 168$ CNOT gates (CNOT depth 3), and $(56 + 70) = 126$ Toffoli gates.

- **Third Layer:** At this stage, we have: 70 quartic, 56 cubic, 196 quadratic, and 56 single-variable qubits. We now build all degree-5, 6, and 7 monomials. The 8 septics $\binom{8}{7} = 8$) are computed by combining eight degree-4 and eight degree-3 terms. The 56 quintics ($\binom{8}{5}$) are obtained by combining 56 degree-4 and degree-1 terms. The 28 sextics ($\binom{8}{6}$) require combinations of four degree-4 and degree-2, and 24 degree-3 and degree-3 terms. In total, constructing and storing these 92 higher-degree monomials requires $(56 + 28 + 8) = 92$ ancilla qubits and 92 Toffoli gates.

Across all eight coordinate functions, 1001 monomials appear in total, requiring 1001 CNOT gates and 8 ancilla qubits for output storage. The CNOT depth is dominated by the coordinate function with the most monomials, which is 145.

Using the T depth 1 Toffoli-to-T decomposition (see Figure 2.18), implementing 246 Toffoli gates require $(246 \times 4) = 984$ T gates and $(246 \times 9) = 2214$ CNOT gates with a CNOT depth of $(9 \times 3) = 27$. Since Toffoli gates are executed in three different layers (with a maximum of 126 in the second layer), this incurs an additional (reusable) ancilla count of 126. Although, the Toffoli gates can be uncomputed without extra cost, uncomputing intermediate qubits requires $(48 + 168) = 216$ CNOT gates.

Hence, the optimal T depth implementation of the AES S-box requires a total $(76 + 294 + 92 + 8 + 126) = 596$ ancilla qubits, $(48 + 168 + 1001 + 2214 + 216) = 3647$ CNOT gates, with a CNOT depth of $2(3 + 3) + 27 + 145 = 184$, and 984 T gates, achieving the T depth 3.

Alternatively, replacing the T depth 1 design with Gidney's logical-AND decomposition (see Figure 2.19) reduces the ancilla count by 126 and the CNOT count by $(246 \times 3) = 738$, at the cost of increasing the T depth to $\lceil \log_2 7 \rceil + 1 = 4$. In

this case, each Toffoli gate has CNOT depth 6, resulting in a total CNOT depth of $2 \times (3 + 3) + 145 + (3 \times 6) = 175$.

Toffoli-to-T	Ancilla count	CNOT count	CNOT depth	T count	T depth
Figure 2.19	470	2909	175	984	4
Figure 2.18	596	3647	184	984	3

Table 7.2: Quantum circuits for AES S-box with corresponding resource estimates.

In the SubBytes transformation, 16 AES S-boxes are executed in parallel, resulting in a 16-fold increase in the number of ancilla qubits, CNOT gates, and T gates compared to a single S-box, while the circuit depths remain unchanged. The ShiftRows transformation introduces no additional quantum resource overhead. According to [99, Table 5], the MixColumns transformation can be implemented in-place (i.e., without additional ancilla) using 98 CNOT gates with a CNOT depth of 13. As MixColumns is applied in parallel to the four columns of the internal state, the total CNOT count becomes $4 \times 98 = 392$, while the CNOT depth remains 13. Finally, the bit-wise XOR with the round key requires 128 parallel CNOT gates, contributing a CNOT depth of 1. The overall quantum resource requirements for a single round of AES are summarized in Table 7.3.

Toffoli-to-T	Ancilla count	CNOT count	CNOT depth	T count	T depth
Figure 2.19	7520	47064	189	15744	4
Figure 2.18	9536	58872	198	15744	3

Table 7.3: Quantum implementation of a single round of AES with corresponding resource estimates

Assume that AES operates for $r \in 10, 12, 14$ rounds, depending on the key length. Ancilla qubits used in the S-boxes of each round are reclaimed for subsequent rounds through uncomputation, except for the 8 qubits required to store the functional output. As a result, the ancilla count increases by $8 \times 16 = 128$ per round, yielding a total ancilla requirement of $9536 + 128(r - 1)$. Furthermore, since the final round omits the MixColumns operation, the total CNOT count is given by $58872r - 392$, and the CNOT depth is $198r - 13$. Both the T count and T depth scale linearly with the number of rounds.

Table 7.4 presents the resource estimates for complete quantum implementations of AES with different key sizes, using our optimal T depth construction (see Figure 2.18). The T depth achieved in this construction is provably optimal, as no quantum circuit for AES, when the rounds are implemented sequentially, can attain a lower T depth, irrespective of the number of ancilla qubits or other resource overheads.

Key size	No. of round	Ancilla	CNOT count	CNOT depth	T count	T depth
128	10	10688	588328	1967	157440	30
192	12	10944	706072	2363	188928	36
256	14	11200	823816	2759	220416	42

Table 7.4: Optimal T depth quantum implementation of full round AES with corresponding resources.

This additional resource overhead in Table 7.4 demonstrates the practical viability of our optimal T depth quantum circuit constructions for AES. While earlier implementations have reported fewer ancilla qubits, lower gate counts, and in some cases, comparable T depths (e.g., [53, Table 7] and [100, Table VI] report a T depth of 60 for AES and AES† combined, i.e., 30 per instance, and [55, Table 5] reports T depths of 30, 36, and 42 for AES-128, AES-192, and AES-256, respectively), none of these works formally establish the optimality of their constructions. Given that presently AES is one of the most popular ciphers in symmetric-key cryptography, numerous heuristic and brute-force efforts have been made to optimize its quantum implementation. The T depth reductions observed in prior works primarily derived from those efforts, rather than from systematic constructions with provable optimality over a general class of block ciphers.

In this context, we acknowledge the recent work of Huang et al. [54], which also reports a T depth 3 implementation of the AES S-box ([54, Table 3]) and establishes its minimality, similar in spirit to our own findings. However, we emphasize that our work is independent and presents a generic quantum circuit construction method that achieves minimal the T depth for any arbitrary Boolean function derived from its Algebraic Normal Form. This construction provides a naive yet complete upper bound on ancillary quantum resources, thereby setting new benchmarks applicable to a broad class of S-boxes beyond AES. Furthermore, our approach is a generalization of the optimal T depth MCT decomposition (via Clifford plus Toffoli gates), as outlined in Corollary 6.3.1 (see our paper [37]). Consequently, optimization strategies such as logical-AND and conditionally clean ancilla techniques can also be integrated to reduce quantum resource overheads significantly, at the cost of a marginal increase in T depth beyond the optimal bound.

It is important to note that a T depth 3 implementation of the AES S-box does not necessarily imply the optimality for the full-round AES circuit. Algebraic manipulations across multiple rounds may reduce the combined T depth below the additive bound and must be analyzed individually for each block cipher. In contrast, we propose a generic, step-by-step construction framework that guarantees optimal T depth for a broad class of block ciphers beyond AES, establishing a definitive benchmark for quantum cryptographic implementations.

For reference, Table 7.5 summarizes prior AES implementations focused on T depth reduction, along with associated resource estimates.

Key size	References	Ancilla	CNOT count	CNOT depth	T count	T depth
128	[56, Table 4]	4244	284420	NA	54400	120
128	[51, Table 2]	9384	NA	NA	33600	50
128	[61, Table 13]	3689	132376	NA	27200	40
128	[53, Table 7]	5576	285393	NA	62400	30
128	[100, Table VI]	NA	228020	NA	52800	30
128	[54, Table 8]	NA	176580	NA	33600	30
128	[55, Table 5]	6128	120812	NA	117984	30
128	**This work** [31]	10688	588328	1967	157440	30
192	[56, Table 4]	4564	321021	NA	60928	144
192	[51, Table 2]	10456	NA	NA	37632	60
192	[61, Table 13]	3945	149256	NA	30464	48
192	[55, Table 5]	6448	136812	NA	132960	36
192	**This work** [31]	10944	706072	2363	188928	36
256	[56, Table 4]	4884	393534	NA	75072	168
256	[51, Table 2]	46368	NA	NA	12704	70
256	[61, Table 13]	4457	187128	NA	38080	56
256	[55, Table 5]	6768	168548	NA	165264	42
256	**This work** [31]	11200	823816	2759	220416	42

Table 7.5: Optimal T depth quantum implementation of full round AES: A comparison with earlier works.

7.3 Cryptanalytic implications

We know that the Shor's factoring algorithm [87] poses the most critical threat to existing classical public key cryptographic protocols due to its implications in attacking the RSA and the discrete logarithm-based schemes. On the other hand, the impact of Grover's algorithm [49] can be temporarily mitigated by doubling the key size. Still there exist numerous instances where the Grover's search, often in conjunction with other quantum algorithms, has demonstrated significant cryptanalytic potential. For example, in [60], Grover's algorithm is combined with Simon's hidden shift technique to break the security of Even-Mansour constructions. Needless to mention that the resource requirements for a full-scale key recovery attack on symmetric ciphers using Grover's algorithm remain infeasible in the near term. Still, substantial work has been conducted on Grover-based cryptanalysis, particularly on AES [53, 54, 55, 56, 61, 86, 100]. In this context, we present the following result on the optimality of T depth required for full-round cryptanalysis

using Grover's search on a block cipher $\mathbf{B}$, where the only source of nonlinearity generates from a single S-box.

Suppose $\mathbf{B}$ is a block cipher whose only nonlinearity arises from an n-bit S-box $\mathbf{S}$. Then, from Theorem 7.1.1, implementing the n-bit S-box $\mathbf{S}$ requires an optimal T depth of $\lceil \log_2 n \rceil$. Since the S-box is the sole source of nonlinearity in the cipher, each round incurs an optimal T depth of $\lceil \log_2 n \rceil$. Assuming the cipher runs for r rounds and produces a ciphertext of length m, the quantum implementation of the full-round cipher $\mathbf{B}$ has a total T depth of $r\lceil \log_2 n \rceil$.

To perform an exhaustive key search using Grover's algorithm, one must implement $\mathbf{B}$, compare the output with a known m-bit ciphertext using an m-MCT gate (which requires an optimal T depth of $\lceil \log_2 m \rceil$ [37]), and then apply the inverse of the full-round cipher, $\mathbf{B}^\dagger$, which adds another $r\lceil \log_2 n \rceil$ to the T depth. Therefore, a single Grover iteration requires an optimal T depth of $2r\lceil \log_2 n \rceil + \lceil \log_2 m \rceil$. Since Grover's algorithm requires $2^{k/2}$ iterations for a key of length k, the exhaustive key recovery attack using sequential Grover's search requires an overall T depth of $(2r\lceil \log_2 n \rceil + \lceil \log_2 m \rceil)\, 2^{k/2}$.

As an immediate corollary, an exhaustive key recovery attack on AES using Grover's algorithm can be executed with an optimal T depth of $(2r\lceil \log_2 8 \rceil + \lceil \log_2 128 \rceil)\, 2^{k/2}$, which evaluates to 67×2^{64}, 79×2^{96}, and 91×2^{128} for key lengths of 128, 192, and 256 bits, respectively. In this regard, we make the following remarks based on different execution scenarios of Grover's algorithm.

Remark 7.3.1.

- *If the block cipher $\mathbf{B}$ is implemented out-of-place, then the inverse full-round cipher can be realized via measurement-based uncomputation, which does not incur any additional T depth. Consequently, Grover's search can be executed with an optimal T depth of $(r\lceil \log_2 n \rceil + \lceil \log_2 m \rceil)\, 2^{k/2}$. The corresponding T depth requirements for AES would be 37×2^{64}, 43×2^{96}, and 49×2^{128} for key lengths of 128, 192, and 256 bits, respectively.*

- *Furthermore, if Grover's search is executed in parallel, i.e., all the 2^k copies of the oracle $\mathbf{B}$ is running concurrently, the T depth for the parallel Grover's search becomes $(r\lceil \log_2 n \rceil + \lceil \log_2 m \rceil)$. Accordingly, for AES, the corresponding minimal T depth would be 37, 43, and 49 for key lengths of 128, 192, and 256 bits, respectively.*

7.4 Conclusions

Given the ANF of an arbitrary n-input, m-output Boolean function f having algebraic degree k, this work presents the construction of an optimal T depth quantum circuit for f, via Toffoli decomposition, along with a complete resource estimation. The primary focus of this paper is on minimizing T depth, an essential metric in quantum circuit

design due to its direct impact on circuit latency and coherence time. While the overall resource usage may be high, we argue that establishing the benchmark T depth should be the first step in any quantum circuit synthesis process, after which other resource parameters may be optimized. This work conclusively settles the minimum achievable T depth for Boolean functions and demonstrates its practical relevance through block cipher constructions, as well as in cryptanalysis.

Chapter 8

Conclusion

In this chapter, we summarize the overall contribution of our thesis along with the related open problems for future research. The thesis mainly considers two aspects. Firstly, we consider various quantum algorithms related to Boolean functions, which are motivated by the famous Forrelation formulation. This is related to various spectra related to Boolean functions. Consequently, we study the implementation of Boolean functions with low Toffoli or T depth. This is motivated by the reason that the Toffoli gate is among the most significant quantum gates, and such gates greatly influence the efficiency of fault-tolerant quantum circuits. A lower depth is thus quite important in this regard, and we connect this to the Algebraic Normal Form of Boolean functions to obtain certain benchmarks.

8.1 Summary of the thesis

The thesis begins with an introduction and background (Chapters 1 and 2 respectively).

In the first contributory chapter (Chapter 3), we explore quantum algorithms to analyze various cryptographically significant spectra of Boolean functions, in particular, the Walsh–Hadamard spectrum, the crosscorrelation spectrum, and the autocorrelation spectrum. First, we study the connection between the 2-fold Forrelation formulation with the bent duality-based promise problems as desirable instantiations. Next, we explore the 3-fold Forrelation formulation, which helps obtain a unifying framework for evaluating different spectra. Additionally, our setup helps in estimating the Walsh–Hadamard spectrum of $f \in \mathcal{B}_n$ at the points in $S \subseteq \mathbb{F}_2^n$ using the 3-fold Forrelation algorithm, achieving a higher sampling probability compared to the standard Deutsch–Jozsa algorithm. Further, we also achieve certain improvements in query complexity over the existing results to determine the resiliency of a Boolean function. Finally, we relate the results with autocorrelation and crosscorrelation spectra and observed improvements over the existing results.

Next, we introduce the concept of nega-Forrelation in Chapter 4. Noting the efficiency achieved in various quantum algorithms following Forrelation, it is natural to study a similar formulation towards the efficient sampling of nega-Hadamard transforms. In this chapter, we extend the Forrelation framework to define the (3-fold) nega-Forrelation and provide related results. We introduce a novel strategy to sample small values of the nega-Hadamard transform more efficiently than the extended Deutsch-Jozsa algorithm in terms of the required number of queries, as presented in recent literature. In this regard, as in Chapter 3, we also consider studying the spectra for a point or a subset of points. Results related to nega-crosscorrelation spectrum are also described. Finally, we consider the problems related to identifying shifts in bent as well as negabent functions, extending certain state-of-the-art results.

Then we move to Chapter 5, where we focus on explaining the existing frameworks in terms of further generalized Boolean functions' spectra. We analyze new unitaries, exploring their implications and establishing connections to existing results. We extend the formulation with different cryptographically significant spectra, including the Walsh-Hadamard, crosscorrelation, and autocorrelation spectra, to a generalized variation with the m-th primitive root of unity, for any $m \in \mathbb{N}$. In the process, we examine previously unexamined classes of Hadamard transforms that lies between the Walsh-Hadamard and nega-Hadamard transformations, which were not present in the existing literature.

All the relevant algorithms discussed here have been implemented and tested using the IBMQ simulator, and the results are verified with the theoretical ones.

Next, in a different but related direction, we move towards exact circuit implementation in Chapter 6. As we know, the quantum gates are the fundamental building blocks of quantum circuits and are inherently reversible. Such quantum gates are mathematically represented by unitary matrices. We have already discussed in great detail that the doubly-controlled X-gate, known as the Toffoli gate too, is among the most significant quantum gates, with various applications in computing and circuit implementation. In this direction, we consider the optimized implementation of multi-controlled Toffoli (MCT) using the Clifford+T gate sets. In particular, we explicitly quantify the trade-off (with concrete formulae) between the Toffoli depth (the depth using the classical 2-controlled Toffoli) and the number of clean ancilla qubits. We achieve a reduced Toffoli depth (and consequently, T depth), which is an extension of certain very recent techniques. From the point of view of a theoretically negative result, we show that using such conditionally clean ancilla techniques, Toffoli depth can never achieve the lower bound of $\lceil \log_2 n \rceil$, though it remains in the same order.

Finally, in Chapter 7, we present a generic construction to design an optimal T depth quantum circuit for any arbitrary n-input m-output Boolean function. This is a generalization of the technique used in Chapter 6, mostly exploiting the ideas taken from Algebraic Normal Form and binary tree representation. We present optimal Toffoli (consequently T) depth quantum circuits implementation for evaluating any Boolean function that subsumes the idea of Chapter 6 and it is achieved through the fundamental

observation that the XOR ($\mathbb{F}_2$ addition) of AND ($\mathbb{F}_2$ multiplication) in ANF can be realized through the tree implementation. Our construction achieves optimal T depth, but it incurs exponential overhead in ancilla qubits and CNOT gates with respect to the number of variables. That is, our contribution here is that, during any circuit design, first it must be understood what should be the benchmark T depth of the circuit and only then the other parameters may be evaluated. We explain our results with the examples related to block cipher implementations.

8.2 Future research directions

In Chapter 3, we employed the 3-fold Forrelation framework to sample several cryptographically relevant spectra of Boolean functions, including Walsh-Hadamard, cross-correlation, and autocorrelation, by relating it to the square of the Walsh-Hadamard transform. A natural direction for future work is to investigate whether this technique can be extended to sample higher-order moments of the Walsh coefficients, potentially by adapting the k-fold Forrelation algorithm of [2]. In particular, estimating the fourth moment could offer insights into special classes of Boolean functions such as APN functions. This generalization may facilitate deeper analysis of Boolean function properties via moment-based techniques. Additionally, improving the efficiency of the resiliency checking algorithm, beyond the method presented in Chapter 3, remains an open question.

In Chapter 4, we studied the hidden shift finding algorithms for bent and negabent functions using Forrelation-like methods [78]. However, for bent (or negabent) functions related by an orthogonal transformation $g(\mathbf{x}) = f(A\mathbf{x})$, no known quantum algorithm can efficiently recover the matrix A. Investigating whether Forrelation or its variants could be extended to extract such transformations presents another promising avenue for future research.

Chapter 5 opens up further questions on the cryptographic implications of generalized m-bent functions, particularly their relationships with other 2^k-bent classes and the conditions under which such transformations can be interchanged or composed.

Another critical direction is the efficient implementation of these quantum algorithms on noisy intermediate-scale quantum (NISQ) devices. Developing fault-tolerant versions that are resilient to noise, as preliminarily explored in [33, 64], is essential for making these techniques practically viable.

In Chapter 6, we provided optimal Toffoli (and thus T) depth decompositions for multi-controlled Toffoli (MCT) gates under a 1D nearest-neighbor qubit architecture. Future extensions could focus on adapting these results to more advanced hardware layouts, such as 2D or 3D qubit topologies, which are increasingly supported by contemporary quantum hardware platforms. Although such networks come with architectural

constraints [50], they may also unlock greater parallelism and reduced latency. For instance, related advancements in low-depth quantum implementations of activation functions for machine learning, such as constant T depth ReLU circuits, have been explored in [101].

Finally, Chapter 7 establishes a benchmark for optimal T depth circuit construction from the ANF of a Boolean function. While this construction minimizes T depth, it incurs exponential overhead in ancilla qubits and CNOT gates. Future research could explore meaningful trade-offs, aiming to reduce these overheads with only marginal increases in T depth, thereby improving the overall efficiency and practicality of quantum oracle construction.

8.3 Final comments

In this thesis, we have studied certain problems in the domain of quantum algorithms and circuits related to Boolean functions. The fields of computing and communication use Boolean functions to a great extent. As it is well known, Boolean functions were introduced by the famous English mathematician George Boole (2nd November 1815 – 8th December 1864) in his monograph "An Investigation of the Laws of Thought" (1854). The subject has developed for more than one hundred and eighty years. We add a few results in this vast domain in the context of quantum algorithms and circuits.

Bibliography

[1] S. Aaronson. *BQP and the Polynomial Hierarchy*. In Proceedings of the 42nd ACM Symposium on Theory of Computing - STOC '10, ACM, pp. 141–150, 2010. doi:10.1145/1806689.1806711

[2] S. Aaronson and A. Ambainis. *Forrelation: A Problem that Optimally Separates Quantum from Classical Computing*. In Proceedings of the 47th ACM Symposium on Theory of Computing - STOC '15, ACM, pp. 307–316, 2015. Siam J. Comput., vol. 47, no. 3, pp. 982–1038, 2018. doi:10.1145/2746539.2746547

[3] M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. *Ciphers for MPC and FHE*. Advances in Cryptology - EUROCRYPT '15, Springer LNCS, vol. 9056, pp. 430–454, 2015. doi:10.1007/978-3-662-46800-5_17

[4] M. Amy, D. Maslov, M. Mosca, and M. Roetteler. *A Meet-in-the-middle Algorithm for Fast Synthesis of Depth-optimal Quantum Circuits*. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, issue 6, pp. 818–830, 2013. doi:10.1109/TCAD.2013.2244643

[5] R. Anand, A. Maitra, S. Maitra, C. S. Mukherjee, and S. Mukhopadhyay. *Quantum Resource Estimation for FSR Based Symmetric Ciphers and Related Grover's Attacks*. Progress in Cryptology – INDOCRYPT '21, Springer LNCS, vol. 13143, pp. 179–198, 2021. doi:10.1007/978-3-030-92518-5_9

[6] J. M. Baker, C. Duckering, A. Hoover, and F. T. Chong. *Decomposing Quantum Generalized Toffoli with an Arbitrary Number of Ancilla*. arXiv: 1904.01671, 2019.

[7] A. Baksi, S. Bhasin, J. Breier, M. Khairallah, T. Peyrin, S. Sarkar, and S. M. Sim. *DEFAULT: Cipher Level Resistance Against Differential Fault Attack*. Advances in Cryptology – ASIACRYPT '21, Springer LNCS, vol. 13091, pp. 124–156, 2021. doi:10.1007/978-3-030-92075-3_5

[8] S. Balauca and A. Arusoaie. *Efficient Constructions for Simulating Multi Controlled Quantum Gates*. International Conference on Computational Science - ICCS '22, Springer LNCS, vol 13353, pp. 179–194, 2022. doi:10.1007/978-3-031-08760-8_16

[9] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo. *GIFT: A Small Present*. Cryptographic Hardware and Embedded Systems - CHES '17, Springer LNCS, vol. 10529, pp. 321–345, 2017. doi:10.1007/978-3-319-66787-4_16

[10] N. Bansal and M. Sinha. *k-Forrelation Optimally Separates Quantum and Classical Query Complexity*. In Proceedings of the 53rd ACM SIGACT Symposium on Theory of Computing - STOC '21, ACM, pp. 1303–1316, 2021. doi:10.1145/3406325.3451040

[11] B. Bathe, R. Anand, and S. Dutta. *Evaluation of Grover's Algorithm toward Quantum Cryptanalysis on ChaCha*. Quantum Information Processing, vol. 20, article no. 394, 2021. doi:10.1007/s11128-021-03322-7

[12] D. Bera, S. Maitra, and S. Tharrmashastha. *Efficient Quantum Algorithms Related to Autocorrelation Spectrum*. Progress in Cryptology – INDOCRYPT '19, Springer LNCS, vol. 11898, pp. 415–432, 2019. doi:10.1007/978-3-030-35423-7_21

[13] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe. *PRESENT: An Ultra-Lightweight Block Cipher*. Cryptographic Hardware and Embedded Systems - CHES '07, Springer LNCS, vol. 4727, pp. 450–466, 2007. doi:10.1007/978-3-540-74735-2_31

[14] G. Boole. *An Investigation of the Laws of Thought*. Walton & Maberly, 1854. doi:10.5962/bhl.title.29413

[15] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, and T. Yalçın. *PRINCE – A Low-Latency Block Cipher for Pervasive Computing Applications*. Advances in Cryptology – ASIACRYPT '12, Springer LNCS, vol. 7658, pp. 208–225, 2012. doi:10.1007/978-3-642-34961-4_14

[16] G. Brassard, P. Hoyer, M. Mosca, and A. Tapp. *Quantum Amplitude Amplification and Estimation*. Quantum Computation and Quantum Information, AMS Contemporary Mathematics, vol. 305, pp. 53–74, 2002. doi:10.1090/conm/305/05215

[17] S. Bravyi and A. Kitaev. *Universal Quantum Computation with Ideal Clifford Gates and Noisy Ancillas*. Physical Review A, 71, 022316, 2005. doi:10.1103/PhysRevA.71.022316

[18] C. Carlet. *Two New Classes of Bent Functions*. Advances in Cryptology — EUROCRYPT '93, Springer LNCS, vol. 765, pp. 77–101, 1994. doi:10.1007/3-540-48285-7_8

[19] C. Carlet. *Boolean Functions for Cryptography and Coding Theory*. Cambridge University Press, Cambridge, 2020. doi:10.1017/9781108606806

[20] K. Chakraborty and S. Maitra. *Application of Grover's Algorithm to Check Non-Resiliency of a Boolean Function*. Cryptography and Communications, vol. 8, no. 3, pp. 401–413, 2016. doi:10.1007/s12095-015-0156-3

[21] M. Chun, A. Baksi, and A. Chattopadhyay. *DORCIS: Depth Optimized Quantum Implementation of Substitution Boxes*. Cryptology ePrint Archive: 2023/286, 2023.

[22] B. Claudon, J. Zylberman, C. Feniou, F. Debbasch, A. Peruzzo, and J. Piquemal. *Polylogarithmic-depth Controlled-NOT Gates without Ancilla Qubits*. Nature Communication, vol. 5, art. no. 5886, 2024. doi:10.1038/s41467-024-50065-x

[23] T. Cusick and P. Stănică. *Cryptographic Boolean Functions and Applications*. Second edition, Elsevier publication, 2017. doi:10.1016/B978-0-12-374890-4.X0001-8

[24] J. Daemen and V. Rijmen. *The design of Rijndael*. Second edition, Information Security and Cryptography, Springer, 2020. doi:10.1007/978-3-662-60769-5

[25] V. A. Dasu, A. Baksi, S. Sarkar, and A. Chattopadhyay. *LIGHTER-R: Optimized Reversible Circuit Implementation For SBoxes*. 32nd IEEE International System-on-Chip Conference, pp. 260–265, 2019. doi:10.1109/SOCC46988.2019.1570548320

[26] D. Deutsch. *Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer*. Proceedings: Mathematical and Physical Sciences, Royal Society London A, vol. 400, issue 1818, pp. 97–117, 1985. doi:10.1098/rspa.1985.0070

[27] D. Deutsch and R. Jozsa. *Rapid Solution of Problems by Quantum Computation*. Proceedings: Mathematical and Physical Sciences, Royal Society London A, vol. 439, issue 1907, pp. 553–558, 1992. doi:10.1098/rspa.1992.0167

[28] J. F. Dillon. *Elementary Hadamard Difference Sets*. Ph.D. thesis, University of Maryland, 1974. doi:10.13016/M2MS3K194

[29] P. A. M. Dirac. *The Principles of Quantum Mechanics*. Fourth edition, Oxford University Press, Ely House, London, 1958.

[30] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer. *ASCON v1.2: Lightweight Authenticated Encryption and Hashing*, Journal of Cryptology, vol. 34, art. no. 33, 2021. doi:10.1007/s00145-021-09398-9

[31] S. Dutta, A. Basu Bhaumik, A. Chattopadhyay, and S. Maitra. *Optimal T-depth Quantum Circuits for Implementing Arbitrary Boolean Functions*. arXiv: 2506.01542, 2025.

[32] S. Dutta, A. Ghatak, A. Chattopadhyay, and S. Maitra. *Quantum Cryptanalysis of ZUC and Related Resource Estimation*. Progress in Cryptology – INDOCRYPT '24, Springer LNCS, vol. 15495. pp. 329–355, 2024. doi:10.1007/978-3-031-80308-6_15

[33] S. Dutta, A. Jaiswal, S. Maitra, and D. Roy. *On (Noisy) Simon's (Quantum) Algorithm for Multi-shift Boolean Functions*. Security and Privacy, ICSP '24, Springer CCIS, vol. 2489, pp. 62–76, 2025. doi:10.1007/978-3-031-90587-2_5

[34] S. Dutta and S. Maitra. *Introducing Nega-Forrelation: Quantum Algorithms in Analyzing Nega-Hadamard and Nega-crosscorrelation Spectra*. Design, Codes and Cryptography, vol. 92, pp. 863–883, 2024. doi:10.1007/s10623-023-01346-x

[35] S. Dutta, S. Maitra, and C. S. Mukherjee. *Following Forrelation – Quantum Algorithms in Exploring Boolean Functions' Spectra*. Advances in Mathematics of Communications, vol. 18, issue 1, pp. 1–25, 2024. doi:10.3934/amc.2021067

[36] S. Dutta, S. Maitra, and P. Stanica. *Extending Forrelation: Quantum Algorithms Related to Generalized Fourier-Correlation*. arXiv: 2507.07231, 2025.

[37] S. Dutta, S. Wang, A. Baksi, A. Chattopadhyay, and S. Maitra. *Exact Space-Depth Trade-offs in Multicontrolled Toffoli Decomposition*. Physical Review A, 111, 052611, 2025. doi:10.1103/PhysRevA.111.052611

[38] A. Einstein. *On a Heuristic Viewpoint Concerning the Production and Transformation of Light*. Annals of Physics, vol. 322, issue 6, pp. 132–148, 1905. doi:10.1002/andp.19053220607

[39] R. P. Feynman. *Simulating Physics with Computers*. International Journal of Theoretical Physics, vol. 21, pp. 467–488, 1982. doi:10.1007/BF02650179

[40] S. Gangopadhyay, S. Maitra, N. Sinha, and P. Stănică. *Quantum Algorithms Related to HN-Transforms of Boolean Functions*. Codes, Cryptology and Information Security - C2SI '17, Springer LNCS, vol. 10194, pp. 314–327, 2017. doi:10.1007/978-3-319-55589-8_21

[41] D. Gavinsky, M. Roetteler, and J. Roland. *Quantum Algorithm for the Boolean Hidden Shift Problem*. Computing and Combinatorics - COCOON '11, Springer LNCS, vol. 6842, pp. 158–167, 2011. doi:10.1007/978-3-642-22685-4_14

[42] C. Gidney. *Constructing Large Controlled Nots*. (Part: 1–3), 2015. url: https://algassert.com/circuits/2015/06/05/Constructing-Large-Controlled-Nots.html, Accessed: November, 2024.

[43] C. Gidney. *Halving the Cost of Quantum Addition*. Quantum, 2, 74, 2018. doi:10.22331/q-2018-06-18-74

[44] C. Gidney. *How to Factor 2048 bit RSA Integers with Less than a Million Noisy Qubits*. arXiv: 2505.15917, 2025.

[45] C. Gidney and M. Ekera. *How to Factor 2048 bit RSA Integers in 8 Hours using 20 Million Noisy Qubits*. Quantum, 5, 433, 2021. doi:10.22331/q-2021-04-15-433

[46] C. Gidney and A. G. Fowler. *Flexible Layout of Surface Code Computations Using AutoCCZ States*. arXiv: 1905.08916, 2019.

[47] C. Gidney and N. C. Jones. *A CCCZ Gate Performed with 6 T Gates*. arXiv: 2106.11513, 2021.

[48] M. Grassl, B. Langenberg, M. Rötteler, and R. Steinwandt. *Applying Grover's Algorithm to AES: Quantum Resource Estimates*. Post-Quantum Cryptography - PQCrypto '16, Springer LNCS, vol. 9606, pp. 29–43, 2016. doi:10.1007/978-3-319-29360-8_3

[49] L. K. Grover. *A fast Quantum Mechanical Algorithm for Database Search*. In Proceedings of the 28th ACM Symposium on Theory of Computing - STOC '96, ACM, pp. 212–219, 1996. doi:10.1145/237814.237866

[50] F. Hahn, A. Dahlberg, J. Eisert, and A. Pappa. *Limitations of Nearest-neighbor Quantum Networks*. Physical Review A, 106, L010401, 2022. doi:10.1103/PhysRevA.106.L010401

[51] T. Häner and M. Soeken. *Lowering the T-depth of Quantum Circuits via Logic Network Optimization*. ACM Transactions on Quantum Computing, vol. 3, issue 2, art. no. 6, pp. 1–15, 2022. doi:10.1145/3501334

[52] W. Heisenberg. *Quantum-Theoretical Re-Interpretation of Kinematic and Mechanical Relations*. Zeitschrift fur Physik, vol. 33, pp. 879–893, 1925. doi:10.1007/BF01328377

[53] Z. Huang and S. Sun. *Synthesizing Quantum Circuits of AES with Lower T-depth and Less Qubits*. Advances in Cryptology – ASIACRYPT '22, Springer LNCS, vol. 13793, pp. 614–644, 2022. doi:10.1007/978-3-031-22969-5_21

[54] Z. Huang, F. Zhang, and D. Lin. *Constructing Quantum Implementations with the Minimal T-depth or Minimal Width and Their Applications*. Advances in Cryptology – EUROCRYPT '25, Springer LNCS, vol. 15601. pp. 155–185, 2025. doi:10.1007/978-3-031-91107-1_6

[55] K. Jang, A. Baksi, H. Kim, G. Song, H. Seo, and A. Chattopadhyay. *Quantum analysis of AES*. IACR Communications in Cryptology, vol. 2, issue 1, 2025. doi:10.62056/ay11zo-3y

[56] S. Jaques, M. Naehrig, M. Roetteler, F. Virdia. *Implementing Grover Oracles for Quantum Key Search on AES and LowMC*. Advances in Cryptology – EUROCRYPT '20, Springer LNCS, vol. 12106. pp. 280–310, 2020. doi:10.1007/978-3-030-45724-2_10

[57] J. Jiang, X. Sun, S. Teng, B. Wu, K. Wu, and J. Zhang. *Optimal Space-Depth Trade-Off of CNOT Circuits in Quantum Logic Synthesis*. In Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms - SODA '20, pp. 213–229, 2020. doi:10.1137/1.9781611975994.13

[58] C. Jones. *Low-overhead Constructions for the Fault-tolerant Toffoli Gate*. Physical Review A, 87, 022328, 2013. doi:10.1103/PhysRevA.87.022328

[59] T. Khattar and C. Gidney. *Rise of Conditionally Clean Ancillae for Efficient Quantum Circuit Constructions*. Quantum, 9, 1752, 2025. doi:10.22331/q-2025-05-21-1752

[60] G. Leander and A. May. *Grover Meets Simon – Quantumly Attacking the FX-construction*. Advances in Cryptology – ASIACRYPT '17, Springer LNCS, vol. 10625, pp. 161–178, 2017. doi:10.1007/978-3-319-70697-9_6

[61] Q. Liu, B. Preneel, Z. Zhao, and M. Wang. *Improved Quantum Circuits for AES: Reducing the Depth and the Number of Qubits*. Advances in Cryptology – ASIACRYPT '23, Springer LNCS, vol. 14440, pp. 67–98, 2023. doi:10.1007/978-981-99-8727-6_3

[62] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error Correcting Codes*. North-Holland, Amsterdam, 1977.

[63] D. Maslov. *On the Advantages of Using Relative Phase Toffolis with an Application to Multiple Control Toffoli Optimization*. Physical Review A, 93, 022311, 2016. doi:10.1103/PhysRevA.93.022311

[64] A. May, L. Schlieper, and J. Schwinger. *Noisy Simon Period Finding*. Topics in Cryptology – CT-RSA '21, Springer LNCS, vol. 12704, pp. 75–99, 2021. doi:10.1007/978-3-030-75539-3_4

[65] L. A. Medina, M. G. Parker, C. Riera, and P. Stănică. *Root-Hadamard Transforms and Complementary Sequences*. Cryptography and Communications, vol. 12, pp. 1035–1049, 2020. doi:10.1007/s12095-020-00440-4

[66] D. M. Miller, R. Wille, and Z. Sasanian. *Elementary Quantum Gate Realizations for Multiple-Control Toffoli Gates*. 41st IEEE International Symposium on Multiple-Valued Logic, Finland, pp. 288–293, 2011. doi:10.1109/ISMVL.2011.54

[67] C. Moore and M. Nilsson. *Parallel Quantum Computation and Quantum Codes*. SIAM Journal on Computing, vol. 31, issue 3, pp. 799–815, 2001. doi:10.1137/S0097539799355053

[68] C. S. Mukherjee, S. Maitra, V. Gaurav, and D. Roy. *Preparing Dicke States on a Quantum Computer*. In IEEE Transactions on Quantum Engineering, vol. 1, art. no. 3102517, pp. 1–17, 2020. doi:10.1109/TQE.2020.3041479

[69] K. M. Nakanishi and S. Todo. *Systematic Construction of Multi-controlled Pauli Gate Decompositions with Optimal T-count*. arXiv: 2410.00910, 2024.

[70] J. Nie, W. Zi, and X. Sun. *Quantum Circuit for Multi-qubit Toffoli Gate with Optimal Resource*. arXiv: 2402.05053, 2024.

[71] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. 10th Anniversary Edition, Cambridge University Press, Cambridge, 2011. doi:10.1017/CBO9780511976667

[72] A. Paler, O. Oumarou, and R. Basmadjian. *On the Realistic Worst-Case Analysis of Quantum Arithmetic Circuits*. IEEE Transactions on Quantum Engineering, vol. 3, art. no. 3101311, pp. 1–11, 2022. doi:10.1109/TQE.2022.3163624

[73] M. G. Parker, A. Pott. *On Boolean Functions Which are Bent and Negabent*. Sequences, Subsequences, and Consequences, Springer LNCS, vol. 4893, pp. 9–23, 2007. doi:10.1007/978-3-540-77404-4_2

[74] M. Planck. *On the Law of Distribution of Energy in the Normal Spectrum*. Annals of Physics, vol. 309, issue 3, pp. 553–563, 1901. doi:10.1002/andp.19013090310

[75] J. Preskill. *Quantum Computing in the NISQ Era and Beyond*. Quantum 2, 79, 2018. doi:10.22331/q-2018-08-06-79

[76] M. Remaud and V. Vandaele. *Ancilla-free Quantum Adder with Sublinear Depth*. arXiv: 2501.16802, 2025.

[77] C. Riera and M. G. Parker. *Generalized Bent Criteria for Boolean Functions (I)*. IEEE Transactions on Information Theory, vol. 52, issue 9, pp. 4142–4159, 2006. doi:10.1109/TIT.2006.880069

[78] M. Rötteler. *Quantum Algorithms for Highly Non-linear Boolean Functions*. In Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms - SODA '10, pp. 448–457, 2010. doi:10.1137/1.9781611973075.37

[79] M. Saeedi and M. Pedram. *Linear-depth Quantum Circuits for n-Qubit Toffoli Gates with No Ancilla*. Physical Review A, 87, 062318, 2013. doi:10.1103/PhysRevA.87.062318

[80] P. Sarkar and S. Maitra. *Cross-Correlation Analysis of Cryptographically Useful Boolean Functions and S-Boxes*. Theory of Computing Systems, vol. 35(1), pp. 39–57, 2002. doi:10.1007/s00224-001-1019-1

[81] P. Sarkar and S. Maitra. *Construction of Nonlinear Resilient Boolean Functions Using 'Small' Affine Functions*. IEEE Transactions on Information Theory, vol. 50, issue 9, pp. 2185–2193, 2004. doi:10.1109/TIT.2004.833366

[82] K. U. Schmidt, M. G. Parker, and A. Pott. *Negabent Functions in the Maiorana–McFarland Class*. Sequences and Their Applications - SETA '08, Springer LNCS, vol. 5203, pp. 390–402, 2008. doi:10.1007/978-3-540-85912-3_34

[83] E. Schrödinger. *An Undulatory Theory of the Mechanics of Atoms and Molecules*. Physical Review, 28, 1049, 1926. doi:10.1103/PhysRev.28.1049

[84] P. Selinger. *Quantum Circuits of T-depth One*. Physical Review A, 87, 042302, 2013. doi:10.1103/PhysRevA.87.042302

[85] C. E. Shannon. *A Symbolic Analysis of Relay and Switching Circuits*. Transactions of the American Institute of Electrical Engineers, vol. 57, no. 12, pp. 713–723, 1938 doi:10.1109/T-AIEE.1938.5057767

[86] H. Shi and X. Feng. *Quantum Circuits of AES with a Low-Depth Linear Layer and a New Structure*. Advances in Cryptology – ASIACRYPT '24, Springer LNCS, vol. 15491, pp. 358–395, 2024. doi:10.1007/978-981-96-0944-4_12

[87] P. W. Shor. *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*. SIAM Journal on Computing, vol. 26, no. 5, pp. 1484–1509, 1997. doi:10.1137/S0097539795293172

[88] D. R. Simon. *On the Power of Quantum Computation*. SIAM Journal on Computing, vol. 26, no. 5, pp. 1474–1483, 1997. doi:10.1137/S0097539796298637

[89] P. Stănică. *On Weak and Strong 2^k-Bent Boolean Functions*. IEEE Transactions on Information Theory, vol. 62, issue 5, pp. 2827–2835, 2016. doi:10.1109/TIT.2016.2539971

[90] P. Stănică, S. Gangopadhyay, A. Chaturvedi, A. K. Gangopadhyay, and S. Maitra. *Nega-Hadamard Transform, Bent and Negabent Functions*. Sequences and Their Applications - SETA '10, Springer LNCS, vol. 6338, pp. 359–372, 2010. doi:10.1007/978-3-642-15874-2_31

[91] P. Stănică, S. Gangopadhyay, A. Chaturvedi, A. K. Gangopadhyay, and S. Maitra. *Investigations on Bent and Negabent Functions via the Nega-Hadamard Transform*. IEEE Transactions on Information Theory, vol. 58, issue 6, pp. 4064–4072, 2012. doi:10.1109/TIT.2012.2186785

[92] P. Stănică, B. Mandal, and S. Maitra. *The Connection between Quadratic Bent-negabent Functions and the Kerdock Code*. Applicable Algebra in Engineering, Communication and Computing, vol. 30(5), pp. 387–401, 2019. doi:10.1007/s00200-019-00380-4

[93] A. Tal. *Towards Optimal Separations between Quantum and Randomized Query Complexities*. IEEE 61st Annual Symposium on Foundations of Computer Science - FOCS '20, Durham, USA, pp. 228–239, 2020. doi:10.1109/FOCS46700.2020.00030

[94] C. Tang, C. Xiang, Y. Qi, and K. Feng, *Complete characterization of generalized bent and 2^k-bent Boolean functions*. IEEE Transactions on Information Theory, vol. 63, issue 7, pp. 4668–4674, 2017. doi:10.1109/TIT.2017.2686987

[95] S. Wang, A. Baksi, and A. Chattopadhyay. *A Higher Radix Architecture for Quantum Carry-lookahead Adder*. Scientific Reports, vol. 13, art. no. 16338, 2023. doi:10.1038/s41598-023-41122-4

[96] S. Wang, A. Mondal, and A. Chattopadhyay. *Optimal Toffoli-Depth Quantum Adder*. ACM Transactions on Quantum Computing, 2025. doi:10.1145/3743691

[97] S. Wang, E. Lim, X. Li, J. Feng, and A. Chattopadhyay. *Minimum Depth Quantum Modular Addition Through Carry-Save Architecture*. IFIP/IEEE 32nd International Conference on Very Large Scale Integration (VLSI-SoC), pp. 1–6, 2024. doi:10.1109/VLSI-SoC62099.2024.10767796

[98] G. -Z. Xiao and J. L. Massey. *A Spectral Characterization of Correlation Immune Combining Functions*. IEEE Transactions on Information Theory, vol. 34, issue 3, pp. 569–571, 1988. doi:10.1109/18.6037

[99] Y. Yuan, W. Wu, T. Shi, L. Zhang, and Y. Zhang. *A Framework to Improve the Implementations of Linear Layers*. IACR Trans. Symmetric Cryptol., vol. 2024, no. 2, pp. 322–347, 2024. doi:10.46586/tosc.v2024.i2.322-347

[100] M. Zhang, T. Shi, W. Wu, and H. Sui. *Optimized Quantum Circuit of AES With Interlacing-Uncompute Structure*. IEEE Transactions on Computers, vol. 73, no. 11, pp. 2563–2575, 2024. doi:10.1109/TC.2024.3449094

[101] W. Zi, S. Wang, H. Kim, X. Sun, A. Chattopadhyay, and P. Rebentrost. *Efficient Quantum Circuits for Machine Learning Activation Functions including Constant T-depth ReLU*. Physical Review Research, 6, 043048, 2024. doi:10.1103/PhysRevResearch.6.043048